
PyFFI

Release 2.2.4.dev4

Amorilia

Jan 06, 2020

CONTENTS

1	PyFFI	3
1.1	Download	3
1.2	Developing	3
1.3	Testing	4
1.4	Documentation	4
1.5	Examples	4
1.6	Questions? Suggestions?	4
1.7	Documentation	4
1.8	Indices and tables	267
	Python Module Index	269
	Index	271

Release 2.2.4.dev4

Date Jan 06, 2020

The Python File Format Interface, briefly PyFFI, is an open source Python library for processing block structured binary files:

- **Simple:** Reading, writing, and manipulating complex binary files in a Python environment is easy! Currently, PyFFI supports the NetImmerse/Gamebryo NIF and KFM formats, CryTek's CGF format, the FaceGen EGM format, the DDS format, and the TGA format.
- **Batteries included:** Many tools for files used by 3D games, such as optimizers, stripifier, tangent space calculator, 2d/3d hull algorithms, inertia calculator, as well as a general purpose file editor QSkope (using [PyQt4](#)), are included.
- **Modular:** Its highly modular design makes it easy to add support for new formats, and also to extend existing functionality.

1.1 Download

Get PyFFI from [Github](#), or install it with:

```
easy_install -U PyFFI
```

or:

```
pip3 install PyFFI
```

1.2 Developing

To get the latest (but possibly unstable) code, clone PyFFI from its [Git repository](#):

```
git clone --recursive git://github.com/nifttools/pyffi.git
virtualenv -p python3 venv
source venv/bin/activate
pip install -r requirements/requirements-dev.txt
```

Be sure to use the `--recursive` flag to ensure that you also get all of the submodules.

If you wish to code on PyFFI and send your contributions back upstream, get a [github account](#) and [fork PyFFI](#).

1.3 Testing

We love tests, they help guarantee that things keep working they way they should. You can run them yourself with the following:

```
source venv/bin/activate
nosetest -v test
```

or:

```
source venv/bin/activate
py.test -v tests
```

1.4 Documentation

All our documentation is written in ReST and can be generated into HTML, LaTeX, PDF and more thanks to Sphinx. You can generate it yourself:

```
source venv/bin/activate
cd docs
make html -a
```

1.5 Examples

- The [Blender NIF Plugin](#)
- QSkope PyFFI's general purpose file editor.
- The niftoaster (PyFFI's "swiss army knife") can for instance [optimize NIF files](#), and much more.

1.6 Questions? Suggestions?

- Open an issue at the [issue tracker](#).

1.7 Documentation

1.7.1 Introduction

Example and Problem Description

Consider an application which processes images stored in for instance the Targa format:

```
>>> # read the file
>>> stream = open("tests/tga/test.tga", "rb")
>>> data = bytearray(stream.read()) # read bytes
>>> stream.close()
>>> # do something with the data...
```

(continues on next page)

(continued from previous page)

```

>>> data[8] = 20 # change x origin
>>> data[10] = 20 # change y origin
>>> # etc... until we are finished processing the data
>>> # write the file
>>> from tempfile import TemporaryFile
>>> stream = TemporaryFile()
>>> dummy = stream.write(data) # py3k returns number of bytes written
>>> stream.close()

```

This methodology will work for any file format, but it is usually not very convenient. For complex file formats, the *do something with the data* part of the program would be necessarily quite complicated for the programmer. For this reason, it is convenient to convert the data (a sequence of bytes) into an organized collection of Python objects (a class suits this purpose perfectly) that clearly reveal what is stored in the data. Such organized collection is called an *interface*:

```

>>> import struct
>>> from tempfile import TemporaryFile
>>> class TgaFile:
...     """A simple class for reading and writing Targa files."""
...     def read(self, filename):
...         """Read tga file from stream."""
...         stream = open(filename, "rb")
...         self.image_id_length, self.colormap_type, self.image_type, \
...         self.colormap_index, self.colormap_length, self.colormap_size, \
...         self.x_origin, self.y_origin, self.width, self.height, \
...         self.pixel_size, self.flags = struct.unpack("<BBBHHBHHHHBB",
...                                                     stream.read(18))
...         self.image_id = stream.read(self.image_id_length)
...         if self.colormap_type:
...             self.colormap = [
...                 stream.read(self.colormap_size >> 3)
...                 for i in range(self.colormap_length)]
...         else:
...             self.colormap = []
...         self.image = [[stream.read(self.pixel_size >> 3)
...                         for i in range(self.width)]
...                        for j in range(self.height)]
...         stream.close()
...     def write(self, filename=None):
...         """Read tga file from stream."""
...         if filename:
...             stream = open(filename, "wb")
...         else:
...             stream = TemporaryFile()
...         stream.write(struct.pack("<BBBHHBHHHHBB",
...                                 self.image_id_length, self.colormap_type, self.image_type,
...                                 self.colormap_index, self.colormap_length,
...                                 self.colormap_size,
...                                 self.x_origin, self.y_origin, self.width, self.height,
...                                 self.pixel_size, self.flags))
...         stream.write(self.image_id)
...         for entry in self.colormap:
...             stream.write(entry)
...         for line in self.image:
...             for pixel in line:
...                 stream.write(pixel)

```

(continues on next page)

(continued from previous page)

```
...         stream.close()
>>> data = TgaFile()
>>> # read the file
>>> data.read("tests/tga/test.tga")
>>> # do something with the data...
>>> data.x_origin = 20
>>> data.y_origin = 20
>>> # etc... until we are finished processing the data
>>> # write the file
>>> data.write()
```

The reading and writing part of the code has become a lot more complicated, but the benefit is immediately clear: instead of working with a sequence of bytes, we can directly work with the members of our `TgaFile` class, and our code no longer depends on how exactly image data is organized in a Targa file. In other words, our code can now use the semantics of the `TgaFile` class, and is consequently much easier to understand and to maintain.

In practice, however, when taking the above approach as given, the additional code that enables this semantic translation is often difficult to maintain, for the following reasons:

- **Duplication:** Any change in the reader part must be reflected in the writer part, and vice versa. Moreover, the same data types tend to occur again and again, leading to nearly identical code for each read/write operation. A partial solution to this problem would be to create an additional class for each data type, each with its read and write method.
- **No validation:** What if `test/tga/test.tga` is not a Targa file at all, or is corrupted? What if `image_id` changes length but `image_id_length` is not updated accordingly? Can we catch such bugs and prevent data to become corrupted?
- **Boring:** Writing *interface* code gets boring very quickly.

What is PyFFI?

PyFFI aims to solve all of the above problems:

- The *interface* classes are *generated at runtime*, from an easy to maintain description of the file format. The generated classes provides semantic access to *all* information in the files.
- Validation is automatically enforced by the generated classes, except in a few rare cases when automatic validation might cause substantial overhead. These cases are well documented and simply require an explicit call to the validation method.
- The generated classes can easily be extended with additional class methods, for instance to provide common calculations (for example: converting a single pixel into greyscale).
- Very high level functions can be implemented as *spells* (for example: convert a height map into a normal map).

1.7.2 Installation

Requirements

To run PyFFI's graphical file editor QSkope, you need `PyQt4`.

Using the Windows installer

Simply download and run the Windows installer provided in our [Releases](#).

Manual installation

If you install PyFFI manually, and you already have an older version of PyFFI installed, then you **must** uninstall (see *Uninstall*) the old version before installing the new one.

Installing via setuptools

If you have [setuptools](#) installed, simply run:

```
easy_install -U PyFFI
```

at the command prompt.

Installing from source package

First, get the [source package](#). Untar or unzip the source package via either:

```
tar xfvz PyFFI-x.x.x.tar.gz
```

or:

```
unzip PyFFI-x.x.x.zip
```

Change to the PyFFI directory and run the setup script:

```
cd PyFFI-x.x.x  
python setup.py install
```

Uninstall

You can uninstall PyFFI manually simply by deleting the `pyffi` folder from your Python's `site-packages` folder, which is typically at:

```
C:\Python25\Lib\site-packages\pyffi
```

or:

```
/usr/lib/python2.5/site-packages/pyffi
```

1.7.3 pyffi — Interfacing block structured files

pyffi.formats — File format interfaces

When experimenting with any of the supported file formats, you can specify an alternate location where you store your modified format description by means of an environment variable. For instance, to tell the library to use your version of `cgf.xml`, set the `CGFXMLPATH` environment variable to the directory where `cgf.xml` can be found. The environment variables `NIFXMLPATH`, `KFXMLPATH`, `DDSXMLPATH`, and `TGAXMLPATH` work similarly.

Supported formats

pyffi.formats.bsa — Bethesda Archive (.bsa)

Warning: This module is still a work in progress, and is not yet ready for production use.

A .bsa file is an archive format used by Bethesda (Morrowind, Oblivion, Fallout 3).

Implementation

```
class pyffi.formats.bsa.BsaFormat
    Bases: pyffi.object_models.xml.FileFormat
    This class implements the BSA format.

class BZString (**kwargs)
    Bases: pyffi.object_models.common.SizedString

    get_size (data=None)
        Return number of bytes this type occupies in a file.
        Returns Number of bytes.

    read (stream, data=None)
        Read string from stream.
        Parameters stream (file) – The stream to read from.

    write (stream, data=None)
        Write string to stream.
        Parameters stream (file) – The stream to write to.

Data
    alias of Header

class FileVersion (**kwargs)
    Bases: pyffi.object_models.common.UInt
    Basic type which implements the header of a BSA file.

    get_size (data=None)
        Return number of bytes the header string occupies in a file.
        Returns Number of bytes.

    read (stream, data)
        Read header string from stream and check it.
        Parameters stream (file) – The stream to read from.
```

```

write (stream, data)
    Write the header string to stream.
    Parameters stream (file) – The stream to write to.

class Hash (**kwargs)
    Bases: pyffi.object_models.common.UInt64

    get_detail_display ()
        Return an object that can be used to display the instance.

class Header (template=None, argument=None, parent=None)
    Bases: pyffi.formats.bsa._Header, pyffi.object_models.Data

    A class to contain the actual bsa data.

    inspect (stream)
        Quickly checks if stream contains BSA data, and reads the header.
        Parameters stream (file) – The stream to inspect.

    inspect_quick (stream)
        Quickly checks if stream contains BSA data, and gets the version, by looking at the first 8 bytes.
        Parameters stream (file) – The stream to inspect.

    read (stream)
        Read a bsa file.
        Parameters stream (file) – The stream from which to read.

    write (stream)
        Write a bsa file.
        Parameters stream (file) – The stream to which to write.

UInt32
    alias of pyffi.object_models.common.UInt

class ZString (**kwargs)
    Bases: pyffi.object_models.xml.basic.BasicBase, pyffi.object_models.editable.EditableLineEdit

    String of variable length (null terminated).

```

```

>>> from tempfile import TemporaryFile
>>> f = TemporaryFile()
>>> s = ZString()
>>> if f.write('abcdefghijklmnopqrst\x00'.encode("ascii")): pass # b'abc...'
>>> if f.seek(0): pass # ignore result for py3k
>>> s.read(f)
>>> str(s)
'abcdefghijklmnopqrst'
>>> if f.seek(0): pass # ignore result for py3k
>>> s.set_value('Hi There!')
>>> s.write(f)
>>> if f.seek(0): pass # ignore result for py3k
>>> m = ZString()
>>> m.read(f)
>>> str(m)
'Hi There!'

```

```

get_hash (data=None)
    Return a hash value for this string.
    Returns An immutable object that can be used as a hash.

```

get_size (*data=None*)

Return number of bytes this type occupies in a file.

Returns Number of bytes.

get_value ()

Return the string.

Returns The stored string.

Return type C{bytes}

read (*stream, data=None*)

Read string from stream.

Parameters **stream** (*file*) – The stream to read from.

set_value (*value*)

Set string to C{value}.

Parameters **value** (*str* (will be encoded as default) or C{bytes}) – The value to assign.

write (*stream, data=None*)

Write string to stream.

Parameters **stream** (*file*) – The stream to write to.

static version_number (*version_str*)

Converts version string into an integer.

Parameters **version_str** (*str*) – The version string.

Returns A version integer.

```
>>> BsaFormat.version_number('103')
103
>>> BsaFormat.version_number('XXX')
-1
```

Regression tests

Read a BSA file

```
>>> # check and read bsa file
>>> from os.path import dirname
>>> dirpath = __file__
>>> for i in range(4): #recurse up to root repo dir
...     dirpath = dirname(dirpath)
>>> repo_root = dirpath
>>> format_root = os.path.join(repo_root, 'tests', 'formats', 'bsa')
>>> stream = open(os.path.join(format_root, 'test.bsa'), 'rb')
>>> data = BsaFormat.Data()
>>> data.inspect_quick(stream)
>>> data.version
103
>>> data.inspect(stream)
>>> data.folders_offset
36
>>> hex(data.archive_flags.get_attributes_values(data))
'0x703'
>>> data.num_folders
1
>>> data.num_files
```

(continues on next page)

(continued from previous page)

```

7
>>> #data.read(stream)
>>> # TODO check something else...

```

Parse all BSA files in a directory tree

```

>>> for stream, data in BsaFormat.walkData(format_root):
...     try:
...         # the replace call makes the doctest also pass on windows
...         os_path = stream.name
...         split = (os_path.split(os.sep))[-4:]
...         rejoin = os.path.join(*split).replace(os.sep, "/")
...         print("reading %s" % rejoin)
...         data.read(stream)
...     except Exception:
...         print(
...             "Warning: read failed due corrupt file,"
...             " corrupt format description, or bug.")
reading tests/formats/bsa/test.bsa

```

Create an BSA file from scratch and write to file

```

>>> data = BsaFormat.Data()
>>> # TODO store something...
>>> from tempfile import TemporaryFile
>>> stream = TemporaryFile()
>>> #data.write(stream)

```

pyffi.formats.cgf — Crytek (.cgf and .cga)

Implementation

```

class pyffi.formats.cgf.CgfFormat
    Bases: pyffi.object_models.xml.FileFormat
    Stores all information about the cgf file format.

class AbstractMtlChunk (template=None, argument=None, parent=None)
    Bases: pyffi.formats.cgf.Chunk
    Common parent for MtlChunk and MtlNameChunk.

class AbstractObjectChunk (template=None, argument=None, parent=None)
    Bases: pyffi.formats.cgf.Chunk
    Common parent for HelperChunk and MeshChunk.

class BoneLink (template=None, argument=None, parent=None)
    Bases: pyffi.object_models.xml.struct_.StructBase
    A bone link.

```

exception CgfErrorBases: `Exception`

Exception for CGF specific errors.

class Chunk (*template=None, argument=None, parent=None*)Bases: `pyffi.formats.cgf._Chunk, object`**apply_scale** (*scale*)

Apply scale factor on data.

tree (*block_type=None, follow_all=True*)A generator for parsing all blocks in the tree (starting from and including `C{self}`).**Parameters**

- **block_type** – If not `None`, yield only blocks of the type `C{block_type}`.
- **follow_all** – If `C{block_type}` is not `None`, then if this is `True` the function will parse the whole tree. Otherwise, the function will not follow branches that start by a non-`C{block_type}` block.

class ChunkHeader (*template=None, argument=None, parent=None*)Bases: `pyffi.object_models.xml.struct_.StructBase`

A CGF chunk header.

class ChunkTable (*template=None, argument=None, parent=None*)Bases: `pyffi.formats.cgf._ChunkTable, object`**get_chunk_types** ()

Iterate all chunk types (in the form of Python classes) referenced in this table.

class ChunkType (***kwargs*)Bases: `pyffi.object_models.xml.enum.EnumBase`

An unsigned 32-bit integer, describing the chunk type.

class ChunkVersion (***kwargs*)Bases: `pyffi.object_models.common.UInt`

The version of a particular chunk, or the version of the chunk table.

class Data (*filetype=4294901760, game='Far Cry'*)Bases: `pyffi.object_models.Data`

A class to contain the actual cgf data.

Note that `L{versions}` and `L{chunk_table}` are not automatically kept in sync with the `L{chunks}`, but they are resynchronized when calling `L{write}`.

Variables

- **game** – The cgf game.
- **header** – The cgf header.
- **chunks** – List of chunks (the actual data).
- **versions** – List of chunk versions.

get_detail_child_names (*edge_filter=(True, True)*)

Generator which yields all child names of this item in the detail view.

Override this method if the node has children.

Returns Generator for detail tree child names.**Return type** generator yielding `strs`

get_detail_child_nodes (*edge_filter=(True, True)*)
 Generator which yields all children of this item in the detail view (by default, all acyclic and active ones).
 Override this method if the node has children.
Parameters **edge_filter** (*EdgeFilter* or type (*None*)) – The edge type to include.
Returns Generator for detail tree child nodes.
Return type generator yielding *DetailNodes*

get_global_child_nodes (*edge_filter=(True, True)*)
 Returns chunks without parent.

inspect (*stream*)
 Quickly checks whether the stream appears to contain cgf data, and read the cgf header and chunk table. Resets stream to original position.
 Call this function if you only need to inspect the header and chunk table.
Parameters **stream** (*file*) – The file to inspect.

inspect_version_only (*stream*)
 This function checks the version only, and is faster than the usual inspect function (which reads the full chunk table). Sets the L{header} and L{game} instance variables if the stream contains a valid cgf file.
 Call this function if you simply wish to check that a file is a cgf file without having to parse even the header.
Raises **ValueError** – If the stream does not contain a cgf file.
Parameters **stream** (*file*) – The stream from which to read.

read (*stream*)
 Read a cgf file. Does not reset stream position.
Parameters **stream** (*file*) – The stream from which to read.

replace_global_node (*oldbranch, newbranch, edge_filter=(True, True)*)
 Replace a particular branch in the graph.

update_versions ()
 Update L{versions} for the given chunks and game.

write (*stream*)
 Write a cgf file. The L{header} and L{chunk_table} are recalculated from L{chunks}. Returns number of padding bytes written (this is for debugging purposes only).
Parameters **stream** (*file*) – The stream to which to write.
Returns Number of padding bytes written.

class DataStreamChunk (*template=None, argument=None, parent=None*)
 Bases: *pyffi.formats.cgf._DataStreamChunk, object*
apply_scale (*scale*)
 Apply scale factor on data.

class ExportFlagsChunk (*template=None, argument=None, parent=None*)
 Bases: *pyffi.formats.cgf.Chunk*
 Export information.

class FRGB (*template=None, argument=None, parent=None*)
 Bases: *pyffi.object_models.xml.struct_.StructBase*
 R32G32B32 (float).

```
class Face (template=None, argument=None, parent=None)
    Bases: pyffi.object_models.xml.struct_.StructBase
    A mesh face.

class FileOffset (**kwargs)
    Bases: pyffi.object_models.common.Int
    Points to a position in a file.

class FileSignature (**kwargs)
    Bases: pyffi.object_models.xml.basic.BasicBase
    The CryTek file signature with which every cgf file starts.

    get_hash (data=None)
        Return a hash value for the signature.
        Returns An immutable object that can be used as a hash.

    get_size (data=None)
        Return number of bytes that the signature occupies in a file.
        Returns Number of bytes.

    get_value ()
        Get signature.
        Returns The signature.

    read (stream, data)
        Read signature from stream.
        Parameters stream (file) – The stream to read from.

    set_value (value)
        Not implemented.

    write (stream, data)
        Write signature to stream.
        Parameters stream (file) – The stream to read from.

class FileType (**kwargs)
    Bases: pyffi.object_models.xml.enum.EnumBase
    An unsigned 32-bit integer, describing the file type.

class GeomNameListChunk (template=None, argument=None, parent=None)
    Bases: pyffi.object_models.xml.struct_.StructBase
    Obsolete, not decoded.

class Header (template=None, argument=None, parent=None)
    Bases: pyffi.object_models.xml.struct_.StructBase
    The CGF header.

class IRGB (template=None, argument=None, parent=None)
    Bases: pyffi.object_models.xml.struct_.StructBase
    R8G8B8.

class IRGBA (template=None, argument=None, parent=None)
    Bases: pyffi.object_models.xml.struct_.StructBase
    R8G8B8A8.

class InitialPosMatrix (template=None, argument=None, parent=None)
    Bases: pyffi.object_models.xml.struct_.StructBase
```

A bone initial position matrix.

```
class MRMChunk (template=None, argument=None, parent=None)
    Bases: pyffi.object_models.xml.struct_.StructBase
```

Obsolete, not decoded.

```
class Matrix33 (template=None, argument=None, parent=None)
    Bases: pyffi.formats.cgf._Matrix33, object
```

```
as_list ()
    Return matrix as 3x3 list.
```

```
as_tuple ()
    Return matrix as 3x3 tuple.
```

```
get_copy ()
    Return a copy of the matrix.
```

```
get_determinant ()
    Return determinant.
```

```
get_inverse ()
    Get inverse (assuming is_scale_rotation is true!).
```

```
get_scale ()
    Gets the scale (assuming is_scale_rotation is true!).
```

```
get_scale_quat ()
    Decompose matrix into scale and quaternion.
```

```
get_scale_rotation ()
    Decompose the matrix into scale and rotation, where scale is a float and rotation is a C{Matrix33}.
    Returns a pair (scale, rotation).
```

```
get_transpose ()
    Get transposed of the matrix.
```

```
is_identity ()
    Return True if the matrix is close to identity.
```

```
is_rotation ()
    Returns True if the matrix is a rotation matrix (a member of SO(3)).
```

```
is_scale_rotation ()
    Returns true if the matrix decomposes nicely into scale * rotation.
```

```
set_identity ()
    Set to identity matrix.
```

```
set_scale_rotation (scale, rotation)
    Compose the matrix as the product of scale * rotation.
```

```
class Matrix44 (template=None, argument=None, parent=None)
    Bases: pyffi.formats.cgf._Matrix44, object
```

```
as_list ()
    Return matrix as 4x4 list.
```

```
as_tuple ()
    Return matrix as 4x4 tuple.
```

```
get_copy ()
    Create a copy of the matrix.
```

get_inverse (*fast=True*)
Calculates inverse (fast assumes is_scale_rotation_translation is True).

get_matrix_33 ()
Returns upper left 3x3 part.

get_translation ()
Returns lower left 1x3 part.

is_identity ()
Return True if the matrix is close to identity.

set_identity ()
Set to identity matrix.

set_matrix_33 (*m*)
Sets upper left 3x3 part.

set_rows (*row0, row1, row2, row3*)
Set matrix from rows.

set_translation (*translation*)
Returns lower left 1x3 part.

sup_norm ()
Calculate supremum norm of matrix (maximum absolute value of all entries).

class Mt1ListChunk (*template=None, argument=None, parent=None*)
Bases: `pyffi.object_models.xml.struct_.StructBase`
Obsolete, not decoded.

class PatchMeshChunk (*template=None, argument=None, parent=None*)
Bases: `pyffi.object_models.xml.struct_.StructBase`
Obsolete, not decoded.

class Ptr (***kwargs*)
Bases: `pyffi.formats.cgf.Ref`
Reference to a chunk, down the hierarchy.

get_refs (*data=None*)
Ptr does not point down, so get_refs returns empty list.
Returns `C{[]}`

class Quat (*template=None, argument=None, parent=None*)
Bases: `pyffi.object_models.xml.struct_.StructBase`
A quaternion (x,y,z,w).

class Ref (***kwargs*)
Bases: `pyffi.object_models.xml.basic.BasicBase`
Reference to a chunk, up the hierarchy.

fix_links (*data*)
Resolve chunk index into a chunk.
Keyword Arguments **block_dct** – Dictionary mapping block index to block.

get_hash (*data=None*)
Return a hash value for the chunk referred to.
Returns An immutable object that can be used as a hash.

get_links (*data=None*)
 Return the chunk reference.
Returns Empty list if no reference, or single item list containing the reference.

get_refs (*data=None*)
 Return the chunk reference.
Returns Empty list if no reference, or single item list containing the reference.

get_size (*data=None*)
 Return number of bytes this type occupies in a file.
Returns Number of bytes.

get_value ()
 Get chunk being referred to.
Returns The chunk being referred to.

read (*stream, data*)
 Read chunk index.
Parameters **stream** (*file*) – The stream to read from.

set_value (*value*)
 Set chunk reference.
Parameters **value** (*L{Cgfformat.Chunk}*) – The value to assign.

write (*stream, data*)
 Write chunk index.
Parameters **stream** (*file*) – The stream to write to.

class ScenePropsChunk (*template=None, argument=None, parent=None*)
 Bases: `pyffi.object_models.xml.struct_.StructBase`
 Not decoded. Nowhere used?

class SizedString (***kwargs*)
 Bases: `pyffi.object_models.xml.basic.BasicBase`, `pyffi.object_models.editable.EditableLineEdit`
 Basic type for strings. The type starts with an unsigned int which describes the length of the string.

```
>>> from tempfile import TemporaryFile
>>> f = TemporaryFile()
>>> from pyffi.object_models import FileFormat
>>> data = FileFormat.Data()
>>> s = SizedString()
>>> if f.write('\x07\x00\x00\x00abcdefg'.encode("ascii")): pass # ignore_
↳ result for py3k
>>> if f.seek(0): pass # ignore result for py3k
>>> s.read(f, data)
>>> str(s)
'abcdefg'
>>> if f.seek(0): pass # ignore result for py3k
>>> s.set_value('Hi There')
>>> s.write(f, data)
>>> if f.seek(0): pass # ignore result for py3k
>>> m = SizedString()
>>> m.read(f, data)
>>> str(m)
'Hi There'
```

get_hash (*data=None*)
 Return a hash value for this string.

Returns An immutable object that can be used as a hash.

get_size (*data=None*)

Return number of bytes this type occupies in a file.

Returns Number of bytes.

get_value ()

Return the string.

Returns The stored string.

read (*stream, data*)

Read string from stream.

Parameters **stream** (*file*) – The stream to read from.

set_value (*value*)

Set string to C{value}.

Parameters **value** (*str*) – The value to assign.

write (*stream, data*)

Write string to stream.

Parameters **stream** (*file*) – The stream to write to.

String

alias of `pyffi.object_models.common.ZString`

class String128 (***kwargs*)

Bases: `pyffi.object_models.common.FixedString`

String of fixed length 128.

class String16 (***kwargs*)

Bases: `pyffi.object_models.common.FixedString`

String of fixed length 16.

class String256 (***kwargs*)

Bases: `pyffi.object_models.common.FixedString`

String of fixed length 256.

class String32 (***kwargs*)

Bases: `pyffi.object_models.common.FixedString`

String of fixed length 32.

class String64 (***kwargs*)

Bases: `pyffi.object_models.common.FixedString`

String of fixed length 64.

class Tangent (*template=None, argument=None, parent=None*)

Bases: `pyffi.object_models.xml.struct_.StructBase`

Tangents. Divide each component by 32767 to get the actual value.

class UV (*template=None, argument=None, parent=None*)

Bases: `pyffi.object_models.xml.struct_.StructBase`

Texture coordinate.

class UVFace (*template=None, argument=None, parent=None*)

Bases: `pyffi.object_models.xml.struct_.StructBase`

A texture face (vertex indices).

```
class UnknownAAFC0005Chunk (template=None, argument=None, parent=None)
    Bases: pyffi.formats.cgf.Chunk
```

Unknown. An extra block written by the XSI exporter.

```
class Vector3 (template=None, argument=None, parent=None)
    Bases: pyffi.formats.cgf._Vector3, object
```

```
bool
    alias of pyffi.object_models.common.Bool
```

```
byte
    alias of pyffi.object_models.common.Byte
```

```
char
    alias of pyffi.object_models.common.Char
```

```
float
    alias of pyffi.object_models.common.Float
```

```
int
    alias of pyffi.object_models.common.Int
```

```
short
    alias of pyffi.object_models.common.Short
```

```
ubyte
    alias of pyffi.object_models.common.UByte
```

```
uint
    alias of pyffi.object_models.common.UInt
```

```
ushort
    alias of pyffi.object_models.common.USHort
```

```
static version_number (version_str)
    Converts version string into an integer.
```

Parameters **version_str** (*str*) – The version string.

Returns A version integer.

```
>>> hex(CgfFormat.version_number('744'))
'0x744'
```

Regression tests

Read a CGF file

```
>>> # get file version and file type, and read cgf file
>>> from os.path import dirname
>>> dirpath = __file__
>>> for i in range(4): #recurse up to root repo dir
...     dirpath = dirname(dirpath)
>>> repo_root = dirpath
>>> format_root = os.path.join(repo_root, 'tests', 'formats', 'cgf')
>>> stream = open(os.path.join(format_root, 'test.cgf'), 'rb')
>>> data = CgfFormat.Data()
>>> # read chunk table only
```

(continues on next page)

(continued from previous page)

```

>>> data.inspect(stream)
>>> # check chunk types
>>> list(chunktype.__name__ for chunktype in data.chunk_table.get_chunk_types())
['SourceInfoChunk', 'TimingChunk']
>>> data.chunks # no chunks yet
[]
>>> # read full file
>>> data.read(stream)
>>> # get all chunks
>>> for chunk in data.chunks:
...     print(chunk)
<class '...SourceInfoChunk'> instance at ...
* source_file : <None>
* date : Fri Sep 28 22:40:44 2007
* author : blender@BLENDER

<class '...TimingChunk'> instance at ...
* secs_per_tick : 0.0002083333...
* ticks_per_frame : 160
* global_range :
  <class '...RangeEntity'> instance at ...
  * name : GlobalRange
  * start : 0
  * end : 100
* num_sub_ranges : 0

```

Parse all CGF files in a directory tree

```

>>> for stream, data in CgfFormat.walkData(format_root):
...     try:
...         # the replace call makes the doctest also pass on windows
...         os_path = stream.name
...         split = (os_path.split(os.sep))[-4:]
...         rejoin = os.path.join(*split).replace(os.sep, "/")
...         print("reading %s" % rejoin)
...         data.read(stream)
...     except Exception:
...         print("Warning: read failed due corrupt file, corrupt format description,
↳ or bug.")
...         print(len(data.chunks))
...         # do something with the chunks
...         for chunk in data.chunks:
...             chunk.apply_scale(2.0)
reading tests/formats/cgf/invalid.cgf
Warning: read failed due corrupt file, corrupt format description, or bug.
0
reading tests/formats/cgf/monkey.cgf
14
reading tests/formats/cgf/test.cgf
2
reading tests/formats/cgf/vcols.cgf
6

```


Create a CGF file from scratch

```
>>> from pyffi.formats.cgf import CgfFormat
>>> node1 = CgfFormat.NodeChunk()
>>> node1.name = "hello"
>>> node2 = CgfFormat.NodeChunk()
>>> node1.num_children = 1
>>> node1.children.update_size()
>>> node1.children[0] = node2
>>> node2.name = "world"
>>> from tempfile import TemporaryFile
>>> stream = TemporaryFile()
>>> data = CgfFormat.Data() # default is far cry
>>> data.chunks = [node1, node2]
>>> # note: write returns number of padding bytes
>>> data.write(stream)
0
>>> # py3k returns 0 on seek; this hack removes return code from doctest
>>> if stream.seek(0): pass
>>> data.inspect_version_only(stream)
>>> hex(data.header.version)
'0x744'
>>> data.read(stream)
>>> # get all chunks
>>> for chunk in data.chunks:
...     print(chunk)
<class 'pyffi.formats.cgf.NodeChunk'> instance at 0x...
* name : hello
* object : None
* parent : None
* num_children : 1
* material : None
* is_group_head : False
* is_group_member : False
* reserved_1 :
  <class 'pyffi.object_models.xml.array.Array'> instance at 0x...
  0: 0
  1: 0
* transform :
  [ 0.000 0.000 0.000 0.000 ]
  [ 0.000 0.000 0.000 0.000 ]
  [ 0.000 0.000 0.000 0.000 ]
  [ 0.000 0.000 0.000 0.000 ]
* pos : [ 0.000 0.000 0.000 ]
* rot :
  <class 'pyffi.formats.cgf.Quat'> instance at 0x...
  * x : 0.0
  * y : 0.0
  * z : 0.0
  * w : 0.0
* scl : [ 0.000 0.000 0.000 ]
* pos_ctrl : None
* rot_ctrl : None
* scl_ctrl : None
* property_string : <None>
* children :
  <class 'pyffi.object_models.xml.array.Array'> instance at 0x...
```

(continues on next page)

(continued from previous page)

```

0: <class 'pyffi.formats.cgf.NodeChunk'> instance at 0x...

<class 'pyffi.formats.cgf.NodeChunk'> instance at 0x...
* name : world
* object : None
* parent : None
* num_children : 0
* material : None
* is_group_head : False
* is_group_member : False
* reserved_1 :
    <class 'pyffi.object_models.xml.array.Array'> instance at 0x...
    0: 0
    1: 0
* transform :
    [ 0.000  0.000  0.000  0.000 ]
    [ 0.000  0.000  0.000  0.000 ]
    [ 0.000  0.000  0.000  0.000 ]
    [ 0.000  0.000  0.000  0.000 ]
* pos : [ 0.000  0.000  0.000 ]
* rot :
    <class 'pyffi.formats.cgf.Quat'> instance at 0x...
    * x : 0.0
    * y : 0.0
    * z : 0.0
    * w : 0.0
* scl : [ 0.000  0.000  0.000 ]
* pos_ctrl : None
* rot_ctrl : None
* scl_ctrl : None
* property_string : <None>
* children : <class 'pyffi.object_models.xml.array.Array'> instance at 0x...

```

pyffi.formats.dae — COLLADA (.dae)

Warning: This module is not yet fully implemented, and is certainly not yet useful in its current state.

Implementation

class pyffi.formats.dae.DaeFormat

Bases: pyffi.object_models.xsd.FileFormat

This class implements the DAE format.

class Data (version=17039616)

Bases: pyffi.object_models.Data

A class to contain the actual collada data.

getVersion()

Get the collada version, as integer (for instance, 1.4.1 would be 0x01040100).

Returns The version, as integer.

inspect (*stream*)

Quickly checks whether the stream appears to contain collada data. Resets stream to original position. If the stream turns out to be collada, `L{getVersion}` is guaranteed to return the version.

Call this function if you simply wish to check that a file is a collada file without having to parse it completely.

Parameters **stream** (*file*) – The file to inspect.

Returns `True` if stream is collada, `False` otherwise.

read (*stream*)

Read collada data from stream.

Parameters **stream** (*file*) – The file to read from.

write (*stream*)

Write collada data to stream.

Parameters **stream** (*file*) – The file to write to.

Regression tests

Create a DAE file

```
>>> daedata = DaeFormat.Data()
>>> print(daedata.collada)
<...Collada object at ...>
```

Read a DAE file

```
>>> from os.path import dirname
>>> dirpath = __file__
>>> for i in range(4): #recurse up to root repo dir
...     dirpath = dirname(dirpath)
>>> repo_root = dirpath
>>> format_root = os.path.join(repo_root, 'tests', 'formats', 'dae')
>>> # check and read dae file
>>> stream = open(os.path.join(format_root, 'cube.dae'), 'rb')
>>> daedata = DaeFormat.Data()
>>> daedata.read(stream)
Traceback (most recent call last):
...
NotImplementedError
>>> # get DAE file root element
>>> #print(daedata.getRootElement())
>>> stream.close()
```

Parse all DAE files in a directory tree

```
>>> for stream, data in DaeFormat.walkData(format_root):
...     try:
...         # the replace call makes the doctest also pass on windows
...         os_path = stream.name
...         split = (os_path.split(os.sep))[-4:]
...         rejoin = os.path.join(*split).replace(os.sep, "/")
...         print("reading %s" % rejoin)
...         data.read(stream)
...     except Exception:
...         print("Warning: read failed due corrupt file, corrupt format description,
↳ or bug.")
reading tests/formats/dae/cube.dae
Warning: read failed due corrupt file, corrupt format description, or bug.
```

Create a DAE file from scratch and write to file

```
>>> daedata = DaeFormat.Data()
>>> from tempfile import TemporaryFile
>>> stream = TemporaryFile()
>>> daedata.write(stream)
Traceback (most recent call last):
...
NotImplementedError
```

pyffi.formats.dds — DirectDraw Surface (.dds)

Implementation

class pyffi.formats.dds.DdsFormat

Bases: pyffi.object_models.xml.FileFormat

This class implements the DDS format.

class Data (version=150994944)

Bases: pyffi.object_models.Data

A class to contain the actual dds data.

get_detail_child_names (edge_filter=(True, True))

Generator which yields all child names of this item in the detail view.

Override this method if the node has children.

Returns Generator for detail tree child names.

Return type generator yielding strs

get_detail_child_nodes (edge_filter=(True, True))

Generator which yields all children of this item in the detail view (by default, all acyclic and active ones).

Override this method if the node has children.

Parameters **edge_filter** (EdgeFilter or type (None)) – The edge type to include.

Returns Generator for detail tree child nodes.

Return type generator yielding DetailNodes

inspect (*stream*)
Quickly checks if stream contains DDS data, and reads the header.
Parameters **stream** (*file*) – The stream to inspect.

inspect_quick (*stream*)
Quickly checks if stream contains DDS data, and gets the version, by looking at the first 8 bytes.
Parameters **stream** (*file*) – The stream to inspect.

read (*stream*, *verbose*=0)
Read a dds file.
Parameters

- **stream** (*file*) – The stream from which to read.
- **verbose** (*int*) – The level of verbosity.

write (*stream*, *verbose*=0)
Write a dds file.
Parameters

- **stream** (*file*) – The stream to which to write.
- **verbose** (*int*) – The level of verbosity.

class FourCC (***kwargs*)
Bases: `pyffi.object_models.xml.enum.EnumBase`
An unsigned 32-bit integer, describing the compression type.

class HeaderString (***kwargs*)
Bases: `pyffi.object_models.xml.basic.BasicBase`
Basic type which implements the header of a DDS file.

get_detail_display ()
Return an object that can be used to display the instance.

get_hash (*data*=None)
Return a hash value for this value.
Returns An immutable object that can be used as a hash.

get_size (*data*=None)
Return number of bytes the header string occupies in a file.
Returns Number of bytes.

read (*stream*, *data*)
Read header string from stream and check it.
Parameters **stream** (*file*) – The stream to read from.

write (*stream*, *data*)
Write the header string to stream.
Parameters **stream** (*file*) – The stream to write to.

PixelData
alias of `pyffi.object_models.common.UndecodedData`

byte
alias of `pyffi.object_models.common.Byte`

char
alias of `pyffi.object_models.common.Char`

float
alias of `pyffi.object_models.common.Float`

int
alias of `pyffi.object_models.common.Int`

short
alias of `pyffi.object_models.common.Short`

ubyte
alias of `pyffi.object_models.common.UByte`

uint
alias of `pyffi.object_models.common.UInt`

ushort
alias of `pyffi.object_models.common.UShort`

static version_number (*version_str*)
Converts version string into an integer.

Parameters **version_str** (*str*) – The version string.

Returns A version integer.

```
>>> hex(DdsFormat.version_number('DX10'))  
'0xa000000'
```

Regression tests

Read a DDS file

```
>>> # check and read dds file  
>>> from os.path import dirname  
>>> dirpath = __file__  
>>> for i in range(4): #recurse up to root repo dir  
...     dirpath = dirname(dirpath)  
>>> repo_root = dirpath  
>>> format_root = os.path.join(repo_root, 'tests', 'formats', 'dds')  
>>> file = os.path.join(format_root, 'test.dds')  
>>> stream = open(file, 'rb')  
>>> data = DdsFormat.Data()  
>>> data.inspect(stream)  
>>> data.header.pixel_format.size  
32  
>>> data.header.height  
20  
>>> data.read(stream)  
>>> len(data.pixeldata.get_value())  
888
```

Parse all DDS files in a directory tree

```
>>> for stream, data in DdsFormat.walkData(format_root):
...     try:
...         # the replace call makes the doctest also pass on windows
...         os_path = stream.name
...         split = (os_path.split(os.sep))[-4:]
...         rejoin = os.path.join(*split).replace(os.sep, "/")
...         print("reading %s" % rejoin)
...     except Exception:
...         print(
...             "Warning: read failed due corrupt file,"
...             " corrupt format description, or bug.")
reading tests/formats/dds/test.dds
```

Create a DDS file from scratch and write to file

```
>>> data = DdsFormat.Data()
>>> from tempfile import TemporaryFile
>>> stream = TemporaryFile()
>>> data.write(stream)
```

Get list of versions

```
>>> for vnum in sorted(DdsFormat.versions.values()):
...     print('0x%08X' % vnum)
0x09000000
0x0A000000
```

pyffi.formats.egm — EGM (.egm)

An .egm file contains facial shape modifiers, that is, morphs that modify static properties of the face, such as nose size, chin shape, and so on.

Implementation

```
class pyffi.formats.egm.EgmFormat
    Bases: pyffi.object_models.xml.FileFormat
```

This class implements the EGM format.

```
class Data (version=2, num_vertices=0)
    Bases: pyffi.object_models.Data
```

A class to contain the actual egm data.

```
add_asym_morph()
    Add an asymmetric morph, and return it.
```

```
add_sym_morph()
    Add a symmetric morph, and return it.
```

apply_scale (*scale*)
Apply scale factor to all morphs.

get_detail_child_names (*edge_filter=(True, True)*)
Generator which yields all child names of this item in the detail view.

Override this method if the node has children.
Returns Generator for detail tree child names.
Return type generator yielding `strs`

get_detail_child_nodes (*edge_filter=(True, True)*)
Generator which yields all children of this item in the detail view (by default, all acyclic and active ones).

Override this method if the node has children.
Parameters **edge_filter** (`EdgeFilter` or `type (None)`) – The edge type to include.
Returns Generator for detail tree child nodes.
Return type generator yielding `DetailNodes`

get_global_child_nodes (*edge_filter=(True, True)*)
Generator which yields all children of this item in the global view, of given edge type (default is edges of type 0).

Override this method.
Returns Generator for global node children.

inspect (*stream*)
Quickly checks if stream contains EGM data, and reads the header.
Parameters **stream** (*file*) – The stream to inspect.

inspect_quick (*stream*)
Quickly checks if stream contains EGM data, and gets the version, by looking at the first 8 bytes.
Parameters **stream** (*file*) – The stream to inspect.

read (*stream*)
Read a egm file.
Parameters **stream** (*file*) – The stream from which to read.

write (*stream*)
Write a egm file.
Parameters **stream** (*file*) – The stream to which to write.

class FileSignature (***kwargs*)
Bases: `pyffi.object_models.xml.basic.BasicBase`
Basic type which implements the header of a EGM file.

get_detail_display ()
Return an object that can be used to display the instance.

get_hash (*data=None*)
Return a hash value for this value.
Returns An immutable object that can be used as a hash.

get_size (*data=None*)
Return number of bytes the header string occupies in a file.
Returns Number of bytes.

read (*stream, data*)
Read header string from stream and check it.
Parameters **stream** (*file*) – The stream to read from.

write(*stream*, *data*)

Write the header string to stream.

Parameters *stream*(*file*) – The stream to write to.

class FileVersion(*template=None*, *argument=None*, *parent=None*)

Bases: `pyffi.object_models.xml.basic.BasicBase`

get_detail_display()

Return an object that can be used to display the instance.

get_hash(*data=None*)

Returns a hash value (an immutable object) that can be used to identify the object uniquely.

get_size(*data=None*)

Returns size of the object in bytes.

get_value()

Return object value.

read(*stream*, *data*)

Read object from file.

set_value(*value*)

Set object value.

write(*stream*, *data*)

Write object to file.

class MorphRecord(*template=None*, *argument=None*, *parent=None*)

Bases: `pyffi.formats.egm._MorphRecord`, `object`

```
>>> # create morph with 3 vertices.
>>> morph = EgmFormat.MorphRecord(argument=3)
>>> morph.set_relative_vertices(
...     [(3, 5, 2), (1, 3, 2), (-9, 3, -1)])
>>> # scale should be 9/32768.0 = 0.0002746...
>>> morph.scale
0.0002746...
>>> for vert in morph.get_relative_vertices():
...     print([int(1000 * x + 0.5) for x in vert])
[3000, 5000, 2000]
[1000, 3000, 2000]
[-8999, 3000, -999]
```

apply_scale(*scale*)

Apply scale factor to data.

```
>>> # create morph with 3 vertices.
>>> morph = EgmFormat.MorphRecord(argument=3)
>>> morph.set_relative_vertices(
...     [(3, 5, 2), (1, 3, 2), (-9, 3, -1)])
>>> morph.apply_scale(2)
>>> for vert in morph.get_relative_vertices():
...     print([int(1000 * x + 0.5) for x in vert])
[6000, 10000, 4000]
[2000, 6000, 4000]
[-17999, 6000, -1999]
```

byte

alias of `pyffi.object_models.common.Byte`

char
alias of `pyffi.object_models.common.Char`

float
alias of `pyffi.object_models.common.Float`

int
alias of `pyffi.object_models.common.Int`

short
alias of `pyffi.object_models.common.Short`

ubyte
alias of `pyffi.object_models.common.UByte`

uint
alias of `pyffi.object_models.common.UInt`

ushort
alias of `pyffi.object_models.common.USHort`

static version_number (*version_str*)
Converts version string into an integer.

Parameters **version_str** (*str*) – The version string.

Returns A version integer.

```
>>> EgmFormat.version_number('002')
2
>>> EgmFormat.version_number('XXX')
-1
```

Regression tests

Read a EGM file

```
>>> # check and read egm file
>>> from os.path import dirname
>>> dirpath = __file__
>>> for i in range(4): #recurse up to root repo dir
...     dirpath = dirname(dirpath)
>>> repo_root = dirpath
>>> format_root = os.path.join(repo_root, 'tests', 'formats', 'egm')
>>> file = os.path.join(format_root, 'mmouthxivilai.egm')
>>> stream = open(file, 'rb')
>>> data = EgmFormat.Data()
>>> data.inspect_quick(stream)
>>> data.version
2
>>> data.inspect(stream)
>>> data.header.num_vertices
89
>>> data.header.num_sym_morphs
50
>>> data.header.num_asym_morphs
30
>>> data.header.time_date_stamp
```

(continues on next page)

(continued from previous page)

```

2001060901
>>> data.read(stream)
>>> data.sym_morphs[0].vertices[0].x
17249

```

Parse all EGM files in a directory tree

```

>>> for stream, data in EgmFormat.walkData(format_root):
...     try:
...         # the replace call makes the doctest also pass on windows
...         os_path = stream.name
...         split = (os_path.split(os.sep))[-4:]
...         rejoin = os.path.join(*split).replace(os.sep, "/")
...         print("reading %s" % rejoin)
...     except Exception:
...         print(
...             "Warning: read failed due corrupt file,"
...             " corrupt format description, or bug.")
reading tests/formats/egm/mmouthevilai.egm

```

Create an EGM file from scratch and write to file

```

>>> data = EgmFormat.Data(num_vertices=10)
>>> data.header.num_vertices
10
>>> morph = data.add_sym_morph()
>>> len(morph.vertices)
10
>>> morph.scale = 0.4
>>> morph.vertices[0].z = 123
>>> morph.vertices[9].x = -30000
>>> morph = data.add_asym_morph()
>>> morph.scale = 2.3
>>> morph.vertices[3].z = -5
>>> morph.vertices[4].x = 99
>>> from tempfile import TemporaryFile
>>> stream = TemporaryFile()
>>> data.write(stream)

```

pyffi.formats.egt — EGT (.egt)

An .egt file contains texture tones for the different races.

Implementation

class pyffi.formats.egt.EgtFormat

Bases: pyffi.object_models.xml.FileFormat

This class implements the EGT format.

Data

alias of *Header*

class FileSignature (**kwargs)

Bases: pyffi.object_models.xml.basic.BasicBase

Basic type which implements the header of a EGT file.

get_detail_display()

Return an object that can be used to display the instance.

get_hash(data=None)

Return a hash value for this value.

Returns An immutable object that can be used as a hash.

get_size(data=None)

Return number of bytes the header segtng occupies in a file.

Returns Number of bytes.

read(stream, data)

Read header string from stream and check it.

Parameters **stream** (*file*) – The stream to read from.

write(stream, data)

Write the header segtng to stream.

Parameters **stream** (*file*) – The stream to write to.

class FileVersion (template=None, argument=None, parent=None)

Bases: pyffi.object_models.xml.basic.BasicBase

get_detail_display()

Return an object that can be used to display the instance.

get_hash(data=None)

Returns a hash value (an immutable object) that can be used to identify the object uniquely.

get_size(data=None)

Returns size of the object in bytes.

get_value()

Return object value.

read(stream, data)

Read object from file.

set_value(value)

Set object value.

write(stream, data)

Write object to file.

```
class Header (template=None, argument=None, parent=None)
    Bases: pyffi.formats.egt._Header, pyffi.object_models.Data

    A class to contain the actual egt data.

get_global_child_nodes (edge_filter=(True, True))
    Generator which yields all children of this item in the global view, of given edge type (default is edges
    of type 0).

    Override this method.
        Returns Generator for global node children.

inspect (stream)
    Quickly checks if stream contains EGT data, and reads everything up to the arrays.
        Parameters stream (file) – The stream to inspect.

inspect_quick (stream)
    Quickly checks if stream contains EGT data, by looking at the first 8 bytes. Reads the signature and
    the version.
        Parameters stream (file) – The stream to inspect.

read (stream)
    Read a egt file.
        Parameters stream (file) – The stream from which to read.

write (stream)
    Write a egt file.
        Parameters stream (file) – The stream to which to write.

byte
    alias of pyffi.object_models.common.Byte

char
    alias of pyffi.object_models.common.Char

float
    alias of pyffi.object_models.common.Float

int
    alias of pyffi.object_models.common.Int

short
    alias of pyffi.object_models.common.Short

ubyte
    alias of pyffi.object_models.common.UByte

uint
    alias of pyffi.object_models.common.UInt

ushort
    alias of pyffi.object_models.common.USHort

static version_number (version_str)
    Converts version segtnng into an integer.

        Parameters version_str (str) – The version segtnng.

        Returns A version integer.
```

```
>>> EgtFormat.version_number('003')
3
```

(continues on next page)

(continued from previous page)

```
>>> EgtFormat.version_number('XXX')
-1
```

Regression tests

Read a EGT file

```
>>> # check and read egt file
>>> from os.path import dirname
>>> dirpath = __file__
>>> for i in range(4): #recurse up to root repo dir
...     dirpath = dirname(dirpath)
>>> repo_root = dirpath
>>> format_root = os.path.join(repo_root, 'tests', 'formats', 'egt')
>>> file = os.path.join(format_root, 'test.egt')
>>> stream = open(file, 'rb')
>>> data = EgtFormat.Data()
>>> data.inspect(stream)
>>> # do some stuff with header?
>>> data.read(stream)
>>> # do more stuff?
```

Parse all EGT files in a directory tree

```
>>> for stream, data in EgtFormat.walkData(format_root):
...     try:
...         # the replace call makes the doctest also pass on windows
...         os_path = stream.name
...         split = (os_path.split(os.sep))[-4:]
...         rejoin = os.path.join(*split).replace(os.sep, "/")
...         print("reading %s" % rejoin)
...     except Exception:
...         print(
...             "Warning: read failed due corrupt file,"
...             " corrupt format description, or bug.")
reading tests/formats/egt/test.egt
```

Create an EGT file from scratch and write to file

```
>>> data = EgtFormat.Data()
>>> from tempfile import TemporaryFile
>>> stream = TemporaryFile()
>>> data.write(stream)
```

pyffi.formats.esp — Elder Scrolls plugin/master/save files (.esp, .esm, and .ess)**Implementation**

class pyffi.formats.esp.EspFormat

Bases: pyffi.object_models.xml.FileFormat

This class implements the ESP format.

class Data

Bases: pyffi.object_models.Data

A class to contain the actual esp data.

get_detail_child_names (*edge_filter=(True, True)*)

Generator which yields all child names of this item in the detail view.

Override this method if the node has children.

Returns Generator for detail tree child names.

Return type generator yielding `str`s

get_detail_child_nodes (*edge_filter=(True, True)*)

Generator which yields all children of this item in the detail view (by default, all acyclic and active ones).

Override this method if the node has children.

Parameters *edge_filter* (EdgeFilter or type (None)) – The edge type to include.

Returns Generator for detail tree child nodes.

Return type generator yielding DetailNodes

get_global_child_nodes (*edge_filter=(True, True)*)

Generator which yields all children of this item in the global view, of given edge type (default is edges of type 0).

Override this method.

Returns Generator for global node children.

inspect (*stream*)

Quickly checks if stream contains ESP data, and reads the header.

Parameters *stream* (*file*) – The stream to inspect.

inspect_quick (*stream*)

Quickly checks if stream contains ESP data, and gets the version, by looking at the first 8 bytes.

Parameters *stream* (*file*) – The stream to inspect.

read (*stream*)

Read a esp file.

Parameters *stream* (*file*) – The stream from which to read.

write (*stream*)

Write a esp file.

Parameters *stream* (*file*) – The stream to which to write.

class GRUP

Bases: pyffi.formats.esp._GRUP, object

get_global_child_nodes (*edge_filter=(True, True)*)

Generator which yields all children of this item in the global view, of given edge type (default is edges of type 0).

Override this method.

Returns Generator for global node children.

read(*stream, data*)
Read structure from stream.

write(*stream, data*)
Write structure to stream.

class Record

Bases: `pyffi.formats.esp._Record`, `object`

get_global_child_nodes(*edge_filter=(True, True)*)
Generator which yields all children of this item in the global view, of given edge type (default is edges of type 0).

Override this method.

Returns Generator for global node children.

get_sub_record(*sub_record_type*)
Find first subrecord of given type.

read(*stream, data*)
Read structure from stream.

write(*stream, data*)
Write structure to stream.

class RecordType(***kwargs*)

Bases: `pyffi.object_models.common.FixedString`

class SubRecord(*template=None, argument=None, parent=None*)

Bases: `pyffi.object_models.xml.struct_.StructBase`

A subrecord.

class ZString(***kwargs*)

Bases: `pyffi.object_models.xml.basic.BasicBase`, `pyffi.object_models.editable.EditableLineEdit`

String of variable length (null terminated).

```
>>> from tempfile import TemporaryFile
>>> f = TemporaryFile()
>>> s = ZString()
>>> if f.write('abcdefghijklmnopqrst\x00'.encode("ascii")): pass # b'abc...'
>>> if f.seek(0): pass # ignore result for py3k
>>> s.read(f)
>>> str(s)
'abcdefghijklmnopqrst'
>>> if f.seek(0): pass # ignore result for py3k
>>> s.set_value('Hi There!')
>>> s.write(f)
>>> if f.seek(0): pass # ignore result for py3k
>>> m = ZString()
>>> m.read(f)
>>> str(m)
'Hi There!'
```

get_hash(*data=None*)
Return a hash value for this string.

Returns An immutable object that can be used as a hash.

get_size (*data=None*)

Return number of bytes this type occupies in a file.

Returns Number of bytes.

get_value ()

Return the string.

Returns The stored string.

Return type C{bytes}

read (*stream, data=None*)

Read string from stream.

Parameters **stream** (*file*) – The stream to read from.

set_value (*value*)

Set string to C{value}.

Parameters **value** (*str* (will be encoded as default) or C{bytes}) – The value to assign.

write (*stream, data=None*)

Write string to stream.

Parameters **stream** (*file*) – The stream to write to.

byte

alias of `pyffi.object_models.common.Byte`

char

alias of `pyffi.object_models.common.Char`

float

alias of `pyffi.object_models.common.Float`

int

alias of `pyffi.object_models.common.Int`

short

alias of `pyffi.object_models.common.Short`

ubyte

alias of `pyffi.object_models.common.UByte`

uint

alias of `pyffi.object_models.common.UInt`

uint64

alias of `pyffi.object_models.common.UInt64`

ushort

alias of `pyffi.object_models.common.USHort`

static version_number (*version_str*)

Converts version string into an integer.

Parameters **version_str** (*str*) – The version string.

Returns A version integer.

```
>>> hex(EspFormat.version_number('1.2'))
'0x102'
```

Regression tests

Read a ESP file

```
>>> # check and read esp file
>>> from os.path import dirname
>>> dirpath = __file__
>>> for i in range(4): #recurse up to root repo dir
...     dirpath = dirname(dirpath)
>>> repo_root = dirpath
>>> format_root = os.path.join(repo_root, 'tests', 'formats', 'esp')
>>> file = os.path.join(format_root, 'test.esp')
>>> stream = open(file, 'rb')
>>> data = EspFormat.Data()
>>> data.inspect(stream)
>>> # do some stuff with header?
>>> #data.header....
>>> data.read(stream)
>>> # do some stuff...
```

Parse all ESP files in a directory tree

```
>>> for stream, data in EspFormat.walkData(format_root):
...     try:
...         # the replace call makes the doctest also pass on windows
...         os_path = stream.name
...         split = (os_path.split(os.sep))[-4:]
...         rejoin = os.path.join(*split).replace(os.sep, "/")
...         print("reading %s" % rejoin)
...     except Exception:
...         print(
...             "Warning: read failed due corrupt file,"
...             " corrupt format description, or bug.")
reading tests/formats/esp/test.esp
```

Create an ESP file from scratch and write to file

```
>>> data = EspFormat.Data()
>>> from tempfile import TemporaryFile
>>> stream = TemporaryFile()
>>> data.write(stream)
```

pyffi.formats.kfm — NetImmerse/Gamebryo Keyframe Motion (.kfm)**Implementation****class** pyffi.formats.kfm.KfmFormat

Bases: pyffi.object_models.xml.FileFormat

This class implements the kfm file format.

class Data (version=33685515)

Bases: pyffi.object_models.Data

A class to contain the actual kfm data.

get_global_child_nodes (edge_filter=(True, True))

Generator which yields all children of this item in the global view, of given edge type (default is edges of type 0).

Override this method.

Returns Generator for global node children.**get_global_display** ()

Display the KFM file name.

inspect (stream)

Quick heuristic check if stream contains KFM data, by looking at the first 64 bytes. Sets version and reads header string.

Parameters stream (file) – The stream to inspect.**read** (stream)

Read a kfm file.

Parameters stream (file) – The stream from which to read.**write** (stream)

Write a kfm file.

Parameters stream (file) – The stream to which to write.**class** FilePath (**kwargs)

Bases: pyffi.object_models.common.SizedString

get_hash (data=None)

Return a hash value for this value. For file paths, the hash value is case insensitive.

Returns An immutable object that can be used as a hash.**class** HeaderString (**kwargs)

Bases: pyffi.object_models.xml.basic.BasicBase

The kfm header string.

get_detail_display ()

Return an object that can be used to display the instance.

get_hash (data=None)

Return a hash value for this value.

Returns An immutable object that can be used as a hash.**get_size** (data=None)

Return number of bytes the header string occupies in a file.

Returns Number of bytes.**get_value** ()

Return object value.

read(*stream*, *data*)

Read header string from stream and check it.

Parameters

- **stream** (*file*) – The stream to read from.
- **data** (`pyffi.formats.kfm.KfmFormat.Data`) – The `KfmFormat.Data()`

set_value(*value*)

Set object value.

static version_string(*version*)

Transforms version number into a version string.

Parameters **version** (*int*) – The version number.

Returns A version string.

```
>>> KfmFormat.HeaderString.version_string(0x0202000b)
';Gamebryo KFM File Version 2.2.0.0b'
>>> KfmFormat.HeaderString.version_string(0x01024b00)
';Gamebryo KFM File Version 1.2.4b'
```

write(*stream*, *data*)

Write the header string to stream.

Parameters

- **stream** (*file*) – The stream to write to.
- **data** (`Data`) – The fileformat data to use

class SizedString (***kwargs*)

Bases: `pyffi.object_models.xml.basic.BasicBase`, `pyffi.object_models.editable.EditableLineEdit`

Basic type for strings. The type starts with an unsigned int which describes the length of the string.

```
>>> from tempfile import TemporaryFile
>>> f = TemporaryFile()
>>> from pyffi.object_models import FileFormat
>>> data = FileFormat.Data()
>>> s = SizedString()
>>> if f.write('\x07\x00\x00\x00abcdefg'.encode("ascii")): pass # ignore_
↳ result for py3k
>>> if f.seek(0): pass # ignore result for py3k
>>> s.read(f, data)
>>> str(s)
'abcdefg'
>>> if f.seek(0): pass # ignore result for py3k
>>> s.set_value('Hi There')
>>> s.write(f, data)
>>> if f.seek(0): pass # ignore result for py3k
>>> m = SizedString()
>>> m.read(f, data)
>>> str(m)
'Hi There'
```

get_hash(*data=None*)

Return a hash value for this string.

Returns An immutable object that can be used as a hash.

get_size(*data=None*)

Return number of bytes this type occupies in a file.

Returns Number of bytes.

get_value()
Return the string.
Returns The stored string.

read(stream, data)
Read string from stream.
Parameters **stream** (*file*) – The stream to read from.

set_value(value)
Set string to C{value}.
Parameters **value** (*str*) – The value to assign.

write(stream, data)
Write string to stream.
Parameters **stream** (*file*) – The stream to write to.

TextString

alias of `pyffi.object_models.common.UndecodedData`

byte

alias of `pyffi.object_models.common.UByte`

char

alias of `pyffi.object_models.common.Char`

float

alias of `pyffi.object_models.common.Float`

int

alias of `pyffi.object_models.common.Int`

short

alias of `pyffi.object_models.common.Short`

uint

alias of `pyffi.object_models.common.UInt`

ushort

alias of `pyffi.object_models.common.USHort`

static version_number(version_str)

Converts version string into an integer.

Parameters **version_str** (*str*) – The version string.

Returns A version integer.

```
>>> hex(KfmFormat.version_number('1.0'))
'0x1000000'
>>> hex(KfmFormat.version_number('1.2.4b'))
'0x1024b00'
>>> hex(KfmFormat.version_number('2.2.0.0b'))
'0x202000b'
```

Regression tests

Read a KFM file

```
>>> # read kfm file
>>> from os.path import dirname
>>> dirpath = __file__
>>> for i in range(4): #recurse up to root repo dir
...     dirpath = dirname(dirpath)
>>> repo_root = dirpath
>>> files_dir = os.path.join(repo_root, 'tests', 'spells', 'kfm', 'files')
>>> file = os.path.join(files_dir, 'test.kfm')
>>> stream = open(file, 'rb')
>>> data = KfmFormat.Data()
>>> data.inspect(stream)
>>> data.read(stream)
>>> stream.close()
>>> print(data.nif_file_name.decode("ascii"))
Test.nif
>>> # get all animation file names
>>> for anim in data.animations:
...     print(anim.kf_file_name.decode("ascii"))
Test_MD_Idle.kf
Test_MD_Run.kf
Test_MD_Walk.kf
Test_MD_Die.kf
```

Parse all KFM files in a directory tree

```
>>> for stream, data in KfmFormat.walkData(files_dir):
...     try:
...         # the replace call makes the doctest also pass on windows
...         os_path = stream.name
...         split = (os_path.split(os.sep))[-5:]
...         rejoin = os.path.join(*split).replace("\\", "/")
...         print("reading %s" % rejoin)
...     except Exception:
...         print(
...             "Warning: read failed due corrupt file,"
...             " corrupt format description, or bug.")
reading tests/spells/kfm/files/invalid.kfm
reading tests/spells/kfm/files/test.kfm
```

Create a KFM model from scratch and write to file

```
>>> data = KfmFormat.Data()
>>> data.nif_file_name = "Test.nif"
>>> data.num_animations = 4
>>> data.animations.update_size()
>>> data.animations[0].kf_file_name = "Test_MD_Idle.kf"
>>> data.animations[1].kf_file_name = "Test_MD_Run.kf"
>>> data.animations[2].kf_file_name = "Test_MD_Walk.kf"
```

(continues on next page)

(continued from previous page)

```
>>> data.animations[3].kf_file_name = "Test_MD_Die.kf"
>>> from tempfile import TemporaryFile
>>> stream = TemporaryFile()
>>> data.write(stream)
>>> stream.close()
```

Get list of versions and games

```
>>> for vnum in sorted(KfmFormat.versions.values()):
...     print('0x%08X' % vnum)
0x01000000
0x01024B00
0x0200000B
0x0201000B
0x0202000B
0x0202001B
>>> for game, versions in sorted(KfmFormat.games.items(),
...                               key=lambda x: x[0]):
...     print("%s " % game + " ".join('0x%08X' % vnum for vnum in versions))
Civilization IV 0x01000000 0x01024B00 0x0200000B
Dragonica 0x0202001B
Emerge 0x0201000B 0x0202000B
Loki 0x01024B00
Megami Tensei: Imagine 0x0201000B
Oblivion 0x01024B00
Prison Tycoon 0x01024B00
Pro Cycling Manager 0x01024B00
Red Ocean 0x01024B00
Sid Meier's Railroads 0x0200000B
The Guild 2 0x01024B00
```

pyffi.formats.nif — NetImmerse/Gamebryo (.nif and .kf)

Implementation

```
class pyffi.formats.nif.NifFormat
    Bases: pyffi.object_models.xml.FileFormat

    This class contains the generated classes from the xml.

    ARCHIVE_CLASSES = [<class 'pyffi.formats.bsa.BsaFormat'>]

    class ATextureRenderData (template=None, argument=None, parent=None)
        Bases: pyffi.formats.nif._ATextureRenderData, object

        save_as_dds (stream)
            Save image as DDS file.

    class AVObject (template=None, argument=None, parent=None)
        Bases: pyffi.object_models.xml.struct_.StructBase

        Used in NiDefaultAVObjectPalette.

        property av_object
```

```
    property name

class AbstractAdditionalGeometryData (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiObject

class AdditionalDataBlock (template=None, argument=None, parent=None)
    Bases: pyffi.object_models.xml.struct_.StructBase

    property block_offsets
    property block_size
    property data
    property data_sizes
    property has_data
    property num_blocks
    property num_data

class AdditionalDataInfo (template=None, argument=None, parent=None)
    Bases: pyffi.object_models.xml.struct_.StructBase

    property block_index
    property channel_offset
    property data_type
    property num_channel_bytes
    property num_channel_bytes_per_element
    property num_total_bytes_per_element
    property unknown_byte_1

class AlphaFormat (**kwargs)
    Bases: pyffi.object_models.xml.enum.EnumBase

    An unsigned 32-bit integer, describing how transparency is handled in a texture.

    ALPHA_BINARY = 1
    ALPHA_DEFAULT = 3
    ALPHA_NONE = 0
    ALPHA_SMOOTH = 2

class AnimationType (**kwargs)
    Bases: pyffi.object_models.xml.enum.EnumBase

    Animation type used on this position. This specifies the function of this position.

    Lean = 4
    Sit = 1
    Sleep = 2

class ApplyMode (**kwargs)
    Bases: pyffi.object_models.xml.enum.EnumBase

    An unsigned 32-bit integer, describing the apply mode of a texture.

    APPLY_DECAL = 1
```



```

APPLY_HILIGHT = 3
APPLY_HILIGHT2 = 4
APPLY_MODULATE = 2
APPLY_REPLACE = 0

class ArkTexture (template=None, argument=None, parent=None)
    Bases: pyffi.object_models.xml.struct_.StructBase
    A texture reference used by NiArkTextureExtraData.
    property texture_name
    property texturing_property
    property unknown_bytes
    property unknown_int_3
    property unknown_int_4

class AvoidNode (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiNode
    Morrowind specific?

class BSAanimNotes (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiObject
    Bethesda-specific node.
    property unknown_short_1

class BSBehaviorGraphExtraData (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiExtraData
    Links a nif with a Havok Behavior .hbx animation file
    property behaviour_graph_file
    property controls_base_skeleton

class BSblastNode (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiNode
    Bethesda-Specific node.
    property unknown_byte_1
    property unknown_short_2

class BSBoneLODExtraData (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiExtraData
    Unknown
    property bone_1_o_d_count
    property bone_1_o_d_info

class BSBound (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif._BSBound, object
    apply_scale (scale)
        Scale data.

```

```
class BSDamageStage (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiNode

    Bethesda-Specific node.

    property unknown_byte_1
    property unknown_short_2

class BSDebrisNode (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiNode

    Bethesda-Specific node.

    property unknown_byte_1
    property unknown_short_2

class BSDecalPlacementVectorExtraData (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiExtraData

    Bethesda-specific node. (for dynamic decal projection?)

    property num_vector_blocks
    property unknown_float_1
    property vector_blocks

class BSDismemberBodyPartType (**kwargs)
    Bases: pyffi.object_models.xml.enum.EnumBase

    Biped bodypart data used for visibility control of triangles. Options are Fallout 3, except where marked
    for Skyrim (uses SBP prefix) Skyrim BP names are listed only for vanilla names, different creatures have
    different definitions for naming.

    BP_BRAIN = 13
    BP_HEAD = 1
    BP_HEAD2 = 2
    BP_LEFTARM = 3
    BP_LEFTARM2 = 4
    BP_LEFTLEG = 7
    BP_LEFTLEG2 = 8
    BP_LEFTLEG3 = 9
    BP_RIGHTARM = 5
    BP_RIGHTARM2 = 6
    BP_RIGHTLEG = 10
    BP_RIGHTLEG2 = 11
    BP_RIGHTLEG3 = 12
    BP_SECTIONCAP_BRAIN = 113
    BP_SECTIONCAP_HEAD = 101
    BP_SECTIONCAP_HEAD2 = 102
    BP_SECTIONCAP_LEFTARM = 103
```

```
BP_SECTIONCAP_LEFTARM2 = 104
BP_SECTIONCAP_LEFTLEG = 107
BP_SECTIONCAP_LEFTLEG2 = 108
BP_SECTIONCAP_LEFTLEG3 = 109
BP_SECTIONCAP_RIGHTARM = 105
BP_SECTIONCAP_RIGHTARM2 = 106
BP_SECTIONCAP_RIGHTLEG = 110
BP_SECTIONCAP_RIGHTLEG2 = 111
BP_SECTIONCAP_RIGHTLEG3 = 112
BP_TORSO = 0
BP_TORSOCAP_BRAIN = 213
BP_TORSOCAP_HEAD = 201
BP_TORSOCAP_HEAD2 = 202
BP_TORSOCAP_LEFTARM = 203
BP_TORSOCAP_LEFTARM2 = 204
BP_TORSOCAP_LEFTLEG = 207
BP_TORSOCAP_LEFTLEG2 = 208
BP_TORSOCAP_LEFTLEG3 = 209
BP_TORSOCAP_RIGHTARM = 205
BP_TORSOCAP_RIGHTARM2 = 206
BP_TORSOCAP_RIGHTLEG = 210
BP_TORSOCAP_RIGHTLEG2 = 211
BP_TORSOCAP_RIGHTLEG3 = 212
BP_TORSOSECTION_BRAIN = 13000
BP_TORSOSECTION_HEAD = 1000
BP_TORSOSECTION_HEAD2 = 2000
BP_TORSOSECTION_LEFTARM = 3000
BP_TORSOSECTION_LEFTARM2 = 4000
BP_TORSOSECTION_LEFTLEG = 7000
BP_TORSOSECTION_LEFTLEG2 = 8000
BP_TORSOSECTION_LEFTLEG3 = 9000
BP_TORSOSECTION_RIGHTARM = 5000
BP_TORSOSECTION_RIGHTARM2 = 6000
BP_TORSOSECTION_RIGHTLEG = 10000
BP_TORSOSECTION_RIGHTLEG2 = 11000
BP_TORSOSECTION_RIGHTLEG3 = 12000
```

```
SBP_130_HEAD = 130
SBP_131_HAIR = 131
SBP_141_LONGHAIR = 141
SBP_142_CIRCLET = 142
SBP_143_EARS = 143
SBP_150_DECAPITATEDHEAD = 150
SBP_230_HEAD = 230
SBP_30_HEAD = 30
SBP_31_HAIR = 31
SBP_32_BODY = 32
SBP_33_HANDS = 33
SBP_34_FOREARMS = 34
SBP_35_AMULET = 35
SBP_36_RING = 36
SBP_37_FEET = 37
SBP_38_CALVES = 38
SBP_39_SHIELD = 39
SBP_40_TAIL = 40
SBP_41_LONGHAIR = 41
SBP_42_CIRCLET = 42
SBP_43_EARS = 43
SBP_44_DRAGON_BLOODHEAD_OR_MOD_MOUTH = 44
SBP_45_DRAGON_BLOODWINGL_OR_MOD_NECK = 45
SBP_46_DRAGON_BLOODWINGR_OR_MOD_CHEST_PRIMARY = 46
SBP_47_DRAGON_BLOODTAIL_OR_MOD_BACK = 47
SBP_48_MOD_MISC1 = 48
SBP_49_MOD_PELVIS_PRIMARY = 49
SBP_50_DECAPITATEDHEAD = 50
SBP_51_DECAPITATE = 51
SBP_52_MOD_PELVIS_SECONDARY = 52
SBP_53_MOD_LEG_RIGHT = 53
SBP_54_MOD_LEG_LEFT = 54
SBP_55_MOD_FACE_JEWELRY = 55
SBP_56_MOD_CHEST_SECONDARY = 56
SBP_57_MOD_SHOULDER = 57
SBP_58_MOD_ARM_LEFT = 58
```

```

SBP_59_MOD_ARM_RIGHT = 59

SBP_60_MOD_MISC2 = 60

SBP_61_FX01 = 61

class BSDismemberSkinInstance (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif._BSDismemberSkinInstance, object

    get_dismember_partitions()
        Return triangles and body part indices.

class BSDistantTreeShaderProperty (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.BSShaderProperty

    Bethesda-specific node.

class BSEffectShaderProperty (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiProperty

    Skyrim non-PP shader model, used primarily for transparency effects, often as decal.

    property emissive_color
    property emissive_multiple
    property falloff_start_angle
    property falloff_start_opacity
    property falloff_stop_angle
    property falloff_stop_opacity
    property greyscale_texture
    property shader_flags_1
    property shader_flags_2
    property soft_falloff_depth
    property source_texture
    property texture_clamp_mode
    property uv_offset
    property uv_scale

class BSEffectShaderPropertyColorController (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiFloatInterpController

    This controller is used to animate colors in BSEffectShaderProperty.

    property type_of_controlled_color

class BSEffectShaderPropertyFloatController (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiFloatInterpController

    This controller is used to animate float variables in BSEffectShaderProperty.

    property type_of_controlled_variable

class BSFadeNode (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiNode

```

Bethesda-specific fade node.

```
class BSFrustumFOVController (template=None, argument=None, parent=None)  
    Bases: pyffi.formats.nif.NiTimeController
```

Bethesda-specific node.

```
    property interpolator
```

```
class BSFurnitureMarker (template=None, argument=None, parent=None)  
    Bases: pyffi.formats.nif.NiExtraData
```

Unknown. Marks furniture sitting positions?

```
    property num_positions
```

```
    property positions
```

```
class BSFurnitureMarkerNode (template=None, argument=None, parent=None)  
    Bases: pyffi.formats.nif.BSFurnitureMarker
```

Furniture Marker for actors

```
class BSInvMarker (template=None, argument=None, parent=None)  
    Bases: pyffi.formats.nif.NiExtraData
```

Orientation marker for Skyrim's inventory view. How to show the nif in the player's inventory. Typically attached to the root node of the nif tree. If not present, then Skyrim will still show the nif in inventory, using the default values. Name should be 'INV' (without the quotes). For rotations, a short of "4712" appears as "4.712" but "959" appears as "0.959" meshesweaponsdaedricdaedricbowskinned.nif

```
    property rotation_x
```

```
    property rotation_y
```

```
    property rotation_z
```

```
    property zoom
```

```
class BSKeyframeController (template=None, argument=None, parent=None)  
    Bases: pyffi.formats.nif.NiKeyframeController
```

An extended keyframe controller.

```
    property data_2
```

```
class BSLODTriShape (template=None, argument=None, parent=None)  
    Bases: pyffi.formats.nif.NiTriBasedGeom
```

A variation on NiTriShape, for visibility control over vertex groups.

```
    property level_0_size
```

```
    property level_1_size
```

```
    property level_2_size
```

```
class BSLagBoneController (template=None, argument=None, parent=None)  
    Bases: pyffi.formats.nif.NiTimeController
```

A controller that trails a bone behind an actor.

```
    property linear_rotation
```

```
    property linear_velocity
```

```
    property maximum_distance
```

```

class BSLeafAnimNode (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiNode

    Unknown, related to trees.

class BSLightingShaderProperty (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiProperty

    Skyrim PP shader for assigning material/shader/texture.

    property alpha
    property emissive_color
    property emissive_multiple
    property environment_map_scale
    property eye_cubemap_scale
    property glossiness
    property hair_tint_color
    property left_eye_reflection_center
    property lighting_effect_1
    property lighting_effect_2
    property max_passes
    property parallax_envmap_strength
    property parallax_inner_layer_texture_scale
    property parallax_inner_layer_thickness
    property parallax_refraction_scale
    property refraction_strength
    property right_eye_reflection_center
    property scale
    property shader_flags_1
    property shader_flags_2
    property skin_tint_color
    property sparkle_parameters
    property specular_color
    property specular_strength
    property texture_clamp_mode
    property texture_set
    property uv_offset
    property uv_scale

class BSLightingShaderPropertyColorController (template=None, argument=None,
                                                parent=None)
    Bases: pyffi.formats.nif.NiFloatInterpController

    This controller is used to animate colors in BSLightingShaderProperty.

```

```
    property type_of_controlled_color

class BSLightingShaderPropertyFloatController (template=None, argument=None,
                                              parent=None)
    Bases: pyffi.formats.nif.NiFloatInterpController
    This controller is used to animate float variables in BSLightingShaderProperty.

    property type_of_controlled_variable

class BSLightingShaderPropertyShaderType (**kwargs)
    Bases: pyffi.object_models.xml.enum.EnumBase
    Values for configuring the shader type in a BSLightingShaderProperty

    Default = 0
    Heightmap = 3
    WorldMap1 = 9
    WorldMap2 = 13
    WorldMap3 = 15
    WorldMap4 = 18

class BSMasterParticleSystem (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiNode
    Bethesda-Specific node.

    property max_emitter_objects
    property num_particle_systems
    property particle_systems

class BSMaterialEmittanceMultController (template=None, argument=None, par-
                                         ent=None)
    Bases: pyffi.formats.nif.NiFloatInterpController
    Bethesda-Specific node.

class BSMultiBound (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiObject
    Bethesda-specific node.

    property data

class BSMultiBoundAABB (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.BSMultiBoundData
    Bethesda-specific node.

    property extent
    property position

class BSMultiBoundData (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiObject
    Abstract base type for bounding data.

class BSMultiBoundNode (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiNode
    Bethesda-specific node.
```



```

    property multi_bound
    property unknown_int

class BSMultiBoundOBB (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.BSMultiBoundData
    Oriented bounding box.
    property center
    property rotation
    property size

class BSMultiBoundSphere (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.BSMultiBoundData
    Bethesda-specific node.
    property radius
    property unknown_int_1
    property unknown_int_2
    property unknown_int_3

class BSNiAlphaPropertyTestRefController (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiAlphaController
    Unkown

class BSOrderedNode (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiNode
    Bethesda-Specific node.
    property alpha_sort_bound
    property is_static_bound

class BSPSysArrayEmitter (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiPSysVolumeEmitter
    Particle emitter that uses a node, its children and subchildren to emit from. Emission will be evenly spread along points from nodes leading to their direct parents/children only.

class BSPSysHavokUpdateModifier (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiPSysModifier
    property modifier
    property nodes
    property num_nodes

class BSPSysInheritVelocityModifier (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiPSysModifier
    property unknown_float_1
    property unknown_float_2
    property unknown_float_3
    property unknown_int_1

```

```
class BSPSysLODModifier (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiPSysModifier

    property unknown_float_1
    property unknown_float_2
    property unknown_float_3
    property unknown_float_4

class BSPSysMultiTargetEmitterCtrl (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiPSysModifierCtrl

    Particle system (multi?) emitter controller.

    property data
    property unknown_int_1
    property unknown_short_1
    property visibility_interpolator

class BSPSysRecycleBoundModifier (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiPSysModifier

    property unknown_float_1
    property unknown_float_2
    property unknown_float_3
    property unknown_float_4
    property unknown_float_5
    property unknown_float_6
    property unknown_int_1

class BSPSysScaleModifier (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiPSysModifier

    property floats
    property num_floats

class BSPSysSimpleColorModifier (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiPSysModifier

    Bethesda-Specific Particle node.

    property color_1_end_percent
    property color_1_start_percent
    property color_2_end_percent
    property color_2_start_percent
    property colors
    property fade_in_percent
    property fade_out_percent
```

```

class BSPSysStripUpdateModifier (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiPSysModifier

    Bethesda-Specific (mesh?) Particle System Modifier.

    property update_delta_time

class BSPSysSubTexModifier (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiPSysModifier

    Similar to a Flip Controller, this handles particle texture animation on a single texture atlas

    property end_frame
    property frame_count
    property frame_count_fudge
    property loop_start_frame
    property loop_start_frame_fudge
    property start_frame
    property start_frame_fudge

class BSPackedAdditionalDataBlock (template=None, argument=None, parent=None)
    Bases: pyffi.object_models.xml.struct_.StructBase

    property atom_sizes
    property block_offsets
    property data
    property has_data
    property num_atoms
    property num_blocks
    property num_total_bytes
    property num_total_bytes_per_element
    property unknown_int_1

class BSPackedAdditionalGeometryData (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.AbstractAdditionalGeometryData

    property block_infos
    property blocks
    property num_block_infos
    property num_blocks
    property num_vertices

class BSParentVelocityModifier (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiPSysModifier

    Particle modifier that adds a blend of object space translation and rotation to particles born in world space.

    property damping

```

```
class BSPartFlag (template=None, argument=None, parent=None)
    Bases: pyffi.object_models.xml.bit_struct.BitStructBase
    Editor flags for the Body Partitions.

    property pf_editor_visible
    property pf_start_net_boneset
    property reserved_bits_1

class BSProceduralLightningController (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiFloatInterpController
    Skyrim, Paired with dummy TriShapes, this controller generates lightning shapes for special effects. First
    interpolator controls Generation.

    property byte_1
    property byte_2
    property byte_3
    property distance_weight
    property float_2
    property float_5
    property fork
    property interpolator_10
    property interpolator_2_mutation
    property interpolator_3
    property interpolator_4
    property interpolator_5
    property interpolator_6
    property interpolator_7
    property interpolator_8
    property interpolator_9_arc_offset
    property strip_width
    property unknown_short_1
    property unknown_short_2
    property unknown_short_3

class BSRefractionFirePeriodController (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiTimeController
    Bethesda-specific node.

    property interpolator

class BSRefractionStrengthController (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiFloatInterpController
    Bethesda-Specific node.
```

```

class BSRotAccumTransfInterpolator (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiTransformInterpolator

class BSSegment (template=None, argument=None, parent=None)
    Bases: pyffi.object_models.xml.struct_.StructBase

    Bethesda-specific node.

    property flags
    property internal_index
    property unknown_byte_1

class BSSegmentFlags (template=None, argument=None, parent=None)
    Bases: pyffi.object_models.xml.bit_struct.BitStructBase

    An unsigned 32-bit integer, describing what's inside the segment.

    property bsseg_water
    property reserved_bits_0

class BSSegmentedTriShape (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiTriShape

    Bethesda-specific node.

    property num_segments
    property segment

class BSShaderFlags (template=None, argument=None, parent=None)
    Bases: pyffi.object_models.xml.bit_struct.BitStructBase

    Shader Property Flags

    property sf_alpha_texture
    property sf_decals_single_pass
    property sf_dynamic_alpha
    property sf_dynamic_decals_single_pass
    property sf_empty
    property sf_environment_mapping
    property sf_external_emittance
    property sf_eye_environment_mapping
    property sf_face_gen
    property sf_fire_refraction
    property sf_hair
    property sf_localmap_hide_secret
    property sf_low_detail
    property sf_multiple_textures
    property sf_non_projective_shadows
    property sf_parallax_occlusion
    property sf_parallax_shader_index_15

```

```
property sf_refraction
property sf_remappable_textures
property sf_shadow_frustum
property sf_shadow_map
property sf_single_pass
property sf_skinned
property sf_specular
property sf_tree_billboard
property sf_unknown_1
property sf_unknown_2
property sf_unknown_3
property sf_unknown_4
property sf_vertex_alpha
property sf_window_environment_mapping
property sf_z_buffer_test

class BSShaderFlags2 (template=None, argument=None, parent=None)
    Bases: pyffi.object_models.xml.bit_struct.BitStructBase
    Shader Property Flags 2
    property sf_2_1_st_light_is_point_light
    property sf_2_2_nd_light
    property sf_2_3_rd_light
    property sf_2_alpha_decals
    property sf_2_billboard_and_envmap_light_fade
    property sf_2_envmap_light_fade
    property sf_2_fit_slope
    property sf_2_lod_building
    property sf_2_lod_landscape
    property sf_2_no_fade
    property sf_2_no_lod_land_blend
    property sf_2_no_transparency_multisampling
    property sf_2_premult_alpha
    property sf_2_refraction_tint
    property sf_2_show_in_local_map
    property sf_2_skip_normal_maps
    property sf_2_uniform_scale
    property sf_2_unknown_1
```

```

property sf_2_unknown_10
property sf_2_unknown_2
property sf_2_unknown_3
property sf_2_unknown_4
property sf_2_unknown_5
property sf_2_unknown_6
property sf_2_unknown_7
property sf_2_unknown_8
property sf_2_unknown_9
property sf_2_vats_selection
property sf_2_vertex_colors
property sf_2_vertex_lighting
property sf_2_wireframe
property sf_2_z_buffer_write

class BSShaderLightingProperty (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.BSShaderProperty
    Bethesda-specific property.

    property texture_clamp_mode

class BSShaderNoLightingProperty (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.BSShaderLightingProperty
    Bethesda-specific property.

    property falloff_start_angle
    property falloff_start_opacity
    property falloff_stop_angle
    property falloff_stop_opacity
    property file_name

class BSShaderPPLightingProperty (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.BSShaderLightingProperty
    Bethesda-specific Shade node.

    property emissive_color
    property refraction_fire_period
    property refraction_strength
    property texture_set
    property unknown_float_4
    property unknown_float_5

```

```
class BSShaderProperty (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiProperty
    Bethesda-specific Property node

    property environment_map_scale
    property shader_flags
    property shader_flags_2
    property shader_type
    property smooth

class BSShaderTextureSet (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiObject
    Bethesda-specific Texture Set.

    property num_textures
    property textures

class BSShaderType (**kwargs)
    Bases: pyffi.object_models.xml.enum.EnumBase
    The type of animation interpolation (blending) that will be used on the associated key frames.

    SHADER_DEFAULT = 1
    SHADER_LIGHTING30 = 29
    SHADER_NOLIGHTING = 33
    SHADER_SKIN = 14
    SHADER_SKY = 10
    SHADER_TALL_GRASS = 0
    SHADER_TILE = 32
    SHADER_WATER = 17

class BSSkyShaderProperty (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiProperty
    Skyrim Sky shader block.

    property shader_flags_1
    property shader_flags_2
    property sky_object_type
    property source_texture
    property uv_offset
    property uv_scale

class BSStripPSysData (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiPSysData
    Bethesda-Specific (mesh?) Particle System Data.

    property unknown_byte_6
    property unknown_float_8
```



```

    property unknown_int_7
    property unknown_short_5
class BSStripParticleSystem (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiParticleSystem
    Bethesda-Specific (mesh?) Particle System.
class BSTreadTransfInterpolator (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiInterpolator
    Bethesda-specific node.
    property data
    property num_tread_transforms
    property tread_transforms
class BSTreadTransform (template=None, argument=None, parent=None)
    Bases: pyffi.object_models.xml.struct_.StructBase
    Bethesda-specific node.
    property name
    property transform_1
    property transform_2
class BSTreadTransformData (template=None, argument=None, parent=None)
    Bases: pyffi.object_models.xml.struct_.StructBase
    Bethesda-specific node.
    property rotation
    property scale
    property translation
class BSTreeNode (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiNode
    Node for handling Trees, Switches branch configurations for variation?
    property bones
    property bones_1
    property num_bones_1
    property num_bones_2
class BSValueNode (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiNode
    Bethesda-Specific node. Found on fxFire effects
    property unknown_byte
    property value
class BSWArray (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiExtraData
    Bethesda-specific node.

```

```
    property items
    property num_items

class BSWaterShaderProperty (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiProperty

    Skyrim water shader property, different from “WaterShaderProperty” seen in Fallout.

    property shader_flags_1
    property shader_flags_2
    property unknown_short_3
    property uv_offset
    property uv_scale
    property water_direction
    property water_shader_flags

class BSWindModifier (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiPSysModifier

    Particle Modifier that uses the wind value from the gamedata to alter the path of particles.

    property strength

class BSXFlags (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiIntegerExtraData

    Controls animation and collision. Integer holds flags: Bit 0 : enable havok, bAnimated(Skyrim) Bit 1
    : enable collision, bHavok(Skyrim) Bit 2 : is skeleton nif?, bRagdoll(Skyrim) Bit 3 : enable animation,
    bComplex(Skyrim) Bit 4 : FlameNodes present, bAddon(Skyrim) Bit 5 : EditorMarkers present Bit 6 :
    bDynamic(Skyrim) Bit 7 : bArticulated(Skyrim) Bit 8 : bIKTarget(Skyrim) Bit 9 : Unknown(Skyrim)

class BallAndSocketDescriptor (template=None, argument=None, parent=None)
    Bases: pyffi.object_models.xml.struct_.StructBase

    property unknown_4_bytes
    property unknown_floats_1
    property unknown_floats_2
    property unknown_int_1

class BillboardMode (**kwargs)
    Bases: pyffi.object_models.xml.enum.EnumBase

    Determines the way the billboard will react to the camera. Billboard mode is stored in lowest 3 bits
    although Oblivion vanilla nifs uses values higher than 7.

    ALWAYS_FACE_CAMERA = 0
    ALWAYS_FACE_CENTER = 3
    BSROTATE_ABOUT_UP = 5
    RIGID_FACE_CAMERA = 2
    RIGID_FACE_CENTER = 4
    ROTATE_ABOUT_UP = 1
    ROTATE_ABOUT_UP2 = 9
```

```

BlockTypeIndex
    alias of pyffi.object_models.common.UShort

class BodyPartList (template=None, argument=None, parent=None)
    Bases: pyffi.object_models.xml.struct_.StructBase
    Body part list for DismemberSkinInstance

    property body_part
    property part_flag

class BoneLOD (template=None, argument=None, parent=None)
    Bases: pyffi.object_models.xml.struct_.StructBase
    Stores Bone Level of Detail info in a BSBoneLODExtraData

    property bone_name
    property distance

class BoundVolumeType (**kwargs)
    Bases: pyffi.object_models.xml.enum.EnumBase

    BASE_BV = 4294967295
    BOX_BV = 1
    CAPSULE_BV = 2
    HALFSPACE_BV = 5
    SPHERE_BV = 0
    UNION_BV = 4

class BoundingBox (template=None, argument=None, parent=None)
    Bases: pyffi.object_models.xml.struct_.StructBase
    Bounding box.

    property radius
    property rotation
    property translation
    property unknown_int

class BoundingVolume (template=None, argument=None, parent=None)
    Bases: pyffi.object_models.xml.struct_.StructBase

    property box
    property capsule
    property collision_type
    property half_space
    property sphere
    property union

class BoxBV (template=None, argument=None, parent=None)
    Bases: pyffi.object_models.xml.struct_.StructBase
    Box Bounding Volume

```

```
    property axis
    property center
    property extent
class ByteArray (**kwargs)
    Bases: pyffi.object_models.xml.basic.BasicBase

    Array (list) of bytes. Implemented as basic type to speed up reading and also to prevent data to be dumped
    by __str__.

    get_hash (data=None)
        Returns a hash value (an immutable object) that can be used to identify the object uniquely.

    get_size (data=None)
        Returns size of the object in bytes.

    get_value ()
        Return object value.

    read (stream, data)
        Read object from file.

    set_value (value)
        Set object value.

    write (stream, data)
        Write object to file.

class ByteColor3 (template=None, argument=None, parent=None)
    Bases: pyffi.object_models.xml.struct_.StructBase

    A color without alpha (red, green, blue).

    property b
    property g
    property r

class ByteColor4 (template=None, argument=None, parent=None)
    Bases: pyffi.object_models.xml.struct_.StructBase

    A color with alpha (red, green, blue, alpha).

    property a
    property b
    property g
    property r

class ByteMatrix (**kwargs)
    Bases: pyffi.object_models.xml.basic.BasicBase

    Matrix of bytes. Implemented as basic type to speed up reading and to prevent data being dumped by
    __str__.

    get_hash (data=None)
        Returns a hash value (an immutable object) that can be used to identify the object uniquely.

    get_size (data=None)
        Returns size of the object in bytes.
```

```

    get_value()
        Return object value.

    read(stream, data)
        Read object from file.

    set_value(value)
        Set object value.

    write(stream, data)
        Write object to file.

class CStreamableAssetData (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiObject

    property root

    property unknown_bytes

class CapsuleBV (template=None, argument=None, parent=None)
    Bases: pyffi.object_models.xml.struct_.StructBase

    Capsule Bounding Volume

    property center

    property origin

    property unknown_float_1

    property unknown_float_2

class ChannelConvention (**kwargs)
    Bases: pyffi.object_models.xml.enum.EnumBase

    CC_COMPRESSED = 4

    CC_EMPTY = 5

    CC_FIXED = 0

    CC_INDEX = 3

class ChannelData (template=None, argument=None, parent=None)
    Bases: pyffi.object_models.xml.struct_.StructBase

    Channel data

    property bits_per_channel

    property convention

    property type

    property unknown_byte_1

class ChannelType (**kwargs)
    Bases: pyffi.object_models.xml.enum.EnumBase

    CHNL_ALPHA = 3

    CHNL_BLUE = 2

    CHNL_COMPRESSED = 4

    CHNL_EMPTY = 19

    CHNL_GREEN = 1

```

```
CHNL_INDEX = 16
CHNL_RED = 0

class CloningBehavior (**kwargs)
    Bases: pyffi.object_models.xml.enum.EnumBase
    Sets how objects are to be cloned.
    CLONING_BLANK_COPY = 2
    CLONING_COPY = 1
    CLONING_SHARE = 0

class CollisionMode (**kwargs)
    Bases: pyffi.object_models.xml.enum.EnumBase
    CM_NOTEST = 3
    CM_USE_ABV = 2
    CM_USE_NIBOUND = 4
    CM_USE_OBB = 0
    CM_USE_TRI = 1

class Color3 (template=None, argument=None, parent=None)
    Bases: pyffi.object_models.xml.struct_.StructBase
    A color without alpha (red, green, blue).
    property b
    property g
    property r

class Color4 (template=None, argument=None, parent=None)
    Bases: pyffi.object_models.xml.struct_.StructBase
    A color with alpha (red, green, blue, alpha).
    property a
    property b
    property g
    property r

class ComponentFormat (**kwargs)
    Bases: pyffi.object_models.xml.enum.EnumBase
    The data format of components.
    F_FLOAT16_1 = 66097
    F_FLOAT16_2 = 131634
    F_FLOAT16_3 = 197171
    F_FLOAT16_4 = 262708
    F_FLOAT32_1 = 66613
    F_FLOAT32_2 = 132150
    F_FLOAT32_3 = 197687
```

```
F_FLOAT32_4 = 263224
F_INT16_1 = 66065
F_INT16_2 = 131602
F_INT16_3 = 197139
F_INT16_4 = 262676
F_INT32_1 = 66593
F_INT32_2 = 132130
F_INT32_3 = 197667
F_INT32_4 = 263204
F_INT8_1 = 65793
F_INT8_2 = 131330
F_INT8_3 = 196867
F_INT8_4 = 262404
F_NORMINT16_1 = 66073
F_NORMINT16_2 = 131610
F_NORMINT16_3 = 197147
F_NORMINT16_4 = 262684
F_NORMINT32_1 = 66601
F_NORMINT32_2 = 132138
F_NORMINT32_3 = 197675
F_NORMINT32_4 = 263212
F_NORMINT8_1 = 65801
F_NORMINT8_2 = 131338
F_NORMINT8_3 = 196875
F_NORMINT8_4 = 262412
F_NORMINT_10_10_10_2 = 66621
F_NORMINT_10_10_10_L1 = 66618
F_NORMINT_11_11_10 = 66619
F_NORMUINT16_1 = 66077
F_NORMUINT16_2 = 131614
F_NORMUINT16_3 = 197151
F_NORMUINT16_4 = 262688
F_NORMUINT32_1 = 66605
F_NORMUINT32_2 = 132142
F_NORMUINT32_3 = 197679
F_NORMUINT32_4 = 263216
```

```
F_NORMUINT8_1 = 65805
F_NORMUINT8_2 = 131342
F_NORMUINT8_3 = 196879
F_NORMUINT8_4 = 262416
F_NORMUINT8_4_BGRA = 262460
F_UINT16_1 = 66069
F_UINT16_2 = 131606
F_UINT16_3 = 197143
F_UINT16_4 = 262680
F_UINT32_1 = 66597
F_UINT32_2 = 132134
F_UINT32_3 = 197671
F_UINT32_4 = 263208
F_UINT8_1 = 65797
F_UINT8_2 = 131334
F_UINT8_3 = 196871
F_UINT8_4 = 262408
F_UINT_10_10_10_2 = 66622
F_UINT_10_10_10_L1 = 66617
F_UNKNOWN = 0
```

```
class ConsistencyType (**kwargs)
```

```
    Bases: pyffi.object_models.xml.enum.EnumBase
```

Used by NiGeometryData to control the volatility of the mesh. While they appear to be flags they behave as an enum.

```
CT_MUTABLE = 0
```

```
CT_STATIC = 16384
```

```
CT_VOLATILE = 32768
```

```
class ControllerLink (template=None, argument=None, parent=None)
```

```
    Bases: pyffi.formats.nif._ControllerLink, object
```

```
>>> from pyffi.formats.nif import NifFormat
>>> link = NifFormat.ControllerLink()
>>> link.node_name_offset
-1
>>> link.set_node_name("Bip01")
>>> link.node_name_offset
0
>>> link.get_node_name()
b'Bip01'
>>> link.node_name
b'Bip01'
>>> link.set_node_name("Bip01 Tail")
```

(continues on next page)

(continued from previous page)

```
>>> link.node_name_offset
6
>>> link.get_node_name()
b'Bip01 Tail'
>>> link.node_name
b'Bip01 Tail'
```

get_controller_type()

get_node_name()

Return the node name.

```
>>> # a doctest
>>> from pyffi.formats.nif import NifFormat
>>> link = NifFormat.ControllerLink()
>>> link.string_palette = NifFormat.NiStringPalette()
>>> palette = link.string_palette.palette
>>> link.node_name_offset = palette.add_string("Bip01")
>>> link.get_node_name()
b'Bip01'
```

```
>>> # another doctest
>>> from pyffi.formats.nif import NifFormat
>>> link = NifFormat.ControllerLink()
>>> link.node_name = "Bip01"
>>> link.get_node_name()
b'Bip01'
```

get_property_type()

get_variable_1()

get_variable_2()

set_controller_type(text)

set_node_name(text)

set_property_type(text)

set_variable_1(text)

set_variable_2(text)

class CoordGenType(kwargs)**

Bases: pyffi.object_models.xml.enum.EnumBase

Determines the way that UV texture coordinates are generated.

CG_DIFFUSE_CUBE_MAP = 4

CG_SPECULAR_CUBE_MAP = 3

CG_SPHERE_MAP = 2

CG_WORLD_PARALLEL = 0

CG_WORLD_PERSPECTIVE = 1

class CycleType(kwargs)**

Bases: pyffi.object_models.xml.enum.EnumBase

The animation cycle behavior.

```
CYCLE_CLAMP = 2
CYCLE_LOOP = 0
CYCLE_REVERSE = 1
```

```
class Data (version=67108866, user_version=0, user_version_2=0)
```

Bases: `pyffi.object_models.Data`

A class to contain the actual nif data.

Note that `L{header}` and `L{blocks}` are not automatically kept in sync with the rest of the nif data, but they are resynchronized when calling `L{write}`.

Variables

- **version** – The nif version.
- **user_version** – The nif user version.
- **user_version_2** – The nif user version 2.
- **roots** – List of root blocks.
- **header** – The nif header.
- **blocks** – List of blocks.
- **modification** – Neo Steam (“neosteam”) or Ndoors (“nddoors”) or Joymaster Interactive Howling Sword (“jmihs1”) or Laxe Lore (“laxelore”) style nif?

```
class VersionUInt (**kwargs)
```

Bases: `pyffi.object_models.common.UInt`

get_detail_display()

Return an object that can be used to display the instance.

set_value(value)

Set value to `C{value}`. Calls `C{int(value)}` to convert to integer.

Parameters **value** (*int*) – The value to assign.

get_detail_child_names (*edge_filter=(True, True)*)

Generator which yields all child names of this item in the detail view.

Override this method if the node has children.

Returns Generator for detail tree child names.

Return type generator yielding `strs`

get_detail_child_nodes (*edge_filter=(True, True)*)

Generator which yields all children of this item in the detail view (by default, all acyclic and active ones).

Override this method if the node has children.

Parameters **edge_filter** (`EdgeFilter` or `type(None)`) – The edge type to include.

Returns Generator for detail tree child nodes.

Return type generator yielding `DetailNodes`

get_global_child_nodes (*edge_filter=(True, True)*)

Generator which yields all children of this item in the global view, of given edge type (default is edges of type 0).

Override this method.

Returns Generator for global node children.

inspect (*stream*)
 Quickly checks whether the stream appears to contain nif data, and read the nif header. Resets stream to original position.
 Call this function if you only need to inspect the header of the nif.
Parameters **stream** (*file*) – The file to inspect.

inspect_version_only (*stream*)
 This function checks the version only, and is faster than the usual inspect function (which reads the full header). Sets the L{version} and L{user_version} instance variables if the stream contains a valid NIF file.
 Call this function if you simply wish to check that a file is a NIF file without having to parse even the header.
Raises **ValueError** – If the stream does not contain a NIF file.
Parameters **stream** (*file*) – The stream from which to read.

read (*stream*)
 Read a NIF file. Does not reset stream position.
Parameters **stream** (*file*) – The stream from which to read.

replace_global_node (*oldbranch, newbranch, edge_filter=(True, True)*)
 Replace a particular branch in the graph.

property user_version
property user_version_2
property version

write (*stream*)
 Write a NIF file. The L{header} and the L{blocks} are recalculated from the tree at L{roots} (e.g. list of block types, number of blocks, list of block types, list of strings, list of block sizes etc.).
Parameters **stream** (*file*) – The stream to which to write.

class DataStreamAccess (*template=None, argument=None, parent=None*)
 Bases: `pyffi.object_models.xml.bit_struct.BitStructBase`
 Determines how the data stream is accessed?
property cpu_read
property cpu_write_mutable
property cpu_write_static
property cpu_write_static_initialized
property cpu_write_volatile
property gpu_read
property gpu_write

class DataStreamUsage (***kwargs*)
 Bases: `pyffi.object_models.xml.enum.EnumBase`
 Determines how a data stream is used?
USAGE_SHADER_CONSTANT = 2
USAGE_USER = 3
USAGE_VERTEX = 1
USAGE_VERTEX_INDEX = 0

```
class DeactivatorType (**kwargs)
    Bases: pyffi.object_models.xml.enum.EnumBase

    DEACTIVATOR_INVALID = 0
    DEACTIVATOR_NEVER = 1
    DEACTIVATOR_SPATIAL = 2

class DecalVectorArray (template=None, argument=None, parent=None)
    Bases: pyffi.object_models.xml.struct_.StructBase

    Array of Vectors for Decal placement in BSDecalPlacementVectorExtraData.

    property normals
    property num_vectors
    property points

class DecayType (**kwargs)
    Bases: pyffi.object_models.xml.enum.EnumBase

    Determines decay function. Used by NiPSysBombModifier.

    DECAY_EXPONENTIAL = 2
    DECAY_LINEAR = 1
    DECAY_NONE = 0

class DistantLODShaderProperty (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.BSShaderProperty

    Bethesda-specific node.

EPSILON = 0.0001

class EffectShaderControlledColor (**kwargs)
    Bases: pyffi.object_models.xml.enum.EnumBase

    An unsigned 32-bit integer, describing which color in BSEffectShaderProperty to animate.

class EffectShaderControlledVariable (**kwargs)
    Bases: pyffi.object_models.xml.enum.EnumBase

    An unsigned 32-bit integer, describing which float variable in BSEffectShaderProperty to animate.

    EmissiveMultiple = 0

class EffectType (**kwargs)
    Bases: pyffi.object_models.xml.enum.EnumBase

    The type of information that's store in a texture used by a NiTextureEffect.

    EFFECT_ENVIRONMENT_MAP = 2
    EFFECT_FOG_MAP = 3
    EFFECT_PROJECTED_LIGHT = 0
    EFFECT_PROJECTED_SHADOW = 1

class ElementReference (template=None, argument=None, parent=None)
    Bases: pyffi.object_models.xml.struct_.StructBase

    property normalize_flag
    property semantic
```

```

class EmitFrom(**kwargs)
    Bases: pyffi.object_models.xml.enum.EnumBase
    Controls which parts of the mesh that the particles are emitted from.
    EMIT_FROM_EDGE_CENTER = 2
    EMIT_FROM_EDGE_SURFACE = 4
    EMIT_FROM_FACE_CENTER = 1
    EMIT_FROM_FACE_SURFACE = 3
    EMIT_FROM_VERTICES = 0

class EndianType(**kwargs)
    Bases: pyffi.object_models.xml.enum.EnumBase
    ENDIAN_BIG = 0
    ENDIAN_LITTLE = 1

class ExportInfo(template=None, argument=None, parent=None)
    Bases: pyffi.object_models.xml.struct_.StructBase
    Information about how the file was exported
    property creator
    property export_info_1
    property export_info_2
    property unknown

class ExtraMeshDataEpicMickey(template=None, argument=None, parent=None)
    Bases: pyffi.object_models.xml.struct_.StructBase
    property unknown_int_1
    property unknown_int_2
    property unknown_int_3
    property unknown_int_4
    property unknown_int_5
    property unknown_int_6

class ExtraMeshDataEpicMickey2(template=None, argument=None, parent=None)
    Bases: pyffi.object_models.xml.struct_.StructBase
    property end
    property start
    property unknown_shorts

class ExtraVectorsFlags(**kwargs)
    Bases: pyffi.object_models.xml.enum.EnumBase
    None = 0
    Tangents_Bitangents = 16

```

```
class FaceDrawMode (**kwargs)
    Bases: pyffi.object_models.xml.enum.EnumBase

    This enum lists the different face culling options.

    DRAW_BOTH = 3
    DRAW_CCW = 1
    DRAW_CCW_OR_BOTH = 0
    DRAW_CW = 2

class Fallout3HavokMaterial (**kwargs)
    Bases: pyffi.object_models.xml.enum.EnumBase

    A material, used by havok shape objects in Fallout 3. Bit 5: flag for PLATFORM (for values 32-63
    subtract 32 to know material number) Bit 6: flag for STAIRS (for values 64-95 subtract 64 to know
    material number) Bit 5+6: flag for STAIRS+PLATFORM (for values 96-127 subtract 96 to know material
    number)

    MAT_BABY_RATTLE = 30
    MAT_BARREL = 23
    MAT_BOTTLE = 24
    MAT_BOTTLECAP = 14
    MAT_BROKEN_CONCRETE = 19
    MAT_CHAIN = 13
    MAT_CLOTH = 1
    MAT_DIRT = 2
    MAT_ELEVATOR = 15
    MAT_GLASS = 3
    MAT_GRASS = 4
    MAT_HEAVY_METAL = 11
    MAT_HEAVY_STONE = 10
    MAT_HEAVY_WOOD = 12
    MAT_HOLLOW_METAL = 16
    MAT_LUNCHBOX = 29
    MAT_METAL = 5
    MAT_ORGANIC = 6
    MAT_PISTOL = 26
    MAT_RIFLE = 27
    MAT_RUBBER BALL = 31
    MAT_SAND = 18
    MAT_SHEET_METAL = 17
    MAT_SHOPPING_CART = 28
    MAT_SKIN = 7
```

```

MAT_SODA_CAN = 25
MAT_STONE = 0
MAT_VEHICLE_BODY = 20
MAT_VEHICLE_PART_HOLLOW = 22
MAT_VEHICLE_PART_SOLID = 21
MAT_WATER = 8
MAT_WOOD = 9
class Fallout3Layer(**kwargs)
    Bases: pyffi.object_models.xml.enum.EnumBase
    Sets mesh color in Fallout 3 GECK. Anything higher than 72 is also null.
    ACOUSTIC_SPACE = 21
    ACTORZONE = 22
    ADDONARM = 71
    ADDONCHEST = 70
    ADDONHEAD = 69
    ADDONLEG = 72
    ANIM_STATIC = 2
    AVOIDBOX = 31
    BIPED = 29
    BODY = 46
    CAMERAPICK = 35
    CAMERASPHERE = 33
    CHAIN = 68
    CHARCONTROLLER = 30
    CLOUD_TRAP = 16
    CLUTTER = 4
    COLLISIONBOX = 32
    CUSTOMPICK1 = 39
    CUSTOMPICK2 = 40
    DEBRIS_LARGE = 20
    DEBRIS_SMALL = 19
    DOORDETECTION = 34
    DROPPINGPICK = 42
    GASTRAP = 24
    GROUND = 17
    HEAD = 45

```

```
INVISIBLE_WALL = 27
ITEMPICK = 36
LINEOFSIGHT = 37
L_CALF = 53
L_FOOT = 54
L_FORE_ARM = 50
L_HAND = 51
L_THIGH = 52
L_UPPER_ARM = 49
NONCOLLIDABLE = 15
NULL = 43
OTHER = 44
PACK = 67
PATHPICK = 38
PONYTAIL = 65
PORTAL = 18
PROJECTILE = 6
PROJECTILEZONE = 23
PROPS = 10
QUIVER = 63
R_CALF = 59
R_FOOT = 60
R_FORE_ARM = 56
R_HAND = 57
R_THIGH = 58
R_UPPER_ARM = 55
SHELLCASING = 25
SHIELD = 62
SPELL = 7
SPELLEXPLOSION = 41
SPINE1 = 47
SPINE2 = 48
STATIC = 1
TAIL = 61
TERRAIN = 13
TRANSPARENT = 3
```



```

TRANSPARENT_SMALL = 26
TRANSPARENT_SMALL_ANIM = 28
TRAP = 14
TREES = 9
TRIGGER = 12
UNIDENTIFIED = 0
WATER = 11
WEAPON = 64
WING = 66

class FieldType(**kwargs)
    Bases: pyffi.object_models.xml.enum.EnumBase
    The force field's type.
    FIELD_POINT = 1
    FIELD_WIND = 0

class FilePath(**kwargs)
    Bases: pyffi.formats.nif.string
    A file path.
    get_hash(data=None)
        Returns a case insensitive hash value.

class FileVersion(**kwargs)
    Bases: pyffi.object_models.common.UInt
    get_detail_display()
        Return an object that can be used to display the instance.
    read(stream, data)
        Read value from stream.
        Parameters stream (file) – The stream to read from.
    set_value()
        Set value to C{value}. Calls C{int(value)} to convert to integer.
        Parameters value (int) – The value to assign.
    write(stream, data)
        Write value to stream.
        Parameters stream (file) – The stream to write to.

class Flags(**kwargs)
    Bases: pyffi.object_models.common.UShort

class Footer(template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif._Footer, object
    read(stream, data)
        Read structure from stream.
    write(stream, data)
        Write structure to stream.

```

```
class ForceType (**kwargs)
    Bases: pyffi.object_models.xml.enum.EnumBase

    The type of force? May be more valid values.

    FORCE_PLANAR = 0
    FORCE_SPHERICAL = 1
    FORCE_UNKNOWN = 2

class FurnitureEntryPoints (template=None, argument=None, parent=None)
    Bases: pyffi.object_models.xml.bit_struct.BitStructBase

    Furniture entry points. It specifies the direction(s) from where the actor is able to enter (and leave) the position.

    property behind
    property front
    property left
    property right
    property up

class FurniturePosition (template=None, argument=None, parent=None)
    Bases: pyffi.object_models.xml.struct_.StructBase

    Describes a furniture position?

    property animation_type
    property entry_properties
    property heading
    property offset
    property orientation
    property position_ref_1
    property position_ref_2

class FxButton (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.FxWidget

    Unknown.

class FxRadioButton (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.FxWidget

    Unknown.

    property buttons
    property num_buttons
    property unknown_int_1
    property unknown_int_2
    property unknown_int_3
```

```

class FxWidget (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiNode
    Firaxis-specific UI widgets?
    property unknown_292_bytes
    property unknown_3

class HairShaderProperty (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.BSShaderProperty
    Bethesda-specific node.

class HalfSpaceBV (template=None, argument=None, parent=None)
    Bases: pyffi.object_models.xml.struct_.StructBase
    property center
    property normal
    property unknown_float_1

class HavokColFilter (template=None, argument=None, parent=None)
    Bases: pyffi.object_models.xml.struct_.StructBase
    ColFilter property for Havok. It contains Layer, Flags and Part Number
    property flags_and_part_number
    property layer
    property unknown_short

class HavokMaterial (template=None, argument=None, parent=None)
    Bases: pyffi.object_models.xml.struct_.StructBase
    property material

class Header (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif._Header, object
    has_block_type (block_type)
        Check if header has a particular block type.
        Raises ValueError – If number of block types is zero (only nif versions 10.0.1.0 and up
            store block types in header).
        Parameters block_type (L{NiFormat.NiObject}) – The block type.
        Returns True if the header's list of block types has the given block type, or a subclass of it.
            False otherwise.
        Return type bool

class HeaderString (template=None, argument=None, parent=None)
    Bases: pyffi.object_models.xml.basic.BasicBase
    get_detail_display ()
        Return an object that can be used to display the instance.
    get_hash (data=None)
        Returns a hash value (an immutable object) that can be used to identify the object uniquely.
    get_size (data=None)
        Returns size of the object in bytes.
    read (stream, data)
        Read object from file.

```

static version_string(*version, modification=None*)

Transforms version number into a version string.

```
>>> NifFormat.HeaderString.version_string(0x03000300)
'NetImmerse File Format, Version 3.03'
>>> NifFormat.HeaderString.version_string(0x03010000)
'NetImmerse File Format, Version 3.1'
>>> NifFormat.HeaderString.version_string(0x0A000100)
'NetImmerse File Format, Version 10.0.1.0'
>>> NifFormat.HeaderString.version_string(0x0A010000)
'Gamebryo File Format, Version 10.1.0.0'
>>> NifFormat.HeaderString.version_string(0x0A010000,
...                                     modification="neosteam")
'NS'
>>> NifFormat.HeaderString.version_string(0x14020008,
...                                     modification="nddoors")
'NDSNIF....@....@...., Version 20.2.0.8'
>>> NifFormat.HeaderString.version_string(0x14030009,
...                                     modification="jmihs1")
'Joymaster Hs1 Object Format - (JMI), Version 20.3.0.9'
```

write(*stream, data*)

Write object to file.

class HingeDescriptor(*template=None, argument=None, parent=None*)

Bases: `pyffi.object_models.xml.struct_.StructBase`

This constraint allows rotation about a specified axis.

property axle_a

property axle_b

property perp_2_axle_in_a_1

property perp_2_axle_in_a_2

property perp_2_axle_in_b_1

property perp_2_axle_in_b_2

property pivot_a

property pivot_b

class ImageType(***kwargs*)

Bases: `pyffi.object_models.xml.enum.EnumBase`

Determines how the raw image data is stored in `NiRawImageData`.

RGB = 1

RGBA = 2

class InertiaMatrix(*template=None, argument=None, parent=None*)

Bases: `pyffi.formats.nif._InertiaMatrix, object`

as_list()

Return matrix as 3x3 list.

as_tuple()

Return matrix as 3x3 tuple.

get_copy()

Return a copy of the matrix.

```

is_identity()
    Return True if the matrix is close to identity.

set_identity()
    Set to identity matrix.

class Key (template=None, argument=None, parent=None)
    Bases: pyffi.object_models.xml.struct_.StructBase

    A generic key with support for interpolation. Type 1 is normal linear interpolation, type 2 has forward
    and backward tangents, and type 3 has tension, bias and continuity arguments. Note that color4 and byte
    always seem to be of type 1.

    property backward
    property forward
    property tbc
    property time
    property value

class KeyGroup (template=None, argument=None, parent=None)
    Bases: pyffi.object_models.xml.struct_.StructBase

    Array of vector keys (anything that can be interpolated, except rotations).

    property interpolation
    property keys
    property num_keys

class KeyType (**kwargs)
    Bases: pyffi.object_models.xml.enum.EnumBase

    The type of animation interpolation (blending) that will be used on the associated key frames.

    CONST_KEY = 5
    LINEAR_KEY = 1
    QUADRATIC_KEY = 2
    TBC_KEY = 3
    XYZ_ROTATION_KEY = 4

class LODRange (template=None, argument=None, parent=None)
    Bases: pyffi.object_models.xml.struct_.StructBase

    The distance range where a specific level of detail applies.

    property far_extent
    property near_extent
    property unknown_ints

class LightMode (**kwargs)
    Bases: pyffi.object_models.xml.enum.EnumBase

    An unsigned 32-bit integer, describing how vertex colors influence lighting.

    LIGHT_MODE_EMISSIVE = 0
    LIGHT_MODE_EMI_AMB_DIF = 1

```

```
class Lighting30ShaderProperty (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.BSShaderPPLightingProperty

    Bethesda-specific node.

class LightingShaderControlledColor (**kwargs)
    Bases: pyffi.object_models.xml.enum.EnumBase

    An unsigned 32-bit integer, describing which color in BSLightingShaderProperty to animate.

class LightingShaderControlledVariable (**kwargs)
    Bases: pyffi.object_models.xml.enum.EnumBase

    An unsigned 32-bit integer, describing which float variable in BSLightingShaderProperty to animate.

Alpha = 12
Glossiness = 9

class LimitedHingeDescriptor (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif._LimitedHingeDescriptor, object

    update_a_b (transform)
        Update B pivot and axes from A using the given transform.

class LineString (**kwargs)
    Bases: pyffi.object_models.xml.basic.BasicBase

    Basic type for strings ending in a newline character (0x0a).
```

```
>>> from tempfile import TemporaryFile
>>> f = TemporaryFile()
>>> l = NifFormat.LineString()
>>> f.write('abcdefg\x0a'.encode())
8
>>> f.seek(0)
0
>>> l.read(f)
>>> str(l)
'abcdefg'
>>> f.seek(0)
0
>>> l.set_value('Hi There')
>>> l.write(f)
>>> f.seek(0)
0
>>> m = NifFormat.LineString()
>>> m.read(f)
>>> str(m)
'Hi There'
```

```
get_hash (data=None)
    Returns a hash value (an immutable object) that can be used to identify the object uniquely.

get_size (data=None)
    Returns size of the object in bytes.

get_value ()
    Return object value.

read (stream, data=None)
    Read object from file.
```

```

set_value (value)
    Set object value.

write (stream, data=None)
    Write object to file.

class MTransform (template=None, argument=None, parent=None)
    Bases: pyffi.object_models.xml.struct_.StructBase

    property rotation

    property scale

    property translation

class MatchGroup (template=None, argument=None, parent=None)
    Bases: pyffi.object_models.xml.struct_.StructBase

    Group of vertex indices of vertices that match.

    property num_vertices

    property vertex_indices

class MaterialData (template=None, argument=None, parent=None)
    Bases: pyffi.object_models.xml.struct_.StructBase

    Data stored per-material by NiRenderObject

    property material_extra_data

    property material_name

class Matrix22 (template=None, argument=None, parent=None)
    Bases: pyffi.object_models.xml.struct_.StructBase

    A 2x2 matrix of float values. Stored in OpenGL column-major format.

    property m_11

    property m_12

    property m_21

    property m_22

class Matrix33 (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif._Matrix33, object

    as_list ()
        Return matrix as 3x3 list.

    as_tuple ()
        Return matrix as 3x3 tuple.

    get_copy ()
        Return a copy of the matrix.

    get_determinant ()
        Return determinant.

    get_inverse ()
        Get inverse (assuming is_scale_rotation is true!).

    get_scale ()
        Gets the scale (assuming is_scale_rotation is true!).

```

```
get_scale_quat()
    Decompose matrix into scale and quaternion.

get_scale_rotation()
    Decompose the matrix into scale and rotation, where scale is a float and rotation is a C{Matrix33}.
    Returns a pair (scale, rotation).

get_transpose()
    Get transposed of the matrix.

is_identity()
    Return True if the matrix is close to identity.

is_rotation()
    Returns True if the matrix is a rotation matrix (a member of SO(3)).

is_scale_rotation()
    Returns true if the matrix decomposes nicely into scale * rotation.

set_identity()
    Set to identity matrix.

set_scale_rotation(scale, rotation)
    Compose the matrix as the product of scale * rotation.

sup_norm()
    Calculate supremum norm of matrix (maximum absolute value of all entries).

class Matrix44 (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif._Matrix44, object

as_list()
    Return matrix as 4x4 list.

as_tuple()
    Return matrix as 4x4 tuple.

get_copy()
    Create a copy of the matrix.

get_inverse(fast=True)
    Calculates inverse (fast assumes is_scale_rotation_translation is True).

get_matrix_33()
    Returns upper left 3x3 part.

get_scale_quat_translation()

get_scale_rotation_translation()

get_translation()
    Returns lower left 1x3 part.

is_identity()
    Return True if the matrix is close to identity.

is_scale_rotation_translation()

set_identity()
    Set to identity matrix.

set_matrix_33(m)
    Sets upper left 3x3 part.
```



```

set_rows (row0, row1, row2, row3)
    Set matrix from rows.

set_scale_rotation_translation (scale, rotation, translation)

set_translation (translation)
    Returns lower left 1x3 part.

sup_norm ()
    Calculate supremum norm of matrix (maximum absolute value of all entries).

class MeshData (template=None, argument=None, parent=None)
    Bases: pyffi.object_models.xml.struct_.StructBase

    property component_semantics

    property is_per_instance

    property num_components

    property num_submeshes

    property stream

    property submesh_to_region_map

class MeshPrimitiveType (**kwargs)
    Bases: pyffi.object_models.xml.enum.EnumBase

    Describes the type of primitives stored in a mesh object.

    MESH_PRIMITIVE_LINESTRIPS = 2

    MESH_PRIMITIVE_POINTS = 4

    MESH_PRIMITIVE_QUADS = 3

    MESH_PRIMITIVE_TRIANGLES = 0

    MESH_PRIMITIVE_TRISTRIPS = 1

class MipMap (template=None, argument=None, parent=None)
    Bases: pyffi.object_models.xml.struct_.StructBase

    Description of a MipMap within a NiPixelData object.

    property height

    property offset

    property width

class MipMapFormat (**kwargs)
    Bases: pyffi.object_models.xml.enum.EnumBase

    An unsigned 32-bit integer, describing how mipmaps are handled in a texture.

    MIP_FMT_DEFAULT = 2

    MIP_FMT_NO = 0

    MIP_FMT_YES = 1

class MoppDataBuildType (**kwargs)
    Bases: pyffi.object_models.xml.enum.EnumBase

    A byte describing if MOPP Data is organized into chunks (PS3) or not (PC)

    BUILD_NOT_SET = 2

```

```
BUILT_WITHOUT_CHUNK_SUBDIVISION = 1
BUILT_WITH_CHUNK_SUBDIVISION = 0
class Morph (template=None, argument=None, parent=None)
    Bases: pyffi.object_models.xml.struct_.StructBase
    Geometry morphing data component.
    property frame_name
    property interpolation
    property keys
    property num_keys
    property unknown_int
    property vectors
class MorphWeight (template=None, argument=None, parent=None)
    Bases: pyffi.object_models.xml.struct_.StructBase
    property interpolator
    property weight
class MotionQuality (**kwargs)
    Bases: pyffi.object_models.xml.enum.EnumBase
    The motion type. Determines quality of motion?
    MO_QUAL_BULLET = 6
    MO_QUAL_CHARACTER = 8
    MO_QUAL_CRITICAL = 5
    MO_QUAL_DEBRIS = 3
    MO_QUAL_FIXED = 1
    MO_QUAL_INVALID = 0
    MO_QUAL_KEYFRAMED = 2
    MO_QUAL_KEYFRAMED_REPORT = 9
    MO_QUAL_MOVING = 4
    MO_QUAL_USER = 7
class MotionSystem (**kwargs)
    Bases: pyffi.object_models.xml.enum.EnumBase
    The motion system. 4 (Box) is used for everything movable. 7 (Keyframed) is used on statics and animated
    stuff.
    MO_SYS_BOX = 4
    MO_SYS_BOX_STABILIZED = 5
    MO_SYS_CHARACTER = 9
    MO_SYS_DYNAMIC = 1
    MO_SYS_FIXED = 7
    MO_SYS_INVALID = 0
```

```

MO_SYS_KEYFRAMED = 6
MO_SYS_SPHERE = 2
MO_SYS_SPHERE_INERTIA = 3
MO_SYS_THIN_BOX = 8

class MotorDescriptor (template=None, argument=None, parent=None)
    Bases: pyffi.object_models.xml.struct_.StructBase
    property unknown_byte_1
    property unknown_float_1
    property unknown_float_2
    property unknown_float_3
    property unknown_float_4
    property unknown_float_5
    property unknown_float_6

class MultiTextureElement (template=None, argument=None, parent=None)
    Bases: pyffi.object_models.xml.struct_.StructBase
    property clamp
    property filter
    property has_image
    property image
    property ps_2_k
    property ps_2_l
    property unknown_short_3
    property uv_set

class Ni3dsAlphaAnimator (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiObject
    Unknown.
    property num_1
    property num_2
    property parent
    property unknown_1
    property unknown_2

class Ni3dsAnimationNode (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiObject
    Unknown. Only found in 2.3 nifs.
    property child
    property count
    property has_data

```

```
    property name
    property unknown_array
    property unknown_floats_1
    property unknown_floats_2
    property unknown_short

class Ni3dsColorAnimator (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiObject
    Unknown!
    property unknown_1

class Ni3dsMorphShape (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiObject
    Unknown!
    property unknown_1

class Ni3dsParticleSystem (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiObject
    Unknown!
    property unknown_1

class Ni3dsPathController (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiObject
    Unknown!
    property unknown_1

class NiAVObject (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif._NiAVObject, object
```

```
>>> from pyffi.formats.nif import NifFormat
>>> node = NifFormat.NiNode()
>>> prop1 = NifFormat.NiProperty()
>>> prop1.name = "hello"
>>> prop2 = NifFormat.NiProperty()
>>> prop2.name = "world"
>>> node.get_properties()
[]
>>> node.set_properties([prop1, prop2])
>>> [prop.name for prop in node.get_properties()]
[b'hello', b'world']
>>> [prop.name for prop in node.properties]
[b'hello', b'world']
>>> node.set_properties([])
>>> node.get_properties()
[]
>>> # now set them the other way around
>>> node.set_properties([prop2, prop1])
>>> [prop.name for prop in node.get_properties()]
[b'world', b'hello']
>>> [prop.name for prop in node.properties]
[b'world', b'hello']
>>> node.remove_property(prop2)
```

(continues on next page)

(continued from previous page)

```
>>> [prop.name for prop in node.properties]
[b'hello']
>>> node.add_property(prop2)
>>> [prop.name for prop in node.properties]
[b'hello', b'world']
```

add_property (*prop*)

Add the given property to the property list.

Parameters **prop** (*L{NifFormat.NiProperty}*) – The property block to add.**apply_scale** (*scale*)

Apply scale factor on data.

Parameters **scale** – The scale factor.**get_properties** ()

Return a list of the properties of the block.

Returns The list of properties.**Return type** list of *L{NifFormat.NiProperty}***get_transform** (*relative_to=None*)

Return scale, rotation, and translation into a single 4x4 matrix, relative to the C{relative_to} block (which should be another NiAVObject connecting to this block). If C{relative_to} is None, then returns the transform stored in C{self}, or equivalently, the target is assumed to be the parent.

Parameters **relative_to** – The block relative to which the transform must be calculated.

If None, the local transform is returned.

remove_property (*prop*)

Remove the given property to the property list.

Parameters **prop** (*L{NifFormat.NiProperty}*) – The property block to remove.**set_properties** (*proplist*)

Set the list of properties from the given list (destroys existing list).

Parameters **proplist** (list of *L{NifFormat.NiProperty}*) – The list of property blocks to set.**set_transform** (*m*)

Set rotation, translation, and scale, from a 4x4 matrix.

Parameters **m** – The matrix to which the transform should be set.**class NiAVObjectPalette** (*template=None, argument=None, parent=None*)Bases: *pyffi.formats.nif.NiObject*

Unknown.

class NiAdditionalGeometryData (*template=None, argument=None, parent=None*)Bases: *pyffi.formats.nif.AbstractAdditionalGeometryData***property_block_infos****property_blocks****property_num_block_infos****property_num_blocks****property_num_vertices****class NiAlphaController** (*template=None, argument=None, parent=None*)Bases: *pyffi.formats.nif.NiFloatInterpController*

Time controller for transparency.

```
    property data
class NiAlphaProperty (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiProperty
    Transparency. Flags 0x00ED.
    property flags
    property threshold
    property unknown_int_2
    property unknown_short_1
class NiAmbientLight (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiLight
    Ambient light source.
class NiArkAnimationExtraData (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiExtraData
    Unknown node.
    property unknown_bytes
    property unknown_ints
class NiArkImporterExtraData (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiExtraData
    Unknown node.
    property importer_name
    property unknown_bytes
    property unknown_floats
    property unknown_int_1
    property unknown_int_2
class NiArkShaderExtraData (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiExtraData
    Unknown node.
    property unknown_int
    property unknown_string
class NiArkTextureExtraData (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiExtraData
    Unknown node.
    property num_textures
    property textures
    property unknown_byte
    property unknown_int_2
    property unknown_ints_1
```

```

class NiArkViewportInfoExtraData (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiExtraData

    Unknown node.

    property unknown_bytes

class NiAutoNormalParticles (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiParticles

    Unknown.

class NiAutoNormalParticlesData (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiParticlesData

    Particle system data object (with automatic normals?).

class NiBSAnimationNode (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiNode

    Bethesda-specific extension of Node with animation properties stored in the flags, often 42?

class NiBSBoneLODController (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiBoneLODController

    A simple LOD controller for bones.

class NiBSPArrayController (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiParticleSystemController

    A particle system controller, used by BS in conjunction with NiBSParticleNode.

class NiBSParticleNode (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiNode

    Unknown.

class NiBSplineBasisData (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiObject

    Stores the number of control points of a B-spline.

    property num_control_points

class NiBSplineCompFloatInterpolator (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiBSplineFloatInterpolator

    Unknown.

    property base
    property bias
    property multiplier
    property offset

class NiBSplineCompPoint3Interpolator (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiBSplinePoint3Interpolator

    Unknown.

class NiBSplineCompTransformEvaluator (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiObject

```

```
class NiBSplineCompTransformInterpolator(template=None, argument=None, parent=None)
```

Bases: `pyffi.formats.nif._NiBSplineCompTransformInterpolator, object`

```
apply_scale(scale)
```

Apply scale factor on data.

```
get_rotations()
```

Return an iterator over all rotation keys.

```
get_scales()
```

Return an iterator over all scale keys.

```
get_translations()
```

Return an iterator over all translation keys.

```
class NiBSplineData(template=None, argument=None, parent=None)
```

Bases: `pyffi.formats.nif._NiBSplineData, object`

```
>>> # a doctest
>>> from pyffi.formats.nif import NifFormat
>>> block = NifFormat.NiBSplineData()
>>> block.num_short_control_points = 50
>>> block.short_control_points.update_size()
>>> for i in range(block.num_short_control_points):
...     block.short_control_points[i] = 20 - i
>>> list(block.get_short_data(12, 4, 3))
[(8, 7, 6), (5, 4, 3), (2, 1, 0), (-1, -2, -3)]
>>> offset = block.append_short_data([(1,2), (4,3), (13,14), (8,2), (33,33)])
>>> offset
50
>>> list(block.get_short_data(offset, 5, 2))
[(1, 2), (4, 3), (13, 14), (8, 2), (33, 33)]
>>> list(block.get_comp_data(offset, 5, 2, 10.0, 32767.0))
[(11.0, 12.0), (14.0, 13.0), (23.0, 24.0), (18.0, 12.0), (43.0, 43.0)]
>>> block.append_float_data([(1.0,2.0), (3.0,4.0), (0.5,0.25)])
0
>>> list(block.get_float_data(0, 3, 2))
[(1.0, 2.0), (3.0, 4.0), (0.5, 0.25)]
>>> block.append_comp_data([(1,2), (4,3)])
(60, 2.5, 1.5)
>>> list(block.get_short_data(60, 2, 2))
[(-32767, -10922), (32767, 10922)]
>>> list(block.get_comp_data(60, 2, 2, 2.5, 1.5))
[(1.0, 2.00...), (4.0, 2.99...)]
```

```
append_comp_data(data)
```

Append data as compressed list.

Parameters *data* – A list of elements, where each element is a tuple of integers. (Note: cannot be an iterator; maybe this restriction will be removed in a future version.)

Returns The offset, bias, and multiplier.

```
append_float_data(data)
```

Append data.

Parameters *data* – A list of elements, where each element is a tuple of floats. (Note: cannot be an iterator; maybe this restriction will be removed in a future version.)

Returns The offset at which the data was appended.

```
append_short_data(data)
```

Append data.

Parameters **data** – A list of elements, where each element is a tuple of integers. (Note: cannot be an iterator; maybe this restriction will be removed in a future version.)

Returns The offset at which the data was appended.

get_comp_data (*offset, num_elements, element_size, bias, multiplier*)

Get an iterator to the data, converted to float with extra bias and multiplication factor. If $C\{x\}$ is the short value, then the returned value is $C\{\text{bias} + x * \text{multiplier} / 32767.0\}$.

Parameters

- **offset** – The offset in the data where to start.
- **num_elements** – Number of elements to get.
- **element_size** – Size of a single element.
- **bias** – Value bias.
- **multiplier** – Value multiplier.

Returns A list of $C\{\text{num_elements}\}$ tuples of size $C\{\text{element_size}\}$.

get_float_data (*offset, num_elements, element_size*)

Get an iterator to the data.

Parameters

- **offset** – The offset in the data where to start.
- **num_elements** – Number of elements to get.
- **element_size** – Size of a single element.

Returns A list of $C\{\text{num_elements}\}$ tuples of size $C\{\text{element_size}\}$.

get_short_data (*offset, num_elements, element_size*)

Get an iterator to the data.

Parameters

- **offset** – The offset in the data where to start.
- **num_elements** – Number of elements to get.
- **element_size** – Size of a single element.

Returns A list of $C\{\text{num_elements}\}$ tuples of size $C\{\text{element_size}\}$.

class NiBSplineFloatInterpolator (*template=None, argument=None, parent=None*)

Bases: `pyffi.formats.nif.NiBSplineInterpolator`

Unknown.

class NiBSplineInterpolator (*template=None, argument=None, parent=None*)

Bases: `pyffi.formats.nif._NiBSplineInterpolator`, `object`

get_times ()

Return an iterator over all key times.

@todo: When code for calculating the bsplines is ready, this function will return exactly `self.basis_data.num_control_points - 1` time points, and not `self.basis_data.num_control_points` as it is now.

class NiBSplinePoint3Interpolator (*template=None, argument=None, parent=None*)

Bases: `pyffi.formats.nif.NiBSplineInterpolator`

Unknown.

property unknown_floats

class NiBSplineTransformInterpolator (*template=None, argument=None, parent=None*)

Bases: `pyffi.formats.nif._NiBSplineTransformInterpolator`, `object`

apply_scale (*scale*)

Apply scale factor on data.

get_rotations ()

Return an iterator over all rotation keys.

```
get_scales()
    Return an iterator over all scale keys.

get_translations()
    Return an iterator over all translation keys.

class NiBezierMesh (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiAVObject

    Unknown

    property bezier_triangle
    property count_1
    property count_2
    property data_2
    property num_bezier_triangles
    property points_1
    property points_2
    property unknown_3
    property unknown_4
    property unknown_5
    property unknown_6

class NiBezierTriangle4 (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiObject

    Sub data of NiBezierMesh

    property matrix
    property unknown_1
    property unknown_2
    property unknown_3
    property unknown_4
    property unknown_5
    property unknown_6
    property vector_1
    property vector_2

class NiBillboardNode (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiNode

    These nodes will always be rotated to face the camera creating a billboard effect for any attached objects.

    In pre-10.1.0.0 the Flags field is used for BillboardMode. Bit 0: hidden Bits 1-2: collision mode Bit 3:
    unknown (set in most official meshes) Bits 5-6: billboard mode

    Collision modes: 00 NONE 01 USE_TRIANGLES 10 USE_OBBS 11 CONTINUE

    Billboard modes:      00 ALWAYS_FACE_CAMERA      01 ROTATE_ABOUT_UP      10
    RIGID_FACE_CAMERA 11 ALWAYS_FACE_CENTER
```

```

    property billboard_mode
class NiBinaryExtraData (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiExtraData
    Binary extra data object. Used to store tangents and bitangents in Oblivion.
    property binary_data
class NiBinaryVoxelData (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiObject
    Voxel data object.
    property num_unknown_bytes_2
    property num_unknown_vectors
    property unknown_5_ints
    property unknown_7_floats
    property unknown_bytes_1
    property unknown_bytes_2
    property unknown_short_1
    property unknown_short_2
    property unknown_short_3
    property unknown_vectors
class NiBinaryVoxelExtraData (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiExtraData
    Voxel extra data object.
    property data
    property unknown_int
class NiBlendBoolInterpolator (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiBlendInterpolator
    An interpolator for a bool.
    property bool_value
class NiBlendFloatInterpolator (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiBlendInterpolator
    An interpolator for a float.
    property float_value
class NiBlendInterpolator (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiInterpolator
    An extended type of interpolater.
    property unknown_int
    property unknown_short

```

```
class NiBlendPoint3Interpolator (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiBlendInterpolator

    Interpolates a point?

    property point_value

class NiBlendTransformInterpolator (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiBlendInterpolator

    Unknown.

class NiBone (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiNode

    A NiNode used as a skeleton bone?

class NiBoneLODController (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiTimeController

    Level of detail controller for bones. Priority is arranged from low to high.

    property node_groups
    property num_node_groups
    property num_node_groups_2
    property num_shape_groups
    property num_shape_groups_2
    property shape_groups_1
    property shape_groups_2
    property unknown_int_1
    property unknown_int_2
    property unknown_int_3

class NiBoolData (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiObject

    Timed boolean data.

    property data

class NiBoolInterpController (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiSingleInterpController

    A controller that interpolates floating point numbers?

class NiBoolInterpolator (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiKeyBasedInterpolator

    Unknown.

    property bool_value
    property data

class NiBoolTimelineInterpolator (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiBoolInterpolator

    Unknown.
```

```
class NiBooleanExtraData (template=None, argument=None, parent=None)
```

```
    Bases: pyffi.formats.nif.NiExtraData
```

```
    Boolean extra data.
```

```
    property boolean_data
```

```
class NiCamera (template=None, argument=None, parent=None)
```

```
    Bases: pyffi.formats.nif.NiAVObject
```

```
    Camera object.
```

```
    property frustum_bottom
```

```
    property frustum_far
```

```
    property frustum_left
```

```
    property frustum_near
```

```
    property frustum_right
```

```
    property frustum_top
```

```
    property lod_adjust
```

```
    property unknown_int
```

```
    property unknown_int_2
```

```
    property unknown_int_3
```

```
    property unknown_link
```

```
    property unknown_short
```

```
    property use_orthographic_projection
```

```
    property viewport_bottom
```

```
    property viewport_left
```

```
    property viewport_right
```

```
    property viewport_top
```

```
class NiClod (template=None, argument=None, parent=None)
```

```
    Bases: pyffi.formats.nif.NiTriBasedGeom
```

```
    A shape node that holds continuous level of detail information. Seems to be specific to Freedom Force.
```

```
class NiClodData (template=None, argument=None, parent=None)
```

```
    Bases: pyffi.formats.nif.NiTriBasedGeomData
```

```
    Holds mesh data for continuous level of detail shapes. Presumably a progressive mesh with triangles specified by edge splits. Seems to be specific to Freedom Force. The structure of this is uncertain and highly experimental at this point. No file with this data can currently be read properly.
```

```
    property unknown_clod_shorts_1
```

```
    property unknown_clod_shorts_2
```

```
    property unknown_clod_shorts_3
```

```
    property unknown_count_1
```

```
    property unknown_count_2
```

```
    property unknown_count_3
```

```
property unknown_float
property unknown_short
property unknown_shorts

class NiClodSkinInstance (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiSkinInstance

    A copy of NiSkinInstance for use with NiClod meshes.

class NiCollisionData (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiCollisionObject

    Collision box.

    property bounding_volume
    property collision_mode
    property propagation_mode
    property use_abv

class NiCollisionObject (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiObject

    This is the most common collision object found in NIF files. It acts as a real object that is visible and
    possibly (if the body allows for it) interactive. The node itself is simple, it only has three properties. For
    this type of collision object, bhkRigidBody or bhkRigidBodyT is generally used.

    property target

class NiColorData (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiObject

    Color data for material color controller.

    property data

class NiColorExtraData (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiExtraData

    Unknown.

    property data

class NiControllerManager (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiTimeController

    Unknown. Root of all controllers?

    property controller_sequences
    property cumulative
    property num_controller_sequences
    property object_palette

class NiControllerSequence (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif._NiControllerSequence, object

    add_controlled_block ()
        Create new controlled block, and return it.
```

```

>>> seq = NifFormat.NiControllerSequence()
>>> seq.num_controlled_blocks
0
>>> ctrlblock = seq.add_controlled_block()
>>> seq.num_controlled_blocks
1
>>> isinstance(ctrlblock, NifFormat.ControllerLink)
True

```

class NiDataStream (*template=None, argument=None, parent=None*)

Bases: `pyffi.formats.nif.NiObject`

property access

property cloning_behavior

property component_formats

property data

property num_bytes

property num_components

property num_regions

property regions

property streamable

property usage

class NiDefaultAVObjectPalette (*template=None, argument=None, parent=None*)

Bases: `pyffi.formats.nif.NiAVObjectPalette`

Unknown. Refers to a list of objects. Used by `NiControllerManager`.

property num_objs

property objs

property unknown_int

class NiDirectionalLight (*template=None, argument=None, parent=None*)

Bases: `pyffi.formats.nif.NiLight`

Directional light source.

class NiDitherProperty (*template=None, argument=None, parent=None*)

Bases: `pyffi.formats.nif.NiProperty`

Unknown.

property flags

class NiDynamicEffect (*template=None, argument=None, parent=None*)

Bases: `pyffi.formats.nif.NiAVObject`

A dynamic effect such as a light or environment map.

property affected_node_list_pointers

property affected_nodes

property num_affected_node_list_pointers

property num_affected_nodes

```
    property switch_state

class NiEnvMappedTriShape (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiObjectNET
    Unknown
    property child_2
    property child_3
    property children
    property num_children
    property unknown_1
    property unknown_matrix

class NiEnvMappedTriShapeData (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiTriShapeData
    Holds mesh data using a list of singular triangles.

class NiExtraData (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiObject
    A generic extra data object.
    property name
    property next_extra_data

class NiExtraDataController (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiSingleInterpController
    An controller for extra data.

class NiFlipController (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiFloatInterpController
    Texture flipping controller.
    property delta
    property images
    property num_sources
    property sources
    property texture_slot
    property unknown_int_2

class NiFloatData (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiObject
    Possibly the 1D position along a 3D path.
    property data

class NiFloatExtraData (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiExtraData
    Float extra data.
    property float_data
```



```

class NiFloatExtraDataController (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiExtraDataController
    Unknown.
    property controller_data
    property num_extra_bytes
    property unknown_bytes
    property unknown_extra_bytes

class NiFloatInterpController (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiSingleInterpController
    A controller that interpolates floating point numbers?

class NiFloatInterpolator (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiKeyBasedInterpolator
    Unknown.
    property data
    property float_value

class NiFloatsExtraData (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiExtraData
    Unknown.
    property data
    property num_floats

class NiFogProperty (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiProperty
    Describes... fog?
    property flags
    property fog_color
    property fog_depth

class NiFurSpringController (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiTimeController
    property bones
    property bones_2
    property num_bones
    property num_bones_2
    property unknown_float
    property unknown_float_2

class NiGeomMorpherController (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiInterpController
    Time controller for geometry morphing.
    property always_update

```

```

property data
property extra_flags
property interpolator_weights
property interpolators
property num_interpolators
property num_unknown_ints
property unknown_2
property unknown_ints

```

```

class NiGeometry (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif._NiGeometry, object

```

```

>>> from pyffi.formats.nif import NifFormat
>>> id44 = NifFormat.Matrix44()
>>> id44.set_identity()
>>> skelroot = NifFormat.NiNode()
>>> skelroot.name = 'skelroot'
>>> skelroot.set_transform(id44)
>>> bone1 = NifFormat.NiNode()
>>> bone1.name = 'bone1'
>>> bone1.set_transform(id44)
>>> bone2 = NifFormat.NiNode()
>>> bone2.name = 'bone2'
>>> bone2.set_transform(id44)
>>> bone21 = NifFormat.NiNode()
>>> bone21.name = 'bone21'
>>> bone21.set_transform(id44)
>>> bone22 = NifFormat.NiNode()
>>> bone22.name = 'bone22'
>>> bone22.set_transform(id44)
>>> bone211 = NifFormat.NiNode()
>>> bone211.name = 'bone211'
>>> bone211.set_transform(id44)
>>> skelroot.add_child(bone1)
>>> bone1.add_child(bone2)
>>> bone2.add_child(bone21)
>>> bone2.add_child(bone22)
>>> bone21.add_child(bone211)
>>> geom = NifFormat.NiTriShape()
>>> geom.name = 'geom'
>>> geom.set_transform(id44)
>>> geomdata = NifFormat.NiTriShapeData()
>>> skininst = NifFormat.NiSkinInstance()
>>> skindata = NifFormat.NiSkinData()
>>> skelroot.add_child(geom)
>>> geom.data = geomdata
>>> geom.skin_instance = skininst
>>> skininst.skeleton_root = skelroot
>>> skininst.data = skindata
>>> skininst.num_bones = 4
>>> skininst.bones.update_size()
>>> skininst.bones[0] = bone1
>>> skininst.bones[1] = bone2
>>> skininst.bones[2] = bone22

```

(continues on next page)

(continued from previous page)

```

>>> skininst.bones[3] = bone211
>>> skindata.num_bones = 4
>>> skindata.bone_list.update_size()
>>> [child.name for child in skelroot.children]
[b'bone1', b'geom']
>>> skindata.set_transform(id44)
>>> for bonedata in skindata.bone_list:
...     bonedata.set_transform(id44)
>>> affectedbones = geom.flatten_skin()
>>> [bone.name for bone in affectedbones]
[b'bone1', b'bone2', b'bone22', b'bone211']
>>> [child.name for child in skelroot.children]
[b'geom', b'bone1', b'bone21', b'bone2', b'bone22', b'bone211']

```

add_bone (*bone, vert_weights*)

Add bone with given vertex weights. After adding all bones, the geometry skinning information should be set from the current position of the bones using the `L{update_bind_position}` function.

Parameters

- **bone** – The bone NiNode block.
- **vert_weights** – A dictionary mapping each influenced vertex index to a vertex weight.

flatten_skin ()

Reposition all bone blocks and geometry block in the tree to be direct children of the skeleton root.

Returns list of all used bones by the skin.

get_skin_deformation ()

Returns a list of vertices and normals in their final position after skinning, in geometry space.

get_skin_partition ()

Return the skin partition block.

get_vertex_weights ()

Get vertex weights in a convenient format: list bone and weight per vertex.

is_skin ()

Returns True if geometry is skinned.

send_bones_to_bind_position ()

Send all bones to their bind position.

@deprecated: Use `L{NifFormat.NiNode.send_bones_to_bind_position}` instead of this function.

set_skin_partition (*skinpart*)

Set skin partition block.

update_bind_position ()

Make current position of the bones the bind position for this geometry.

Sets the NiSkinData overall transform to the inverse of the geometry transform relative to the skeleton root, and sets the NiSkinData of each bone to the geometry transform relative to the skeleton root times the inverse of the bone transform relative to the skeleton root.

class NiGeometryData (*template=None, argument=None, parent=None*)

Bases: `pyffi.formats.nif._NiGeometryData, object`

```

>>> from pyffi.formats.nif import NifFormat
>>> geomdata = NifFormat.NiGeometryData()

```

(continues on next page)

(continued from previous page)

```

>>> geomdata.num_vertices = 3
>>> geomdata.has_vertices = True
>>> geomdata.has_normals = True
>>> geomdata.has_vertex_colors = True
>>> geomdata.num_uv_sets = 2
>>> geomdata.vertices.update_size()
>>> geomdata.normals.update_size()
>>> geomdata.vertex_colors.update_size()
>>> geomdata.uv_sets.update_size()
>>> geomdata.vertices[0].x = 1
>>> geomdata.vertices[0].y = 2
>>> geomdata.vertices[0].z = 3
>>> geomdata.vertices[1].x = 4
>>> geomdata.vertices[1].y = 5
>>> geomdata.vertices[1].z = 6
>>> geomdata.vertices[2].x = 1.200001
>>> geomdata.vertices[2].y = 3.400001
>>> geomdata.vertices[2].z = 5.600001
>>> geomdata.normals[0].x = 0
>>> geomdata.normals[0].y = 0
>>> geomdata.normals[0].z = 1
>>> geomdata.normals[1].x = 0
>>> geomdata.normals[1].y = 1
>>> geomdata.normals[1].z = 0
>>> geomdata.normals[2].x = 1
>>> geomdata.normals[2].y = 0
>>> geomdata.normals[2].z = 0
>>> geomdata.vertex_colors[1].r = 0.310001
>>> geomdata.vertex_colors[1].g = 0.320001
>>> geomdata.vertex_colors[1].b = 0.330001
>>> geomdata.vertex_colors[1].a = 0.340001
>>> geomdata.uv_sets[0][0].u = 0.990001
>>> geomdata.uv_sets[0][0].v = 0.980001
>>> geomdata.uv_sets[0][2].u = 0.970001
>>> geomdata.uv_sets[0][2].v = 0.960001
>>> geomdata.uv_sets[1][0].v = 0.910001
>>> geomdata.uv_sets[1][0].v = 0.920001
>>> geomdata.uv_sets[1][2].v = 0.930001
>>> geomdata.uv_sets[1][2].v = 0.940001
>>> for h in geomdata.get_vertex_hash_generator():
...     print(h)
(1000, 2000, 3000, 0, 0, 1000, 99000, 98000, 0, 92000, 0, 0, 0, 0)
(4000, 5000, 6000, 0, 1000, 0, 0, 0, 0, 0, 310, 320, 330, 340)
(1200, 3400, 5600, 1000, 0, 0, 97000, 96000, 0, 94000, 0, 0, 0, 0)

```

apply_scale (*scale*)

Apply scale factor on data.

get_vertex_hash_generator (*vertexprecision=3, normalprecision=3, uvprecision=5, vcolprecision=3*)

Generator which produces a tuple of integers for each (vertex, normal, uv, vcol), to ease detection of duplicate vertices. The precision parameters denote number of significant digits behind the comma.

Default for uvprecision should really be high because for very large models the uv coordinates can be very close together.

For vertexprecision, 3 seems usually enough (maybe we'll have to increase this at some point).

Parameters

- **vertexprecision** (*float*) – Precision to be used for vertices.
- **normalprecision** (*float*) – Precision to be used for normals.
- **uvprecision** (*float*) – Precision to be used for uvs.
- **vcolprecision** (*float*) – Precision to be used for vertex colors.

Returns A generator yielding a hash value for each vertex.

update_center_radius ()

Recalculate center and radius of the data.

class NiGravity (*template=None, argument=None, parent=None*)

Bases: `pyffi.formats.nif.NiParticleModifier`

A particle modifier; applies a gravitational field on the particles.

property direction

property force

property position

property type

property unknown_float_1

class NiImage (*template=None, argument=None, parent=None*)

Bases: `pyffi.formats.nif.NiObject`

property file_name

property image_data

property unknown_float

property unknown_int

property use_external

class NiInstancingMeshModifier (*template=None, argument=None, parent=None*)

Bases: `pyffi.formats.nif.NiMeshModifier`

class NiIntegerExtraData (*template=None, argument=None, parent=None*)

Bases: `pyffi.formats.nif.NiExtraData`

Extra integer data.

property integer_data

class NiIntegersExtraData (*template=None, argument=None, parent=None*)

Bases: `pyffi.formats.nif.NiExtraData`

Integers data.

property data

property num_integers

class NiInterpController (*template=None, argument=None, parent=None*)

Bases: `pyffi.formats.nif.NiTimeController`

A controller capable of interpolation?

class NiInterpolator (*template=None, argument=None, parent=None*)

Bases: `pyffi.formats.nif.NiObject`

Interpolator objects - function unknown.

```
class NiKeyBasedInterpolator (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiInterpolator

    Interpolator objects that use keys?

class NiKeyframeController (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiSingleInterpController

    A time controller object for animation key frames.

    property data

class NiKeyframeData (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif._NiKeyframeData, object

    apply_scale (scale)
        Apply scale factor on data.

class NiLODData (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiObject

    Abstract class used for different types of LOD selections.

class NiLODNode (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiSwitchNode

    Level of detail selector. Links to different levels of detail of the same model, used to switch a geometry at
    a specified distance.

    property lod_center

    property lod_level_data

    property lod_levels

    property num_lod_levels

class NiLight (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiDynamicEffect

    Light source.

    property ambient_color

    property diffuse_color

    property dimmer

    property specular_color

class NiLightColorController (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiPoint3InterpController

    Light color animation controller.

class NiLightDimmerController (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiFloatInterpController

    Unknown controller.

class NiLightIntensityController (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiFloatInterpController

    Unknown controller
```

```

class NiLines (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiTriBasedGeom

    Wireframe geometry.

class NiLinesData (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiGeometryData

    Wireframe geometry data.

    property lines

class NiLookAtController (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiTimeController

    Unknown. Start time is 3.4e+38 and stop time is -3.4e+38.

    property look_at_node
    property unknown_1

class NiLookAtInterpolator (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiInterpolator

    Unknown.

    property look_at
    property rotation
    property scale
    property target
    property translation
    property unknown_link_1
    property unknown_link_2
    property unknown_link_3
    property unknown_short

class NiMaterialColorController (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif._NiMaterialColorController, object

    get_target_color()
        Get target color (works for all nif versions).

    set_target_color(target_color)
        Set target color (works for all nif versions).

class NiMaterialProperty (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif._NiMaterialProperty, object

    is_interchangeable(other)
        Are the two material blocks interchangeable?

class NiMesh (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiRenderObject

    property bound
    property datas
    property instancing_enabled

```

```
property modifiers
property num_datas
property num_modifiers
property num_submeshes
property primitive_type
property unknown_100
property unknown_101
property unknown_102
property unknown_103
property unknown_200
property unknown_201
property unknown_250
property unknown_251
property unknown_300
property unknown_301
property unknown_302
property unknown_303
property unknown_350
property unknown_351
property unknown_400
property unknown_51
property unknown_52
property unknown_53
property unknown_54
property unknown_55
property unknown_56

class NiMeshHWInstance (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiObject

class NiMeshModifier (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiObject
    Base class for mesh modifiers.

    property complete_points
    property num_complete_points
    property num_submit_points
    property submit_points
```



```

class NiMeshPSysData (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiPSysData
    Particle meshes data.

    property num_unknown_ints_1
    property unknown_byte_3
    property unknown_int_2
    property unknown_ints_1
    property unknown_node

class NiMeshParticleSystem (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiParticleSystem
    Particle system.

class NiMorphController (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiInterpController
    Unknown! Used by Daoc->'healing.nif'.

class NiMorphData (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif._NiMorphData, object
    apply_scale (scale)
        Apply scale factor on data.

class NiMorphMeshModifier (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiMeshModifier
    Performs linear-weighted blending between a set of target data streams.

    property elements
    property flags
    property num_elements
    property num_targets

class NiMorphWeightsController (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiInterpController
    property interpolators
    property num_interpolators
    property num_targets
    property target_names
    property unknown_2

class NiMorpherController (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiInterpController
    Unknown! Used by Daoc.

    property data

class NiMultiTargetTransformController (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiInterpController
    Unknown.

```

```

    property extra_targets
    property num_extra_targets

class NiMultiTextureProperty (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiProperty

    (note: not quite complete yet... but already reads most of the DAoC ones)

    property flags
    property texture_elements
    property unknown_int

class NiNode (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif._NiNode, object

```

```

>>> from pyffi.formats.nif import NifFormat
>>> x = NifFormat.NiNode()
>>> y = NifFormat.NiNode()
>>> z = NifFormat.NiNode()
>>> x.num_children = 1
>>> x.children.update_size()
>>> y in x.children
False
>>> x.children[0] = y
>>> y in x.children
True
>>> x.add_child(z, front = True)
>>> x.add_child(y)
>>> x.num_children
2
>>> x.children[0] is z
True
>>> x.remove_child(y)
>>> y in x.children
False
>>> x.num_children
1
>>> e = NifFormat.NiSpotLight()
>>> x.add_effect(e)
>>> x.num_effects
1
>>> e in x.effects
True

```

```

>>> from pyffi.formats.nif import NifFormat
>>> node = NifFormat.NiNode()
>>> child1 = NifFormat.NiNode()
>>> child1.name = "hello"
>>> child_2 = NifFormat.NiNode()
>>> child_2.name = "world"
>>> node.get_children()
[]
>>> node.set_children([child1, child_2])
>>> [child.name for child in node.get_children()]
[b'hello', b'world']
>>> [child.name for child in node.children]
[b'hello', b'world']

```

(continues on next page)

(continued from previous page)

```

>>> node.set_children([])
>>> node.get_children()
[]
>>> # now set them the other way around
>>> node.set_children([child_2, child1])
>>> [child.name for child in node.get_children()]
[b'world', b'hello']
>>> [child.name for child in node.children]
[b'world', b'hello']
>>> node.remove_child(child_2)
>>> [child.name for child in node.children]
[b'hello']
>>> node.add_child(child_2)
>>> [child.name for child in node.children]
[b'hello', b'world']

```

```

>>> from pyffi.formats.nif import NifFormat
>>> node = NifFormat.NiNode()
>>> effect1 = NifFormat.NiSpotLight()
>>> effect1.name = "hello"
>>> effect2 = NifFormat.NiSpotLight()
>>> effect2.name = "world"
>>> node.get_effects()
[]
>>> node.set_effects([effect1, effect2])
>>> [effect.name for effect in node.get_effects()]
[b'hello', b'world']
>>> [effect.name for effect in node.effects]
[b'hello', b'world']
>>> node.set_effects([])
>>> node.get_effects()
[]
>>> # now set them the other way around
>>> node.set_effects([effect2, effect1])
>>> [effect.name for effect in node.get_effects()]
[b'world', b'hello']
>>> [effect.name for effect in node.effects]
[b'world', b'hello']
>>> node.remove_effect(effect2)
>>> [effect.name for effect in node.effects]
[b'hello']
>>> node.add_effect(effect2)
>>> [effect.name for effect in node.effects]
[b'hello', b'world']

```

add_child(*child*, *front=False*)

Add block to child list.

Parameters **child** (*L*{*NifFormat.NiAVObject*}) – The child to add.

Keyword Arguments **front** – Whether to add to the front or to the end of the list (default is at end).

add_effect(*effect*)

Add an effect to the list of effects.

Parameters **effect** (*L*{*NifFormat.NiDynamicEffect*}) – The effect to add.

get_children()

Return a list of the children of the block.

Returns The list of children.

Return type list of L{NifFormat.NiAVObject}

get_effects()

Return a list of the effects of the block.

Returns The list of effects.

Return type list of L{NifFormat.NiDynamicEffect}

get_skinned_geometries()

This function yields all skinned geometries which have self as skeleton root.

merge_external_skeleton_root(skelroot)

Attach skinned geometry to self (which will be the new skeleton root of the nif at the given skeleton root). Use this function if you move a skinned geometry from one nif into a new NIF file. The bone links will be updated to point to the tree at self, instead of to the external tree.

merge_skeleton_roots()

This function will look for other geometries whose skeleton root is a (possibly indirect) child of this node. It will then reparent those geometries to this node. For example, it will unify the skeleton roots in Morrowind's cliffcracer.nif file, or of the (official) body skins. This makes it much easier to import skeletons in for instance Blender: there will be only one skeleton root for each bone, over all geometries.

The merge fails for those geometries whose global skin data transform does not match the inverse geometry transform relative to the skeleton root (the maths does not work out in this case!)

Returns list of all new blocks that have been reparented (and added to the skeleton root children list), and a list of blocks for which the merge failed.

remove_child(child)

Remove a block from the child list.

Parameters **child** (L{NifFormat.NiAVObject}) – The child to remove.

remove_effect(effect)

Remove a block from the effect list.

Parameters **effect** (L{NifFormat.NiDynamicEffect}) – The effect to remove.

send_bones_to_bind_position()

This function will send all bones of geometries of this skeleton root to their bind position. For best results, call L{send_geometries_to_bind_position} first.

Returns A number quantifying the remaining difference between bind positions.

Return type float

send_detached_geometries_to_node_position()

Some nifs (in particular in Morrowind) have geometries that are skinned but that do not share bones. In such cases, send_geometries_to_bind_position cannot reposition them. This function will send such geometries to the position of their root node.

Examples of such nifs are the official Morrowind skins (after merging skeleton roots).

Returns list of detached geometries that have been moved.

send_geometries_to_bind_position()

Call this on the skeleton root of geometries. This function will transform the geometries, such that all skin data transforms coincide, or at least coincide partially.

Returns A number quantifying the remaining difference between bind positions.

Return type float

set_children(childlist)

Set the list of children from the given list (destroys existing list).

Parameters **childlist** (*list* of L{NifFormat.NiAVObject}) – The list of child blocks to set.

set_effects (*effectlist*)
Set the list of effects from the given list (destroys existing list).

Parameters **effectlist** (*list* of L{NifFormat.NiDynamicEffect}) – The list of effect blocks to set.

class NiObject (*template=None, argument=None, parent=None*)
Bases: `pyffi.formats.nif._NiObject`, `object`

apply_scale (*scale*)
Scale data in this block. This implementation does nothing. Override this method if it contains geometry data that can be scaled.

find (*block_name=None, block_type=None*)

find_chain (*block, block_type=None*)
Finds a chain of blocks going from C{self} to C{block}. If found, self is the first element and block is the last element. If no branch found, returns an empty list. Does not check whether there is more than one branch; if so, the first one found is returned.

Parameters

- **block** – The block to find a chain to.
- **block_type** – The type that blocks should have in this chain.

is_interchangeable (*other*)
Are the two blocks interchangeable?

@todo: Rely on AnyType, SimpleType, ComplexType, etc. implementation.

tree (*block_type=None, follow_all=True, unique=False*)
A generator for parsing all blocks in the tree (starting from and including C{self}).

Parameters

- **block_type** – If not `None`, yield only blocks of the type C{block_type}.
- **follow_all** – If C{block_type} is not `None`, then if this is `True` the function will parse the whole tree. Otherwise, the function will not follow branches that start by a non-C{block_type} block.
- **unique** – Whether the generator can return the same block twice or not.

class NiObjectNET (*template=None, argument=None, parent=None*)
Bases: `pyffi.formats.nif._NiObjectNET`, `object`

add_controller (*ctrlblock*)
Add block to controller chain and set target of controller to self.

add_extra_data (*extrablock*)
Add block to extra data list and extra data chain. It is good practice to ensure that the extra data has empty next_extra_data field when adding it to avoid loops in the hierarchy.

add_integer_extra_data (*name, value*)
Add a particular extra integer data block.

get_controllers ()
Get a list of all controllers.

get_extra_datas ()
Get a list of all extra data blocks.

remove_extra_data (*extrablock*)
Remove block from extra data list and extra data chain.

```

>>> from pyffi.formats.nif import NifFormat
>>> block = NifFormat.NiNode()
>>> block.num_extra_data_list = 3
>>> block.extra_data_list.update_size()
>>> extrablock = NifFormat.NiStringExtraData()
>>> block.extra_data_list[1] = extrablock
>>> block.remove_extra_data(extrablock)
>>> [extra for extra in block.extra_data_list]
[None, None]

```

set_extra_datas (*extralist*)

Set all extra data blocks from given list (erases existing data).

```

>>> from pyffi.formats.nif import NifFormat
>>> node = NifFormat.NiNode()
>>> extral = NifFormat.NiExtraData()
>>> extral.name = "hello"
>>> extra2 = NifFormat.NiExtraData()
>>> extra2.name = "world"
>>> node.get_extra_datas()
[]
>>> node.set_extra_datas([extral, extra2])
>>> [extra.name for extra in node.get_extra_datas()]
[b'hello', b'world']
>>> [extra.name for extra in node.extra_data_list]
[b'hello', b'world']
>>> node.extra_data is extral
True
>>> extral.next_extra_data is extra2
True
>>> extra2.next_extra_data is None
True
>>> node.set_extra_datas([])
>>> node.get_extra_datas()
[]
>>> # now set them the other way around
>>> node.set_extra_datas([extra2, extral])
>>> [extra.name for extra in node.get_extra_datas()]
[b'world', b'hello']
>>> [extra.name for extra in node.extra_data_list]
[b'world', b'hello']
>>> node.extra_data is extra2
True
>>> extra2.next_extra_data is extral
True
>>> extral.next_extra_data is None
True

```

Parameters **extralist** (list of L{NifFormat.NiExtraData}) – List of extra data blocks to add.

class NiPSBombForce (*template=None, argument=None, parent=None*)

Bases: pyffi.formats.nif.NiObject

property name

property unknown_1

property unknown_10

```

    property unknown_2
    property unknown_3
    property unknown_4
    property unknown_5
    property unknown_6
    property unknown_7
    property unknown_8
    property unknown_9

class NiPSBoundUpdater (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiObject
    property unknown_1
    property unknown_2

class NiPSBoxEmitter (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiObject
    property name
    property unknown_1
    property unknown_10
    property unknown_11
    property unknown_12
    property unknown_13
    property unknown_14
    property unknown_15
    property unknown_16
    property unknown_17
    property unknown_18
    property unknown_19
    property unknown_2
    property unknown_20
    property unknown_21
    property unknown_22
    property unknown_23
    property unknown_24
    property unknown_25
    property unknown_26
    property unknown_27
    property unknown_28
    property unknown_29

```

```
property unknown_3
property unknown_30
property unknown_31
property unknown_32
property unknown_33
property unknown_34
property unknown_35
property unknown_36
property unknown_37
property unknown_38
property unknown_39
property unknown_4
property unknown_40
property unknown_41
property unknown_42
property unknown_43
property unknown_44
property unknown_45
property unknown_46
property unknown_47
property unknown_48
property unknown_5
property unknown_6
property unknown_7
property unknown_8
property unknown_9

class NiPSCylinderEmitter (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiPSSphereEmitter
    property unknown_23

class NiPSDragForce (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiObject
    property unknown_1
    property unknown_10
    property unknown_2
    property unknown_3
    property unknown_4
    property unknown_5
```



```

    property unknown_6
    property unknown_7
    property unknown_8
    property unknown_9
class NiPSEmitParticlesCtrl (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiPSysEmitterCtrlr
class NiPSEmitterDeclinationCtrlr (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiPSysModifierCtrlr
class NiPSEmitterDeclinationVarCtrlr (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiPSEmitterDeclinationCtrlr
class NiPSEmitterLifeSpanCtrlr (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiPSysModifierCtrlr
class NiPSEmitterPlanarAngleCtrlr (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiPSysModifierCtrlr
class NiPSEmitterPlanarAngleVarCtrlr (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiPSEmitterPlanarAngleCtrlr
class NiPSEmitterRadiusCtrlr (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiTimeController
    property interpolator
    property unknown_2
class NiPSEmitterRotAngleCtrlr (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiPSysModifierCtrlr
class NiPSEmitterRotAngleVarCtrlr (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiPSEmitterRotAngleCtrlr
class NiPSEmitterRotSpeedCtrlr (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiPSysModifierCtrlr
class NiPSEmitterRotSpeedVarCtrlr (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiPSEmitterRotSpeedCtrlr
class NiPSEmitterSpeedCtrlr (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiTimeController
    property interpolator
    property unknown_3
class NiPSFacingQuadGenerator (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiObject
    property unknown_1
    property unknown_10
    property unknown_11
    property unknown_12
    property unknown_2
    property unknown_3

```

```
property unknown_4
property unknown_5
property unknown_6
property unknown_7
property unknown_8
property unknown_9
class NiPSForceActiveCtrl (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiTimeController
    property interpolator
    property unknown_2
class NiPSGravityForce (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiObject
    property unknown_1
    property unknown_10
    property unknown_11
    property unknown_12
    property unknown_13
    property unknown_14
    property unknown_15
    property unknown_16
    property unknown_17
    property unknown_18
    property unknown_19
    property unknown_2
    property unknown_20
    property unknown_21
    property unknown_22
    property unknown_23
    property unknown_24
    property unknown_25
    property unknown_26
    property unknown_27
    property unknown_28
    property unknown_29
    property unknown_3
    property unknown_30
    property unknown_31
```

```

    property unknown_32
    property unknown_33
    property unknown_34
    property unknown_35
    property unknown_36
    property unknown_4
    property unknown_5
    property unknown_6
    property unknown_7
    property unknown_8
    property unknown_9

class NiPSGravityStrengthCtrl (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiTimeController
    property unknown_2
    property unknown_3

class NiPSMeshEmitter (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiObject
    property name
    property unknown_1
    property unknown_10
    property unknown_11
    property unknown_12
    property unknown_13
    property unknown_14
    property unknown_15
    property unknown_16
    property unknown_17
    property unknown_18
    property unknown_19
    property unknown_2
    property unknown_20
    property unknown_21
    property unknown_22
    property unknown_23
    property unknown_24
    property unknown_25
    property unknown_26

```

```
property unknown_27
property unknown_28
property unknown_3
property unknown_4
property unknown_5
property unknown_6
property unknown_7
property unknown_8
property unknown_9

class NiPSMeshParticleSystem (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiPSParticleSystem
    property unknown_23
    property unknown_24
    property unknown_25
    property unknown_26

class NiPSParticleSystem (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiAVObject
    property emitter
    property generator
    property simulator
    property unknown_10
    property unknown_11
    property unknown_12
    property unknown_15
    property unknown_16
    property unknown_17
    property unknown_19
    property unknown_20
    property unknown_21
    property unknown_22
    property unknown_27
    property unknown_28
    property unknown_29
    property unknown_3
    property unknown_30
    property unknown_31
    property unknown_32
```

```

    property unknown_33
    property unknown_34
    property unknown_35
    property unknown_36
    property unknown_37
    property unknown_38
    property unknown_39
    property unknown_4
    property unknown_5
    property unknown_6
    property unknown_7
    property unknown_8
    property unknown_9

class NiPSPlanarCollider (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiObject
    property name
    property unknown_byte_4
    property unknown_floats_5
    property unknown_int_1
    property unknown_int_2
    property unknown_link_6
    property unknown_short_3

class NiPSResetOnLoopCtrlr (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiTimeController

class NiPSSimulator (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiMeshModifier
    The mesh modifier that performs all particle system simulation.
    property num_simulation_steps
    property simulation_steps

class NiPSSimulatorCollidersStep (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiPSSimulatorStep
    Encapsulates a floodgate kernel that simulates particle colliders.
    property colliders
    property num_colliders

class NiPSSimulatorFinalStep (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiPSSimulatorStep
    Encapsulates a floodgate kernel that updates particle positions and ages. As indicated by its name, this step
    should be attached last in the NiPSSimulator mesh modifier.

```

```
class NiPSSimulatorForcesStep (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiPSSimulatorStep
    Encapsulates a floodgate kernel that simulates particle forces.

    property forces
    property num_forces

class NiPSSimulatorGeneralStep (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiPSSimulatorStep
    Encapsulates a floodgate kernel that updates particle size, colors, and rotations.

    property color_keys
    property color_loop_behavior
    property grow_generation
    property grow_time
    property num_color_keys
    property num_rotation_keys
    property num_size_keys
    property rotation_keys
    property rotation_loop_behavior
    property shrink_generation
    property shrink_time
    property size_keys
    property size_loop_behavior
    property unknown_1
    property unknown_2
    property unknown_3

class NiPSSimulatorMeshAlignStep (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiPSSimulatorStep
    Encapsulates a floodgate kernel that updates mesh particle alignment and transforms.

    property num_rotation_keys
    property rotation_keys
    property rotation_loop_behavior

class NiPSSimulatorStep (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiObject
    Abstract base class for a single step in the particle system simulation process. It has no seralized data.

class NiPSSpawner (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiObject

class NiPSSphereEmitter (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiObject

    property name
```

```

property unknown_10
property unknown_11
property unknown_12
property unknown_13
property unknown_14
property unknown_15
property unknown_16
property unknown_17
property unknown_18
property unknown_19
property unknown_2
property unknown_20
property unknown_21
property unknown_22
property unknown_3
property unknown_4
property unknown_5
property unknown_6
property unknown_7
property unknown_8
property unknown_9

class NiPSSphericalCollider (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiObject
    property unknown_1
    property unknown_2
    property unknown_3
    property unknown_4
    property unknown_5
    property unknown_6
    property unknown_7

class NiPSysAgeDeathModifier (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiPSysModifier
    Unknown particle modifier.
    property spawn_modifier
    property spawn_on_death

```

```
class NiPSysAirFieldAirFrictionCtrl (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiPSysModifierFloatCtrl

    Particle system controller for air field air friction.

class NiPSysAirFieldInheritVelocityCtrl (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiPSysModifierFloatCtrl

    Particle system controller for air field inherit velocity.

class NiPSysAirFieldModifier (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiPSysFieldModifier

    Particle system modifier, used for controlling the particle velocity in a field like wind.

    property direction
    property unknown_boolean_1
    property unknown_boolean_2
    property unknown_boolean_3
    property unknown_float_2
    property unknown_float_3
    property unknown_float_4

class NiPSysAirFieldSpreadCtrl (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiPSysModifierFloatCtrl

    Particle system controller for air field spread.

class NiPSysBombModifier (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiPSysModifier

    Particle modifier that uses a NiNode to use as a “Bomb Object” to alter the path of particles.

    property bomb_axis
    property bomb_object
    property decay
    property decay_type
    property delta_v
    property symmetry_type

class NiPSysBoundUpdateModifier (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiPSysModifier

    Unknown particle system modifier.

    property update_skip

class NiPSysBoxEmitter (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiPSysVolumeEmitter

    Particle emitter that uses points within a defined Box shape to emit from..

    property depth
    property height
    property width
```



```

class NiPSysCollider (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiObject
    Particle system collider.

    property bounce
    property collider_object
    property die_on_collide
    property next_collider
    property parent
    property spawn_modifier
    property spawn_on_collide

class NiPSysColliderManager (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiPSysModifier
    Particle modifier that adds a defined shape to act as a collision object for particles to interact with.

    property collider

class NiPSysColorModifier (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiPSysModifier
    Particle modifier that adds keyframe data to modify color/alpha values of particles over time.

    property data

class NiPSysCylinderEmitter (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiPSysVolumeEmitter
    Particle emitter that uses points within a defined Cylinder shape to emit from.

    property height
    property radius

class NiPSysData (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiRotatingParticlesData
    Particle system data.

    property aspect_ratio
    property has_subtexture_offset_u_vs
    property has_unknown_floats_3
    property num_subtexture_offset_u_vs
    property particle_descriptions
    property subtexture_offset_u_vs
    property unknown_byte_4
    property unknown_floats_3
    property unknown_int_4
    property unknown_int_5
    property unknown_int_6
    property unknown_short_1

```

```
    property unknown_short_2
    property unknown_short_3
class NiPSysDragFieldModifier (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiPSysFieldModifier
    Particle system modifier, used for controlling the particle velocity in drag space warp.
    property direction
    property use_direction
class NiPSysDragModifier (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiPSysModifier
    Unknown.
    property drag_axis
    property parent
    property percentage
    property range
    property range_falloff
class NiPSysEmitter (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiPSysModifier
    A particle emitter?
    property declination
    property declination_variation
    property initial_color
    property initial_radius
    property life_span
    property life_span_variation
    property planar_angle
    property planar_angle_variation
    property radius_variation
    property speed
    property speed_variation
class NiPSysEmitterCtrlr (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiPSysModifierCtrlr
    Particle system emitter controller.
    property data
    property visibility_interpolator
class NiPSysEmitterCtrlrData (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiObject
    Particle system emitter controller data.
    property float_keys
```

```

    property num_visibility_keys
    property visibility_keys
class NiPSysEmitterDeclinationCtrl (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiPSysModifierFloatCtrl
    Unknown.
class NiPSysEmitterDeclinationVarCtrl (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiPSysModifierFloatCtrl
    Unknown.
class NiPSysEmitterInitialRadiusCtrl (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiPSysModifierFloatCtrl
    Unknown.
class NiPSysEmitterLifeSpanCtrl (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiPSysModifierFloatCtrl
    Unknown.
class NiPSysEmitterPlanarAngleCtrl (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiPSysModifierFloatCtrl
    Particle system controller for emitter planar angle.
class NiPSysEmitterPlanarAngleVarCtrl (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiPSysModifierFloatCtrl
    Particle system controller for emitter planar angle variation.
class NiPSysEmitterSpeedCtrl (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiPSysModifierFloatCtrl
    Unknown.
class NiPSysFieldAttenuationCtrl (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiPSysModifierFloatCtrl
    Particle system controller for force field attenuation.
class NiPSysFieldMagnitudeCtrl (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiPSysModifierFloatCtrl
    Particle system controller for force field magnitude.
class NiPSysFieldMaxDistanceCtrl (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiPSysModifierFloatCtrl
    Particle system controller for force field maximum distance.
class NiPSysFieldModifier (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiPSysModifier
    Base for all force field particle modifiers.
    property attenuation
    property field_object
    property magnitude
    property max_distance

```

```
    property use_max_distance
```

class NiPSysGravityFieldModifier (*template=None, argument=None, parent=None*)
Bases: `pyffi.formats.nif.NiPSysFieldModifier`

Particle system modifier, used for controlling the particle velocity in gravity field.

```
    property direction
```

class NiPSysGravityModifier (*template=None, argument=None, parent=None*)
Bases: `pyffi.formats.nif.NiPSysModifier`

Adds gravity to a particle system, when linked to a NiNode to use as a Gravity Object.

```
    property decay
    property force_type
    property gravity_axis
    property gravity_object
    property strength
    property turbulence
    property turbulence_scale
    property unknown_byte
```

class NiPSysGravityStrengthCtrlr (*template=None, argument=None, parent=None*)
Bases: `pyffi.formats.nif.NiPSysModifierFloatCtrlr`

Unknown.

class NiPSysGrowFadeModifier (*template=None, argument=None, parent=None*)
Bases: `pyffi.formats.nif.NiPSysModifier`

Particle modifier that controls the time it takes to grow a particle from Size=0 to the specified Size in the emitter, and then back to 0. This modifier has no control over alpha settings.

```
    property base_scale
    property fade_generation
    property fade_time
    property grow_generation
    property grow_time
```

class NiPSysInitialRotAngleCtrlr (*template=None, argument=None, parent=None*)
Bases: `pyffi.formats.nif.NiPSysModifierFloatCtrlr`

Particle system controller for emitter initial rotation angle.

class NiPSysInitialRotAngleVarCtrlr (*template=None, argument=None, parent=None*)
Bases: `pyffi.formats.nif.NiPSysModifierFloatCtrlr`

Particle system controller for emitter initial rotation angle variation.

class NiPSysInitialRotSpeedCtrlr (*template=None, argument=None, parent=None*)
Bases: `pyffi.formats.nif.NiPSysModifierFloatCtrlr`

Particle system controller for emitter initial rotation speed.

```

class NiPSysInitialRotSpeedVarCtrl (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiPSysModifierFloatCtrl

    Particle system controller for emitter initial rotation speed variation.

class NiPSysMeshEmitter (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiPSysEmitter

    Particle emitter that uses points on a specified mesh to emit from.

    property emission_axis
    property emission_type
    property emitter_meshes
    property initial_velocity_type
    property num_emitter_meshes

class NiPSysMeshUpdateModifier (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiPSysModifier

    Unknown.

    property meshes
    property num_meshes

class NiPSysModifier (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiObject

    Generic particle system modifier object.

    property active
    property name
    property order
    property target

class NiPSysModifierActiveCtrl (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiPSysModifierBoolCtrl

    Unknown.

    property data

class NiPSysModifierBoolCtrl (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiPSysModifierCtrl

    A particle system modifier controller that deals with boolean data?

class NiPSysModifierCtrl (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiSingleInterpController

    A particle system modifier controller.

    property modifier_name

class NiPSysModifierFloatCtrl (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiPSysModifierCtrl

    A particle system modifier controller that deals with floating point data?

    property data

```

```
class NiPSysPlanarCollider (template=None, argument=None, parent=None)  
    Bases: pyffi.formats.nif.NiPSysCollider  
    Particle Collider object which particles will interact with.  
  
    property height  
    property width  
    property x_axis  
    property y_axis  
  
class NiPSysPositionModifier (template=None, argument=None, parent=None)  
    Bases: pyffi.formats.nif.NiPSysModifier  
    Unknown particle system modifier.  
  
class NiPSysRadialFieldModifier (template=None, argument=None, parent=None)  
    Bases: pyffi.formats.nif.NiPSysFieldModifier  
    Particle system modifier, used for controlling the particle velocity in force field.  
  
    property radial_type  
  
class NiPSysResetOnLoopCtrlr (template=None, argument=None, parent=None)  
    Bases: pyffi.formats.nif.NiTimeController  
    Unknown.  
  
class NiPSysRotationModifier (template=None, argument=None, parent=None)  
    Bases: pyffi.formats.nif.NiPSysModifier  
    Particle modifier that adds rotations to particles.  
  
    property initial_axis  
    property initial_rotation_angle  
    property initial_rotation_angle_variation  
    property initial_rotation_speed  
    property initial_rotation_speed_variation  
    property random_initial_axis  
    property random_rot_speed_sign  
  
class NiPSysSpawnModifier (template=None, argument=None, parent=None)  
    Bases: pyffi.formats.nif.NiPSysModifier  
    Unknown particle modifier.  
  
    property life_span  
    property life_span_variation  
    property max_num_to_spawn  
    property min_num_to_spawn  
    property num_spawn_generations  
    property percentage_spawned  
    property spawn_dir_chaos  
    property spawn_speed_chaos
```

```

    property unknown_int
class NiPSysSphereEmitter (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiPSysVolumeEmitter

    Particle emitter that uses points within a sphere shape to emit from.

    property radius
class NiPSysSphericalCollider (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiPSysCollider

    Particle Collider object which particles will interact with.

    property radius
class NiPSysTrailEmitter (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiPSysEmitter

    Guild 2-Specific node

    property unknown_float_1
    property unknown_float_2
    property unknown_float_3
    property unknown_float_4
    property unknown_float_5
    property unknown_float_6
    property unknown_float_7
    property unknown_int_1
    property unknown_int_2
    property unknown_int_3
    property unknown_int_4
class NiPSysTurbulenceFieldModifier (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiPSysFieldModifier

    Particle system modifier, used for controlling the particle velocity in drag space warp.

    property frequency
class NiPSysUpdateCtrlr (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiTimeController

    Particle system controller, used for ???.

class NiPSysVolumeEmitter (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiPSysEmitter

    An emitter that emits meshes?

    property emitter_object
class NiPSysVortexFieldModifier (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiPSysFieldModifier

    Particle system modifier, used for controlling the particle velocity in force field.

    property direction

```

```
class NiPalette (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiObject
    A color palette.

    property num_entries
    property palette
    property unknown_byte

class NiParticleBomb (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiParticleModifier
    A particle modifier.

    property decay
    property decay_type
    property delta_v
    property direction
    property duration
    property position
    property start
    property symmetry_type

class NiParticleColorModifier (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiParticleModifier
    Unknown.

    property color_data

class NiParticleGrowFade (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiParticleModifier

    This particle system modifier controls the particle size. If it is present the particles start with size 0.0 .
    Then they grow to their original size and stay there until they fade to zero size again at the end of their
    lifetime cycle.

    property fade
    property grow

class NiParticleMeshModifier (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiParticleModifier
    Unknown.

    property num_particle_meshes
    property particle_meshes

class NiParticleMeshes (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiParticles
    Mesh particle node?

class NiParticleMeshesData (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiRotatingParticlesData
    Particle meshes data.
```



```

    property unknown_link_2
class NiParticleModifier (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiObject
    A particle system modifier.
    property controller
    property next_modifier
class NiParticleRotation (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiParticleModifier
    Unknown.
    property initial_axis
    property random_initial_axis
    property rotation_speed
class NiParticleSystem (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiParticles
    A particle system.
    property modifiers
    property num_modifiers
    property unknown_int_1
    property unknown_short_2
    property unknown_short_3
    property world_space
class NiParticleSystemController (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiTimeController
    A generic particle system time controller object.
    property color_data
    property emit_flags
    property emit_rate
    property emit_start_time
    property emit_stop_time
    property emitter
    property horizontal_angle
    property horizontal_direction
    property lifetime
    property lifetime_random
    property num_particles
    property num_valid
    property old_emit_rate

```

```
property old_speed
property particle_extra
property particle_lifetime
property particle_link
property particle_timestamp
property particle_unknown_short
property particle_unknown_vector
property particle_velocity
property particle_vertex_id
property particles
property size
property speed
property speed_random
property start_random
property trailer
property unknown_byte
property unknown_color
property unknown_float_1
property unknown_float_13
property unknown_floats_2
property unknown_int_1
property unknown_int_2
property unknown_link
property unknown_link_2
property unknown_normal
property unknown_short_2
property unknown_short_3
property vertical_angle
property vertical_direction

class NiParticles (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiGeometry
    Generic particle system node.

class NiParticlesData (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiGeometryData
    Generic rotating particles data object.

    property has_radii
    property has_rotation_angles
```

```

    property has_rotation_axes
    property has_rotations
    property has_sizes
    property has_uv_quadrants
    property num_active
    property num_particles
    property num_uv_quadrants
    property particle_radius
    property radii
    property rotation_angles
    property rotation_axes
    property rotations
    property sizes
    property unknown_byte_1
    property unknown_byte_2
    property unknown_link
    property uv_quadrants

class NiPathController (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiTimeController
    Time controller for a path.
    property float_data
    property pos_data
    property unknown_float_2
    property unknown_float_3
    property unknown_int_1
    property unknown_short
    property unknown_short_2

class NiPathInterpolator (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiKeyBasedInterpolator
    Unknown interpolator.
    property float_data
    property pos_data
    property unknown_float_1
    property unknown_float_2
    property unknown_int
    property unknown_short
    property unknown_short_2

```

```
class NiPersistentSrcTextureRendererData (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.ATextureRenderData
    property num_faces
    property num_pixels
    property pixel_data
    property unknown_int_6
    property unknown_int_7

class NiPhysXActorDesc (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiObject
    Unknown PhysX node.
    property shape_description
    property unknown_byte_1
    property unknown_byte_2
    property unknown_int_1
    property unknown_int_2
    property unknown_int_4
    property unknown_int_5
    property unknown_int_6
    property unknown_quat_1
    property unknown_quat_2
    property unknown_quat_3
    property unknown_ref_0
    property unknown_ref_1
    property unknown_ref_2
    property unknown_refs_3

class NiPhysXBodyDesc (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiObject
    Unknown PhysX node.
    property unknown_bytes

class NiPhysXD6JointDesc (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiObject
    Unknown PhysX node.
    property unknown_bytes

class NiPhysXKinematicSrc (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiObject
    Unknown PhysX node.
    property unknown_bytes
```

```
class NiPhysXMaterialDesc (template=None, argument=None, parent=None)
```

```
    Bases: pyffi.formats.nif.NiObject
```

```
    Unknown node.
```

```
    property unknown_byte_1
```

```
    property unknown_byte_2
```

```
    property unknown_int
```

```
class NiPhysXMeshDesc (template=None, argument=None, parent=None)
```

```
    Bases: pyffi.formats.nif.NiObject
```

```
    Unknown PhysX node.
```

```
    property num_vertices
```

```
    property unknown_byte_1
```

```
    property unknown_byte_2
```

```
    property unknown_bytes_0
```

```
    property unknown_bytes_1
```

```
    property unknown_bytes_2
```

```
    property unknown_bytes_3
```

```
    property unknown_float_1
```

```
    property unknown_float_2
```

```
    property unknown_int_1
```

```
    property unknown_int_2
```

```
    property unknown_int_4
```

```
    property unknown_ints_1
```

```
    property unknown_short_1
```

```
    property unknown_short_2
```

```
    property unknown_shorts_1
```

```
    property vertices
```

```
class NiPhysXProp (template=None, argument=None, parent=None)
```

```
    Bases: pyffi.formats.nif.NiObjectNET
```

```
    Unknown PhysX node.
```

```
    property num_dests
```

```
    property prop_description
```

```
    property transform_dests
```

```
    property unknown_byte
```

```
    property unknown_float_1
```

```
    property unknown_int
```

```
    property unknown_int_1
```

```
    property unknown_refs_1
```

```
class NiPhysXPropDesc (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiObject
    Unknown PhysX node.
    property actor_descs
    property joint_descs
    property material_descs
    property num_dests
    property num_joints
    property num_materials
    property unknown_byte_6
    property unknown_int_1
    property unknown_int_2
    property unknown_int_3
    property unknown_int_5
    property unknown_string_4

class NiPhysXShapeDesc (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiObject
    Unknown PhysX node.
    property mesh_description
    property unknown_float_1
    property unknown_float_2
    property unknown_float_3
    property unknown_int_1
    property unknown_int_2
    property unknown_int_3
    property unknown_int_4
    property unknown_int_5
    property unknown_int_7
    property unknown_int_8
    property unknown_quat_1
    property unknown_quat_2
    property unknown_quat_3
    property unknown_short_1
    property unknown_short_2

class NiPhysXTransformDest (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiObject
    Unknown PhysX node.
```

```

    property node
    property unknown_byte_1
    property unknown_byte_2
class NiPixelData (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.ATextureRenderData
    A texture.
    property num_faces
    property num_pixels
    property pixel_data
class NiPlanarCollider (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiParticleModifier
    Unknown.
    property unknown_float_1
    property unknown_float_10
    property unknown_float_11
    property unknown_float_12
    property unknown_float_13
    property unknown_float_14
    property unknown_float_15
    property unknown_float_16
    property unknown_float_2
    property unknown_float_3
    property unknown_float_4
    property unknown_float_5
    property unknown_float_6
    property unknown_float_7
    property unknown_float_8
    property unknown_float_9
    property unknown_short
    property unknown_short_2
class NiPoint3InterpController (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiSingleInterpController
    A controller that interpolates point 3 data?
    property data
    property target_color

```

```
class NiPoint3Interpolator (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiKeyBasedInterpolator
    Unknown.
    property data
    property point_3_value

class NiPointLight (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiLight
    A point light.
    property constant_attenuation
    property linear_attenuation
    property quadratic_attenuation

class NiPortal (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiAVObject
    A Portal
    property num_vertices
    property target
    property unknown_flags
    property unknown_short_2
    property vertices

class NiPosData (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiObject
    Position data.
    property data

class NiProperty (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiObjectNET
    A generic property object.

class NiRangeLODDData (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiLODDData
    Describes levels of detail based on distance of object from camera.
    property lod_center
    property lod_levels
    property num_lod_levels

class NiRawImageData (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiObject
    Raw image data.
    property height
    property image_type
    property rgb_image_data
```



```

    property rgba_image_data
    property width
class NiRenderObject (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiAVObject
    An object that can be rendered.
    property active_material
    property material_data
    property material_needs_update_default
    property num_materials
class NiRollController (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiSingleInterpController
    Unknown.
    property data
class NiRoom (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiNode
    Grouping node for nodes in a Portal
    property in_portals
    property items
    property num_in_portals
    property num_items
    property num_portals_2
    property num_walls
    property portals_2
    property wall_plane
class NiRoomGroup (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiNode
    Grouping node for nodes in a Portal
    property num_rooms
    property rooms
    property shell_link
class NiRotatingParticles (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiParticles
    Unknown.
class NiRotatingParticlesData (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiParticlesData
    Rotating particles data object.
    property has_rotations_2
    property rotations_2

```

```
class NiScreenElements (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiTriShape
    Two dimensional screen elements.

class NiScreenElementsData (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiTriShapeData
    Two dimensional screen elements.

    property max_polygons
    property num_polygons
    property polygon_indices
    property polygons
    property unknown_u_short_1
    property unknown_u_short_2
    property unknown_u_short_3
    property used_triangle_points
    property used_vertices

class NiScreenLODDData (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiLODDData
    Describes levels of detail based on size of object on screen?

    property bound_center
    property bound_radius
    property proportion_count
    property proportion_levels
    property world_center
    property world_radius

class NiSequence (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiObject
    Root node used in some Empire Earth II .kf files (version 4.2.2.0).

    property controlled_blocks
    property name
    property num_controlled_blocks
    property text_keys
    property text_keys_name
    property unknown_int_1
    property unknown_int_4
    property unknown_int_5

class NiSequenceData (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiObject
```

class NiSequenceStreamHelper (*template=None, argument=None, parent=None*)

Bases: `pyffi.formats.nif.NiObjectNET`

Keyframe animation root node, in .kf files.

class NiShadeProperty (*template=None, argument=None, parent=None*)

Bases: `pyffi.formats.nif.NiProperty`

Determines whether flat shading or smooth shading is used on a shape.

property flags

class NiShadowGenerator (*template=None, argument=None, parent=None*)

Bases: `pyffi.formats.nif.NiObject`

property name

property num_unknown_links_1

property target

property unknown_flags

property unknown_links_1

property unkown_byte_5

property unkown_byte_9

property unkown_float_4

property unkown_int_2

property unkown_int_6

property unkown_int_7

property unkown_int_8

class NiSingleInterpController (*template=None, argument=None, parent=None*)

Bases: `pyffi.formats.nif.NiInterpController`

A controller referring to a single interpolator.

property interpolator

class NiSkinData (*template=None, argument=None, parent=None*)

Bases: `pyffi.formats.nif._NiSkinData, object`

apply_scale (*scale*)

Apply scale factor on data.

```
>>> from pyffi.formats.nif import NifFormat
>>> id44 = NifFormat.Matrix44()
>>> id44.set_identity()
>>> skelroot = NifFormat.NiNode()
>>> skelroot.name = 'Scene Root'
>>> skelroot.set_transform(id44)
>>> bone1 = NifFormat.NiNode()
>>> bone1.name = 'bone1'
>>> bone1.set_transform(id44)
>>> bone1.translation.x = 10
>>> skelroot.add_child(bone1)
>>> geom = NifFormat.NiTriShape()
>>> geom.set_transform(id44)
```

(continues on next page)

(continued from previous page)

```

>>> skelroot.add_child(geom)
>>> skininst = NifFormat.NiSkinInstance()
>>> geom.skin_instance = skininst
>>> skininst.skeleton_root = skelroot
>>> skindata = NifFormat.NiSkinData()
>>> skininst.data = skindata
>>> skindata.set_transform(id44)
>>> geom.add_bone(bone1, {})
>>> geom.update_bind_position()
>>> bone1.translation.x
10.0
>>> skindata.bone_list[0].skin_transform.translation.x
-10.0
>>> import pyffi.spells.nif.fix
>>> import pyffi.spells.nif
>>> data = NifFormat.Data()
>>> data.roots = [skelroot]
>>> toaster = pyffi.spells.nif.NifToaster()
>>> toaster.scale = 0.1
>>> pyffi.spells.nif.fix.SpellScale(data=data, toaster=toaster).recurse()
pyffi.toaster:INFO:--- fix_scale ---
pyffi.toaster:INFO: scaling by factor 0.100000
pyffi.toaster:INFO: ~~~ NiNode [Scene Root] ~~~
pyffi.toaster:INFO: ~~~ NiNode [bone1] ~~~
pyffi.toaster:INFO: ~~~ NiTriShape [] ~~~
pyffi.toaster:INFO: ~~~ NiSkinInstance [] ~~~
pyffi.toaster:INFO: ~~~ NiSkinData [] ~~~
>>> bone1.translation.x
1.0
>>> skindata.bone_list[0].skin_transform.translation.x
-1.0

```

get_transform()

Return scale, rotation, and translation into a single 4x4 matrix.

set_transform(mat)

Set rotation, transform, and velocity.

class NiSkinInstance (*template=None, argument=None, parent=None*)

Bases: `pyffi.formats.nif.NiObject`

Skinning instance.

property bones

property data

property num_bones

property skeleton_root

property skin_partition

class NiSkinPartition (*template=None, argument=None, parent=None*)

Bases: `pyffi.formats.nif.NiObject`

Skinning data, optimized for hardware skinning. The mesh is partitioned in submeshes such that each vertex of a submesh is influenced only by a limited and fixed number of bones.

property num_skin_partition_blocks

```

    property skin_partition_blocks
class NiSkinningLODController (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiTimeController
class NiSkinningMeshModifier (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiMeshModifier
    property bone_bounds
    property bone_transforms
    property bones
    property flags
    property num_bones
    property skeleton_root
    property skeleton_transform
class NiSortAdjustNode (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiNode
    Unknown node. Found in Loki.
    property sorting_mode
    property unknown_int_2
class NiSourceCubeMap (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiSourceTexture
    Unknown node. Found in Emerge Demo.
class NiSourceTexture (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiTexture
    Describes texture source and properties.
    property alpha_format
    property direct_render
    property file_name
    property is_static
    property persist_render_data
    property pixel_data
    property pixel_layout
    property unknown_byte
    property unknown_link
    property use_external
    property use_mipmaps
class NiSpecularProperty (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiProperty
    Gives specularity to a shape. Flags 0x0001.
    property flags

```

class NiSphericalCollider (*template=None, argument=None, parent=None*)

Bases: `pyffi.formats.nif.NiParticleModifier`

Unknown.

property `unknown_float_1`

property `unknown_float_2`

property `unknown_float_3`

property `unknown_float_4`

property `unknown_float_5`

property `unknown_short_1`

property `unknown_short_2`

class NiSpotLight (*template=None, argument=None, parent=None*)

Bases: `pyffi.formats.nif.NiPointLight`

A spot.

property `cutoff_angle`

property `exponent`

property `unknown_float`

class NiStencilProperty (*template=None, argument=None, parent=None*)

Bases: `pyffi.formats.nif.NiProperty`

Allows control of stencil testing.

property `draw_mode`

property `fail_action`

property `flags`

property `pass_action`

property `stencil_enabled`

property `stencil_function`

property `stencil_mask`

property `stencil_ref`

property `z_fail_action`

class NiStringExtraData (*template=None, argument=None, parent=None*)

Bases: `pyffi.formats.nif.NiExtraData`

Apparently commands for an optimizer instructing it to keep things it would normally discard. Also refers to NiNode objects (through their name) in animation .kf files.

property `bytes_remaining`

property `string_data`

class NiStringPalette (*template=None, argument=None, parent=None*)

Bases: `pyffi.formats.nif.NiObject`

List of 0x00-separated strings, which are names of controlled objects and controller types. Used in .kf files in conjunction with NiControllerSequence.

```

    property palette
class NiStringsExtraData (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiExtraData
    List of strings; for example, a list of all bone names.
    property data
    property num_strings
class NiSwitchNode (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiNode
    A node used to switch between branches, such as for LOD levels?
    property unknown_flags_1
    property unknown_int_1
class NiTextKeyExtraData (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiExtraData
    Extra data, used to name different animation sequences.
    property num_text_keys
    property text_keys
    property unknown_int_1
class NiTexture (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiObjectNET
    A texture.
class NiTextureEffect (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiDynamicEffect
    Enables environment mapping. Should be in both the children list and effects list of the NiTriShape object.
    For Morrowind: the bump map can be used to bump the environment map (note that the bump map is
    ignored if no NiTextureEffect object is present).
    property clipping_plane
    property coordinate_generation_type
    property image
    property model_projection_matrix
    property model_projection_transform
    property ps_2_k
    property ps_2_1
    property source_texture
    property texture_clamping
    property texture_filtering
    property texture_type
    property unknown
    property unknown_float

```

```
    property unknown_short
    property unknown_vector

class NiTextureModeProperty (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiProperty
    Unknown
    property ps_2_k
    property ps_2_l
    property unknown_ints
    property unknown_short

class NiTextureProperty (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiProperty
    property flags
    property image
    property unknown_ints_1
    property unknown_ints_2

class NiTextureTransformController (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiFloatInterpController
    Texture transformation controller. The target texture slot should have “Has Texture Transform” enabled.
    property data
    property operation
    property texture_slot
    property unknown_2

class NiTexturingProperty (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiProperty
    Describes an object’s textures.
    property apply_mode
    property base_texture
    property bump_map_luma_offset
    property bump_map_luma_scale
    property bump_map_matrix
    property bump_map_texture
    property dark_texture
    property decal_0_texture
    property decal_1_texture
    property decal_2_texture
    property decal_3_texture
    property detail_texture
```



```

property flags
property gloss_texture
property glow_texture
property has_base_texture
property has_bump_map_texture
property has_dark_texture
property has_decal_0_texture
property has_decal_1_texture
property has_decal_2_texture
property has_decal_3_texture
property has_detail_texture
property has_gloss_texture
property has_glow_texture
property has_normal_texture
property has_unknown_2_texture
property normal_texture
property num_shader_textures
property shader_textures
property texture_count
property unknown_2_float
property unknown_2_texture

class NiTimeController (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiObject
    A generic time controller object.
    property flags
    property frequency
    property next_controller
    property phase
    property start_time
    property stop_time
    property target
    property unknown_integer

class NiTransformController (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiKeyframeController
    NiTransformController replaces the NiKeyframeController.

```

```
class NiTransformData (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiKeyframeData

    Mesh animation keyframe data.

class NiTransformEvaluator (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiObject

class NiTransformInterpolator (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif._NiTransformInterpolator, object

    apply_scale (scale)
        Apply scale factor <scale> on data.

class NiTransparentProperty (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiProperty

    Unknown

    property unknown

class NiTriBasedGeom (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif._NiTriBasedGeom, object

    get_interchangeable_tri_shape (triangles=None)
        Returns a NiTriShape block that is geometrically interchangeable. If you do not want to set the
        triangles from the original shape, use the triangles argument.

    get_interchangeable_tri_strips (strips=None)
        Returns a NiTriStrips block that is geometrically interchangeable. If you do not want to set the strips
        from the original shape, use the strips argument.

    get_tangent_space ()
        Return iterator over normal, tangent, bitangent vectors. If the block has no tangent space, then returns
        None.

    update_skin_center_radius ()
        Update centers and radii of all skin data fields.

    update_skin_partition (maxbonesperpartition=4, maxbonespervortex=4, verbose=0, strip-
        ify=True, stitchstrips=False, padbones=False, triangles=None, tri-
        anglepartmap=None, maximize_bone_sharing=False)
        Recalculate skin partition data.
        Deprecated Do not use the verbose argument.

    Parameters
        • maxbonesperpartition – Maximum number of bones in each partition. The
          num_bones field will not exceed this number.
        • maxbonespervortex – Maximum number of bones per vertex. The
          num_weights_per_vertex field will be exactly equal to this number.
        • verbose – Ignored, and deprecated. Set pyffi's log level instead.
        • stripify – If true, stripify the partitions, otherwise use triangles.
        • stitchstrips – If stripify is true, then set this to true to stitch the strips.
        • padbones – Enforces the numbones field to be equal to maxbonesperpartition. Also
          ensures that the bone indices are unique and sorted, per vertex. Raises an exception
          if maxbonespervortex is not equal to maxbonesperpartition (in that case bone indices
          cannot be unique and sorted). This options is required for Freedom Force vs. the 3rd
          Reich skin partitions.
        • triangles – The triangles of the partition (if not specified, then this defaults to
          C{self.data.get_triangles()}.
```

- **trianglepartmap** – Maps each triangle to a partition index. Faces with different indices will never appear in the same partition. If the skin instance is a BSDismemberSkinInstance, then these indices are used as body part types, and the partitions in the BSDismemberSkinInstance are updated accordingly. Note that the faces are counted relative to L{triangles}.
- **maximize_bone_sharing** – Maximize bone sharing between partitions. This option is useful for Fallout 3.

update_tangent_space (*as_extra=None, vertexprecision=3, normalprecision=3*)

Recalculate tangent space data.

Parameters **as_extra** – Whether to store the tangent space data as extra data (as in Oblivion) or not (as in Fallout 3). If not set, switches to Oblivion if an extra data block is found, otherwise does default. Set it to override this detection (for example when using this function to create tangent space data) and force behaviour.

class NiTriBasedGeomData (*template=None, argument=None, parent=None*)

Bases: `pyffi.formats.nif._NiTriBasedGeomData`, `object`

get_triangle_indices (*triangles*)

Yield list of triangle indices (relative to `self.get_triangles()`) of given triangles. Degenerate triangles in the list are assigned index `None`.

```
>>> from pyffi.formats.nif import NifFormat
>>> geomdata = NifFormat.NiTriShapeData()
>>> geomdata.set_triangles([(0,1,2), (1,2,3), (2,3,4)])
>>> list(geomdata.get_triangle_indices([(1,2,3)]))
[1]
>>> list(geomdata.get_triangle_indices([(3,1,2)]))
[1]
>>> list(geomdata.get_triangle_indices([(2,3,1)]))
[1]
>>> list(geomdata.get_triangle_indices([(1,2,0), (4,2,3)]))
[0, 2]
>>> list(geomdata.get_triangle_indices([(0,0,0), (4,2,3)]))
[None, 2]
>>> list(geomdata.get_triangle_indices([(0,3,4), (4,2,3)]))
Traceback (most recent call last):
...
ValueError: ...
```

Parameters **triangles** (*iterator or list of tuples of three ints*) – An iterable of triangles to check.

is_interchangeable (*other*)

Heuristically checks if two `NiTriBasedGeomData` blocks describe the same geometry, that is, if they can be used interchangeably in a NIF file without affecting the rendering. The check is not fool proof but has shown to work in most practical cases.

Parameters **other** (`L{NifFormat.NiTriBasedGeomData}`) (if it has another type then the function will always return `False`) – Another geometry data block.

Returns `True` if the geometries are equivalent, `False` otherwise.

class NiTriShape (*template=None, argument=None, parent=None*)

Bases: `pyffi.formats.nif.NiTriBasedGeom`

A shape node that refers to singular triangle data.

class NiTriShapeData (*template=None, argument=None, parent=None*)

Bases: `pyffi.formats.nif._NiTriShapeData`, `object`

Example usage:

```
>>> from pyffi.formats.nif import NifFormat
>>> block = NifFormat.NiTriShapeData()
>>> block.set_triangles([(0,1,2), (2,1,3), (2,3,4)])
>>> block.get_strips()
[[0, 1, 2, 3, 4]]
>>> block.get_triangles()
[(0, 1, 2), (2, 1, 3), (2, 3, 4)]
>>> block.set_strips([[1,0,1,2,3,4]])
>>> block.get_strips() # stripifier keeps geometry but nothing else
[[0, 2, 1, 3], [2, 4, 3]]
>>> block.get_triangles()
[(0, 2, 1), (1, 2, 3), (2, 4, 3)]
```

get_strips()

get_triangles()

set_strips(strips)

set_triangles(triangles, stitchstrips=False)

class NiTriShapeSkinController(template=None, argument=None, parent=None)

Bases: pyffi.formats.nif.NiTimeController

Old version of skinning instance.

property bone_data

property bones

property num_bones

property vertex_counts

class NiTriStrips(template=None, argument=None, parent=None)

Bases: pyffi.formats.nif.NiTriBasedGeom

A shape node that refers to data organized into strips of triangles

class NiTriStripsData(template=None, argument=None, parent=None)

Bases: pyffi.formats.nif._NiTriStripsData, object

Example usage:

```
>>> from pyffi.formats.nif import NifFormat
>>> block = NifFormat.NiTriStripsData()
>>> block.set_triangles([(0,1,2), (2,1,3), (2,3,4)])
>>> block.get_strips()
[[0, 1, 2, 3, 4]]
>>> block.get_triangles()
[(0, 1, 2), (1, 3, 2), (2, 3, 4)]
>>> block.set_strips([[1,0,1,2,3,4]])
>>> block.get_strips()
[[1, 0, 1, 2, 3, 4]]
>>> block.get_triangles()
[(0, 2, 1), (1, 2, 3), (2, 4, 3)]
```

get_strips()

get_triangles()

set_strips(strips)

set_triangles(triangles, stitchstrips=False)

```

class NiUVController (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiTimeController
    Time controller for texture coordinates.

    property data
    property unknown_short

class NiUVData (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiObject
    Texture coordinate data.

    property uv_groups

class NiVectorExtraData (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiExtraData
    Extra vector data.

    property unknown_float
    property vector_data

class NiVertWeightsExtraData (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiExtraData
    Not used in skinning. Unsure of use - perhaps for morphing animation or gravity.

    property num_bytes
    property num_vertices
    property weight

class NiVertexColorProperty (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiProperty
    Property of vertex colors. This object is referred to by the root object of the NIF file whenever some
    NiTriShapeData object has vertex colors with non-default settings; if not present, vertex colors have ver-
    tex_mode=2 and lighting_mode=1.

    property flags
    property lighting_mode
    property vertex_mode

class NiVisController (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiBoolInterpController
    Time controller for visibility.

    property data

class NiVisData (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiObject
    Visibility data for a controller.

    property keys
    property num_keys

class NiWireframeProperty (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiProperty

```

Unknown.

property flags

class NiZBufferProperty (*template=None, argument=None, parent=None*)

Bases: `pyffi.formats.nif.NiProperty`

This Property controls the Z buffer (OpenGL: depth buffer).

property flags

property function

exception NifError

Bases: `Exception`

Standard nif exception class.

class NodeGroup (*template=None, argument=None, parent=None*)

Bases: `pyffi.object_models.xml.struct_.StructBase`

A group of NiNodes references.

property nodes

property num_nodes

class OblivionHavokMaterial (***kwargs*)

Bases: `pyffi.object_models.xml.enum.EnumBase`

A material, used by havok shape objects in Oblivion.

HAV_MAT_CHAIN = 13

HAV_MAT_CHAIN_STAIRS = 28

HAV_MAT_CLOTH = 1

HAV_MAT_CLOTH_STAIRS = 16

HAV_MAT_DIRT = 2

HAV_MAT_DIRT_STAIRS = 17

HAV_MAT_ELEVATOR = 30

HAV_MAT_GLASS = 3

HAV_MAT_GLASS_STAIRS = 18

HAV_MAT_GRASS = 4

HAV_MAT_GRASS_STAIRS = 19

HAV_MAT_HEAVY_METAL = 11

HAV_MAT_HEAVY_METAL_STAIRS = 26

HAV_MAT_HEAVY_STONE = 10

HAV_MAT_HEAVY_STONE_STAIRS = 25

HAV_MAT_HEAVY_WOOD = 12

HAV_MAT_HEAVY_WOOD_STAIRS = 27

HAV_MAT_METAL = 5

HAV_MAT_METAL_STAIRS = 20

```

HAV_MAT_ORGANIC = 6
HAV_MAT_ORGANIC_STAIRS = 21
HAV_MAT_RUBBER = 31
HAV_MAT_SKIN = 7
HAV_MAT_SKIN_STAIRS = 22
HAV_MAT_SNOW = 14
HAV_MAT_SNOW_STAIRS = 29
HAV_MAT_STONE = 0
HAV_MAT_STONE_STAIRS = 15
HAV_MAT_WATER = 8
HAV_MAT_WATER_STAIRS = 23
HAV_MAT_WOOD = 9
HAV_MAT_WOOD_STAIRS = 24

class OblivionLayer(**kwargs)
    Bases: pyffi.object_models.xml.enum.EnumBase
    Sets mesh color in Oblivion Construction Set. Anything higher than 57 is also null.

    ANIM_STATIC = 2
    AVOID_BOX = 21
    BACK_WEAPON = 53
    BACK_WEAPON2 = 54
    BIPED = 8
    BODY = 34
    CAMERA_PICK = 24
    CHAR_CONTROLLER = 20
    CLOUD_TRAP = 16
    CLUTTER = 4
    CUSTOM_PICK_1 = 28
    CUSTOM_PICK_2 = 29
    DROPPING_PICK = 31
    GROUND = 17
    HEAD = 33
    ITEM_PICK = 25
    LINE_OF_SIGHT = 26
    L_CALF = 41
    L_FOOT = 42
    L_FOREARM = 38

```

```
L_HAND = 39
L_THIGH = 40
L_UPPER_ARM = 37
NONCOLLIDABLE = 15
NULL = 57
OTHER = 32
PATH_PICK = 27
PONYTAIL = 55
PORTAL = 18
PROJECTILE = 6
PROPS = 10
QUIVER = 52
R_CALF = 47
R_FOOT = 48
R_FOREARM = 44
R_HAND = 45
R_THIGH = 46
R_UPPER_ARM = 43
SHIELD = 51
SIDE_WEAPON = 50
SPELL = 7
SPELL_EXPLOSION = 30
SPINE1 = 35
SPINE2 = 36
STAIRS = 19
STATIC = 1
TAIL = 49
TERRAIN = 13
TRANSPARENT = 3
TRAP = 14
TREES = 9
TRIGGER = 12
UNIDENTIFIED = 0
UNKNOWN1 = 22
UNKNOWN2 = 23
WATER = 11
```



```

WEAPON = 5
WING = 56

class OblivionSubShape (template=None, argument=None, parent=None)
    Bases: pyffi.object_models.xml.struct_.StructBase
    Havok Information for packed TriStrip shapes.
    property havok_col_filter
    property material
    property num_vertices

class OldSkinData (template=None, argument=None, parent=None)
    Bases: pyffi.object_models.xml.struct_.StructBase
    Used to store skin weights in NiTriShapeSkinController.
    property unknown_vector
    property vertex_index
    property vertex_weight

class PSLoopBehavior (**kwargs)
    Bases: pyffi.object_models.xml.enum.EnumBase
    PS_LOOP_AGESCALE = 2
    PS_LOOP_CLAMP_BIRTH = 0
    PS_LOOP_CLAMP_DEATH = 1
    PS_LOOP_LOOP = 3
    PS_LOOP_REFLECT = 4

class Particle (template=None, argument=None, parent=None)
    Bases: pyffi.object_models.xml.struct_.StructBase
    particle array entry
    property lifespan
    property lifetime
    property timestamp
    property unknown_short
    property unknown_vector
    property velocity
    property vertex_id

class ParticleDesc (template=None, argument=None, parent=None)
    Bases: pyffi.object_models.xml.struct_.StructBase
    Particle Description.
    property translation
    property unknown_float_1
    property unknown_float_2
    property unknown_float_3

```

```
    property unknown_floats_1
    property unknown_int_1
class PixelFormat (**kwargs)
    Bases: pyffi.object_models.xml.enum.EnumBase
    Specifies the pixel format used by the NiPixelData object to store a texture.
    PX_FMT_DXT1 = 4
    PX_FMT_DXT5 = 5
    PX_FMT_DXT5_ALT = 6
    PX_FMT_PAL8 = 2
    PX_FMT_RGB8 = 0
    PX_FMT_RGBA8 = 1
class PixelLayout (**kwargs)
    Bases: pyffi.object_models.xml.enum.EnumBase
    An unsigned 32-bit integer, describing the color depth of a texture.
    PIX_LAY_BUMPMAP = 4
    PIX_LAY_COMPRESSED = 3
    PIX_LAY_DEFAULT = 6
    PIX_LAY_HIGH_COLOR_16 = 1
    PIX_LAY_PALETTISED = 0
    PIX_LAY_PALETTISED_4 = 5
    PIX_LAY_TRUE_COLOR_32 = 2
class Polygon (template=None, argument=None, parent=None)
    Bases: pyffi.object_models.xml.struct_.StructBase
    Two dimensional screen elements.
    property num_triangles
    property num_vertices
    property triangle_offset
    property vertex_offset
class PrismaticDescriptor (template=None, argument=None, parent=None)
    Bases: pyffi.object_models.xml.struct_.StructBase
    property friction
    property max_distance
    property min_distance
    property pivot_a
    property pivot_b
    property plane_a
    property plane_b
```

```

property rotation_a
property rotation_b
property rotation_matrix_a
property sliding_a
property sliding_b
property unknown_byte_1
class PropagationMode (**kwargs)
    Bases: pyffi.object_models.xml.enum.EnumBase

    PROPAGATE_ALWAYS = 2
    PROPAGATE_NEVER = 3
    PROPAGATE_ON_FAILURE = 1
    PROPAGATE_ON_SUCCESS = 0

class Ptr (**kwargs)
    Bases: pyffi.formats.nif.Ref

    A weak reference to another block, used to point up the hierarchy tree. The reference is not returned by
    the L{get_refs} function to avoid infinite recursion.

    get_hash (data=None)
        Returns a hash value (an immutable object) that can be used to identify the object uniquely.

    get_refs (data=None)
        Return all references (excluding weak pointers) used by this object.

    get_value ()
        Return object value.

    replace_global_node (oldbranch, newbranch, edge_filter=(True, True))

```

```

>>> from pyffi.formats.nif import NifFormat
>>> x = NifFormat.NiNode()
>>> y = NifFormat.NiNode()
>>> z = NifFormat.NiNode()
>>> x.add_child(y)
>>> x.children[0] is y
True
>>> x.children[0] is z
False
>>> x.replace_global_node(y, z)
>>> x.children[0] is y
False
>>> x.children[0] is z
True
>>> x.replace_global_node(z, None)
>>> x.children[0] is None
True

```

```

set_value (value)
    Set object value.

class QTransform (template=None, argument=None, parent=None)
    Bases: pyffi.object_models.xml.struct_.StructBase

```

```
    property rotation
    property scale
    property translation

class QuatKey (template=None, argument=None, parent=None)
    Bases: pyffi.object_models.xml.struct_.StructBase
    A special version of the key type used for quaternions. Never has tangents.
    property tbc
    property time
    property value

class Quaternion (template=None, argument=None, parent=None)
    Bases: pyffi.object_models.xml.struct_.StructBase
    A quaternion.
    property w
    property x
    property y
    property z

class QuaternionXYZW (template=None, argument=None, parent=None)
    Bases: pyffi.object_models.xml.struct_.StructBase
    A quaternion as it appears in the havok objects.
    property w
    property x
    property y
    property z

RE_FILENAME = re.compile('^.*\.(nif|kf|kfa|nifcache|jmi|texcache|pcpatch|nft|item|nif

class RagdollDescriptor (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif._RagdollDescriptor, object
    update_a_b (transform)
        Update B pivot and axes from A using the given transform.

class Ref (**kwargs)
    Bases: pyffi.object_models.xml.basic.BasicBase
    Reference to another block.
    fix_links (data)
        Fix block links.
    get_detail_display ()
        Return an object that can be used to display the instance.
    get_hash (data=None)
        Returns a hash value (an immutable object) that can be used to identify the object uniquely.
    get_links (data=None)
        Return all links referred to in this object.
```

get_refs (*data=None*)
Return all references (excluding weak pointers) used by this object.

get_size (*data=None*)
Returns size of the object in bytes.

get_value ()
Return object value.

read (*stream, data*)
Read object from file.

replace_global_node (*oldbranch, newbranch, edge_filter=(True, True)*)

```
>>> from pyffi.formats.nif import NifFormat
>>> x = NifFormat.NiNode()
>>> y = NifFormat.NiNode()
>>> z = NifFormat.NiNode()
>>> x.add_child(y)
>>> x.children[0] is y
True
>>> x.children[0] is z
False
>>> x.replace_global_node(y, z)
>>> x.children[0] is y
False
>>> x.children[0] is z
True
>>> x.replace_global_node(z, None)
>>> x.children[0] is None
True
```

set_value (*value*)
Set object value.

write (*stream, data*)
Write block reference.

class Region (*template=None, argument=None, parent=None*)
Bases: `pyffi.object_models.xml.struct_.StructBase`
A range of indices, which make up a region (such as a submesh).

property num_indices

property start_index

class RootCollisionNode (*template=None, argument=None, parent=None*)
Bases: `pyffi.formats.nif.NiNode`
Morrowind-specific node for collision mesh.

class SemanticData (*template=None, argument=None, parent=None*)
Bases: `pyffi.object_models.xml.struct_.StructBase`

property index

property name

class ShaderTexDesc (*template=None, argument=None, parent=None*)
Bases: `pyffi.object_models.xml.struct_.StructBase`
An extended texture description for shader textures.

```
property is_used
property map_index
property texture_data
class ShortString (**kwargs)
    Bases: pyffi.object_models.xml.basic.BasicBase
    Another type for strings.
    get_hash (data=None)
        Returns a hash value (an immutable object) that can be used to identify the object uniquely.
    get_size (data=None)
        Returns size of the object in bytes.
    get_value ()
        Return object value.
    read (stream, data)
        Read object from file.
    set_value (value)
        Set object value.
    write (stream, data)
        Write object to file.
class SizedString (**kwargs)
    Bases: pyffi.object_models.xml.basic.BasicBase, pyffi.object_models.
    editable.EditableLineEdit
```

Basic type for strings. The type starts with an unsigned int which describes the length of the string.

```
>>> from tempfile import TemporaryFile
>>> f = TemporaryFile()
>>> from pyffi.object_models import FileFormat
>>> data = FileFormat.Data()
>>> s = SizedString()
>>> if f.write('\x07\x00\x00\x00abcdefg'.encode("ascii")): pass # ignore_
↳result for py3k
>>> if f.seek(0): pass # ignore result for py3k
>>> s.read(f, data)
>>> str(s)
'abcdefg'
>>> if f.seek(0): pass # ignore result for py3k
>>> s.set_value('Hi There')
>>> s.write(f, data)
>>> if f.seek(0): pass # ignore result for py3k
>>> m = SizedString()
>>> m.read(f, data)
>>> str(m)
'Hi There'
```

```
get_hash (data=None)
    Return a hash value for this string.
    Returns An immutable object that can be used as a hash.
get_size (data=None)
    Return number of bytes this type occupies in a file.
    Returns Number of bytes.
```

```

get_value()
    Return the string.
    Returns The stored string.

read(stream, data)
    Read string from stream.
    Parameters stream (file) – The stream to read from.

set_value(value)
    Set string to C{value}.
    Parameters value (str) – The value to assign.

write(stream, data)
    Write string to stream.
    Parameters stream (file) – The stream to write to.

class SkinData (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif._SkinData, object

    get_transform()
        Return scale, rotation, and translation into a single 4x4 matrix.

    set_transform(mat)
        Set rotation, transform, and velocity.

class SkinPartition (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif._SkinPartition, object

    get_mapped_triangles()
        Get list of triangles of this partition (mapping into the geometry data vertex list).

    get_triangles()
        Get list of triangles of this partition.

class SkinShape (template=None, argument=None, parent=None)
    Bases: pyffi.object_models.xml.struct_.StructBase

    Reference to shape and skin instance.

    property shape

    property skin_instance

class SkinShapeGroup (template=None, argument=None, parent=None)
    Bases: pyffi.object_models.xml.struct_.StructBase

    Unknown.

    property link_pairs

    property num_link_pairs

class SkinTransform (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif._SkinTransform, object

    get_transform()
        Return scale, rotation, and translation into a single 4x4 matrix.

    set_transform(mat)
        Set rotation, transform, and velocity.

class SkinWeight (template=None, argument=None, parent=None)
    Bases: pyffi.object_models.xml.struct_.StructBase

    A weighted vertex.

```

```
    property index
    property weight

class SkyObjectType (**kwargs)
    Bases: pyffi.object_models.xml.enum.EnumBase
    Sets what sky function this object fulfills in BSSkyShaderProperty or SkyShaderProperty.
    BSSM_SKY = 2
    BSSM_SKY_CLOUDS = 3
    BSSM_SKY_MOON_STARS_MASK = 7
    BSSM_SKY_STARS = 5
    BSSM_SKY_SUNGLARE = 1
    BSSM_SKY_TEXTURE = 0

class SkyShaderProperty (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.BSShaderLightingProperty
    Bethesda-specific node? Found in Fallout3
    property file_name
    property sky_object_type

class SkyrimHavokMaterial (**kwargs)
    Bases: pyffi.object_models.xml.enum.EnumBase
    A material, used by havok shape objects in Skyrim.
    MAT_BARREL = 732141076
    MAT_BOTTLE = 493553910
    MAT_BROKEN_STONE = 131151687
    MAT_CLOTH = 3839073443
    MAT_DIRT = 3106094762
    MAT_DRAGON = 2518321175
    MAT_GLASS = 3739830338
    MAT_GRASS = 1848600814
    MAT_GRAVEL = 428587608
    MAT_HEAVY_METAL = 2229413539
    MAT_HEAVY_STONE = 1570821952
    MAT_HEAVY_WOOD = 3070783559
    MAT_ICE = 873356572
    MAT_LIGHT_WOOD = 365420259
    MAT_MATERIAL_ARMOR_HEAVY = 3708432437
    MAT_MATERIAL_ARMOR_LIGHT = 3424720541
    MAT_MATERIAL_ARROW = 3725505938
    MAT_MATERIAL_AXE_1HAND = 1305674443
```


MAT_MATERIAL_BASKET = 790784366
MAT_MATERIAL_BLADE_1HAND = 1060167844
MAT_MATERIAL_BLADE_1HAND_SMALL = 2617944780
MAT_MATERIAL_BLADE_2HAND = 2022742644
MAT_MATERIAL_BLUNT_2HAND = 3969592277
MAT_MATERIAL_BONE = 3049421844
MAT_MATERIAL_BOOK = 1264672850
MAT_MATERIAL_BOTTLE_SMALL = 2025794648
MAT_MATERIAL_BOULDER_LARGE = 1885326971
MAT_MATERIAL_BOULDER_MEDIUM = 4283869410
MAT_MATERIAL_BOULDER_SMALL = 1550912982
MAT_MATERIAL_BOWS_STAVES = 1607128641
MAT_MATERIAL_CARPET = 1286705471
MAT_MATERIAL_CERAMIC_MEDIUM = 781661019
MAT_MATERIAL_CHAIN = 3074114406
MAT_MATERIAL_CHAIN_METAL = 438912228
MAT_MATERIAL_COIN = 3589100606
MAT_MATERIAL_SHIELD_HEAVY = 3702389584
MAT_MATERIAL_SHIELD_LIGHT = 3448167928
MAT_MATERIAL_SKIN_LARGE = 2965929619
MAT_MATERIAL_SKIN_SMALL = 2632367422
MAT_MATERIAL_STONE_AS_STAIRS = 1886078335
MAT_MATERIAL_WOOD_AS_STAIRS = 1803571212
MAT_MUD = 1486385281
MAT_ORGANIC = 2974920155
MAT_SAND = 2168343821
MAT_SKIN = 591247106
MAT_SNOW = 398949039
MAT_SOLID_METAL = 1288358971
MAT_STAIRS_BROKEN_STONE = 2892392795
MAT_STAIRS_SNOW = 1560365355
MAT_STAIRS_STONE = 899511101
MAT_STAIRS_WOOD = 1461712277
MAT_STONE = 3741512247
MAT_UNKNOWN_1028101969 = 1028101969
MAT_UNKNOWN_1440721808 = 1440721808

```
MAT_UNKNOWN_1574477864 = 1574477864
MAT_UNKNOWN_1591009235 = 1591009235
MAT_WATER = 1024582599
MAT_WOOD = 500811281
class SkyrimLayer(**kwargs)
    Bases: pyffi.object_models.xml.enum.EnumBase
    Physical purpose of collision object? The setting affects object's havok behavior in game. Anything higher
    than 47 is also null.
    ACOUSTIC_SPACE = 21
    ACTORZONE = 22
    ANIMSTATIC = 2
    AVOIDBOX = 34
    BIPED = 8
    BIPED_NO_CC = 33
    CAMERAPICK = 39
    CAMERASHPERE = 36
    CHARCONTROLLER = 30
    CLOUD_TRAP = 16
    CLUTTER = 4
    COLLISIONBOX = 35
    CONEPROJECTILE = 38
    CUSTOMPICK1 = 43
    CUSTOMPICK2 = 44
    DEADBIP = 32
    DEBRIS_LARGE = 20
    DEBRIS_SMALL = 19
    DOORDETECTION = 37
    DROPPINGPICK = 46
    GASTRAP = 24
    GROUND = 17
    INVISIBLE_WALL = 27
    ITEMPICK = 40
    LINEOFSIGHT = 41
    NONCOLLIDABLE = 15
    NULL = 47
    PATHPICK = 42
    PORTAL = 18
```

```

PROJECTILE = 6
PROJECTILEZONE = 23
PROPS = 10
SHELLCASING = 25
SPELL = 7
SPELLEXPLOSION = 45
STAIRHELPER = 31
STATIC = 1
TERRAIN = 13
TRANSPARENT = 3
TRANSPARENT_SMALL = 26
TRANSPARENT_SMALL_ANIM = 28
TRAP = 14
TREES = 9
TRIGGER = 12
UNIDENTIFIED = 0
WARD = 29
WATER = 11
WEAPON = 5

class SkyrimShaderPropertyFlags1 (template=None, argument=None, parent=None)
    Bases: pyffi.object_models.xml.bit_struct.BitStructBase
    Skyrim Shader Property Flags 1
    property slsf_1_cast_shadows
    property slsf_1_decals
    property slsf_1_dynamic_decals
    property slsf_1_environment_mapping
    property slsf_1_external_emittance
    property slsf_1_eye_environment_mapping
    property slsf_1_face_gen_rgb_tint
    property slsf_1_facegen_detail_map
    property slsf_1_fire_refraction
    property slsf_1_greyscale_to_palette_alpha
    property slsf_1_greyscale_to_palette_color
    property slsf_1_hair_soft_lighting
    property slsf_1_landscape
    property slsf_1_localmap_hide_secret

```

```
property slsf_1_model_space_normals
property slsf_1_multiple_textures
property slsf_1_non_projective_shadows
property slsf_1_own_emit
property slsf_1_parallax
property slsf_1_parallax_occlusion
property slsf_1_projected_uv
property slsf_1_recieve_shadows
property slsf_1_refraction
property slsf_1_remappable_textures
property slsf_1_screendoor_alpha_fade
property slsf_1_skinned
property slsf_1_soft_effect
property slsf_1_specular
property slsf_1_temp_refraction
property slsf_1_use_falloff
property slsf_1_vertex_alpha
property slsf_1_z_buffer_test

class SkyrimShaderPropertyFlags2 (template=None, argument=None, parent=None)
    Bases: pyffi.object_models.xml.bit_struct.BitStructBase
    Skyrim Shader Property Flags 2
    property slsf_2_anisotropic_lighting
    property slsf_2_assume_shadowmask
    property slsf_2_back_lighting
    property slsf_2_billboard
    property slsf_2_cloud_lod
    property slsf_2_double_sided
    property slsf_2_effect_lighting
    property slsf_2_env_map_light_fade
    property slsf_2_fit_slope
    property slsf_2_glow_map
    property slsf_2_hd_lod_objects
    property slsf_2_hide_on_local_map
    property slsf_2_lod_landscape
    property slsf_2_lod_objects
    property slsf_2_multi_index_snow
```

```

property slsf_2_multi_layer_parallax
property slsf_2_no_fade
property slsf_2_no_lod_land_blend
property slsf_2_no_transparency_multisampling
property slsf_2_packed_tangent
property slsf_2_premult_alpha
property slsf_2_rim_lighting
property slsf_2_soft_lighting
property slsf_2_tree_anim
property slsf_2_uniform_scale
property slsf_2_unused_01
property slsf_2_unused_02
property slsf_2_vertex_colors
property slsf_2_vertex_lighting
property slsf_2_weapon_blood
property slsf_2_wireframe
property slsf_2_z_buffer_write

```

class **SkyrimWaterShaderFlags** (*template=None, argument=None, parent=None*)
 Bases: `pyffi.object_models.xml.bit_struct.BitStructBase`
 Skyrim water shader property flags

```

property swsf_1_bypass_refraction_map
property swsf_1_enabled
property swsf_1_highlight_layer_toggle
property swsf_1_unknown_0
property swsf_1_unknown_3
property swsf_1_unknown_4
property swsf_1_unknown_5
property swsf_1_water_toggle

```

class **SolverDeactivation** (***kwargs*)
 Bases: `pyffi.object_models.xml.enum.EnumBase`

A list of possible solver deactivation settings. This value defines how the solver deactivates objects. The solver works on a per object basis. Note: Solver deactivation does not save CPU, but reduces creeping of movable objects in a pile quite dramatically.

```

SOLVER_DEACTIVATION_HIGH = 4
SOLVER_DEACTIVATION_INVALID = 0
SOLVER_DEACTIVATION_LOW = 2
SOLVER_DEACTIVATION_MAX = 5

```

```
SOLVER_DEACTIVATION_MEDIUM = 3
SOLVER_DEACTIVATION_OFF = 1
class SortingMode (**kwargs)
    Bases: pyffi.object_models.xml.enum.EnumBase
    SORTING_INHERIT = 0
    SORTING_OFF = 1
class SphereBV (template=None, argument=None, parent=None)
    Bases: pyffi.object_models.xml.struct_.StructBase
    A sphere.
    property center
    property radius
class StencilAction (**kwargs)
    Bases: pyffi.object_models.xml.enum.EnumBase
    This enum defines the various actions used in conjunction with the stencil buffer. For a detailed description
    of the individual options please refer to the OpenGL docs.
    ACTION_DECREMENT = 4
    ACTION_INCREMENT = 3
    ACTION_INVERT = 5
    ACTION_KEEP = 0
    ACTION_REPLACE = 2
    ACTION_ZERO = 1
class StencilCompareMode (**kwargs)
    Bases: pyffi.object_models.xml.enum.EnumBase
    This enum contains the options for doing stencil buffer tests.
    TEST_ALWAYS = 7
    TEST_EQUAL = 2
    TEST_GREATER = 4
    TEST_GREATER_EQUAL = 6
    TEST_LESS = 1
    TEST_LESS_EQUAL = 3
    TEST_NEVER = 0
    TEST_NOT_EQUAL = 5
class StiffSpringDescriptor (template=None, argument=None, parent=None)
    Bases: pyffi.object_models.xml.struct_.StructBase
    property length
    property pivot_a
    property pivot_b
```

StringIndex

alias of `pyffi.object_models.common.UInt`

class StringOffset (***kwargs*)

Bases: `pyffi.object_models.common.Int`

This is just an integer with -1 as default value.

class StringPalette (*template=None, argument=None, parent=None*)

Bases: `pyffi.formats.nif._StringPalette, object`

add_string (*text*)

Adds string to palette (will recycle existing strings if possible) and return offset to the string in the palette.

```
>>> from pyffi.formats.nif import NifFormat
>>> pal = NifFormat.StringPalette()
>>> pal.add_string("abc")
0
>>> pal.add_string("abc")
0
>>> pal.add_string("def")
4
>>> pal.add_string("")
-1
>>> print(pal.get_string(4).decode("ascii"))
def
```

clear ()

Clear all strings in the palette.

```
>>> from pyffi.formats.nif import NifFormat
>>> pal = NifFormat.StringPalette()
>>> pal.add_string("abc")
0
>>> pal.add_string("def")
4
>>> # pal.palette.decode("ascii") needs lstrip magic for py3k
>>> print(repr(pal.palette.decode("ascii")).lstrip("u"))
'abc\x00def\x00'
>>> pal.clear()
>>> # pal.palette.decode("ascii") needs lstrip magic for py3k
>>> print(repr(pal.palette.decode("ascii")).lstrip("u"))
''
```

get_all_strings ()

Return a list of all strings.

```
>>> from pyffi.formats.nif import NifFormat
>>> pal = NifFormat.StringPalette()
>>> pal.add_string("abc")
0
>>> pal.add_string("def")
4
>>> for x in pal.get_all_strings():
...     print(x.decode("ascii"))
abc
def
>>> # pal.palette.decode("ascii") needs lstrip magic for py3k
```

(continues on next page)

(continued from previous page)

```
>>> print(repr(pal.palette.decode("ascii")).rstrip("u"))
'abc\x00def\x00'
```

get_string (*offset*)

Return string at given offset.

```
>>> from pyffi.formats.nif import NifFormat
>>> pal = NifFormat.StringPalette()
>>> pal.add_string("abc")
0
>>> pal.add_string("def")
4
>>> print(pal.get_string(0).decode("ascii"))
abc
>>> print(pal.get_string(4).decode("ascii"))
def
>>> pal.get_string(5)
pyffi.nif.stringpalette:WARNING:StringPalette: no string starts at offset_
↪5 (string is b'ef', preceeding character is b'd')
b'ef'
>>> pal.get_string(100)
Traceback (most recent call last):
...
ValueError: ...
```

class SubConstraint (*template=None, argument=None, parent=None*)Bases: `pyffi.object_models.xml.struct_.StructBase`**property ball_and_socket****property entities****property hinge****property limited_hinge****property num_entities****property priority****property prismatic****property ragdoll****property stiff_spring****property type****class SymmetryType** (***kwargs*)Bases: `pyffi.object_models.xml.enum.EnumBase`Determines symmetry type used by `NiPSysBombModifier`.**CYLINDRICAL_SYMMETRY = 1****PLANAR_SYMMETRY = 2****SPHERICAL_SYMMETRY = 0****class SyncPoint** (***kwargs*)Bases: `pyffi.object_models.xml.enum.EnumBase`

Specifies the time when an application must synchronize for some reason.


```

    SYNC_ANY = 32768
    SYNC_PHYSICS_COMPLETED = 32864
    SYNC_PHYSICS_SIMULATE = 32848
    SYNC_POST_UPDATE = 32800
    SYNC_REFLECTIONS = 32880
    SYNC_RENDER = 32832
    SYNC_UPDATE = 32784
    SYNC_VISIBLE = 32816

class TBC (template=None, argument=None, parent=None)
    Bases: pyffi.object_models.xml.struct_.StructBase
    Tension, bias, continuity.
    property b
    property c
    property t

class TallGrassShaderProperty (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.BSShaderProperty
    Bethesda-specific node.
    property file_name

class TargetColor (**kwargs)
    Bases: pyffi.object_models.xml.enum.EnumBase
    Used by NiPoint3InterpControllers to select which type of color in the controlled object that will be animated.
    TC_AMBIENT = 0
    TC_DIFFUSE = 1
    TC_SELF_ILLUM = 3
    TC_SPECULAR = 2

class TexClampMode (**kwargs)
    Bases: pyffi.object_models.xml.enum.EnumBase
    Specifies the available texture clamp modes. That is, the behavior of pixels outside the range of the texture.
    CLAMP_S_CLAMP_T = 0
    CLAMP_S_WRAP_T = 1
    WRAP_S_CLAMP_T = 2
    WRAP_S_WRAP_T = 3

class TexCoord (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif._TexCoord, object
    as_list()
    normalize()

```

```
class TexDesc (template=None, argument=None, parent=None)
    Bases: pyffi.object_models.xml.struct_.StructBase
    Texture description.

    property center_offset
    property clamp_mode
    property filter_mode
    property flags
    property has_texture_transform
    property ps_2_k
    property ps_2_l
    property source
    property tiling
    property transform_type
    property translation
    property unknown_1
    property unknown_short
    property uv_set
    property w_rotation

class TexFilterMode (**kwargs)
    Bases: pyffi.object_models.xml.enum.EnumBase
    Specifies the available texture filter modes. That is, the way pixels within a texture are blended together
    when textures are displayed on the screen at a size other than their original dimensions.

    FILTER_BILERP = 1
    FILTER_BILERP_MIPNEAREST = 5
    FILTER_NEAREST = 0
    FILTER_NEAREST_MIPLERP = 4
    FILTER_NEAREST_MIPNEAREST = 3
    FILTER_TRILERP = 2

class TexSource (template=None, argument=None, parent=None)
    Bases: pyffi.object_models.xml.struct_.StructBase
    A texture source.

    property file_name
    property pixel_data
    property unknown_byte
    property unknown_link
    property use_external
```

```

class TextTransform(**kwargs)
    Bases: pyffi.object_models.xml.enum.EnumBase
    Determines how a NiTextureTransformController animates the UV coordinates.

    TT_ROTATE = 2
    TT_SCALE_U = 3
    TT_SCALE_V = 4
    TT_TRANSLATE_U = 0
    TT_TRANSLATE_V = 1

class TextureType(**kwargs)
    Bases: pyffi.object_models.xml.enum.EnumBase
    The type of texture.

    BASE_MAP = 0
    BUMP_MAP = 5
    DARK_MAP = 1
    DECAL_0_MAP = 8
    DECAL_1_MAP = 9
    DECAL_2_MAP = 10
    DECAL_3_MAP = 11
    DETAIL_MAP = 2
    GLOSS_MAP = 3
    GLOW_MAP = 4
    NORMAL_MAP = 6
    UNKNOWN2_MAP = 7

class TileShaderProperty(template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.BSShaderLightingProperty
    Bethesda-specific node.

    property file_name

class Triangle(template=None, argument=None, parent=None)
    Bases: pyffi.object_models.xml.struct_.StructBase
    List of three vertex indices.

    property v_1
    property v_2
    property v_3

class UnionBV(template=None, argument=None, parent=None)
    Bases: pyffi.object_models.xml.struct_.StructBase
    property bounding_volumes
    property num_bv

```

```
class Vector3 (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif._Vector3, object

    as_list ()
    as_tuple ()
    assign (vec)
        Set this vector to values from another object that supports iteration or x,y,z properties
    crossproduct (x)
    get_copy ()
    norm (sqrt=<built-in function sqrt>)
    normalize (ignore_error=False, sqrt=<built-in function sqrt>)
    normalized (ignore_error=False)
```

```
class Vector4 (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif._Vector4, object
```

```
>>> from pyffi.formats.nif import NifFormat
>>> vec = NifFormat.Vector4 ()
>>> vec.x = 1.0
>>> vec.y = 2.0
>>> vec.z = 3.0
>>> vec.w = 4.0
>>> print (vec)
[ 1.000  2.000  3.000  4.000 ]
>>> vec.as_list ()
[1.0, 2.0, 3.0, 4.0]
>>> vec.as_tuple ()
(1.0, 2.0, 3.0, 4.0)
>>> print (vec.get_vector_3 ())
[ 1.000  2.000  3.000 ]
>>> vec2 = NifFormat.Vector4 ()
>>> vec == vec2
False
>>> vec2.x = 1.0
>>> vec2.y = 2.0
>>> vec2.z = 3.0
>>> vec2.w = 4.0
>>> vec == vec2
True
```

```
as_list ()
as_tuple ()
get_copy ()
get_vector_3 ()
```

```
class VelocityType (**kwargs)
    Bases: pyffi.object_models.xml.enum.EnumBase
```

Controls the way the a particle mesh emitter determines the starting speed and direction of the particles that are emitted.

```
VELOCITY_USE_DIRECTION = 2
VELOCITY_USE_NORMALS = 0
```

```

    VELOCITY_USE_RANDOM = 1

class VertMode (**kwargs)
    Bases: pyffi.object_models.xml.enum.EnumBase
    An unsigned 32-bit integer, which describes how to apply vertex colors.

    VERT_MODE_SRC_AMB_DIF = 2
    VERT_MODE_SRC_EMISSIVE = 1
    VERT_MODE_SRC_IGNORE = 0

class VolumetricFogShaderProperty (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.BSShaderProperty
    Bethesda-specific node.

class WaterShaderProperty (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.BSShaderProperty
    Bethesda-specific node? Found in Fallout3

class ZCompareMode (**kwargs)
    Bases: pyffi.object_models.xml.enum.EnumBase
    This enum contains the options for doing z buffer tests.

    ZCOMP_ALWAYS = 0
    ZCOMP_EQUAL = 2
    ZCOMP_GREATER = 4
    ZCOMP_GREATER_EQUAL = 6
    ZCOMP_LESS = 1
    ZCOMP_LESS_EQUAL = 3
    ZCOMP_NEVER = 7
    ZCOMP_NOT_EQUAL = 5

class bhkAabbPhantom (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.bhkShapePhantom
    Bethesda-specific node.

    property unknown_ints_1

class bhkBallAndSocketConstraint (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.bhkConstraint
    A Ball and Socket Constraint.

    property ball_and_socket

class bhkBallSocketConstraintChain (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.bhkSerializable
    A Ball and Socket Constraint chain.

    property floats_1
    property links
    property links_2

```

```
property num_floats
property num_links
property num_links_2
property unknown_float_1
property unknown_float_2
property unknown_int_1
property unknown_int_2
property unknown_int_3

class bhkBlendCollisionObject (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.bhkCollisionObject
    Unknown.
    property unknown_float_1
    property unknown_float_2

class bhkBlendController (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiTimeController
    Unknown. Is apparently only used in skeleton.nif files.
    property unknown_int

class bhkBoxShape (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif._bhkBoxShape, object
    apply_scale (scale)
        Apply scale factor C{scale} on data.
    get_mass_center_inertia (density=1, solid=True)
        Return mass, center, and inertia tensor.

class bhkBreakableConstraint (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.bhkConstraint
    A breakable constraint.
    property remove_if_broken
    property sub_constraint
    property threshold

class bhkBvTreeShape (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.bhkShape
    A tree-like Havok data structure stored in an assembly-like binary code?

class bhkCMSDBigTris (template=None, argument=None, parent=None)
    Bases: pyffi.object_models.xml.struct_.StructBase
    Triangle indices used in pair with “Big Verts” in a bhkCompressedMeshShapeData.
    property triangle_1
    property triangle_2
    property triangle_3
    property unknown_int_1
```

```

    property unknown_short_1
class bhkCMSDChunk (template=None, argument=None, parent=None)
    Bases: pyffi.object_models.xml.struct_.StructBase
    Defines subshape chunks in bhkCompressedMeshShapeData
    property indices
    property indices_2
    property material_index
    property num_indices
    property num_indices_2
    property num_strips
    property num_vertices
    property strips
    property transform_index
    property translation
    property unknown_short_1
    property vertices
class bhkCMSDMaterial (template=None, argument=None, parent=None)
    Bases: pyffi.object_models.xml.struct_.StructBase
    Per-chunk material, used in bhkCompressedMeshShapeData
    property byte_set_to_0
    property layer
    property material
    property short_set_to_0
class bhkCMSDTransform (template=None, argument=None, parent=None)
    Bases: pyffi.object_models.xml.struct_.StructBase
    A set of transformation data: translation and rotation
    property rotation
    property translation
class bhkCapsuleShape (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif._bhkCapsuleShape, object
    apply_scale (scale)
        Apply scale factor <scale> on data.
    get_mass_center_inertia (density=1, solid=True)
        Return mass, center, and inertia tensor.
class bhkCollisionObject (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.bhkNiCollisionObject
    Havok related collision object?

```

```
class bhkCompressedMeshShape (template=None, argument=None, parent=None)
```

```
    Bases: pyffi.formats.nif.bhkShape
```

```
    Compressed collision mesh.
```

```
    property data
```

```
    property radius
```

```
    property scale
```

```
    property target
```

```
    property unknown_4_bytes
```

```
    property unknown_float_1
```

```
    property unknown_float_3
```

```
    property unknown_float_4
```

```
    property unknown_float_5
```

```
    property unknown_floats_1
```

```
    property unknown_int_1
```

```
class bhkCompressedMeshShapeData (template=None, argument=None, parent=None)
```

```
    Bases: pyffi.formats.nif.bhkRefObject
```

```
    A compressed mesh shape for collision in Skyrim.
```

```
    property big_tris
```

```
    property big_verts
```

```
    property bits_per_index
```

```
    property bits_per_w_index
```

```
    property bounds_max
```

```
    property bounds_min
```

```
    property chunk_materials
```

```
    property chunk_transforms
```

```
    property chunks
```

```
    property error
```

```
    property mask_index
```

```
    property mask_w_index
```

```
    property num_big_tris
```

```
    property num_big_verts
```

```
    property num_chunks
```

```
    property num_materials
```

```
    property num_transforms
```

```
    property unknown_byte_1
```

```
    property unknown_byte_2
```

```
    property unknown_int_12
```



```

property unknown_int_3
property unknown_int_4
property unknown_int_5
property unknown_int_6

```

class bhkConstraint (*template=None, argument=None, parent=None*)
 Bases: `pyffi.formats.nif._bhkConstraint`, `object`

get_transform_a_b (*parent*)
 Returns the transform of the first entity relative to the second entity. Root is simply a nif block that is a common parent to both blocks.

class bhkConvexListShape (*template=None, argument=None, parent=None*)
 Bases: `pyffi.formats.nif.bhkShape`

A havok shape. A list of convex shapes.

Do not put a `bhkPackedNiTriStripsShape` in the Sub Shapes. Use a separate collision nodes without a list shape for those.

Also, shapes collected in a `bhkListShape` may not have the correct walking noise, so only use it for non-walkable objects.

```

property material
property num_sub_shapes
property sub_shapes
property unknown_byte_1
property unknown_float_1
property unknown_floats

```

class bhkConvexShape (*template=None, argument=None, parent=None*)
 Bases: `pyffi.formats.nif.bhkSphereRepShape`

A havok shape.

class bhkConvexTransformShape (*template=None, argument=None, parent=None*)
 Bases: `pyffi.formats.nif.bhkTransformShape`

A convex transformed shape?

class bhkConvexVerticesShape (*template=None, argument=None, parent=None*)
 Bases: `pyffi.formats.nif._bhkConvexVerticesShape`, `object`

apply_scale (*scale*)
 Apply scale factor on data.

get_mass_center_inertia (*density=1, solid=True*)
 Return mass, center, and inertia tensor.

class bhkEntity (*template=None, argument=None, parent=None*)
 Bases: `pyffi.formats.nif.bhkWorldObject`

A havok node, describes physical properties.

class bhkHingeConstraint (*template=None, argument=None, parent=None*)
 Bases: `pyffi.formats.nif.bhkConstraint`

A hinge constraint.

```
    property hinge

class bhkLimitedHingeConstraint (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif._bhkLimitedHingeConstraint, object

    apply_scale (scale)
        Scale data.

    update_a_b (parent)
        Update the B data from the A data. The parent argument is simply a common parent to the entities.

class bhkLiquidAction (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.bhkSerializable

    Bethesda-specific node.

    property unknown_float_1
    property unknown_float_2
    property unknown_float_3
    property unknown_float_4
    property unknown_int_1
    property unknown_int_2
    property unknown_int_3

class bhkListShape (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif._bhkListShape, object

    add_shape (shape, front=False)
        Add shape to list.

    get_mass_center_inertia (density=1, solid=True)
        Return center of gravity and area.

    remove_shape (shape)
        Remove a shape from the shape list.

class bhkMalleableConstraint (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif._bhkMalleableConstraint, object

    apply_scale (scale)
        Scale data.

    update_a_b (parent)
        Update the B data from the A data.

class bhkMeshShape (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.bhkShape

    property num_strips_data
    property num_unknown_floats
    property strips_data
    property unknown_1
    property unknown_2
    property unknown_floats
```

```

class bhkMoppBvTreeShape (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif._bhkMoppBvTreeShape, object

    get_mass_center_inertia (density=1, solid=True)
        Return mass, center of gravity, and inertia tensor.

    mopp_from_tree (tree)

    parse_mopp (start=0, depth=0, toffset=0, verbose=False)
        The mopp data is printed to the debug channel while parsed. Returns list of indices into mopp data of
        the bytes processed and a list of triangle indices encountered.

        The verbose argument is ignored (and is deprecated).

    split_triangles (ts, bbox, dir=0)
        Direction 0=X, 1=Y, 2=Z

    update_mopp ()
        Update the MOPP data, scale, and origin, and welding info.

        @deprecated: use update_mopp_welding instead

    update_mopp_welding ()
        Update the MOPP data, scale, and origin, and welding info.

    update_origin_scale ()
        Update scale and origin.

class bhkMultiSphereShape (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif._bhkMultiSphereShape, object

    get_mass_center_inertia (density=1, solid=True)
        Return center of gravity and area.

class bhkNiCollisionObject (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiCollisionObject

    Havok related collision object?

    property body
    property flags

class bhkNiTriStripsShape (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif._bhkNiTriStripsShape, object

    get_interchangeable_packed_shape ()
        Returns a bhkPackedNiTriStripsShape block that is geometrically interchangeable.

    get_mass_center_inertia (density=1, solid=True)
        Return mass, center, and inertia tensor.

class bhkOrientHingedBodyAction (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.bhkSerializable

    Bethesda-Specific node.

    property unknown_ints_1

class bhkPCollisionObject (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.bhkNiCollisionObject

    Unknown.

class bhkPackedNiTriStripsShape (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif._bhkPackedNiTriStripsShape, object

```

add_shape (*triangles, normals, vertices, layer=0, material=0*)

Pack the given geometry.

get_mass_center_inertia (*density=1, solid=True*)

Return mass, center, and inertia tensor.

get_sub_shapes ()

Return sub shapes (works for both Oblivion and Fallout 3).

get_triangle_hash_generator ()

Generator which produces a tuple of integers, or None in degenerate case, for each triangle to ease detection of duplicate triangles.

```
>>> shape = NifFormat.bhkPackedNiTriStripsShape()
>>> data = NifFormat.hkPackedNiTriStripsData()
>>> shape.data = data
>>> data.num_triangles = 6
>>> data.triangles.update_size()
>>> data.triangles[0].triangle.v_1 = 0
>>> data.triangles[0].triangle.v_2 = 1
>>> data.triangles[0].triangle.v_3 = 2
>>> data.triangles[1].triangle.v_1 = 2
>>> data.triangles[1].triangle.v_2 = 1
>>> data.triangles[1].triangle.v_3 = 3
>>> data.triangles[2].triangle.v_1 = 3
>>> data.triangles[2].triangle.v_2 = 2
>>> data.triangles[2].triangle.v_3 = 1
>>> data.triangles[3].triangle.v_1 = 3
>>> data.triangles[3].triangle.v_2 = 1
>>> data.triangles[3].triangle.v_3 = 2
>>> data.triangles[4].triangle.v_1 = 0
>>> data.triangles[4].triangle.v_2 = 0
>>> data.triangles[4].triangle.v_3 = 3
>>> data.triangles[5].triangle.v_1 = 1
>>> data.triangles[5].triangle.v_2 = 3
>>> data.triangles[5].triangle.v_3 = 4
>>> list(shape.get_triangle_hash_generator())
[(0, 1, 2), (1, 3, 2), (1, 3, 2), (1, 2, 3), None, (1, 3, 4)]
```

Returns A generator yielding a hash value for each triangle.

get_vertex_hash_generator (*vertexprecision=3, subshape_index=None*)

Generator which produces a tuple of integers for each vertex to ease detection of duplicate/close enough to remove vertices. The precision parameter denote number of significant digits behind the comma.

For vertexprecision, 3 seems usually enough (maybe we'll have to increase this at some point).

```
>>> shape = NifFormat.bhkPackedNiTriStripsShape()
>>> data = NifFormat.hkPackedNiTriStripsData()
>>> shape.data = data
>>> shape.num_sub_shapes = 2
>>> shape.sub_shapes.update_size()
>>> data.num_vertices = 3
>>> shape.sub_shapes[0].num_vertices = 2
>>> shape.sub_shapes[1].num_vertices = 1
>>> data.vertices.update_size()
>>> data.vertices[0].x = 0.0
>>> data.vertices[0].y = 0.1
```

(continues on next page)

(continued from previous page)

```

>>> data.vertices[0].z = 0.2
>>> data.vertices[1].x = 1.0
>>> data.vertices[1].y = 1.1
>>> data.vertices[1].z = 1.2
>>> data.vertices[2].x = 2.0
>>> data.vertices[2].y = 2.1
>>> data.vertices[2].z = 2.2
>>> list(shape.get_vertex_hash_generator())
[(0, (0, 100, 200)), (0, (1000, 1100, 1200)), (1, (2000, 2100, 2200))]
>>> list(shape.get_vertex_hash_generator(subshape_index=0))
[(0, 100, 200), (1000, 1100, 1200)]
>>> list(shape.get_vertex_hash_generator(subshape_index=1))
[(2000, 2100, 2200)]

```

Parameters `vertexprecision` (*float*) – Precision to be used for vertices.

Returns A generator yielding a hash value for each vertex.

class `bhkPhantom` (*template=None, argument=None, parent=None*)

Bases: `pyffi.formats.nif.bhkWorldObject`

Havok object that do not react with other objects when they collide (causing deflection, etc.) but still trigger collision notifications to the game. Possible uses are traps, portals, AI fields, etc.

class `bhkPrismaticConstraint` (*template=None, argument=None, parent=None*)

Bases: `pyffi.formats.nif.bhkConstraint`

A prismatic constraint.

property `prismatic`

class `bhkRDTConstraint` (*template=None, argument=None, parent=None*)

Bases: `pyffi.object_models.xml.struct_.StructBase`

property `entity_a`

property `entity_b`

property `malleable_constraint`

property `priority`

property `ragdoll`

property `type`

property `unknown_int`

class `bhkRDTMalleableConstraint` (*template=None, argument=None, parent=None*)

Bases: `pyffi.object_models.xml.struct_.StructBase`

A malleable constraint.

property `damping`

property `entity_a`

property `entity_b`

property `hinge`

property `limited_hinge`

property `priority`

property `ragdoll`

```
    property type
    property unknown_int

class bhkRagdollConstraint (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif._bhkRagdollConstraint, object

    apply_scale (scale)
        Scale data.

    update_a_b (parent)
        Update the B data from the A data.

class bhkRagdollTemplate (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiObject

    Found in Fallout 3, more ragdoll info? (meshesragdollconstraint*.rdt)

    property bones
    property name
    property num_bones

class bhkRagdollTemplateData (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.NiObject

    Data for bhkRagdollTemplate

    property constraint
    property flag_or_num_constraints
    property friction
    property mass
    property name
    property radius
    property restitution
    property unknown_int

class bhkRefObject (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif._bhkRefObject, object

    get_shape_mass_center_inertia (density=1, solid=True)
        Return mass, center of gravity, and inertia tensor of this object's shape, if self.shape is not None.

        If self.shape is None, then returns zeros for everything.

class bhkRigidBody (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif._bhkRigidBody, object

    apply_scale (scale)
        Apply scale factor <scale> on data.

    update_mass_center_inertia (density=1, solid=True, mass=None)
        Look at all the objects under this rigid body and update the mass, center of gravity, and inertia tensor
        accordingly. If the C{mass} parameter is given then the C{density} argument is ignored.

class bhkRigidBodyT (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.bhkRigidBody

    Unknown.
```

```

class bhkSPCollisionObject (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.bhkPCollisionObject

    Unknown.

class bhkSerializable (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.bhkRefObject

    Havok objects that can be saved and loaded from disk?

class bhkShape (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.bhkSerializable

    A Havok Shape?

class bhkShapeCollection (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.bhkShape

    Havok collision object that uses multiple shapes?

class bhkShapePhantom (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.bhkPhantom

    A Havok phantom that uses a Havok shape object for its collision volume instead of just a bounding box.

class bhkSimpleShapePhantom (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.bhkShapePhantom

    Unknown shape.

    property unknown_float

    property unknown_floats_2

    property unknown_floats

class bhkSphereRepShape (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.bhkShape

    A havok shape, perhaps with a bounding sphere for quick rejection in addition to more detailed shape data?

    property material

    property radius

class bhkSphereShape (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif._bhkSphereShape, object

    apply_scale (scale)
        Apply scale factor <scale> on data.

    get_mass_center_inertia (density=1, solid=True)
        Return mass, center, and inertia tensor.

class bhkStiffSpringConstraint (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.bhkConstraint

    A spring constraint.

    property stiff_spring

class bhkTransformShape (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif._bhkTransformShape, object

    apply_scale (scale)
        Apply scale factor <scale> on data.

```

```
get_mass_center_inertia (density=1, solid=True)
    Return mass, center, and inertia tensor.

class bhkWorldObject (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif.bhkSerializable

    Havok objects that have a position in the world?

    property havok_col_filter

    property shape

class bool (**kwargs)
    Bases: pyffi.object_models.xml.basic.BasicBase, pyffi.object_models.
    editable.EditableBoolComboBox
```

Basic implementation of a 32-bit (8-bit for versions > 4.0.0.2) boolean type.

```
>>> i = NifFormat.bool()
>>> i.set_value('false')
>>> i.get_value()
False
>>> i.set_value('true')
>>> i.get_value()
True
```

```
get_hash (data=None)
    Returns a hash value (an immutable object) that can be used to identify the object uniquely.

get_size (data=None)
    Returns size of the object in bytes.

get_value ()
    Return object value.

read (stream, data)
    Read object from file.

set_value (value)
    Set object value.

write (stream, data)
    Write object to file.

byte
    alias of pyffi.object_models.common.UByte

char
    alias of pyffi.object_models.common.Char

float
    alias of pyffi.object_models.common.Float

games = {'?': [167772419], 'Atlantica': [335675400], 'Axis and Allies': [167837696]}

class hkConstraintType (**kwargs)
    Bases: pyffi.object_models.xml.enum.EnumBase

    The type of constraint.

    BallAndSocket = 0

    Hinge = 1

    Prismatic = 6
```



```

    Ragdoll = 7
    StiffSpring = 8
class hkPackedNiTriStripsData (template=None, argument=None, parent=None)
    Bases: pyffi.formats.nif._hkPackedNiTriStripsData, object
    apply_scale (scale)
        Apply scale factor on data.
class hkResponseType (**kwargs)
    Bases: pyffi.object_models.xml.enum.EnumBase
    RESPONSE_INVALID = 0
    RESPONSE_NONE = 3
    RESPONSE_REPORTING = 2
    RESPONSE_SIMPLE_CONTACT = 1
class hkTriangle (template=None, argument=None, parent=None)
    Bases: pyffi.object_models.xml.struct_.StructBase
    A triangle with extra data used for physics.
    property normal
    property triangle
    property welding_info
int
    alias of pyffi.object_models.common.Int
class physXMMaterialRef (template=None, argument=None, parent=None)
    Bases: pyffi.object_models.xml.struct_.StructBase
    property material_desc
    property number
    property unknown_byte_1
short
    alias of pyffi.object_models.common.Short
class string (**kwargs)
    Bases: pyffi.object_models.common.SizedString
    get_hash (data=None)
        Return a hash value for this string.
        Returns An immutable object that can be used as a hash.
    get_size (data=None)
        Return number of bytes this type occupies in a file.
        Returns Number of bytes.
    get_strings (data)
        Return all strings used by this object.
    read (stream, data)
        Read string from stream.
        Parameters stream (file) – The stream to read from.

```

write (*stream*, *data*)

Write string to stream.

Parameters **stream** (*file*) – The stream to write to.

uint

alias of `pyffi.object_models.common.UInt`

ulittle32

alias of `pyffi.object_models.common.ULittle32`

ushort

alias of `pyffi.object_models.common.USHort`

static version_number (*version_str*)

Converts version string into an integer.

Parameters **version_str** (*str*) – The version string.

Returns A version integer.

```
>>> hex(NifFormat.version_number('3.14.15.29'))
'0x30e0f1d'
>>> hex(NifFormat.version_number('1.2'))
'0x1020000'
>>> hex(NifFormat.version_number('3.03'))
'0x3000300'
>>> hex(NifFormat.version_number('NS'))
'0xa010000'
```

```
versions = {'10.0.1.0': 167772416, '10.0.1.2': 167772418, '10.0.1.3': 167772419, '1
```

```
xml_alias = []
```

```
xml_bit_struct = [<bit_struct 'BSSegmentFlags'>, <bit_struct 'FurnitureEntryPoints'>, ]
```

```
xml_enum = [<enum 'AlphaFormat'>, <enum 'ApplyMode'>, <enum 'TexType'>, <enum 'KeyType
```

```
xml_file_name = 'nif.xml'
```

```
xml_file_path = [None, '/home/docs/checkouts/readthedocs.org/user_builds/pyffi/checkou
```

```
xml_struct = [<struct 'Color3'>, <struct 'ByteColor3'>, <struct 'Color4'>, <struct 'By
```

Regression tests

These tests are used to check for functionality and bugs in the library. They also provide code examples which you may find useful.

Read a NIF file

```
>>> from os.path import dirname
>>> dirpath = __file__
>>> for i in range(4): #recurse up to root repo dir
...     dirpath = dirname(dirpath)
>>> repo_root = dirpath
>>> format_root = os.path.join(repo_root, 'tests', 'spells', 'nif', 'files')
>>> stream = open(os.path.join(format_root, 'test.nif'), 'rb')
>>> data = NifFormat.Data()
```

(continues on next page)

(continued from previous page)

```

>>> # inspect is optional; it will not read the actual blocks
>>> data.inspect(stream)
>>> hex(data.version)
'0x14010003'
>>> data.user_version
0
>>> for blocktype in data.header.block_types:
...     print(blocktype.decode("ascii"))
NiNode
NiTriShape
NiTriShapeData
>>> data.roots # blocks have not been read yet, so this is an empty list
[]
>>> data.read(stream)
>>> for root in data.roots:
...     for block in root.tree():
...         if isinstance(block, NifFormat.NiNode):
...             print(block.name.decode("ascii"))
test
>>> stream.close()

```

Parse all NIF files in a directory tree

```

>>> for stream, data in NifFormat.walkData(format_root):
...     try:
...         # the replace call makes the doctest also pass on windows
...         os_path = stream.name
...         split = (os_path.split(os.sep))[-5:]
...         rejoin = os.path.join(*split).replace(os.sep, "/")
...         print("reading %s" % rejoin)
...         data.read(stream)
...     except Exception:
...         print(
...             "Warning: read failed due corrupt file,"
...             " corrupt format description, or bug.")
reading tests/spells/nif/files/invalid.nif
Warning: read failed due corrupt file, corrupt format description, or bug.
reading tests/spells/nif/files/nds.nif
reading tests/spells/nif/files/neosteam.nif
reading tests/spells/nif/files/test.nif
reading tests/spells/nif/files/test_centerradius.nif
reading tests/spells/nif/files/test_check_tangentspace1.nif
reading tests/spells/nif/files/test_check_tangentspace2.nif
reading tests/spells/nif/files/test_check_tangentspace3.nif
reading tests/spells/nif/files/test_check_tangentspace4.nif
reading tests/spells/nif/files/test_convexverticesshape.nif
reading tests/spells/nif/files/test_dump_tex.nif
reading tests/spells/nif/files/test_fix_clampmaterialalpha.nif
reading tests/spells/nif/files/test_fix_cleanstringpalette.nif
reading tests/spells/nif/files/test_fix_detachhavoktristripsdata.nif
reading tests/spells/nif/files/test_fix_disableparallax.nif
reading tests/spells/nif/files/test_fix_ffvt3rskinpartition.nif
reading tests/spells/nif/files/test_fix_mergeskeletonroots.nif
reading tests/spells/nif/files/test_fix_tangentspace.nif

```

(continues on next page)

(continued from previous page)

```

reading tests/spells/nif/files/test_fix_texturepath.nif
reading tests/spells/nif/files/test_grid_128x128.nif
reading tests/spells/nif/files/test_grid_64x64.nif
reading tests/spells/nif/files/test_mopp.nif
reading tests/spells/nif/files/test_opt_collision_complex_mopp.nif
reading tests/spells/nif/files/test_opt_collision_mopp.nif
reading tests/spells/nif/files/test_opt_collision_packed.nif
reading tests/spells/nif/files/test_opt_collision_to_boxshape.nif
reading tests/spells/nif/files/test_opt_collision_to_boxshape_notabox.nif
reading tests/spells/nif/files/test_opt_collision_unpacked.nif
reading tests/spells/nif/files/test_opt_delunusedbones.nif

```

```

reading tests/spells/nif/files/test_opt_dupverts.nif reading tests/spells/nif/files/test_opt_emptyproperties.nif read-
ing tests/spells/nif/files/test_opt_grid_layout.nif reading tests/spells/nif/files/test_opt_mergeduplicates.nif read-
ing tests/spells/nif/files/test_opt_vertex_cache.nif reading tests/spells/nif/files/test_opt_zeryscale.nif reading
tests/spells/nif/files/test_skincenterradius.nif reading tests/spells/nif/files/test_vertexcolor.nif

```

Create a NIF model from scratch and write to file

```

>>> root = NifFormat.NiNode()
>>> root.name = 'Scene Root'
>>> blk = NifFormat.NiNode()
>>> root.add_child(blk)
>>> blk.name = 'new block'
>>> blk.scale = 2.4
>>> blk.translation.x = 3.9
>>> blk.rotation.m_11 = 1.0
>>> blk.rotation.m_22 = 1.0
>>> blk.rotation.m_33 = 1.0
>>> ctrl = NifFormat.NiVisController()
>>> ctrl.flags = 0x000c
>>> ctrl.target = blk
>>> blk.add_controller(ctrl)
>>> blk.add_controller(NifFormat.NiAlphaController())
>>> strips = NifFormat.NiTriStrips()
>>> root.add_child(strips, front = True)
>>> strips.name = "hello world"
>>> strips.rotation.m_11 = 1.0
>>> strips.rotation.m_22 = 1.0
>>> strips.rotation.m_33 = 1.0
>>> data = NifFormat.NiTriStripsData()
>>> strips.data = data
>>> data.num_vertices = 5
>>> data.has_vertices = True
>>> data.vertices.update_size()
>>> for i, v in enumerate(data.vertices):
...     v.x = 1.0+i/10.0
...     v.y = 0.2+1.0/(i+1)
...     v.z = 0.03
>>> data.update_center_radius()
>>> data.num_strips = 2
>>> data.strip_lengths.update_size()
>>> data.strip_lengths[0] = 3
>>> data.strip_lengths[1] = 4
>>> data.has_points = True

```

(continues on next page)

(continued from previous page)

```

>>> data.points.update_size()
>>> data.points[0][0] = 0
>>> data.points[0][1] = 1
>>> data.points[0][2] = 2
>>> data.points[1][0] = 1
>>> data.points[1][1] = 2
>>> data.points[1][2] = 3
>>> data.points[1][3] = 4
>>> data.num_uv_sets = 1
>>> data.has_uv = True
>>> data.uv_sets.update_size()
>>> for i, v in enumerate(data.uv_sets[0]):
...     v.u = 1.0-i/10.0
...     v.v = 1.0/(i+1)
>>> data.has_normals = True
>>> data.normals.update_size()
>>> for i, v in enumerate(data.normals):
...     v.x = 0.0
...     v.y = 0.0
...     v.z = 1.0
>>> strips.update_tangent_space()
>>> from tempfile import TemporaryFile
>>> stream = TemporaryFile()
>>> nifdata = NifFormat.Data(version=0x14010003, user_version=10)
>>> nifdata.roots = [root]
>>> nifdata.write(stream)
>>> stream.close()

```

Get list of versions and games

```

>>> for vnum in sorted(NifFormat.versions.values()):
...     print('0x%08X' % vnum)
0x02030000
0x03000000
0x03000300
0x03010000
0x0303000D
0x04000000
0x04000002
0x0401000C
0x04020002
0x04020100
0x04020200
0x0A000100
0x0A000102
0x0A000103
0x0A010000
0x0A010065
0x0A01006A
0x0A020000
0x0A020001
0x0A040001
0x14000004
0x14000005

```

(continues on next page)

(continued from previous page)

```

0x14010003
0x14020007
0x14020008
0x14030001
0x14030002
0x14030003
0x14030006
0x14030009
0x14050000
0x14060000
0x14060500
0x1E000002
0x1E010003
>>> for game, versions in sorted(NifFormat.games.items(), key=lambda x: x[0]):
...     print("%s " % game + " ".join('0x%08X' % vnum for vnum in versions))
? 0x0A000103
Atlantica 0x14020008
Axis and Allies 0x0A010000
Bully SE 0x14030009
Civilization IV 0x04020002 0x04020100 0x04020200 0x0A000100 0x0A010000 0x0A020000_
↳0x14000004
Culpa Innata 0x04020200
Dark Age of Camelot 0x02030000 0x03000300 0x03010000 0x0401000C 0x04020100 0x04020200_
↳0x0A010000
Divinity 2 0x14030009
Emerge 0x14020007 0x14020008 0x14030001 0x14030002 0x14030003 0x14030006 0x1E000002
Empire Earth II 0x04020200 0x0A010000
Empire Earth III 0x14020007 0x14020008
Entropia Universe 0x0A010000
Epic Mickey 0x14060500
Fallout 3 0x14020007
Freedom Force 0x04000000 0x04000002
Freedom Force vs. the 3rd Reich 0x0A010000
Howling Sword 0x14030009
Kohan 2 0x0A010000
KrazyRain 0x14050000 0x14060000
Lazeska 0x14030009
Loki 0x0A020000
Megami Tensei: Imagine 0x14010003
Morrowind 0x04000002
NeoSteam 0x0A010000
Oblivion 0x0303000D 0x0A000100 0x0A000102 0x0A010065 0x0A01006A 0x0A020000 0x14000004_
↳0x14000005
Prison Tycoon 0x0A020000
Pro Cycling Manager 0x0A020000
Red Ocean 0x0A020000
Rocksmith 0x1E010003
Rocksmith 2014 0x1E010003
Sid Meier's Railroads 0x14000004
Skyrim 0x14020007
Star Trek: Bridge Commander 0x03000000 0x03010000
The Guild 2 0x0A010000
Warhammer 0x14030009
Wildlife Park 2 0x0A010000 0x0A020000
Worldshift 0x0A020001 0x0A040001
Zoo Tycoon 2 0x0A000100

```

Reading an unsupported NIF file

```
>>> file = os.path.join(format_root, 'invalid.nif')
>>> stream = open(file, 'rb')
>>> data = NifFormat.Data()
>>> data.inspect(stream) # the file seems ok on inspection
>>> data.read(stream)
Traceback (most recent call last):
...
ValueError: ...
>>> stream.close()
```

Template types

```
>>> block = NifFormat.NiTextKeyExtraData()
>>> block.num_text_keys = 1
>>> block.text_keys.update_size()
>>> block.text_keys[0].time = 1.0
>>> block.text_keys[0].value = 'hi'
```

Links

```
>>> NifFormat.NiNode._has_links
True
>>> NifFormat.NiBone._has_links
True
>>> skelroot = NifFormat.NiNode()
>>> geom = NifFormat.NiTriShape()
>>> geom.skin_instance = NifFormat.NiSkinInstance()
>>> geom.skin_instance.skeleton_root = skelroot
>>> [block.__class__.__name__ for block in geom.get_refs()]
['NiSkinInstance']
>>> [block.__class__.__name__ for block in geom.get_links()]
['NiSkinInstance']
>>> [block.__class__.__name__ for block in geom.skin_instance.get_refs()]
[]
>>> [block.__class__.__name__ for block in geom.skin_instance.get_links()]
['NiNode']
```

Strings

```
>>> extra = NifFormat.NiTextKeyExtraData()
>>> extra.num_text_keys = 2
>>> extra.text_keys.update_size()
>>> extra.text_keys[0].time = 0.0
>>> extra.text_keys[0].value = "start"
>>> extra.text_keys[1].time = 2.0
>>> extra.text_keys[1].value = "end"
>>> for extrastr in extra.get_strings(None):
...     print(extrastr.decode("ascii"))
```

(continues on next page)

(continued from previous page)

```
start
end
```

`pyffi.formats.tga` — Targa (.tga)

Implementation

class `pyffi.formats.tga.TgaFormat`

Bases: `pyffi.object_models.xml.FileFormat`

This class implements the TGA format.

class `ColorMapType` (***kwargs*)

Bases: `pyffi.object_models.xml.enum.EnumBase`

An unsigned 8-bit integer, describing the color map type.

class `Data`

Bases: `pyffi.object_models.Data`

get_global_child_nodes (*edge_filter=(True, True)*)

Generator which yields all children of this item in the global view, of given edge type (default is edges of type 0).

Override this method.

Returns Generator for global node children.

inspect (*stream*)

Quick heuristic check if stream contains Targa data, by looking at the first 18 bytes.

Parameters **stream** (*file*) – The stream to inspect.

read (*stream*)

Read a tga file.

Parameters **stream** (*file*) – The stream from which to read.

write (*stream*)

Write a tga file.

Parameters **stream** (*file*) – The stream to write to.

class `FooterString` (*template=None, argument=None, parent=None*)

Bases: `pyffi.object_models.xml.basic.BasicBase`

The Targa footer signature.

get_hash (*data=None*)

Return a hash value for the signature.

Returns An immutable object that can be used as a hash.

get_size (*data=None*)

Return number of bytes that the signature occupies in a file.

Returns Number of bytes.

get_value ()

Get signature.

Returns The signature.

read (*stream, data*)

Read signature from stream.

Parameters *stream* (*file*) – The stream to read from.

set_value (*value*)
Set signature.

Parameters *value* (*str*) – The value to assign.

write (*stream*, *data*)
Write signature to stream.

Parameters *stream* (*file*) – The stream to read from.

class Image
Bases: `pyffi.utils.graph.GlobalNode`

get_detail_child_names (*edge_filter*=(*True*, *True*))
Generator which yields all child names of this item in the detail view.

Override this method if the node has children.

Returns Generator for detail tree child names.

Return type generator yielding `str`s

get_detail_child_nodes (*edge_filter*=(*True*, *True*))
Generator which yields all children of this item in the detail view (by default, all acyclic and active ones).

Override this method if the node has children.

Parameters *edge_filter* (`EdgeFilter` or type (`None`)) – The edge type to include.

Returns Generator for detail tree child nodes.

Return type generator yielding `DetailNodes`

class ImageType (***kwargs*)
Bases: `pyffi.object_models.xml.enum.EnumBase`

An unsigned 8-bit integer, describing the image type.

PixelData
alias of `pyffi.object_models.common.UndecodedData`

byte
alias of `pyffi.object_models.common.Byte`

char
alias of `pyffi.object_models.common.Char`

float
alias of `pyffi.object_models.common.Float`

int
alias of `pyffi.object_models.common.Int`

short
alias of `pyffi.object_models.common.Short`

ubyte
alias of `pyffi.object_models.common.UByte`

uint
alias of `pyffi.object_models.common.UInt`

ushort
alias of `pyffi.object_models.common.USHort`

Regression tests

Read a TGA file

```
>>> # check and read tga file
>>> import os
>>> from os.path import dirname
>>> dirpath = __file__
>>> for i in range(4): #recurse up to root repo dir
...     dirpath = dirname(dirpath)
>>> repo_root = dirpath
>>> format_root = os.path.join(repo_root, 'tests', 'formats', 'tga')
>>> file = os.path.join(format_root, 'test.tga').replace("\\", "/")
>>> stream = open(file, 'rb')
>>> data = TgaFormat.Data()
>>> data.inspect(stream)
>>> data.read(stream)
>>> stream.close()
>>> data.header.width
60
>>> data.header.height
20
```

Parse all TGA files in a directory tree

```
>>> for stream, data in TgaFormat.walkData(format_root):
...     try:
...         # the replace call makes the doctest also pass on windows
...         os_path = stream.name
...         split = (os_path.split(os.sep))[-4:]
...         rejoin = os.path.join(*split).replace("\\", "/")
...         print("reading %s" % rejoin)
...     except Exception:
...         print(
...             "Warning: read failed due corrupt file,"
...             " corrupt format description, or bug.")
reading tests/formats/tga/test.tga
reading tests/formats/tga/test_footer.tga
```

Create a TGA file from scratch and write to file

```
>>> data = TgaFormat.Data()
>>> from tempfile import TemporaryFile
>>> stream = TemporaryFile()
>>> data.write(stream)
>>> stream.close()
```

pyffi.formats.tri — TRI (.tri)

A .tri file contains facial expression data, that is, morphs for dynamic expressions such as smile, frown, and so on.

Implementation

```
class pyffi.formats.tri.TriFormat
```

Bases: `pyffi.object_models.xml.FileFormat`

This class implements the TRI format.

Data

alias of *Header*

```
class FileSignature (**kwargs)
```

Bases: `pyffi.object_models.xml.basic.BasicBase`

Basic type which implements the header of a TRI file.

```
get_detail_display()
```

Return an object that can be used to display the instance.

```
get_hash(data=None)
```

Return a hash value for this value.

Returns An immutable object that can be used as a hash.

```
get_size(data=None)
```

Return number of bytes the header string occupies in a file.

Returns Number of bytes.

```
read(stream, data)
```

Read header string from stream and check it.

Parameters *stream* (*file*) – The stream to read from.

```
write(stream, data)
```

Write the header string to stream.

Parameters *stream* (*file*) – The stream to write to.

```
class FileVersion (template=None, argument=None, parent=None)
```

Bases: `pyffi.object_models.xml.basic.BasicBase`

```
get_detail_display()
```

Return an object that can be used to display the instance.

```
get_hash(data=None)
```

Returns a hash value (an immutable object) that can be used to identify the object uniquely.

```
get_size(data=None)
```

Returns size of the object in bytes.

```
get_value()
```

Return object value.

```
read(stream, data)
```

Read object from file.

```
set_value(value)
```

Set object value.

```
write(stream, data)
```

Write object to file.

```
class Header (template=None, argument=None, parent=None)
    Bases: pyffi.formats.tri._Header, pyffi.object_models.Data

    A class to contain the actual tri data.

    add_modifier (name=None, relative_vertices=None)
        Add a modifier.

    add_morph (name=None, relative_vertices=None)
        Add a morph.

    get_global_child_nodes (edge_filter=(True, True))
        Generator which yields all children of this item in the global view, of given edge type (default is edges
        of type 0).

        Override this method.
        Returns Generator for global node children.

    inspect (stream)
        Quickly checks if stream contains TRI data, and reads everything up to the arrays.
        Parameters stream (file) – The stream to inspect.

    inspect_quick (stream)
        Quickly checks if stream contains TRI data, by looking at the first 8 bytes. Reads the signature and
        the version.
        Parameters stream (file) – The stream to inspect.

    read (stream)
        Read a tri file.
        Parameters stream (file) – The stream from which to read.

    write (stream)
        Write a tri file.
        Parameters stream (file) – The stream to which to write.

class ModifierRecord (template=None, argument=None, parent=None)
    Bases: pyffi.object_models.xml.struct_.StructBase

    A modifier replaces the vertices from the base model (Header.vertices) with those in
    Header.modifier_vertices. Note that Header.modifier_vertices counts up from the first modifier on-
    wards. For example, if you were to take the 4th vertex to be modified in the 2nd modifier and
    the size of the 1st modifier was 10 vertices, then you would need Header.modifier_vertices[14].
    Therefore, Header.num_modifier_vertices = sum(modifier.num_vertices_to_modify for modifier in
    Header.modifiers).

class MorphRecord (template=None, argument=None, parent=None)
    Bases: pyffi.formats.tri._MorphRecord, object
```

```
>>> # create morph with 3 vertices.
>>> morph = TriFormat.MorphRecord(argument=3)
>>> morph.set_relative_vertices(
...     [(3, 5, 2), (1, 3, 2), (-9, 3, -1)])
>>> # scale should be 9/32768.0 = 0.0002746...
>>> morph.scale
0.0002746...
>>> for vert in morph.get_relative_vertices():
...     print([int(1000 * x + 0.5) for x in vert])
[3000, 5000, 2000]
[1000, 3000, 2000]
[-8999, 3000, -999]
```

apply_scale (*scale*)
 Apply scale factor to data.

```
>>> # create morph with 3 vertices.
>>> morph = TriFormat.MorphRecord(argument=3)
>>> morph.set_relative_vertices(
...     [(3, 5, 2), (1, 3, 2), (-9, 3, -1)])
>>> morph.apply_scale(2)
>>> for vert in morph.get_relative_vertices():
...     print([int(1000 * x + 0.5) for x in vert])
[6000, 10000, 4000]
[2000, 6000, 4000]
[-17999, 6000, -1999]
```

class QuadFace (*template=None, argument=None, parent=None*)
 Bases: `pyffi.object_models.xml.struct_.StructBase`
 Not used by the Oblivion engine as it's tri based.

class SizedStringZ (***kwargs*)
 Bases: `pyffi.object_models.common.SizedString`

get_size (*data=None*)
 Return number of bytes this type occupies in a file.
Returns Number of bytes.

read (*stream, data*)
 Read string from stream.
Parameters **stream** (*file*) – The stream to read from.

write (*stream, data*)
 Write string to stream.
Parameters **stream** (*file*) – The stream to write to.

byte
 alias of `pyffi.object_models.common.Byte`

char
 alias of `pyffi.object_models.common.Char`

float
 alias of `pyffi.object_models.common.Float`

int
 alias of `pyffi.object_models.common.Int`

short
 alias of `pyffi.object_models.common.Short`

ubyte
 alias of `pyffi.object_models.common.UByte`

uint
 alias of `pyffi.object_models.common.UInt`

ushort
 alias of `pyffi.object_models.common.USHort`

static version_number (*version_str*)
 Converts version string into an integer.
Parameters **version_str** (*str*) – The version string.

Returns A version integer.

```
>>> TriFormat.version_number('003')
3
>>> TriFormat.version_number('XXX')
-1
```

Regression tests

Read a TRI file

```
>>> # check and read tri file
>>> from os.path import dirname
>>> dirpath = __file__
>>> for i in range(4): #recurse up to root repo dir
...     dirpath = dirname(dirpath)
>>> repo_root = dirpath
>>> format_root = os.path.join(repo_root, 'tests', 'formats', 'tri')
>>> file = os.path.join(format_root, 'mmouthxivilai.tri')
>>> stream = open(file, 'rb')
>>> data = TriFormat.Data()
>>> data.inspect(stream)
>>> # do some stuff with header?
>>> data.num_vertices
89
>>> data.num_tri_faces
215
>>> data.num_quad_faces
0
>>> data.num_uvs
89
>>> data.num_morphs
18
>>> data.read(stream)
>>> print([str(morph.name.decode("ascii")) for morph in data.morphs])
['Fear', 'Surprise', 'Aah', 'BigAah', 'BMP', 'ChJSh', 'DST', 'Eee', 'Eh', 'FV', 'I',
↪ 'K', 'N', 'Oh', 'OohQ', 'R', 'Th', 'W']
```

Parse all TRI files in a directory tree

```
>>> for stream, data in TriFormat.walkData(format_root):
...     try:
...         # the replace call makes the doctest also pass on windows
...         os_path = stream.name
...         split = (os_path.split(os.sep))[-4:]
...         rejoin = os.path.join(*split).replace(os.sep, "/")
...         print("reading %s" % rejoin)
...     except Exception:
...         print(
...             "Warning: read failed due corrupt file,"
...             " corrupt format description, or bug.")
reading tests/formats/tri/mmouthxivilai.tri
```

Create an TRI file from scratch and write to file

```
>>> data = TriFormat.Data()
>>> from tempfile import TemporaryFile
>>> stream = TemporaryFile()
>>> data.write(stream)
```

Adding new formats

This section tries to explain how you can implement your own format in pyffi.

Getting Started

Note that the files which make up the following example can all be found in the examples/simple directory of the source distribution of pyffi.

Suppose you have a simple file format, which consists of an integer, followed by a list of integers as many as described by the first integer. We start by creating an XML file, call it `simple.xml`, which describes this format in a way that pyffi can understand:

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE fileformat>
<fileformat version="1.0">
  <basic name="Int">A signed 32-bit integer.</basic>
  <struct name="Example">
    <add name="Num Integers" type="Int">
      Number of integers that follow.
    </add>
    <add name="Integers" type="Int" arr1="Num Integers">
      A list of integers.
    </add>
  </struct>
</fileformat>
```

What pyffi does is convert this simple XML description into Python classes which automatically can read and write the structure you've just described. Say this is the contents of `simple.py`:

```
import os
import pyffi.object_models.xml
import pyffi.object_models.common

class SimpleFormat(pyffi.object_models.xml.FileFormat):
    xml_file_name = 'simple.xml'
    xml_file_path = [ os.path.dirname(__file__) ]

    # basic types

    Int = pyffi.object_models.common.Int

    # extensions of generated types

    class Data(pyffi.object_models.xml.FileFormat.Data):
        def __init__(self):
            self.example = SimpleFormat.Example()
```

(continues on next page)

(continued from previous page)

```

def read(self, stream):
    self.example.read(stream, self)

def write(self, stream):
    self.example.write(stream, self)

class Example:
    def addInteger(self, x):
        self.numIntegers += 1
        self.integers.update_size()
        self.integers[self.numIntegers-1] = x

```

What happens in this piece of code?

- The `pyffi.object_models.xml.FileFormat` base class triggers the transformation of xml into Python classes; how these classes can be used will be explained further.
- The `xml_file_name` class attribute provides the name of the xml file that describes the structures we wish to generate. The `xml_file_path` attribute gives a list of locations of where to look for this file; in our case we have simply chosen to put `simple.xml` in the same directory as `simple.py`.
- The `SimpleFormat.Example` class provides the generated class with a function `addInteger()` in addition to the attributes `numIntegers` and `integers` which have been created from the XML.
- Finally, the `pyffi.object_models.common` module implements the most common basic types, such as integers, characters, and floats. In the above example we have taken advantage of `pyffi.object_models.common.Int`, which defines a signed 32-bit integer, exactly the type we need.

Reading and Writing Files

To read the contents of a file of the format described by `simple.xml`:

```

from simple import SimpleFormat
x = SimpleFormat.Data()
f = open('somefile.simple', 'rb')
x.read(f)
f.close()
print(x.example)

```

Or, to create a new file in this format:

```

from simple import SimpleFormat
x = SimpleFormat.Data()
x.example.num_integers = 5
x.example.integers.update_size()
x.example.integers[0] = 3
x.example.integers[1] = 1
x.example.integers[2] = 4
x.example.integers[3] = 1
x.example.integers[4] = 5
f = open('pi.simple', 'wb')
x.write(f)
f.close()

```


Further References

With the above simple example in mind, you may wish to browse through the source code of `pyffi.formats.cgf` or `pyffi.formats.nif` to see how pyffi works for more complex file formats.

pyffi.spells — High level file operations

Note: This module is based on wz’s NifTester module, although nothing of wz’s original code is left in this module.

A *toaster*, implemented by subclasses of the abstract *Toaster* class, walks over all files in a folder, and applies one or more transformations on each file. Such transformations are called *spells*, and are implemented by subclasses of the abstract *Spell* class.

A *spell* can also run independently of a *toaster* and be applied on a branch directly. The recommended way of doing this is via the `Spell.recurse()` method.

Supported spells

For format specific spells, refer to the corresponding module.

pyffi.spells.cgf — Crytek Geometry/Animation (.cgf/.cga) spells

Todo: Write documentation.

pyffi.spells.dds — DirectDraw Surface spells

There are no spells yet.

pyffi.spells.kfm — NetImmerse/Gamebryo Keyframe Motion (.kfm) spells

pyffi.spells.nif — NetImmerse/Gamebryo File/Keyframe (.nif/.kf/.kfa) spells

Module which contains all spells that check something in a NIF file.

Spells for dumping particular blocks from nifs.

pyffi.spells.nif.fix — spells to fix errors

Module which contains all spells that fix something in a nif.

Implementation

```
class pyffi.spells.nif.fix.SpellDelTangentSpace (toaster=None, data=None,
                                                stream=None)
```

Bases: `pyffi.spells.nif.NifSpell`

Delete tangentspace if it is present.

branchentry (*branch*)

Cast the spell on the given branch. First called with branch equal to data's children, then the grandchildren, and so on. The default implementation simply returns True.

Typically, you will override this function to perform an operation on a particular block type and/or to stop recursion at particular block types.

Parameters **branch** (`GlobalNode`) – The branch to cast the spell on.

Returns True if the children must be processed, False otherwise.

Return type bool

branchinspect (*branch*)

Like `_branchinspect()`, but for customization: can be overridden to perform an extra inspection (the default implementation always returns True).

Parameters **branch** (`GlobalNode`) – The branch to check.

Returns True if the branch must be processed, False otherwise.

Return type bool

datainspect ()

This is called after `pyffi.object_models.FileFormat.Data.inspect()` has been called, and before `pyffi.object_models.FileFormat.Data.read()` is called. Override this function for customization.

Returns True if the file must be processed, False otherwise.

Return type bool

```
class pyffi.spells.nif.fix.SpellAddTangentSpace (toaster=None, data=None,
                                                stream=None)
```

Bases: `pyffi.spells.nif.NifSpell`

Add tangentspace if none is present.

branchentry (*branch*)

Cast the spell on the given branch. First called with branch equal to data's children, then the grandchildren, and so on. The default implementation simply returns True.

Typically, you will override this function to perform an operation on a particular block type and/or to stop recursion at particular block types.

Parameters **branch** (`GlobalNode`) – The branch to cast the spell on.

Returns True if the children must be processed, False otherwise.

Return type bool

branchinspect (*branch*)

Like `_branchinspect()`, but for customization: can be overridden to perform an extra inspection (the default implementation always returns `True`).

Parameters **branch** (`GlobalNode`) – The branch to check.

Returns `True` if the branch must be processed, `False` otherwise.

Return type `bool`

datainspect ()

This is called after `pyffi.object_models.FileFormat.Data.inspect()` has been called, and before `pyffi.object_models.FileFormat.Data.read()` is called. Override this function for customization.

Returns `True` if the file must be processed, `False` otherwise.

Return type `bool`

```
class pyffi.spells.nif.fix.SpellFFVT3RSkinPartition (toaster=None,      data=None,
                                                    stream=None)
```

Bases: `pyffi.spells.nif.NifSpell`

Create or update skin partition, with settings that work for Freedom Force vs. The 3rd Reich.

branchentry (*branch*)

Cast the spell on the given branch. First called with `branch` equal to `data`'s children, then the grandchildren, and so on. The default implementation simply returns `True`.

Typically, you will override this function to perform an operation on a particular block type and/or to stop recursion at particular block types.

Parameters **branch** (`GlobalNode`) – The branch to cast the spell on.

Returns `True` if the children must be processed, `False` otherwise.

Return type `bool`

branchinspect (*branch*)

Like `_branchinspect()`, but for customization: can be overridden to perform an extra inspection (the default implementation always returns `True`).

Parameters **branch** (`GlobalNode`) – The branch to check.

Returns `True` if the branch must be processed, `False` otherwise.

Return type `bool`

datainspect ()

This is called after `pyffi.object_models.FileFormat.Data.inspect()` has been called, and before `pyffi.object_models.FileFormat.Data.read()` is called. Override this function for customization.

Returns `True` if the file must be processed, `False` otherwise.

Return type `bool`

```
class pyffi.spells.nif.fix.SpellFixTexturePath (toaster=None,      data=None,
                                                    stream=None)
```

Bases: `pyffi.spells.nif.fix.SpellParseTexturePath`

Fix the texture path. Transforms `0x0a` into `n` and `0x0d` into `r`. This fixes a bug in nifs saved with older versions of nifskope. Also transforms `/` into `.`. This fixes problems when packing files into a bsa archive. Also if the version is 20.0.0.4 or higher it will check for bad texture path form of e.g. `c:program filesbethsofttexturesfilepath.dds` and replace it with e.g. `texturesfilepath.dds`.

substitute (*old_path*)

Helper function to allow subclasses of this spell to change part of the path with minimum of code. This implementation returns path unmodified.

```
class pyffi.spells.nif.fix.SpellDetachHavokTriStripsData (*args, **kwargs)
```

Bases: `pyffi.spells.nif.NifSpell`

For NiTriStrips if their NiTriStripsData also occurs in a bhkNiTriStripsShape, make deep copy of data in havok. This is mainly useful as a preperation for other spells that act on NiTriStripsData, to ensure that the havok data remains untouched.

branchentry (*branch*)

Cast the spell on the given branch. First called with branch equal to data's children, then the grandchildren, and so on. The default implementation simply returns True.

Typically, you will override this function to perform an operation on a particular block type and/or to stop recursion at particular block types.

Parameters **branch** (GlobalNode) – The branch to cast the spell on.

Returns True if the children must be processed, False otherwise.

Return type bool

branchinspect (*branch*)

Like `_branchinspect()`, but for customization: can be overridden to perform an extra inspection (the default implementation always returns True).

Parameters **branch** (GlobalNode) – The branch to check.

Returns True if the branch must be processed, False otherwise.

Return type bool

dataentry ()

Called before all blocks are recursed. The default implementation simply returns True. You can access the data via `data`, and unlike in the `datainspect()` method, the full file has been processed at this stage.

Typically, you will override this function to perform a global operation on the file data.

Returns True if the children must be processed, False otherwise.

Return type bool

datainspect ()

This is called after `pyffi.object_models.FileFormat.Data.inspect()` has been called, and before `pyffi.object_models.FileFormat.Data.read()` is called. Override this function for customization.

Returns True if the file must be processed, False otherwise.

Return type bool

```
class pyffi.spells.nif.fix.SpellClampMaterialAlpha (toaster=None, data=None,
                                                    stream=None)
```

Bases: `pyffi.spells.nif.NifSpell`

Clamp corrupted material alpha values.

branchentry (*branch*)

Cast the spell on the given branch. First called with branch equal to data's children, then the grandchildren, and so on. The default implementation simply returns True.

Typically, you will override this function to perform an operation on a particular block type and/or to stop recursion at particular block types.

Parameters **branch** (GlobalNode) – The branch to cast the spell on.

Returns True if the children must be processed, False otherwise.

Return type bool

branchinspect (*branch*)

Like `_branchinspect()`, but for customization: can be overridden to perform an extra inspection (the default implementation always returns True).

Parameters **branch** (GlobalNode) – The branch to check.

Returns True if the branch must be processed, False otherwise.

Return type bool

datainspect ()

This is called after `pyffi.object_models.FileFormat.Data.inspect()` has been called, and before `pyffi.object_models.FileFormat.Data.read()` is called. Override this function for customization.

Returns True if the file must be processed, False otherwise.

Return type bool

```
class pyffi.spells.nif.fix.SpellSendGeometriesToBindPosition (toaster=None,
                                                             data=None,
                                                             stream=None)
```

Bases: `pyffi.spells.nif.SpellVisitSkeletonRoots`

Transform skinned geometries so similar bones have the same bone data, and hence, the same bind position, over all geometries.

skelrootentry (*branch*)

Do something with a skeleton root. Return value is ignored.

```
class pyffi.spells.nif.fix.SpellSendDetachedGeometriesToNodePosition (toaster=None,
                                                                       data=None,
                                                                       stream=None)
```

Bases: `pyffi.spells.nif.SpellVisitSkeletonRoots`

Transform geometries so each set of geometries that shares bones is aligned with the transform of the root bone of that set.

skelrootentry (*branch*)

Do something with a skeleton root. Return value is ignored.

```
class pyffi.spells.nif.fix.SpellSendBonesToBindPosition (toaster=None, data=None,
                                                         stream=None)
```

Bases: `pyffi.spells.nif.SpellVisitSkeletonRoots`

Transform bones so bone data agrees with bone transforms, and hence, all bones are in bind position.

skelrootentry (*branch*)

Do something with a skeleton root. Return value is ignored.

```
class pyffi.spells.nif.fix.SpellMergeSkeletonRoots (toaster=None, data=None,
                                                     stream=None)
```

Bases: `pyffi.spells.nif.NifSpell`

Merges skeleton roots in the NIF file so that no skeleton root has another skeleton root as child. Warns if merge is impossible (this happens if the global skin data of the geometry is not the unit transform).

branchentry (*branch*)

Cast the spell on the given branch. First called with `branch` equal to `data`'s children, then the grandchildren, and so on. The default implementation simply returns `True`.

Typically, you will override this function to perform an operation on a particular block type and/or to stop recursion at particular block types.

Parameters **branch** (`GlobalNode`) – The branch to cast the spell on.

Returns `True` if the children must be processed, `False` otherwise.

Return type `bool`

branchinspect (*branch*)

Like `_branchinspect()`, but for customization: can be overridden to perform an extra inspection (the default implementation always returns `True`).

Parameters **branch** (`GlobalNode`) – The branch to check.

Returns `True` if the branch must be processed, `False` otherwise.

Return type `bool`

dataentry ()

Called before all blocks are recursed. The default implementation simply returns `True`. You can access the data via `data`, and unlike in the `datainspect()` method, the full file has been processed at this stage.

Typically, you will override this function to perform a global operation on the file data.

Returns `True` if the children must be processed, `False` otherwise.

Return type `bool`

datainspect ()

This is called after `pyffi.object_models.FileFormat.Data.inspect()` has been called, and before `pyffi.object_models.FileFormat.Data.read()` is called. Override this function for customization.

Returns `True` if the file must be processed, `False` otherwise.

Return type `bool`

```
class pyffi.spells.nif.fix.SpellApplySkinDeformation (toaster=None, data=None, stream=None)
```

Bases: `pyffi.spells.nif.NifSpell`

Apply skin deformation to nif.

```
class pyffi.spells.nif.fix.SpellScale (toaster=None, data=None, stream=None)
```

Bases: `pyffi.spells.nif.NifSpell`

Scale a model.

branchentry (*branch*)

Cast the spell on the given branch. First called with `branch` equal to `data`'s children, then the grandchildren, and so on. The default implementation simply returns `True`.

Typically, you will override this function to perform an operation on a particular block type and/or to stop recursion at particular block types.

Parameters **branch** (`GlobalNode`) – The branch to cast the spell on.

Returns `True` if the children must be processed, `False` otherwise.

Return type `bool`

branchinspect (*branch*)

Like `_branchinspect()`, but for customization: can be overridden to perform an extra inspection (the default implementation always returns `True`).

Parameters **branch** (`GlobalNode`) – The branch to check.

Returns `True` if the branch must be processed, `False` otherwise.

Return type `bool`

dataentry ()

Called before all blocks are recursed. The default implementation simply returns `True`. You can access the data via `data`, and unlike in the `datainspect()` method, the full file has been processed at this stage.

Typically, you will override this function to perform a global operation on the file data.

Returns `True` if the children must be processed, `False` otherwise.

Return type `bool`

classmethod toastentry (*toaster*)

Called just before the toaster starts processing all files. If it returns `False`, then the spell is not used. The default implementation simply returns `True`.

For example, if the spell only acts on a particular block type, but that block type is excluded, then you can use this function to flag that this spell can be skipped. You can also use this function to initialize statistics data to be aggregated from files, to initialize a log file, and so.

Parameters **toaster** (`Toaster`) – The toaster this spell is called from.

Returns `True` if the spell applies, `False` otherwise.

Return type `bool`

```
class pyffi.spells.nif.fix.SpellFixCenterRadius (toaster=None, data=None,  
                                              stream=None)
```

Bases: `pyffi.spells.nif.check.SpellCheckCenterRadius`

Recalculate geometry centers and radii.

```
class pyffi.spells.nif.fix.SpellFixSkinCenterRadius (toaster=None, data=None,  
                                              stream=None)
```

Bases: `pyffi.spells.nif.check.SpellCheckSkinCenterRadius`

Recalculate skin centers and radii.

```
class pyffi.spells.nif.fix.SpellFixMopp (toaster=None, data=None, stream=None)
```

Bases: `pyffi.spells.nif.check.SpellCheckMopp`

Recalculate mopp data from collision geometry.

branchentry (*branch*)

Cast the spell on the given branch. First called with `branch` equal to `data`'s children, then the grandchildren, and so on. The default implementation simply returns `True`.

Typically, you will override this function to perform an operation on a particular block type and/or to stop recursion at particular block types.

Parameters **branch** (`GlobalNode`) – The branch to cast the spell on.

Returns `True` if the children must be processed, `False` otherwise.

Return type `bool`

```
class pyffi.spells.nif.fix.SpellCleanStringPalette (toaster=None, data=None,
                                                    stream=None)
```

Bases: `pyffi.spells.nif.NifSpell`

Remove unused strings from string palette.

branchentry (*branch*)

Parses string palette of either a single controller sequence, or of all controller sequences in a controller manager.

```
>>> seq = NifFormat.NiControllerSequence()
>>> seq.string_palette = NifFormat.NiStringPalette()
>>> block = seq.add_controlled_block()
>>> block.string_palette = seq.string_palette
>>> block.set_variable_1("there")
>>> block.set_node_name("hello")
>>> block.string_palette.palette.add_string("test")
12
>>> seq.string_palette.palette.get_all_strings()
[b'there', b'hello', b'test']
>>> SpellCleanStringPalette().branchentry(seq)
pyffi.toaster:INFO:parsing string palette
False
>>> seq.string_palette.palette.get_all_strings()
[b'hello', b'there']
>>> block.get_variable_1()
b'there'
>>> block.get_node_name()
b'hello'
```

branchinspect (*branch*)

Like `_branchinspect()`, but for customization: can be overridden to perform an extra inspection (the default implementation always returns `True`).

Parameters **branch** (`GlobalNode`) – The branch to check.

Returns `True` if the branch must be processed, `False` otherwise.

Return type `bool`

datainspect ()

This is called after `pyffi.object_models.FileFormat.Data.inspect()` has been called, and before `pyffi.object_models.FileFormat.Data.read()` is called. Override this function for customization.

Returns `True` if the file must be processed, `False` otherwise.

Return type `bool`

substitute (*old_string*)

Helper function to substitute strings in the string palette, to allow subclasses of this spell can modify the strings. This implementation returns string unmodified.

```
class pyffi.spells.nif.fix.SpellDelUnusedRoots (toaster=None, data=None,
                                                    stream=None)
```

Bases: `pyffi.spells.nif.NifSpell`

Remove root branches that shouldn't be root branches and are unused in the file such as `NiProperty` branches that are not properly parented.

dataentry ()

Called before all blocks are recursed. The default implementation simply returns `True`. You can access

the data via `data`, and unlike in the `datainspect()` method, the full file has been processed at this stage.

Typically, you will override this function to perform a global operation on the file data.

Returns True if the children must be processed, False otherwise.

Return type bool

datainspect()

This is called after `pyffi.object_models.FileFormat.Data.inspect()` has been called, and before `pyffi.object_models.FileFormat.Data.read()` is called. Override this function for customization.

Returns True if the file must be processed, False otherwise.

Return type bool

```
class pyffi.spells.nif.fix.SpellFixEmptySkeletonRoots (toaster=None, data=None,
                                                         stream=None)
```

Bases: `pyffi.spells.nif.NifSpell`

Fix empty skeleton roots in an as sane as possible way.

branchentry (branch)

Cast the spell on the given branch. First called with branch equal to `data`'s children, then the grandchildren, and so on. The default implementation simply returns True.

Typically, you will override this function to perform an operation on a particular block type and/or to stop recursion at particular block types.

Parameters **branch** (GlobalNode) – The branch to cast the spell on.

Returns True if the children must be processed, False otherwise.

Return type bool

branchinspect (branch)

Like `_branchinspect()`, but for customization: can be overridden to perform an extra inspection (the default implementation always returns True).

Parameters **branch** (GlobalNode) – The branch to check.

Returns True if the branch must be processed, False otherwise.

Return type bool

dataentry()

Called before all blocks are recursed. The default implementation simply returns True. You can access the data via `data`, and unlike in the `datainspect()` method, the full file has been processed at this stage.

Typically, you will override this function to perform a global operation on the file data.

Returns True if the children must be processed, False otherwise.

Return type bool

datainspect()

This is called after `pyffi.object_models.FileFormat.Data.inspect()` has been called, and before `pyffi.object_models.FileFormat.Data.read()` is called. Override this function for customization.

Returns True if the file must be processed, False otherwise.

Return type bool

Regression tests

Spells for optimizing NIF files.

```
class pyffi.spells.nif.optimize.SpellCleanRefLists (toaster=None,      data=None,
                                                    stream=None)
```

Bases: `pyffi.spells.nif.NifSpell`

Remove empty and duplicate entries in reference lists.

branchentry (*branch*)

Cast the spell on the given branch. First called with branch equal to data's children, then the grandchildren, and so on. The default implementation simply returns True.

Typically, you will override this function to perform an operation on a particular block type and/or to stop recursion at particular block types.

Parameters **branch** (`GlobalNode`) – The branch to cast the spell on.

Returns True if the children must be processed, False otherwise.

Return type bool

branchinspect (*branch*)

Like `_branchinspect()`, but for customization: can be overridden to perform an extra inspection (the default implementation always returns True).

Parameters **branch** (`GlobalNode`) – The branch to check.

Returns True if the branch must be processed, False otherwise.

Return type bool

cleanreflist (*reflist, category*)

Return a cleaned copy of the given list of references.

dataentry ()

Called before all blocks are recursed. The default implementation simply returns True. You can access the data via `data`, and unlike in the `datainspect()` method, the full file has been processed at this stage.

Typically, you will override this function to perform a global operation on the file data.

Returns True if the children must be processed, False otherwise.

Return type bool

datainspect ()

This is called after `pyffi.object_models.FileFormat.Data.inspect()` has been called, and before `pyffi.object_models.FileFormat.Data.read()` is called. Override this function for customization.

Returns True if the file must be processed, False otherwise.

Return type bool

```
class pyffi.spells.nif.optimize.SpellMergeDuplicates (*args, **kwargs)
```

Bases: `pyffi.spells.nif.NifSpell`

Remove duplicate branches.

branchentry (*branch*)

Cast the spell on the given branch. First called with branch equal to data's children, then the grandchildren, and so on. The default implementation simply returns True.

Typically, you will override this function to perform an operation on a particular block type and/or to stop recursion at particular block types.

Parameters **branch** (GlobalNode) – The branch to cast the spell on.

Returns True if the children must be processed, False otherwise.

Return type bool

branchinspect (*branch*)

Like `_branchinspect()`, but for customization: can be overridden to perform an extra inspection (the default implementation always returns True).

Parameters **branch** (GlobalNode) – The branch to check.

Returns True if the branch must be processed, False otherwise.

Return type bool

datainspect ()

This is called after `pyffi.object_models.FileFormat.Data.inspect()` has been called, and before `pyffi.object_models.FileFormat.Data.read()` is called. Override this function for customization.

Returns True if the file must be processed, False otherwise.

Return type bool

class `pyffi.spells.nif.optimize.SpellOptimizeGeometry` (*args, **kwargs)

Bases: `pyffi.spells.nif.NifSpell`

Optimize all geometries: - remove duplicate vertices - triangulate - recalculate skin partition - recalculate tangent space

branchentry (*branch*)

Optimize a NiTriStrips or NiTriShape block:

- remove duplicate vertices
- retriangulate for vertex cache
- recalculate skin partition
- recalculate tangent space

Todo: Limit the size of shapes (see operation optimization mod for Oblivion!)

branchinspect (*branch*)

Like `_branchinspect()`, but for customization: can be overridden to perform an extra inspection (the default implementation always returns True).

Parameters **branch** (GlobalNode) – The branch to check.

Returns True if the branch must be processed, False otherwise.

Return type bool

datainspect ()

This is called after `pyffi.object_models.FileFormat.Data.inspect()` has been called, and before `pyffi.object_models.FileFormat.Data.read()` is called. Override this function for customization.

Returns True if the file must be processed, False otherwise.

Return type bool

class `pyffi.spells.nif.optimize.SpellOptimize` (*toaster=None*, *data=None*,
stream=None)

Bases: `pyffi.spells.SpellCleanFarNifSpellDelUnusedRootsSpellCleanRefListsSpellDetachHavol`

Global fixer and optimizer spell.

class `pyffi.spells.nif.optimize.SpellDelUnusedBones` (*toaster=None*, *data=None*,
stream=None)

Bases: `pyffi.spells.nif.NifSpell`

Remove nodes that are not used for anything.

branchentry (*branch*)

Cast the spell on the given branch. First called with branch equal to data's children, then the grandchildren, and so on. The default implementation simply returns True.

Typically, you will override this function to perform an operation on a particular block type and/or to stop recursion at particular block types.

Parameters **branch** (`GlobalNode`) – The branch to cast the spell on.

Returns True if the children must be processed, False otherwise.

Return type bool

branchinspect (*branch*)

Like `_branchinspect()`, but for customization: can be overridden to perform an extra inspection (the default implementation always returns True).

Parameters **branch** (`GlobalNode`) – The branch to check.

Returns True if the branch must be processed, False otherwise.

Return type bool

dataentry ()

Called before all blocks are recursed. The default implementation simply returns True. You can access the data via `data`, and unlike in the `datainspect()` method, the full file has been processed at this stage.

Typically, you will override this function to perform a global operation on the file data.

Returns True if the children must be processed, False otherwise.

Return type bool

datainspect ()

This is called after `pyffi.object_models.FileFormat.Data.inspect()` has been called, and before `pyffi.object_models.FileFormat.Data.read()` is called. Override this function for customization.

Returns True if the file must be processed, False otherwise.

Return type bool

pyffi.spells.nif.modify — spells to make modifications

Module which contains all spells that modify a nif.

```
class pyffi.spells.nif.modify.SpellTexturePath (toaster=None, data=None,
                                              stream=None)
```

Bases: `pyffi.spells.nif.fix.SpellParseTexturePath`

Changes the texture path while keeping the texture names.

substitute (*old_path*)

Helper function to allow subclasses of this spell to change part of the path with minimum of code. This implementation returns path unmodified.

classmethod toastentry (*toaster*)

Called just before the toaster starts processing all files. If it returns `False`, then the spell is not used. The default implementation simply returns `True`.

For example, if the spell only acts on a particular block type, but that block type is excluded, then you can use this function to flag that this spell can be skipped. You can also use this function to initialize statistics data to be aggregated from files, to initialize a log file, and so.

Parameters **toaster** (`Toaster`) – The toaster this spell is called from.

Returns `True` if the spell applies, `False` otherwise.

Return type `bool`

```
class pyffi.spells.nif.modify.SpellSubstituteTexturePath (toaster=None,
                                                         data=None,
                                                         stream=None)
```

Bases: `pyffi.spells.nif.fix.SpellFixTexturePath`

Runs a regex replacement on texture paths.

substitute (*old_path*)

Returns modified texture path, and reports if path was modified.

classmethod toastentry (*toaster*)

Called just before the toaster starts processing all files. If it returns `False`, then the spell is not used. The default implementation simply returns `True`.

For example, if the spell only acts on a particular block type, but that block type is excluded, then you can use this function to flag that this spell can be skipped. You can also use this function to initialize statistics data to be aggregated from files, to initialize a log file, and so.

Parameters **toaster** (`Toaster`) – The toaster this spell is called from.

Returns `True` if the spell applies, `False` otherwise.

Return type `bool`

```
class pyffi.spells.nif.modify.SpellLowResTexturePath (toaster=None, data=None,
                                                         stream=None)
```

Bases: `pyffi.spells.nif.modify.SpellSubstituteTexturePath`

Changes the texture path by replacing 'textures*' with 'textureslowres*' - used mainly for making _far.nifs

substitute (*old_path*)

Returns modified texture path, and reports if path was modified.

classmethod toastentry (*toaster*)

Called just before the toaster starts processing all files. If it returns `False`, then the spell is not used. The default implementation simply returns `True`.

For example, if the spell only acts on a particular block type, but that block type is excluded, then you can use this function to flag that this spell can be skipped. You can also use this function to initialize statistics data to be aggregated from files, to initialize a log file, and so.

Parameters **toaster** (*Toaster*) – The toaster this spell is called from.

Returns *True* if the spell applies, *False* otherwise.

Return type *bool*

```
class pyffi.spells.nif.modify.SpellCollisionType (toaster=None, data=None,  
                                                stream=None)
```

Bases: *pyffi.spells.nif.NifSpell*

Sets the object collision to be a different type

branchentry (*branch*)

Cast the spell on the given branch. First called with *branch* equal to *data*'s children, then the grandchildren, and so on. The default implementation simply returns *True*.

Typically, you will override this function to perform an operation on a particular block type and/or to stop recursion at particular block types.

Parameters **branch** (*GlobalNode*) – The branch to cast the spell on.

Returns *True* if the children must be processed, *False* otherwise.

Return type *bool*

branchinspect (*branch*)

Like *_branchinspect()*, but for customization: can be overridden to perform an extra inspection (the default implementation always returns *True*).

Parameters **branch** (*GlobalNode*) – The branch to check.

Returns *True* if the branch must be processed, *False* otherwise.

Return type *bool*

datainspect ()

This is called after *pyffi.object_models.FileFormat.Data.inspect()* has been called, and before *pyffi.object_models.FileFormat.Data.read()* is called. Override this function for customization.

Returns *True* if the file must be processed, *False* otherwise.

Return type *bool*

classmethod toastentry (*toaster*)

Called just before the toaster starts processing all files. If it returns *False*, then the spell is not used. The default implementation simply returns *True*.

For example, if the spell only acts on a particular block type, but that block type is excluded, then you can use this function to flag that this spell can be skipped. You can also use this function to initialize statistics data to be aggregated from files, to initialize a log file, and so.

Parameters **toaster** (*Toaster*) – The toaster this spell is called from.

Returns *True* if the spell applies, *False* otherwise.

Return type *bool*

```
class pyffi.spells.nif.modify.SpellCollisionMaterial (toaster=None, data=None,  
                                                  stream=None)
```

Bases: *pyffi.spells.nif.NifSpell*

Sets the object's collision material to be a different type

branchentry (*branch*)

Cast the spell on the given branch. First called with branch equal to data's children, then the grandchildren, and so on. The default implementation simply returns True.

Typically, you will override this function to perform an operation on a particular block type and/or to stop recursion at particular block types.

Parameters **branch** (GlobalNode) – The branch to cast the spell on.

Returns True if the children must be processed, False otherwise.

Return type bool

branchinspect (*branch*)

Like `_branchinspect()`, but for customization: can be overridden to perform an extra inspection (the default implementation always returns True).

Parameters **branch** (GlobalNode) – The branch to check.

Returns True if the branch must be processed, False otherwise.

Return type bool

datainspect ()

This is called after `pyffi.object_models.FileFormat.Data.inspect()` has been called, and before `pyffi.object_models.FileFormat.Data.read()` is called. Override this function for customization.

Returns True if the file must be processed, False otherwise.

Return type bool

classmethod toastentry (*toaster*)

Called just before the toaster starts processing all files. If it returns False, then the spell is not used. The default implementation simply returns True.

For example, if the spell only acts on a particular block type, but that block type is excluded, then you can use this function to flag that this spell can be skipped. You can also use this function to initialize statistics data to be aggregated from files, to initialize a log file, and so.

Parameters **toaster** (Toaster) – The toaster this spell is called from.

Returns True if the spell applies, False otherwise.

Return type bool

```
class pyffi.spells.nif.modify.SpellScaleAnimationTime (toaster=None, data=None,
                                                    stream=None)
```

Bases: `pyffi.spells.nif.NifSpell`

Scales the animation time.

branchentry (*branch*)

Cast the spell on the given branch. First called with branch equal to data's children, then the grandchildren, and so on. The default implementation simply returns True.

Typically, you will override this function to perform an operation on a particular block type and/or to stop recursion at particular block types.

Parameters **branch** (GlobalNode) – The branch to cast the spell on.

Returns True if the children must be processed, False otherwise.

Return type bool

branchinspect (*branch*)

Like `_branchinspect()`, but for customization: can be overridden to perform an extra inspection (the default implementation always returns `True`).

Parameters **branch** (`GlobalNode`) – The branch to check.

Returns `True` if the branch must be processed, `False` otherwise.

Return type `bool`

datainspect ()

This is called after `pyffi.object_models.FileFormat.Data.inspect()` has been called, and before `pyffi.object_models.FileFormat.Data.read()` is called. Override this function for customization.

Returns `True` if the file must be processed, `False` otherwise.

Return type `bool`

classmethod toastentry (*toaster*)

Called just before the toaster starts processing all files. If it returns `False`, then the spell is not used. The default implementation simply returns `True`.

For example, if the spell only acts on a particular block type, but that block type is excluded, then you can use this function to flag that this spell can be skipped. You can also use this function to initialize statistics data to be aggregated from files, to initialize a log file, and so.

Parameters **toaster** (`Toaster`) – The toaster this spell is called from.

Returns `True` if the spell applies, `False` otherwise.

Return type `bool`

class `pyffi.spells.nif.modify.SpellReverseAnimation` (*toaster=None, data=None, stream=None*)

Bases: `pyffi.spells.nif.NifSpell`

Reverses the animation by reversing datas in relation to the time.

branchentry (*branch*)

Cast the spell on the given branch. First called with `branch` equal to `data`'s children, then the grandchildren, and so on. The default implementation simply returns `True`.

Typically, you will override this function to perform an operation on a particular block type and/or to stop recursion at particular block types.

Parameters **branch** (`GlobalNode`) – The branch to cast the spell on.

Returns `True` if the children must be processed, `False` otherwise.

Return type `bool`

branchinspect (*branch*)

Like `_branchinspect()`, but for customization: can be overridden to perform an extra inspection (the default implementation always returns `True`).

Parameters **branch** (`GlobalNode`) – The branch to check.

Returns `True` if the branch must be processed, `False` otherwise.

Return type `bool`

datainspect ()

This is called after `pyffi.object_models.FileFormat.Data.inspect()` has been called, and before `pyffi.object_models.FileFormat.Data.read()` is called. Override this function for customization.

Returns `True` if the file must be processed, `False` otherwise.

Return type `bool`

```
class pyffi.spells.nif.modify.SpellSubstituteStringPalette (toaster=None,
                                                         data=None,
                                                         stream=None)
```

Bases: `pyffi.spells.nif.fix.SpellCleanStringPalette`

Substitute strings in a string palette.

substitute (*old_string*)

Returns modified string, and reports if string was modified.

classmethod toastentry (*toaster*)

Called just before the toaster starts processing all files. If it returns `False`, then the spell is not used. The default implementation simply returns `True`.

For example, if the spell only acts on a particular block type, but that block type is excluded, then you can use this function to flag that this spell can be skipped. You can also use this function to initialize statistics data to be aggregated from files, to initialize a log file, and so.

Parameters **toaster** (`Toaster`) – The toaster this spell is called from.

Returns `True` if the spell applies, `False` otherwise.

Return type `bool`

```
class pyffi.spells.nif.modify.SpellChangeBonePriorities (toaster=None, data=None,
                                                         stream=None)
```

Bases: `pyffi.spells.nif.NifSpell`

Changes controlled block priorities based on controlled block name.

branchentry (*branch*)

Cast the spell on the given branch. First called with *branch* equal to *data*'s children, then the grandchildren, and so on. The default implementation simply returns `True`.

Typically, you will override this function to perform an operation on a particular block type and/or to stop recursion at particular block types.

Parameters **branch** (`GlobalNode`) – The branch to cast the spell on.

Returns `True` if the children must be processed, `False` otherwise.

Return type `bool`

branchinspect (*branch*)

Like `_branchinspect()`, but for customization: can be overridden to perform an extra inspection (the default implementation always returns `True`).

Parameters **branch** (`GlobalNode`) – The branch to check.

Returns `True` if the branch must be processed, `False` otherwise.

Return type `bool`

datainspect ()

This is called after `pyffi.object_models.FileFormat.Data.inspect()` has been called, and before `pyffi.object_models.FileFormat.Data.read()` is called. Override this function for customization.

Returns `True` if the file must be processed, `False` otherwise.

Return type `bool`

classmethod `toastentry` (*toaster*)

Called just before the toaster starts processing all files. If it returns `False`, then the spell is not used. The default implementation simply returns `True`.

For example, if the spell only acts on a particular block type, but that block type is excluded, then you can use this function to flag that this spell can be skipped. You can also use this function to initialize statistics data to be aggregated from files, to initialize a log file, and so.

Parameters **toaster** (`Toaster`) – The toaster this spell is called from.

Returns `True` if the spell applies, `False` otherwise.

Return type `bool`

```
class pyffi.spells.nif.modify.SpellSetInterpolatorTransRotScale (toaster=None,  
                                                                data=None,  
                                                                stream=None)
```

Bases: `pyffi.spells.nif.NifSpell`

Changes specified bone(s) translations/rotations in their `NiTransformInterpolator`.

branchentry (*branch*)

Cast the spell on the given branch. First called with `branch` equal to `data`'s children, then the grandchildren, and so on. The default implementation simply returns `True`.

Typically, you will override this function to perform an operation on a particular block type and/or to stop recursion at particular block types.

Parameters **branch** (`GlobalNode`) – The branch to cast the spell on.

Returns `True` if the children must be processed, `False` otherwise.

Return type `bool`

branchinspect (*branch*)

Like `_branchinspect()`, but for customization: can be overridden to perform an extra inspection (the default implementation always returns `True`).

Parameters **branch** (`GlobalNode`) – The branch to check.

Returns `True` if the branch must be processed, `False` otherwise.

Return type `bool`

datainspect ()

This is called after `pyffi.object_models.FileFormat.Data.inspect()` has been called, and before `pyffi.object_models.FileFormat.Data.read()` is called. Override this function for customization.

Returns `True` if the file must be processed, `False` otherwise.

Return type `bool`

classmethod `toastentry` (*toaster*)

Called just before the toaster starts processing all files. If it returns `False`, then the spell is not used. The default implementation simply returns `True`.

For example, if the spell only acts on a particular block type, but that block type is excluded, then you can use this function to flag that this spell can be skipped. You can also use this function to initialize statistics data to be aggregated from files, to initialize a log file, and so.

Parameters **toaster** (`Toaster`) – The toaster this spell is called from.

Returns `True` if the spell applies, `False` otherwise.

Return type `bool`

```
class pyffi.spells.nif.modify.SpellDelInterpolatorTransformData (toaster=None,  
                                                                data=None,  
                                                                stream=None)
```

Bases: `pyffi.spells.nif.NifSpell`

Deletes the specified bone(s) `NiTransformData(s)`.

branchentry (*branch*)

Cast the spell on the given branch. First called with `branch` equal to `data`'s children, then the grandchildren, and so on. The default implementation simply returns `True`.

Typically, you will override this function to perform an operation on a particular block type and/or to stop recursion at particular block types.

Parameters **branch** (`GlobalNode`) – The branch to cast the spell on.

Returns `True` if the children must be processed, `False` otherwise.

Return type `bool`

branchinspect (*branch*)

Like `_branchinspect()`, but for customization: can be overridden to perform an extra inspection (the default implementation always returns `True`).

Parameters **branch** (`GlobalNode`) – The branch to check.

Returns `True` if the branch must be processed, `False` otherwise.

Return type `bool`

datainspect ()

This is called after `pyffi.object_models.FileFormat.Data.inspect()` has been called, and before `pyffi.object_models.FileFormat.Data.read()` is called. Override this function for customization.

Returns `True` if the file must be processed, `False` otherwise.

Return type `bool`

classmethod toastentry (*toaster*)

Called just before the toaster starts processing all files. If it returns `False`, then the spell is not used. The default implementation simply returns `True`.

For example, if the spell only acts on a particular block type, but that block type is excluded, then you can use this function to flag that this spell can be skipped. You can also use this function to initialize statistics data to be aggregated from files, to initialize a log file, and so.

Parameters **toaster** (`Toaster`) – The toaster this spell is called from.

Returns `True` if the spell applies, `False` otherwise.

Return type `bool`

```
class pyffi.spells.nif.modify.SpellDelBranches (toaster=None,          data=None,  
                                                stream=None)
```

Bases: `pyffi.spells.nif.NifSpell`

Delete blocks that match the exclude list.

branchentry (*branch*)

Strip branch if it is flagged for deletion.

is_branch_to_be_deleted (*branch*)

Returns `True` for those branches that must be deleted. The default implementation returns `True` for

branches that are not admissible as specified by include/exclude options of the toaster. Override in subclasses that must delete specific branches.

```
class pyffi.spells.nif.modify._SpellDelBranchClasses (toaster=None,      data=None,
                                                    stream=None)
```

Bases: `pyffi.spells.nif.modify.SpellDelBranches`

Delete blocks that match a given list. Only useful as base class for other spells.

```
BRANCH_CLASSES_TO_BE_DELETED = ()
```

List of branch classes that have to be deleted.

```
datainspect ()
```

This is called after `pyffi.object_models.FileFormat.Data.inspect()` has been called, and before `pyffi.object_models.FileFormat.Data.read()` is called. Override this function for customization.

Returns True if the file must be processed, False otherwise.

Return type bool

```
is_branch_to_be_deleted (branch)
```

Returns True for those branches that must be deleted. The default implementation returns True for branches that are not admissible as specified by include/exclude options of the toaster. Override in subclasses that must delete specific branches.

```
class pyffi.spells.nif.modify.SpellDelSkinShapes (toaster=None,      data=None,
                                                    stream=None)
```

Bases: `pyffi.spells.nif.modify.SpellDelBranches`

Delete any geometries with a material name of 'skin'

```
branchinspect (branch)
```

Like `_branchinspect()`, but for customization: can be overridden to perform an extra inspection (the default implementation always returns True).

Parameters **branch** (GlobalNode) – The branch to check.

Returns True if the branch must be processed, False otherwise.

Return type bool

```
is_branch_to_be_deleted (branch)
```

Returns True for those branches that must be deleted. The default implementation returns True for branches that are not admissible as specified by include/exclude options of the toaster. Override in subclasses that must delete specific branches.

```
class pyffi.spells.nif.modify.SpellDisableParallax (toaster=None,      data=None,
                                                    stream=None)
```

Bases: `pyffi.spells.nif.NifSpell`

Disable parallax shader (for Oblivion, but may work on other nifs too).

```
branchentry (branch)
```

Cast the spell on the given branch. First called with branch equal to data's children, then the grandchildren, and so on. The default implementation simply returns True.

Typically, you will override this function to perform an operation on a particular block type and/or to stop recursion at particular block types.

Parameters **branch** (GlobalNode) – The branch to cast the spell on.

Returns True if the children must be processed, False otherwise.

Return type bool

branchinspect (*branch*)

Like `_branchinspect()`, but for customization: can be overridden to perform an extra inspection (the default implementation always returns `True`).

Parameters **branch** (`GlobalNode`) – The branch to check.

Returns `True` if the branch must be processed, `False` otherwise.

Return type `bool`

datainspect ()

This is called after `pyffi.object_models.FileFormat.Data.inspect()` has been called, and before `pyffi.object_models.FileFormat.Data.read()` is called. Override this function for customization.

Returns `True` if the file must be processed, `False` otherwise.

Return type `bool`

```
class pyffi.spells.nif.modify.SpellAddStencilProperty (toaster=None, data=None,
                                                    stream=None)
```

Bases: `pyffi.spells.nif.NifSpell`

Adds a `NiStencilProperty` to each geometry if it is not present.

branchentry (*branch*)

Cast the spell on the given branch. First called with `branch` equal to `data`'s children, then the grandchildren, and so on. The default implementation simply returns `True`.

Typically, you will override this function to perform an operation on a particular block type and/or to stop recursion at particular block types.

Parameters **branch** (`GlobalNode`) – The branch to cast the spell on.

Returns `True` if the children must be processed, `False` otherwise.

Return type `bool`

branchinspect (*branch*)

Like `_branchinspect()`, but for customization: can be overridden to perform an extra inspection (the default implementation always returns `True`).

Parameters **branch** (`GlobalNode`) – The branch to check.

Returns `True` if the branch must be processed, `False` otherwise.

Return type `bool`

datainspect ()

This is called after `pyffi.object_models.FileFormat.Data.inspect()` has been called, and before `pyffi.object_models.FileFormat.Data.read()` is called. Override this function for customization.

Returns `True` if the file must be processed, `False` otherwise.

Return type `bool`

```
class pyffi.spells.nif.modify.SpellDelVertexColor (toaster=None, data=None,
                                                    stream=None)
```

Bases: `pyffi.spells.nif.modify.SpellDelBranches`

Delete vertex color properties and vertex color data.

branchentry (*branch*)

Strip branch if it is flagged for deletion.

branchinspect (*branch*)

Like `_branchinspect()`, but for customization: can be overridden to perform an extra inspection (the default implementation always returns `True`).

Parameters **branch** (`GlobalNode`) – The branch to check.

Returns `True` if the branch must be processed, `False` otherwise.

Return type `bool`

datainspect ()

This is called after `pyffi.object_models.FileFormat.Data.inspect()` has been called, and before `pyffi.object_models.FileFormat.Data.read()` is called. Override this function for customization.

Returns `True` if the file must be processed, `False` otherwise.

Return type `bool`

is_branch_to_be_deleted (*branch*)

Returns `True` for those branches that must be deleted. The default implementation returns `True` for branches that are not admissible as specified by include/exclude options of the toaster. Override in subclasses that must delete specific branches.

```
class pyffi.spells.nif.modify.SpellMakeSkinlessNif (toaster=None, data=None,
                                                    stream=None)
```

Bases: `pyffi.spells.SpellDelSkinShapesSpellAddStencilProperty`

Spell to make fleshless CMR (Custom Model Races) clothing/armour type nifs.

```
class pyffi.spells.nif.modify.SpellCleanFarNif (toaster=None, data=None,
                                                stream=None)
```

Bases: `pyffi.spells.SpellDelVertexColorPropertySpellDelAlphaPropertySpellDelSpecularProp`

Spell to clean `_far` type nifs (for even more optimizations, combine this with the `optimize spell`).

datainspect ()

Inspect every spell with `L{Spell.datainspect}` and keep those spells that must be cast.

```
class pyffi.spells.nif.modify.SpellMakeFarNif (toaster=None, data=None,
                                                stream=None)
```

Bases: `pyffi.spells.SpellDelVertexColorPropertySpellDelAlphaPropertySpellDelSpecularProp`

Spell to make `_far` type nifs (for even more optimizations, combine this with the `optimize spell`).

pyffi.spells.tga — Targa spells

There are no spells yet.

Some spells are applicable on every file format, and those are documented here.

```
class pyffi.spells.SpellApplyPatch (toaster=None, data=None, stream=None)
```

Bases: `pyffi.spells.Spell`

A spell for applying a patch on files.

datainspect ()

There is no need to read the whole file, so we apply the patch already at inspection stage, and stop the spell process by returning `False`.

Returns `False`

Return type `bool`

Adding new spells

To create new spells, derive your custom spells from the `Spell` class, and include them in the `Toaster.SPELLS` attribute of your toaster.

```
class pyffi.spells.Spell (toaster=None, data=None, stream=None)
```

Bases: `object`

Spell base class. A spell takes a data file and then does something useful with it. The main entry point for spells is `recurse()`, so if you are writing new spells, start with reading the documentation with `recurse()`.

READONLY = True

A `bool` which determines whether the spell is read only or not. Default value is `True`. Override this class attribute, and set to `False`, when subclassing a spell that must write files back to the disk.

SPELLNAME = None

A `str` describing how to refer to the spell from the command line. Override this class attribute when subclassing.

```
__init__ (toaster=None, data=None, stream=None)
```

Initialize the spell data.

Parameters

- **data** (`Data`) – The file `data`.
- **stream** (`file`) – The file `stream`.
- **toaster** (`Toaster`) – The `toaster` this spell is called from (optional).

```
_branchinspect (branch)
```

Check if spell should be cast on this branch or not, based on `exclude` and `include` options passed on the command line. You should not need to override this function: if you need additional checks on whether a branch must be parsed or not, override the `branchinspect()` method.

Parameters **branch** (`GlobalNode`) – The branch to check.

Returns `True` if the branch must be processed, `False` otherwise.

Return type `bool`

```
_datainspect ()
```

This is called after `pyffi.object_models.FileFormat.Data.inspect()` has been called, and before `pyffi.object_models.FileFormat.Data.read()` is called.

Returns `True` if the file must be processed, `False` otherwise.

Return type `bool`

```
branchentry (branch)
```

Cast the spell on the given branch. First called with `branch` equal to `data`'s children, then the grandchildren, and so on. The default implementation simply returns `True`.

Typically, you will override this function to perform an operation on a particular block type and/or to stop recursion at particular block types.

Parameters **branch** (`GlobalNode`) – The branch to cast the spell on.

Returns `True` if the children must be processed, `False` otherwise.

Return type `bool`

```
branchexit (branch)
```

Cast a spell on the given branch, after all its children, grandchildren, have been processed, if `branchentry()` returned `True` on the given branch.

Typically, you will override this function to perform a particular operation on a block type, but you rely on the fact that the children must have been processed first.

Parameters **branch** (GlobalNode) – The branch to cast the spell on.

branchinspect (*branch*)

Like `_branchinspect()`, but for customization: can be overridden to perform an extra inspection (the default implementation always returns `True`).

Parameters **branch** (GlobalNode) – The branch to check.

Returns `True` if the branch must be processed, `False` otherwise.

Return type `bool`

data = None

The *Data* instance this spell acts on.

dataentry ()

Called before all blocks are recursed. The default implementation simply returns `True`. You can access the data via *data*, and unlike in the `datainspect()` method, the full file has been processed at this stage.

Typically, you will override this function to perform a global operation on the file data.

Returns `True` if the children must be processed, `False` otherwise.

Return type `bool`

dataexit ()

Called after all blocks have been processed, if `dataentry()` returned `True`.

Typically, you will override this function to perform a final spell operation, such as writing back the file in a special way, or making a summary log.

datainspect ()

This is called after `pyffi.object_models.FileFormat.Data.inspect()` has been called, and before `pyffi.object_models.FileFormat.Data.read()` is called. Override this function for customization.

Returns `True` if the file must be processed, `False` otherwise.

Return type `bool`

recurse (*branch=None*)

Helper function which calls `_branchinspect()` and `branchinspect()` on the branch, if both successful then `branchentry()` on the branch, and if this is successful it calls `recurse()` on the branch's children, and once all children are done, it calls `branchexit()`.

Note that `_branchinspect()` and `branchinspect()` are not called upon first entry of this function, that is, when called with *data* as branch argument. Use `datainspect()` to stop recursion into this branch.

Do not override this function.

Parameters **branch** (GlobalNode) – The branch to start the recursion from, or `None` to recurse the whole tree.

stream = None

The current file being processed.

classmethod toastentry (*toaster*)

Called just before the toaster starts processing all files. If it returns `False`, then the spell is not used. The default implementation simply returns `True`.

For example, if the spell only acts on a particular block type, but that block type is excluded, then you can use this function to flag that this spell can be skipped. You can also use this function to initialize statistics data to be aggregated from files, to initialize a log file, and so.

Parameters `toaster` (*Toaster*) – The toaster this spell is called from.

Returns `True` if the spell applies, `False` otherwise.

Return type `bool`

toaster = `None`

The *Toaster* instance this spell is called from.

classmethod `toastexit` (*toaster*)

Called when the toaster has finished processing all files.

Parameters `toaster` (*Toaster*) – The toaster this spell is called from.

Grouping spells together

It is also possible to create composite spells, that is, spells that simply execute other spells. The following functions and classes can be used for this purpose.

`pyffi.spells.SpellGroupParallel` (*args)

Class factory for grouping spells in parallel.

`pyffi.spells.SpellGroupSeries` (*args)

Class factory for grouping spells in series.

class `pyffi.spells.SpellGroupBase` (*toaster=None, data=None, stream=None*)

Bases: *pyffi.spells.Spell*

Base class for grouping spells. This implements all the spell grouping functions that fall outside of the actual recursing (`__init__()`, `toastentry()`, `_datainspect()`, `datainspect()`, and `toastexit()`).

ACTIVESPELLCLASSES = []

List of active spells of this group (not instantiated). This list is automatically built when `toastentry()` is called.

SPELLCLASSES = []

List of *Spells* of this group (not instantiated).

datainspect ()

Inspect every spell with `L{Spell.datainspect}` and keep those spells that must be cast.

spells = []

List of active spell instances.

classmethod `toastentry` (*toaster*)

Called just before the toaster starts processing all files. If it returns `False`, then the spell is not used. The default implementation simply returns `True`.

For example, if the spell only acts on a particular block type, but that block type is excluded, then you can use this function to flag that this spell can be skipped. You can also use this function to initialize statistics data to be aggregated from files, to initialize a log file, and so.

Parameters `toaster` (*Toaster*) – The toaster this spell is called from.

Returns `True` if the spell applies, `False` otherwise.

Return type `bool`

classmethod `toastexit (toaster)`

Called when the toaster has finished processing all files.

Parameters `toaster` (*Toaster*) – The toaster this spell is called from.

class `pyffi.spells.SpellGroupParallelBase (toaster=None, data=None, stream=None)`

Bases: `pyffi.spells.SpellGroupBase`

Base class for running spells in parallel (that is, with only a single recursion in the tree).

branchentry (*branch*)

Run all spells.

branchexit (*branch*)

Cast a spell on the given branch, after all its children, grandchildren, have been processed, if `branchentry()` returned `True` on the given branch.

Typically, you will override this function to perform a particular operation on a block type, but you rely on the fact that the children must have been processed first.

Parameters `branch` (*GlobalNode*) – The branch to cast the spell on.

branchinspect (*branch*)

Inspect spells with `Spell.branchinspect()` (not all checks are executed, only keeps going until a spell inspection returns `True`).

property `changed`

`bool(x) -> bool`

Returns `True` when the argument `x` is true, `False` otherwise. The builtins `True` and `False` are the only two instances of the class `bool`. The class `bool` is a subclass of the class `int`, and cannot be subclassed.

dataentry ()

Look into every spell with `Spell.dataentry()`.

dataexit ()

Look into every spell with `Spell.dataexit()`.

class `pyffi.spells.SpellGroupSeriesBase (toaster=None, data=None, stream=None)`

Bases: `pyffi.spells.SpellGroupBase`

Base class for running spells in series.

branchentry (*branch*)

Cast the spell on the given branch. First called with `branch` equal to `data`'s children, then the grandchildren, and so on. The default implementation simply returns `True`.

Typically, you will override this function to perform an operation on a particular block type and/or to stop recursion at particular block types.

Parameters `branch` (*GlobalNode*) – The branch to cast the spell on.

Returns `True` if the children must be processed, `False` otherwise.

Return type `bool`

branchinspect (*branch*)

Like `_branchinspect()`, but for customization: can be overridden to perform an extra inspection (the default implementation always returns `True`).

Parameters `branch` (*GlobalNode*) – The branch to check.

Returns `True` if the branch must be processed, `False` otherwise.

Return type `bool`

property changed

`bool(x) -> bool`

Returns True when the argument `x` is true, False otherwise. The builtins True and False are the only two instances of the class bool. The class bool is a subclass of the class int, and cannot be subclassed.

dataentry ()

Called before all blocks are recursed. The default implementation simply returns True. You can access the data via `data`, and unlike in the `datainspect ()` method, the full file has been processed at this stage.

Typically, you will override this function to perform a global operation on the file data.

Returns True if the children must be processed, False otherwise.

Return type bool

dataexit ()

Called after all blocks have been processed, if `dataentry ()` returned True.

Typically, you will override this function to perform a final spell operation, such as writing back the file in a special way, or making a summary log.

recurse (branch=None)

Recurse spells in series.

Creating toaster scripts

To create a new toaster script, derive your toaster from the `Toaster` class, and set the `Toaster.FILEFORMAT` attribute of your toaster to the file format class of the files it can toast.

class `pyffi.spells.Toaster` (*spellclass=None, options=None, spellnames=None, logger=None*)

Bases: `object`

Toaster base class. Toasters run spells on large quantities of files. They load each file and pass the data structure to any number of spells.

ALIASESDICT = {}

Dictionary with aliases for spells.

DEFAULT_OPTIONS = {'applypatch': False, 'archives': False, 'arg': '', 'createpatch': False}

List of spell classes of the particular `Toaster` instance.

EXAMPLES = ''

Some examples which describe typical use of the toaster.

FILEFORMAT

alias of `pyffi.object_models.FileFormat`

SPELLS = []

List of all available `Spell` classes.

cli ()

Command line interface: initializes spell classes and options from the command line, and run the `toast ()` method.

exclude_types = []

Tuple of types corresponding to the exclude key of `options`.

get_toast_head_root_ext (*filename*)

Get the name of where the input file *filename* would be written to by the toaster: head, root, and extension.

Parameters `filename` (`str`) – The name of the hypothetical file to be toasted.

Returns The head, root, and extension of the destination, or `(None, None, None)` if `--dry-run` is specified.

Return type `tuple` of three `strs`

get_toast_stream (`filename`, `test_exists=False`)

Calls `get_toast_head_root_ext (filename) ()` to determine the name of the toast file, and return a stream for writing accordingly.

Then return a stream where result can be written to, or in case `test_exists` is `True`, test if result would overwrite a file. More specifically, if `test_exists` is `True`, then no streams are created, and `True` is returned if the file already exists, and `False` is returned otherwise.

include_types = []

Tuple of types corresponding to the include key of *options*.

indent = 0

An int which describes the current indentation level for messages.

inspect_filename (`filename`)

Returns whether to toast a filename or not, based on `skip_regexs` and `only_regexs`.

is_admissible_branch_class (`branchtype`)

Helper function which checks whether a given branch type should have spells cast on it or not, based in exclude and include options.

```
>>> from pyffi.formats.nif import NifFormat
>>> class MyToaster(Toaster):
...     FILEFORMAT = NifFormat
>>> toaster = MyToaster() # no include or exclude: all admissible
>>> toaster.is_admissible_branch_class(NifFormat.NiProperty)
True
>>> toaster.is_admissible_branch_class(NifFormat.NiNode)
True
>>> toaster = MyToaster(options={"include": ["NiProperty", "NiNode"], "exclude":
↳ ["NiMaterialProperty", "NiLODNode"]})
>>> toaster.is_admissible_branch_class(NifFormat.NiProperty)
True
>>> toaster.is_admissible_branch_class(NifFormat.NiNode)
True
>>> toaster.is_admissible_branch_class(NifFormat.NiAVObject)
False
>>> toaster.is_admissible_branch_class(NifFormat.NiLODNode)
False
>>> toaster.is_admissible_branch_class(NifFormat.NiSwitchNode)
True
>>> toaster.is_admissible_branch_class(NifFormat.NiMaterialProperty)
False
>>> toaster.is_admissible_branch_class(NifFormat.NiAlphaProperty)
True
```

logger = <Logger pyffi.toaster (WARNING)>

A `logging.Logger` for toaster log messages.

msg (`message`)

Write log message with `logger.info()`, taking into account *indent*.

Parameters `message` (`str`) – The message to write.

msgblockbegin (*message*)
 Acts like *msg()*, but also increases *indent* after writing the message.

msgblockend (*message=None*)
 Acts like *msg()*, but also decreases *indent* before writing the message, but if the message argument is *None*, then no message is printed.

only_regexs = []
 Tuple of regular expressions corresponding to the only key of *options*.

options = {}
 The options of the toaster, as dict.

static parse_inifile (*option, opt, value, parser, toaster=None*)
 Initializes spell classes and options from an ini file.

skip_regexs = []
 Tuple of regular expressions corresponding to the skip key of *options*.

spellnames = []
 A list of the names of the spells.

toast (*top*)
 Walk over all files in a directory tree and cast spells on every file.

Parameters *top* (*str*) – The directory or file to toast.

toast_archives (*top*)
 Toast all files in all archives.

top = ''
 Name of the top folder to toast.

write (*stream, data*)
 Writes the data to data and raises an exception if the write fails, but restores file if fails on overwrite.

writepatch (*stream, data*)
 Creates a binary patch for the updated file.

pyffi.object_models — File format description engines

Warning: The documentation of this module is very incomplete.

This module bundles all file format object models. An object model is a group of classes whose instances can hold the information contained in a file whose format is described in a particular way (xml, xsd, and possibly others).

class `pyffi.object_models.MetaFileFormat`

Bases: `type`

This metaclass is an abstract base class for transforming a file format description into classes which can be directly used to manipulate files in this format.

A file format is implemented as a particular class (a subclass of `FileFormat`) with class members corresponding to different (bit)struct types, enum types, basic types, and aliases.

static `openfile` (*filename, filepaths=None, encoding=None*)

Find *filename* in given *filepaths*, and open it. Raises `IOError` if file cannot be opened.

Parameters

- **filename** (*str*) – The file to open.
- **filepaths** (*list of str*) – List of paths where to look for the file.

class `pyffi.object_models.FileFormat`

Bases: `object`

This class is the base class for all file formats. It implements a number of useful functions such as walking over directory trees (`walkData()`) and a default attribute naming function (`name_attribute()`). It also implements the base class for representing file data (`FileFormat.Data`).

ARCHIVE_CLASSES = []

Override this with a list of archive formats that may contain files of the format.

class `Data`

Bases: `pyffi.utils.graph.GlobalNode`

Base class for representing data in a particular format. Override this class to implement reading and writing.

inspect (*stream*)

Quickly checks whether the stream appears to contain data of a particular format. Resets stream to original position. Call this function if you simply wish to check that a file is of a particular format without having to parse it completely.

Override this method.

Parameters `stream` (*file*) – The file to inspect.

Returns `True` if stream is of particular format, `False` otherwise.

read (*stream*)

Read data of particular format from stream. Override this method.

Parameters `stream` (*file*) – The file to read from.

user_version = `None`

User version (additional version field) of the data.

version = `None`

Version of the data.

write (*stream*)

Write data of particular format to stream. Override this method.

Parameters `stream` (*file*) – The file to write to.

RE_FILENAME = `None`

Override this with a regular expression (the result of a `re.compile` call) for the file extension of the format you are implementing.

classmethod `name_attribute` (*name*)

Converts an attribute name, as in the description file, into a name usable by python.

Parameters `name` (*str*) – The attribute name.

Returns Reformatted attribute name, useable by python.

```
>>> FileFormat.name_attribute('tHis is A Silly naME')
't_this_is_a_silly_na_me'
>>> FileFormat.name_attribute('Test:Something')
'test_something'
>>> FileFormat.name_attribute('unknown?')
'unknown'
```

classmethod `name_class` (*name*)

Converts a class name, as in the xsd file, into a name usable by python.

Parameters **name** (*str*) – The class name.

Returns Reformatted class name, useable by python.

```
>>> FileFormat.name_class('this IS a sillyNAME')
'ThisIsASillyNAME'
```

classmethod **name_parts** (*name*)

Intelligently split a name into parts:

- first, split at non-alphanumeric characters
- next, seperate digits from characters
- finally, if some part has mixed case, it must be camel case so split it further at upper case characters

```
>>> FileFormat.name_parts("hello_world")
['hello', 'world']
>>> FileFormat.name_parts("HELLO_WORLD")
['HELLO', 'WORLD']
>>> FileFormat.name_parts("HelloWorld")
['Hello', 'World']
>>> FileFormat.name_parts("helloWorld")
['hello', 'World']
>>> FileFormat.name_parts("xs:NMTOKEN")
['xs', 'NMTOKEN']
>>> FileFormat.name_parts("xs:NCName")
['xs', 'N', 'C', 'Name']
>>> FileFormat.name_parts('this IS a sillyNAME')
['this', 'IS', 'a', 'silly', 'N', 'A', 'M', 'E']
>>> FileFormat.name_parts('tHis is A Silly naME')
['t', 'His', 'is', 'A', 'Silly', 'na', 'M', 'E']
```

static **version_number** (*version_str*)

Converts version string into an integer. This default implementation simply returns zero at all times, and works for formats that are not versioned.

Override for versioned formats.

Parameters **version_str** (*str*) – The version string.

Returns A version integer. A negative number denotes an invalid version.

classmethod **walk** (*top*, *topdown=True*, *mode='rb'*)

A generator which yields all files in directory *top* whose filename matches the regular expression [RE_FILENAME](#). The argument *top* can also be a file instead of a directory. Errors coming from `os.walk` are ignored.

Note that the caller is not responsible for closing the stream.

This function is for instance used by `pyffi.spells` to implement modifying a file after reading and parsing.

Parameters

- **top** (*str*) – The top folder.
- **topdown** (*bool*) – Determines whether subdirectories should be iterated over first.
- **mode** (*str*) – The mode in which to open files.

classmethod **walkData** (*top*, *topdown=True*, *mode='rb'*)

A generator which yields the data of all files in directory *top* whose filename matches the regular expression

`RE_FILENAME`. The argument `top` can also be a file instead of a directory. Errors coming from `os.walk` are ignored.

Note that the caller is not responsible for closing the stream.

This function is for instance used by `pyffi.spells` to implement modifying a file after reading and parsing.

Parameters

- **top** (`str`) – The top folder.
- **topdown** (`bool`) – Determines whether subdirectories should be iterated over first.
- **mode** (`str`) – The mode in which to open files.

1.7.4 How to contribute

Do you want to fix a bug, improve documentation, or add a new feature?

Get git/msysgit

If you are on windows, you need [msysgit](#). If you are already familiar with subversion, then, in a nutshell, `msysgit` is for git what TortoiseSVN is for subversion. The main difference is that `msysgit` is a command line based tool.

For more information about git and github, the [github help site](#) is a great start.

Track the source

If you simply want to keep track of the latest source code, start a shell (or, the Git Bash on windows), and type (this is like “svn checkout”):

```
git clone git://github.com/niftools/pyffi.git
```

To synchronize your code, type (this is like “svn update”):

```
git pull
```

Development

Create a fork

- Get a [github account](#).
- [Log in](#) on github and [fork PyFFI](#) (yes! merging with git is easy so forking is encouraged!).

Use the source

PyFFI is entirely written in pure Python, hence the source code runs as such on any system that runs Python. Edit the code with your favorite editor, and install your version of PyFFI into your Python tree to enable your PyFFI to be used by other applications such as for instance QSkope, or the Blender NIF Scripts. From within your PyFFI git checkout:

```
C:\Python25\python.exe setup.py install
```

or on linux:

```
python setup.py install
```

To build the documentation:

```
cd docs
make html -a
```

PyFFI has an extensive test suite, which you can run via:

```
python rundoctest.py
```

The Blender NIF Scripts test suite provides additional testing for PyFFI. From within your niftools/blender checkout:

```
./install.sh
blender -P runtest_xxx.py
```

To build the source packages and the Windows installer (on a linux system which has both wine and nsis installed):

```
makezip.sh
```

Submit your updates

Simply do a [pull request](#) if you want your fork to be merged, and your contributions may be included in the next release!

1.7.5 Authors

Main author

- Amorilia (amorilia@users.sourceforge.net)

Previous Developers

- wz (wz_@users.sourceforge.net)
 - niftester (the current spells module is a revamped version of that)
 - nifvis (which hopefully will be embedded into QSkope one day...)
- taarna23 (taarna23@users.sourceforge.net)
 - extraction of DXT compressed embedded textures for the nif format
- tazpn (tazpn@users.sourceforge.net)

- mopp generator using the Havok SDK
 - suggestions for improving the spells module
- Scanti
 - EGM & TRI format info
- Carver13
 - EGM & TRI format xml

Contributors

- PacificMorrowind (pacmorrowind@sourceforge.net)
 - some nif/kf modifying spells
- Ulrim/Arthmoor
 - optimization kit
- seith (seith@users.sourceforge.net)
 - logo design for the Windows installer
- MorCroft
 - Patch for BSSTextureSet texture path substitution

1.7.6 License

Copyright © 2007-2012, Python File Format Interface. All rights reserved.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

- Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
- Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.
- Neither the name of the Python File Format Interface project nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS “AS IS” AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

PyFFI uses Mopper. Copyright 2008 NIF File Format Library and Tools. All rights reserved. Run `pyffi/utils/mopper.exe --license` for details.

Moppor uses Havok. Copyright 1999-2008 Havok.com Inc. (and its Licensors). All rights reserved. See <http://www.havok.com> for details.

PyFFI uses xdelta3. Copyright 2001-2010 Joshua P. MacDonald xdelta3 is released under the GPLv2. See external/xdelta3.0z/COPYING for details.

1.7.7 ChangeLog

Release 2.2.3 (Mar 17, 2014)

- Update spell texture_path_substitution for BSTextureSet blocks (fix contributed by MorCroft)
- Updated to latest nif.xml, submodules moved to github.

Release 2.2.2 (Nov 17, 2012)

- Use VERSION file.
- Fix dump_python for correct default values.
- Fix xml for Fallout 3.

Release 2.2.1 (Nov 11, 2012)

- Added Skyrim version detection.
- The check_readwrite spell now handles file size differences due to empty strings.
- Bugfix in nif write method when entities are None (see Skyrim meshes/actors/character/character assets/hair/hairlonghumanm.nif).
- Various fixes for Fallout 3 and Fallout NV nifs.
- New dump_python spell, to generate python code that recreates a nif file.
- Accept string palette strings that do not have a null character preceeding it (reported by Koniption).
- New modify_getbonepriorities and modify_setbonepriorities spells, which work similar to the kfupdater.
- New fix_fallout3stringoffsets spell to 'fix' empty offsets in Oblivion style kf files, if they are to be used in Fallout 3.
- Installer no longer targets Maya and Blender.

Release 2.2.0 (Nov 26, 2011)

- Added PSK and PSA file support (used by Unreal engine).
- A py3k fix (contributed by infectedsoundsystem).
- Updated installer for Blender 2.5x+.

Release 2.1.11 (Nov 26, 2011)

- Explicitly use wine for running mopper on non-win32 platforms (fixes issue on Arch Linux, reported by ifss000f, see issue #3423990).
- Removed skip list from extra fix_texturepath stage in Oblivion optimization kit.
- Various optimizations (contributed by infectedsoundsystem). The optimizer spell now runs a fair bit faster.
- Garbage collection call after each spell has been removed as profiling showed that a lot of time was spent on it. You can still force the old (slow) behaviour by using the new `-gccollect` command line option or adding `"gccollect = True"` in your ini file.
- Encoding fix for xml and xsd parsing.
- Merge duplicates after optimizing geometry to work around de-duplication during geometry optimization phase (fixes issue #3425637, reported by chacky2).
- Removed xsd object model and dae format (wasn't functional yet anyway); deferred to py3k.

Release 2.1.10 (Oct 10, 2011)

- Fixed bspline data methods to handle invalid kfs with missing basis data (reported by K'Aviash).
- Fixed mass, center, inertia methods to deal with cases where shape is missing (reported by rlibiez, see niftools issue #3248754).
- Fixed center calculation of bhkListShape collisions, and fixed zero division error when creating very small collision shapes (reported by Koniption, see issues #3334577 and #3308638).
- Fixed shortcut to uninstaller in programs folder (reported by neomonkeus, see niftools issue #3397540).
- Fixed geometry optimizer to handle cases where number of morph vertices does not match number of shape vertices (reported by rlibiez, see issue #3395484).
- Merged ulrim's optimization kit, along with arthmoor's improved ini files.
- Integrated far nif optimization with the general purpose optimize spell (requested by Gratis_monsta, see issue #3177316).
- New shell_optimize.ini for configuring optimization as executed from Windows shell.
- Allow .ini files that do not have a [main] or [options] section.
- Fix Windows shell integration to point to the new shell_optimize.ini file (reported by rlibiez, see issue #3415490).
- Fixed zombie process problem on Windows when a toaster was running with multiple jobs (reported by Alphanos, see issue #3390826).

Release 2.1.9 (Mar 26, 2011)

- Improved documentation of .dir/.img unpack and pack scripts.
- Bugfix in .dir format, so it can now also handle single record images.
- Added new script to make and apply patches (functionality is identical to and OnmyojiOmn's and jonwd7's pyffi patcher scripts, but it is written in Python to work on all platforms).
- New fix_emptyskeletonroots spell (automatically included in the optimize spell) to fix issues with nifs that do not have their NiSkinInstance skeleton root set (fixes #3174085, reported by xjdhdr).

- Fixed logging issue on Windows platform with multithreading enabled (fixes issue #3174339, reported by xjd-hdr).
- Fixed QSkope shortcut issue when path contains spaces (reported by Brumbek).
- Added support for BSPackedAdditionalGeometryData (reported by Ghostwalker71, niftools issue #3177847).
- Skip terminal chars in mopper output (fixes issues with running mopper under wine).
- Bugfix in patch_recursive_make/apply scripts for “deep” folder layouts (fixes issue #3193914, reported by xjd-hdr).
- Do not pack collisions in OL_CLUTTER (fixes issue #3194017 reported by Gratis_monsta).
- Fixed issue with skipping terminal chars in mopper output on Windows platform (fixes issue #3205569, reported and diagnosed by ulrim).
- Updates for Bully SE format (fixes issue reported by Tosyk).
- Bully SE nif header reading fix for BBonusB.nft.
- Added initial support for Epic Mickey (reported and test files provided by Tosyk).
- Bugfix for NiMesh read and write.
- Updated dump_pixeldata spell to enable it to export Bully SE’s nft files.
- Disabled copy in patch_recursive_apply script (see issue #3219744, suggested by ulrim).
- Pass additional arguments of patch_recursive_apply/make to the patch command (see issue #3219744, suggested by ulrim).
- Fix nif optimizer in case it contains tangent space data but no uv data (see issue #3218751, reported by krimhorn).
- Handle removal of redundant triangles when updating dismember skin partitions (see issue #3218751, reported by krimhorn).
- Fix vertex cache optimizer to handle more meshes with more than 255 triangles per vertex (see issue #3218751, reported by krimhorn).
- Skipping meshes that have NiAdditionalGeometryData (until we understand what this data does and how to optimize it).
- Sane default settings for bhkRigidBody unknowns to ensure that constraints behave properly (contributed by Koniption).
- Added ini file to unpack Bully SE .nft files.

Release 2.1.8 (Feb 4, 2011)

- Quickhull bugfix for precision argument in degenerate cases (issue #3163949, fix contributed by liuhuanjim013).
- Fixed issue with detecting box shapes on degenerate collision meshes (fixes issue #3145104, reported by Gratis_monsta).
- Ensure that enum has valid default value.
- Added CStreamableAssetData for Divinity 2 (reported by pertinen, niftools issue #3164929).
- NiPixelData.save_as_dds fourcc flag bugfix.
- Added Rockstar .dir format (used in Bully SE).
- Added Rockstar .dir/.img unpack and pack scripts (only tested for Bully SE).

Release 2.1.7 (Jan 23, 2011)

- Added support for Fallout New Vegas (contributed by throttlekitty and saiden).
- Updated geometry optimizer to keep dismember body parts, for Fallout 3 and Fallout New Vegas (fixes issue #3025691 reported by Chaky).
- Added flag to enable debugging in vertex cache algorithm, to assess how suboptimal the solution is on any particular mesh (testing reveals that it finds the globally optimal solution in more than 99% of all iterations, for typical meshes).
- New `check_triangles_atvr` spell to find optimal parameters for vertex cache algorithm by simulated annealing.
- Fixed `send_geometries_to_bind_position`, `send_detached_geometries_to_node_position`, and `send_bones_to_bind_position` in case skin instance has empty bone references (fixes issue #3114079, reported by drakonnen).
- Fix for verbose option in multithread mode (reported by Gratis_monsta).
- Fix optimize spell when no vertices are left after removing duplicate vertices and degenerate triangles (reported by Gratis_monsta).
- Fixed tangent space issue along uv seams (reported by Gratis_monsta and Tommy_H, see issue #3120585).
- Log an error instead of raising an exception on invalid enum values (fixes issue #3127161, reported by rlibiez).
- Disabled 2to3 in Windows installer; the Python 3 version of PyFFI will be released separately.

Release 2.1.6 (Nov 13, 2010)

- The optimize spell now includes two new spells: `opt_collisiongeometry` for optimizing triangle based collisions, and `opt_collisionbox` for optimizing simple box collisions. This should result in faster loads and probably also a small but noticeable performance improvement.
- Fixed `opt_collisiongeometry` for multimaterial mopps (reported by wildcard_25, see issue #3058096).
- New `SpellCleanFarNif` spell (suggested by wildcard_25, see issue #3021629).
- Bad normals are now ignored when packing a `bhkNiTriStripsShape` (fixes issue #3060025, reported by rlibiez).
- The `opt_delunusedbones` spell no longer removes bones if they have a collision object (fixes issue #3064083, reported by wildcard_25).
- If the jobs option is not specified in the toaster, then the number of processors is used—requires Python 2.6 or higher (suggested by chaky2, see issue #3052715, implements issue #3065503).
- New `opt_delzeroscale` spell to delete branches with zero scale (suggested by chaky2, see issue #3013004).
- The `opt_mergeduplicates` spell now ignores (non-special) material names, so identical materials with different names will get merged as well (suggested by chaky2, see issue #3013004).
- New spell to fix subshape counts (see issue #3060025, reported by rlibiez), it is included in the optimize spell.
- New `opt_collisionbox` spell which automatically converts triangle based box collisions to primitive box collisions, which are much faster in-game (contributed by PacificMorrowind).
- Optimizer spell now uses triangles to represent skin partitions (improves in-game fps).
- Better vertex map calculation when calculating skin partitions (improves in-game fps).
- Optimizer now always triangulates (improves in-game fps). Stripification will be readded later in a modularized version of the optimizer spell, for those that want minimal file size rather than maximal performance).
- Much faster implementation of vertex cache algorithm (now runs in linear time instead of quadratic time).

- Check triangle count when converting to box shape (fixes issue #3091150).
- Bugfix in vertex map reordering (fixes most nifs reported in issue #3071616).
- Bugfix in vertex cache algorithm (fixes a nif reported in issue #3071616).
- Cover degenerate case in ATVR calculation when there are no vertices (fixes a nif reported in issue #3071616).
- Cover degenerate case when calculating cache optimized vertex map (fixes a nif reported in issue #3071616).
- Remove branches if they have no triangles (again fixes a nif reported in issue #3071616).

Release 2.1.5 (Jul 18, 2010)

- Improved interface for TRI files, and a bugfix in TRI file writing.
- Added EGT file support.
- The `fix_texturepath` spell now also converts double backslash in single backslash (suggested by Baphometal).
- Bugfix in `save_as_dds` function for newer `NiPixelData` blocks (reported by norocelmiau, issue #2996800).
- Added support for Laxe Lore nifs (reported by bobsobol, issue #2995866).
- New spells:
 - `opt_collisiongeometry`: to optimize collision geometry in nifs (contributed by PacificMorrowind).
 - `check_materialemissivevalue`: checks (and warns) about high values in material emissive settings (contributed by PacificMorrowind).
 - `modify_mirroranimation`: mirrors an animation (specifically left to right and vice versa) - use it to for example turn a right hand punch anim into a left hand punch anim (contributed by PacificMorrowind).
- Added big-endian support.
- Removed `**kwargs` argument passing for faster and more transparant implementation (reading and writing is now about 8% faster).
- Do not merge `BSShaderProperty` blocks (reported by Chaky, niftools issue #3009832).
- Installer now recognizes Maya 2011.
- Fixed `NiPSysData` read and write for Fallout 3 (reported by Chaky, niftools issue #3010861).

Release 2.1.4 (Mar 19, 2010)

- Extra names in `oblivion_optimize.ini` skip list for known mods (contributed by Tommy_H).
- New spells
 - `modify_collisiontomopp`
 - `opt_reducegeometry`
 - `opt_packcollision`
- Windows right-click optimize method now uses `default.ini` and `oblivion_optimize.ini`.
- `fix_texturepath` now fixes paths that include the whole drive path to just the textures/... path.
- The optimize spell has been fixed to update Fallout 3 style tangent space (fixes issue #2941568, reported by xjdhdr).

Release 2.1.3 (Feb 20, 2010)

- Added toaster option to process files in archives (not yet functional).
- Added toaster option to resume, by skipping existing files in the destination folder.
- Toaster now removes incompletely written files on CTRL-C (to avoid corrupted files).
- Fixed makefarnif spell (now no longer deletes vertex colors).
- New spells
 - fix_delunusedroots
 - modify_bonepriorities
 - modify_interpolatortransrotscale
 - modify_delinterpolatortransformdata
 - opt_delunusedbones
- The niftoaster optimize spell now also includes fix_delunusedroots.
- Removed unused pep8 attribute conversion code.
- Toasters can now be configured from an ini file.
- bhkMalleableHinge update_a_b bugfix (reported by Ghostwalker71).

Release 2.1.2 (Jan 16, 2010)

- Fallout 3 skin partition flag bugfix (reported by Ghostwalker71).
- Fixed bug in optimize spell, when has_vertex_colors was False but vertex color array was present (reported by Baphometal, debugged by PacificMorrowind).
- Initial bsa file support (Morrowind, Oblivion, and Fallout 3).

Release 2.1.1 (Jan 11, 2010)

- Accidentally released corrupted nif.xml (affected Fallout 3), so this is just a quick bugfix release including the correct nif.xml.

Release 2.1.0 (Jan 10, 2010)

- Improved windows installer.
- Compatibility fix for Python 2.5 users (reported by mac1415).
- Renamed some internal modules for pep8 compliance.
- All classes and attributes are now in pep8 style. For compatibility, camelCase attributes are generated too (however this will be dropped for py3k).
- Renamed a few niftoaster spells.
 - fix_strip -> modify_delbranches
 - fix_disableparallax -> modify_disableparallax
- New niftoaster spells.

- `fix_cleanstringpalette`: removes unused strings from string palette.
 - `modify_substitutestringpalette`: regular expression substitution of strings in a string palette.
 - `modify_scaleanimationtime`: numeric scaling of animations.
 - `modify_reverseanimation`: reverses an animation (ie useful for making only an open animation and then running this to get a close animation).
 - `modify_collisionmaterial`: sets any collision materials in a nif to specified type.
 - `modify_delskinshapes`: Delete any geometries with a material name of 'skin'
 - `modify_texturepathlowres`: Changes the texture path by replacing 'textures/' with 'textures/lowres/'. used mainly for making `_far`.nifs.
 - `modify_addstencilprop`: Adds a `NiStencilProperty` to each geometry if it is not present.
 - `modify_substitutetexturepath`: regular expression substitution of a texture path.
 - `modify_makeskinlessnif`: Spell to make fleshless CMR (Custom Model Races) clothing/armour type nifs. (runs `modify_delskinshapes` and `modify_addstencilprop`)
 - `modify_makefarnif`: Spell to make `_far` type nifs.
- Bugfix for `niftoaster dump` spell.
 - New `-suffix` option for toaster (similar to the already existing `-prefix` option).
 - New `-skip` and `-only` toaster options to toast files by regular expression.
 - New `-jobs` toaster option which enables multithreaded toasting.
 - New `-source-dir` and `-dest-dir` options to save toasted nifs in a given destination folder.
 - Added workaround for memory leaks (at the moment requires `-jobs >= 2` to be functional).
 - The `niftoaster opt_geometry` spell now always skips NIF files when a similarly named tri or egm file is found.
 - Added support for Atlantica nifs.
 - Added support for Joymaster Interactive Howling Sword nifs.

Release 2.0.5 (Nov 23, 2009)

- Added regression test and fixed rare bug in stripification (reported by PacificMorrowind, see issue #2889048).
- Improved strip stitching algorithm: *much* more efficient, and now rarely needs more than 2 stitches per strip.
- Improved stripifier algorithm: runs about 30% faster, and usually yields slightly better strips.
- Added new `modify_texturepath` and `modify_collisiontype` `niftoaster` spells (contributed by PacificMorrowind).
- Various fixes and improvements for 20.5.0.0+ nifs.
- Check endian type when processing nifs.
- Source release now includes missing `egm.xml` and `tri.xml` files (reported by skomut, fixes issue #2902125).

Release 2.0.4 (Nov 10, 2009)

- Write NaN on float overflow.
- Use pytristrip if it is installed.
- Implemented the FaceGen egm (done) and tri (in progress) file formats with help of Scanti and Carver13.
- The nif dump_pixeldata spell now also dumps NiPersistentSrcTextureRenderData (reported by lusht).
- Set TSpace flags 16 to signal presence of tangent space data (fixes Fallout 3 issue, reported by Miaximus).

Release 2.0.3 (Sep 28, 2009)

- Various bugfixes for the Aion cgf format.
- Updates for nif.xml to support more recent nif versions (20.5.0.0, 20.6.0.0, and 30.0.0.2).

Release 2.0.2 (Aug 12, 2009)

- The source has been updated to be Python 3.x compatible via 2to3.
- New unified installer which works for all versions of Python and Maya at once (at the moment: 2.5, 2.6, 3.0, 3.1) and also for all versions of Maya that use Python 2.5 and 2.6 (2008, 2009, and 2010, including the 64 bit variants).
- Added support for Aion cgf files.
- Added support for NeoSteam header and footer.
- Log warning rather than raising exception on invalid links (fixes issue #2818403 reported by abubakr125).
- Optimizer can now recover from invalid indices in strips (this fixes some nifs mentioned in issue #2795837 by baphometal).
- Skin updater can now recover when some vertices have no weights (this fixes some nifs mentioned in issue #2795837 by baphometal).
- Skip zero weights and add up weights of duplicated bones when calculating vertex weights (this fixes some nifs mentioned in issue #2795837 by baphometal).
- The nif optimizer can now handle NiTriShapeData attached as a NiTriStrips data block (fixes some corrupt nifs provided by baphometal in issue #2795837).
- Optimizer can now recover from NaN values in geometry (sample nifs provided by baphometal).
- Do not attempt to optimize nifs with an insane amount of triangles, but put out a warning instead.
- Log error rather than raising exception when end of NIF file is not reached (fixes issue with sample nif provided by baphometal).

Release 2.0.1 (Jul 22, 2009)

- Added Windows installer for Python 2.6.
- Updated mopper.exe compiled with msvc 2008 sp1 (fixes issue #2802413, reported by pacmorrowind).
- Added pdb session to track circular references and memory leaks (see issues #2787602 and #2795837 reported by alexkapi12 and xfrancis147).
- Added valgrind script to check memory usage, and to allow keeping track of it between releases (see issues #2787602 and #2795837 reported by alexkapi12 and xfrancis147).
- Removed parenting in xml model from everywhere except Array, and using weakrefs to avoid circular references, which helps with garbage collection. Performance should now be slightly improved.
- Updates to xml object model expression syntax.
 - Support for field qualifier ‘.’.
 - Support for addition ‘+’.
- Updates to Targa format.
 - Support for RLE compressed Targa files (test file contributed by Alphax, see issue #2790494).
 - Read Targa footer, if present (test file contributed by Alphax, see issue #2790494).
 - Improved interface: header, image, and footer are now global nodes.
- Updates to xsd object model.
 - Classes and attributes for Collada format are now generated (but not yet functional).

Release 2.0.0 (May 4, 2009)

- Windows installer now detects Maya 2008 and Maya 2009, and their 64 bit variants, and can install itself into every Maya version that is found.
- Updates to the XML object model (affects CGF, DDS, KFM, NIF, and TGA).
 - Class customizers are taken immediately from the format class, and not from separate modules — all code from customization modules has been moved into the main format classes. The result is that parsing is faster by about 50 percent.
 - clsFilePath removed, as it is no longer used.
- Updates and fixes for the KFM format.
 - The Data element inherits from Header, and Header includes also all animations, so it is more straightforward to edit files.
 - The KFM files open again in QSkope.
- Updates for the CGF format.
 - CHUNK_MAP no longer constructed in Data.__init__ but in a metaclass.
 - Deprecated functions in CgfFormat have been removed.
- Updates for the NIF format.
 - Synced nif.xml with nifskope’s xml (includes fixes for Lazeska).
 - Removed deprecated scripts (niftextdump, nifdump, fftv3rskinpartition, nifoptimize).

- Fixed scaling bug on nifs whose tree has duplicate nodes. Scaling now no longer works recursively, unless you use the scaling spell which handles the duplication correctly.
- Updated module names to follow pep8 naming conventions: all modules have lower case names.

Release 1.2.4 (Apr 21, 2009)

- Documentation is being converted to Sphinx. Currently some parts of the documentation are slightly broken with epydoc. Hopefully the migration will be complete in a month or so, resolving this issue.
- removed deprecated PyFFI.Spells code:
 - old style spells no longer supported
 - almost all old spells have been converted to the new spell system (the few remaining ones will be ported for the next release)
- nif:
 - nif optimizer can be run on folders from the windows context menu (right-click on any folder containing nifs and select “Optimize with PyFFI”)
 - synced nif.xml with upstream (adds support for Worldshift, bug fixes)
 - using weak references for Ptr type (this aids garbage collection)
 - added fix_strip niftoaster spell which can remove branches selectively (feature request #2164309)
 - new getTangentSpace function for NiTriBasedGeom (works for both Oblivion and Fallout 3 style tangent spaces)
 - improved mergeSkeletonRoots function (will also merge roots of skins that have no bones in common, this helps a lot with Morrowind imports)
 - new sendDetachedGeometriesToNodePosition function and spell (helps a lot with Morrowind imports)
- tga:
 - added support for color map and image data in the xml
 - uses the new data model
 - works again in QSkope
- xml object model:
 - added support for multiplication and division operators in expressions
- fixes for unicode support (prepares for py3k)

Release 1.2.3 (Apr 2, 2009)

- removed reduce() calls (py3k compatibility)
- started converting print calls (py3k compatibility)
- removed relative imports (py3k compatibility)
- removed BSDiff module (not useful, very slow, use external bsdiff instead)
- nif:
 - fixed the update mopp spell for fallout 3 nifs
 - fixed addShape in bhkPackedNiTriStripsShape for fallout 3 nifs

- niftoaster sends to stdout instead of stderr so output can be captured (reported by razorwing)

Release 1.2.2 (Feb 15, 2009)

- cgf format:
 - fixed various regression bugs that prevented qskope to run on cgf files
 - updated to use the new data system

Release 1.2.1 (Feb 2, 2009)

- nif format:
 - new addIntegerExtraData function for NiObjectNET

Release 1.2.0 (Jan 25, 2009)

- installer directs to Python 2.5.4 if not installed
- using logging module for log messages
- nif format:
 - swapping tangents and binormals in xml; renaming binormals to bitangents (see <http://www.terathon.com/code/tangent.html>)
 - updates for Fallout 3 format
 - updated skin partition algorithm to work for Fallout 3
 - * new triangles argument
 - * new facemap argument to pre-define partitions (they will be split further if needed to meet constraints)
 - * sort vertex weight list by weight in skin partitions (except if padbones is true; then sorted by bone index, to keep compatibility with ffvt3r)
 - * option to maximize bone sharing
 - mopps take material indices into account and compute welding info (attempt to fix mopp multi-material issues, does not yet seem to work though)
 - support for niftools bitflags by converting it to a bitstruct on the fly
 - better algorithm for sending bones to bind position, including spells for automating this function over a large number of nifs
 - disable fast inverse in bind pos functions to increase numerical precision
 - new algorithm to sync geometry bind poses, along with spell (this fixes many issues with Morrowind imports and a few issues with Fallout 3 imports)
 - more doctests for various functions
 - a few more matrix functions (supNorm, subtraction)
- dds format:
 - updated to use the FileFormat.Data method (old inconvenient method removed)
- qskope:

- refactored the tree model
- all parenting functions are delegated to separate DetailTree and GlobalTree classes
- the DetailNode and GlobalNode classes only implement the minimal functions to calculate the hierarchy, but no longer host the more advanced hierarchy functions and data (this will save memory and speed up regular use of pyffi outside qscope)
- EdgeFilter for generic edge type filtering; this is now a parameter for every method that needs to list child nodes

Release 1.1.0 (Nov 18, 2008)

- nif format:
 - a large number of functions have moved from the optimizer spell to the main interface, so they can be easily used in other scripts without having to import this spell module (getInterchangeableTriShape, getInterchangeableTriStrips, isInterchangeable)
 - new convenience functions in NiObjectNET, NiAVObject, and NiNode (setExtraDatas, setProperties, setEffects, setChildren, etc.)
 - updates for Fallout 3
- niftoaster
 - new fix_addtangentspace spell to add missing tangent space blocks
 - new fix_deltangentspace spell to remove tangent space blocks
 - new fix_texturepath spell to change / into \ and to fix corrupted newline characters (which sometimes resulted from older versions of nifskope) in NiSourceTexture file paths
 - new fix_clampmaterialalpha spell
 - new fix_detachhavoktristripsdata spell
 - the ffvt3r skin partition spell is now fix_ffvt3rskinpartition
 - new opt_cleanreflists spell
 - new opt_mergeduplicates spell
 - new opt_geometry spell
 - the optimize spell is now simply implemented as a combination of other spells
- new internal implementation of bsdiff algorithm
- removed cry dae filter (an improved version of this filter is now bundled with ColladaCGF)
- reorganization of file format description code
 - all generic format description specific code has been moved to the PyFFI.ObjectModels.FileFormat module
 - all xml/xsd description specific code has been moved to the PyFFI.ObjectModels.XML/XSD.FileFormat modules
 - new NifFormat.Data class which now implements all the NIF file read and write functions
- completely revamped spell system, which makes it much easier to customize spells, and also enables more efficient implementations (thanks to tazpn for some useful suggestions, see issue #2122196)
 - toaster can call multiple spells at once

- toaster takes spell classes instead of modules
- for backwards compatibility, there is a class factory which turns any old spell module into a new spell class (the Toaster class will automatically convert any modules that it finds in its list of spells, so you do not need to be worried about call the factory explicitly)
- early inspection of the header is possible, to avoid having to read all of the file if no blocks of interest are present
- possibility to prevent the spell to cast itself on particular branches (mostly useful to speed up the spell casting process)
- spells have callbacks for global initialization and finalization of data within the toaster
- possibility to write output to a log file instead of to sys.stdout
- better messaging system (auto indentation, list nif tree as spell runs)
- support for spell hierarchies and spell grouping, in parallel or in series or any combination of these
- replaced ad hoc class customization with partial classes (still wip converting all the classes)
- xml object model expression parser
 - implemented not operator
 - expressions can combine multiple operators (only use this if the result is independent of the order in which these operators are applied)
 - new < and > operators
 - support for vercond attribute for Fallout 3
- started on a new object model based on an ANTLR parser of a grammar aimed at file format descriptions; this parser will eventually yield a more streamlined, more optimized, and more customizable version of the current xml object model (this is not yet bundled with the release, initial code is on svn)

Release 1.0.5 (Sep 27, 2008)

- niftoaster optimize
 - fix for materials named skin, envmap2, etc. (issue #2121098)
 - fix for empty source textures in texdesc (issue #2118481)
- niftoaster
 - new spell to disable parallax (issue #2121283)
- toaster
 - new options –diff and –patch to create and apply patches; internal patcher uses bsdiff format, but you can also specify an arbitrary external diff/patch command via –diff-cmd and –patch-cmd options (the external command must take three arguments: oldfile, newfile, and patchfile); note that this is still in experimental stage, not ready for production use yet

Release 1.0.4 (Sep 18, 2008)

- niftoaster optimize
 - morph data optimization (issue #2116594, fixes “bow” weapons)

Release 1.0.3 (Sep 17, 2008)

- niftoaster optimize
 - detach NiTriStripsData from havok tree when block is shared with geometry data (fixes issue #2065018, MiddleWolfRug01.NIF)
 - fix in case merged properties had controllers (issue #2106668)
- fix writing of block order: bhkConstraint entities now always precede the constraint block (this also fixes the “falling sign” issue with the niftoaster optimize spell, issue #2068090)

Release 1.0.2 (Sep 15, 2008)

- “negative mass” fix in inertia calculation

Release 1.0.1 (Sep 12, 2008)

- small fix in uninstaller (didn’t remove crydaefilter script)
- crydaefilter converts %20 back into spaces (as rc doesn’t recognize %20)
- bugfixes for niftoaster optimize spell (pyffi issue #2065018)

Release 1.0.0 (Jul 24, 2008)

- new NSIS installer (this solves various issues with Vista, and also allows the documentation to be bundled)
- new filter to prepare collada (.dae) files for CryEngine2 resource compiler
 - wraps scenes into CryExportNodes
 - corrects id/sid naming
 - fixes init_from image paths
 - adds phong and lambert shader sid’s
 - enforces material instance symbol to coincide with target
 - sets material names in correct format for material library and physicalization
- started on support for collada format, by parsing the collada xsd schema description (this is still far from functional, but an initial parser is already included with the library, although it does not yet create any classes yet)
- fully optimal mopp generation for Oblivion (using the NifTools mopper.exe which is a command line utility that calls the mopp functions in the havok library, credit for writing the original wrapper goes to tazpn)
- minor updates to the nif.xml format description
- refactoring: library reorganized and some interfaces have been unified, also a lot of code duplication has been reduced; see README.TXT for more details on how to migrate from 0.x.x to 1.x.x
 - main format classes PyFFI.XXX have been moved to PyFFI.Formats.XXX

- “XxxFormat.getVersion(cls, stream)” now always returns two integers, version and user_version
 - “XxxFormat.read(self, stream, version, user_version, ...)” for all formats
 - “XxxFormat.write(self, stream, version, user_version, *readresult, ...)” for all formats
 - in particular, CGF format game argument removed from read and write functions, but there are new CgfFormat.getGame and CgfFormat.getGameVersion functions to convert between (version, user_version) and game
 - also for the CGF format, take care that getVersion no longer returns the file type. It is returned with the CgfFormat.read function, however there is a new CgfFormat.getFileType function, if you need to know the file type but you don’t want to parse the whole file
 - all XxxFormat classes derive from XmlFileFormat base class
 - common nameAttribute, walk, and walkFile functions
 - XxxTester modules have been moved to PyFFI.Spells.XXX, along with a much improved PyFFI.Spells module for toasters with loads of new options
 - some other internal code has been moved around
 - * qskopelib -> PyFFI.QSkope
 - * PyFFI.Bases -> PyFFI.ObjectModels.XML
 - a lot more internal code reorganization is in progress...
- much documentation has been added and improved

Release 0.11.0 (Jun 16, 2008)

- nif:
 - fixed updateTangentSpace for nifs with zero normals
- cfg:
 - a lot of new physics stuff: MeshPhysicsDataChunk mostly decoded (finally!!)
 - fixes for reading and writing caf files (they are missing controller headers)
 - activated BoneMeshChunk and BoneInitialPosChunk for Crysis
- tga:
 - improved tga file detection heuristic

Release 0.10.10 (Jun 8, 2008)

- nif:
 - minor updates in xml
 - NiPixelData saveAsDDS function now also writes DXT compressed formats, that is, pixel formats 4, 5, and 6 (contributed by taarna23)
 - fixed nifoptimize for nifs with particle systems (niftools issue #1965936)
 - fixed nifoptimize for nifs with invalid normals (niftools issue #1987506)

Release 0.10.9 (May 27, 2008)

- nif:
 - bspline interpolator fix if no keys
 - fixed bspline scale bug

Release 0.10.8 (Apr 13, 2008)

- cgf:
 - more decoded of the mesh physics data chunk
- nif:
 - scaling for constraints
 - ported the A -> B spell from nifscope (see the new getTransformAB and updateAB methods)

Release 0.10.7 (Apr 5, 2008)

- cgf:
 - indices are unsigned shorts now (fixes geometry corruption on import of large models)
 - MeshChunk.setGeometry gives useful error message if number of vertices is too large
- nif:
 - nif.xml has minor updates in naming
 - added NiBSplineData access functions (experimental, interface could still change)
 - started on support for compressed B-spline data
 - fixed block order writing of bhkConstraints

Release 0.10.6 (Mar 30, 2008)

- tga: added missing xml file
- nif:
 - removed some question marks so the fields can be accessed easily in python interface
 - ControllerLink and StringPalette functions and doctests
 - quaternion functions in Matrix33 and Matrix44
 - new bspline modules (still to implement later)
 - fixed NiTransformInterpolator scaling bug
- cgf:
 - use tempfile for write test
- quick install batch file for windows

Release 0.10.5 (Mar 27, 2008)

- qskope: make bitstructs editable
- cgf:
 - MeshChunk functions to get vertex colors (game independent).
 - Set vertex colors in setGeometry function.

Release 0.10.4 (Mar 26, 2008)

- cgf:
 - fixed tangent space doctest
 - setGeometry argument sanity checking
 - setGeometry fix for empty material list
 - setGeometry tangent space update fix if there are no uvs

Release 0.10.3 (Mar 24, 2008)

- added support for the TGA format
- tangentspace:
 - validate normals before calculating tangents
 - added new option to get orientation of tangent space relative to texture space (Crysis needs to know about this)
- installer detects Maya 2008 and copies relevant files to Maya Python directory for the Maya scripts to work
- cgf:
 - tangent space cgftoaster
 - new MeshChunk updateTangentSpace function

Release 0.10.2 (Mar 22, 2008)

- cgf:
 - fixed “normals” problem by setting last component of tangents to -1.0
 - meshchunk function to get all material indices, per triangle (game independent)
 - scaling fixes for datastreamchunk, meshchunk, and meshsubsetschunk
 - fixed version of BreakablePhysicsChunk
 - a few new findings in decoding the physics data (position and rotation)

Release 0.10.1 (Mar 21, 2008)

- cgf:
 - some minor xml updates
 - setGeometry function for MeshChunk to set geometry for both Far Cry and Crysic in a unified way
 - uv.v opengl flip fix for Crysic MeshChunk data
- MathUtils: new function to calculate bounding box, center, and radius
- qscope: fixed bug which prevented setting material physics type to NONE

Release 0.10.0 (Mar 8, 2008)

- cgf: ported A LOT of stuff from the Crysic Mod SDK 1.2; the most common CE2 chunks now read and write successfully

Release 0.9.3 (Mar 7, 2008)

- cgf:
 - decoded a lot of geometry data
 - * vertices
 - * normals
 - * vertex colors
 - * uvs
 - * mesh material info
 - started decoding many other chunk types
 - added chr chunk types so files containing them can be processed (the data is ignored)
 - started adding functions to MeshChunk to have unified access to geometry data for both Far Cry and Crysic cgf files
- windows installer registers chr extension with qscope

Release 0.9.2 (Feb 26, 2008)

- full support for the xml enum tag type, with improved editor in qscope
- new common string types (shared between cgf and nif formats)
 - null terminated
 - fixed sized
 - variable sized starting with integer describing length
- qscope: no more duplicate ptr refs in global view
- qscope: refactored delegate editor system to be more transparent and much easier to extend
- cgf: crysis chunks have been partially decoded (still very much wip)
- cgf: added extra chunk size check on read to aid decoding

- dds: register dds extension with qskope on windows install
- nif: nifoptimize clamps material alpha to [0,1]

Release 0.9.1 (Feb 22, 2008)

- full support for the xml bitstruct tag (for types that contain bit flags)
- added PyFFI.Formats.DDS library for dds file format
- nif: new function for NiPixelData to save image as dds file
- niftoaster: script for exporting images from NiPixelData blocks
- nifoptimize:
 - merge identical shape data blocks
 - remove empty NiNode children
 - update skin partition only if block already exists

Release 0.9.0 (Feb 11, 2008)

- added PyFFI.Formats.KFM library for kfm file format
- cgf.xml and nif.xml updates
- new qBlockParent function to assign parents if the parent block does not contain a reference to the child, but the child contains a reference to the parent (as in MeshMorphTargetChunk and BoneInitialPosChunk)
- QSkope: root blocks sorted by reference number
- QSkope: added kfm format
- niftextdump: bug fixed when reading nifs that have textures without source

Release 0.8.2 (Jan 28, 2008)

- fixed installer bug (nifoptimize would not launch from context menu)
- qskope:
 - handle back-references and shared blocks
 - blocks are now numbered
 - improved display references

Release 0.8.1 (Jan 27, 2008)

- deep copy for structs and arrays
- nifoptimize:
 - detects cases where triangulated geometry performs better than stripified geometry (fixes a performance issue with non-smooth geometry reported by Lazarus)
 - can now also optimize NiTriShapes
 - throws away empty and/or duplicate children in NiNode lists

Release 0.8.0 (Jan 27, 2008)

- qskope: new general purpose tool for visualizing files loaded with PyFFI
- cgf: corrected the bool implementation (using True/False rather than an int)
- nif: many xml updates, support for Culpa Innata, updates for emerge demo
- support for forward declaration of types (required for UnionBV)
- PyFFI.__hexversion__ for numeric representation of the version number

Release 0.7.5 (Jan 14, 2008)

- added a DTD for the 'fileformat' document type, to validate the xml
- bits tag for bitstructs, instead of add tag, to allow validation
- cgf: write the chunk header table at start, for crisis
- nifoptimize:
 - new command line option '-x' to exclude blocks per type
 - fixes corrupted texture paths (that is, files that got corrupted with nifskope 1.0 due to the \r \n bug)
 - on windows, the script can now be called from the .nif context menu
 - accept both lower and upper case 'y' for confirmation
 - new command line option '-p' to pause after run
- niftoaster: fix reporting of file size difference in readwrite test
- bug fixed when writing nifs of version <= 3.1
- support for multiple 'Top Level Object' (roots) for nifs of version <= 3.1
- various xml fixes
 - new version 20.3.0.2 from emerge demo
 - NiMeshPSysData bugfix and simplification
 - replaced NiTimeController Target with unknown int to cope with invalid pointers in nif versions <= 3.1
- fixed bug nifmakehsl.py script
- fixed bug in nifdump.py script
- new post installation script for installing/uninstalling registry keys

Release 0.7.4 (Dec 26, 2007)

- fix in nif xml for a long outstanding issue which caused some nifs with mopp shapes to fail
- fixed file size check bug in readwrite test for nif and cgf
- initial read and write support for crisis cgf files
- support for versions in structs
- updates for controller key types 6, 9, and 10, in cgf xml

Release 0.7.3 (Dec 13, 2007)

- nif: fixed error message when encountering empty block type
- nif: dump script with block selection feature
- cgf: fix transform errors, ported matrix and vector operations from nif library

Release 0.7.2 (Dec 3, 2007)

- NifTester: new `raisereaderror` argument which simplifies the older system and yields more instructive backtraces
- nif: better support for recent nif versions, if block sizes do not match with the number of bytes read then the bytes are skipped and a warning is printed, instead of raising an exception

Release 0.7.1 (Nov 27, 2007)

- nif: fixed `applyScale` in `bhkRigidBody`

Release 0.7 (Nov 19, 2007)

- fixed a problem locating the customized functions for Fedora 8 python which does not look in default locations besides `sys.path`
- new vector and matrix library under `Utils` (for internal use)
- new quick hull library for computing convex hulls
- new inertia library for computing mass, center of gravity, and inertia tensors of solid and hollow objects
- nif: fixed order of `bhkCollisionObject` when writing NIF files
- nif: new `bhkRigidBody` function for updating inertia, center of gravity, and mass, for all types of primitives

Release 0.6 (Nov 3, 2007)

- `nifoptimize` removes duplicate property blocks
- reduced memory footprint in skin data center and radius calculation for the nif format
- new option to ignore strings when calculating hash
- code has been cleaned up using `pylint`
- added a lot more documentation
- refactored all common functions to take `**kwargs` as argument
- read and write functions have the file stream as first non-keyword argument
- refactored and simplified attribute parsing, using a common `_filteredAttributeList` method used by all methods that need to parse attributes; the `version` and `user_version` checks are now also consistent over all functions (i.e. `getRefs`, `getLinks`, etc.)
- added more doctests

Release 0.5.2 (Oct 25, 2007)

- added hash functions (useful for identifying and comparing objects)

Release 0.5.1 (Oct 19, 2007)

- fixed a bug in the nif.xml file which prevented Oblivion skeleton.nif files to load

Release 0.5 (Oct 19, 2007)

- new functions to get block size
- various small bugs fixed
- nif: support for new versions (20.2.0.6, 20.2.0.7, 20.2.0.8, 20.3.0.3, 20.3.0.6, 20.3.0.9)
- nif: block sizes are now also written to the NIF files, improving support for writing 20.2.0.7+ nif versions
- nif: fixed flattenSkin bug (reported by Kikai)

Release 0.4.9 (Oct 13, 2007)

- nif: nifoptimize no longer raises an exception on test errors, unless you pass the -r option
- nif: nifoptimize will try to restore the original file if something goes wrong during write, so - in theory - it should no longer leave you with corrupt nifs; still it is recommended to keep your backups around just in case
- nif: niftesters recoded to accept arbitrary argument dictionaries; this could cause incompatibilities for people writing their own scripts, but the upgrade to the new system is fairly simple: check the niftemplate.py script
- nif: fixed bug in updateTangentSpace which caused an exception when uvs or normals were not present
- nif: doctest for unsupported blocks in nifs

Release 0.4.8 (Oct 7, 2007)

- cgf: MeshMorphTargetChunk is now supported too
- nif: new script (niftextdump.py) to dump texture and material info
- nif: added template script for quickly writing new nif scripts

Release 0.4.7 (Oct 4, 2007)

- nif: new optimizer script

Release 0.4.6 (Sep 29, 2007)

- nif and cgf documentation improved
- added a number of new doctests
- nif: new scripts
 - niftoaster.py for testing and modifying NIF files (contributed by wz)
 - nifvisualizer.py for visualizing nif blocks (contributed by wz)
 - nifmakehsl.py for making hex workshop structure libraries for all nif versions
- nif: bundling NifVis and NifTester modules so you can make your own nif toasters and visualizers
- nif: fixed rare issue with skin partition calculation
- cgf: new script
 - cgftoaster.py for testing and modifying cgf files (similar to niftoaster.py)
- cgf: bundling CgfTester module so you can make your own cgf toasters
- cgf: various xml bugs fixed
- cgf: write support improved (but not entirely functional yet)
- cgf: material chunk custom function for extraction material shader and script
- Expression.py: support for empty string check in condition

Release 0.4.5 (Sep 16, 2007)

- issue warning message instead of raising exception for improper rotation matrix in setScaleRotationTranslation
- fixed skin partition bug during merge
- skin partition bone index padding and sorting for Freedom Force vs. the 3rd Reich

Release 0.4.4 (Sep 2, 2007)

- added mopp parser and simple mopp generator

Release 0.4.3 (Aug 17, 2007)

- fixed bug that occurred if userver = 0 in the xml (fixes geometry morph data in NIF versions 20.0.0.4 and up)
- NIF:
 - tree() function has been extended
 - some minor cleanups and more documentation

Release 0.4.2 (Aug 15, 2007)

- kwargs for getRefs
- NIF:
 - fixed bug in skin partition calculation
 - when writing NIF files the refs are written in sequence (instead of the links, so missing links will yield an exception, which is a good thing)
 - new functions to get list of extra data blocks and to add effect

Release 0.4.1 (Aug 14, 2007)

- NIF:
 - new function to add collision geometries to packed tristripshape
 - fixed bug in bhkListShape.addShape

Release 0.4 (Aug 12, 2007)

- NIF:
 - new function updateBindPosition in NiGeometry to fix a geometry rest position from current bone positions
 - removed deprecated functions
 - (!) changed interface of addBone, no longer takes “transform” argument; use the new function updateBindPosition instead

Release 0.3.4 (Aug 11, 2007)

- improved documentation
- fixed the ‘in’ operator in Bases/Array.py
- NIF:
 - doctest for NiNode
 - flatten skin fix for skins that consist of multiple shapes
 - support for the most common oblivion havok blocks

Release 0.3.3 (Aug 8, 2007)

- NIF:
 - fixed a bug in the skin center and radius calculation
 - added copy function to Vector3
 - fixed NiGeometry doctest

Release 0.3.2 (Aug 7, 2007)

- simplified interface (still wip) by using keyword arguments for common functions such as read and write
- NIF:
 - fix for skin partition blocks in older nif versions such as Freedom Force vs. 3rd Reich
 - support for triangle skin partitions
 - added stitchstrips option for skin partitions
 - added a NiGeometry function to send bones to bind pose

Release 0.3.1 (Aug 6, 2007)

- NIF:
 - new function for getting geometry skin deformation in rest pose
 - old rest pose functions are deprecated and will be removed from a future release

Release 0.3 (Aug 2, 2007)

- NIF:
 - fixed an issue with writing skeleton.nif files
- CGF:
 - reading support for the most common blocks in static cgf files; experimental

Release 0.2.1 (Jul 29, 2007)

- NIF:
 - fixed bug in getTransform
 - new option in findChain to fix block type

Release 0.2 (Jul 29, 2007)

- fixed argument passing when writing arrays
- NIF: added getControllers function to NiObjectNET

Release 0.1 (Jul 22, 2007)

- bug fixed when writing array of strings
- NIF
 - new function to add bones
 - XML update, supports newer versions from Emerge Demo

Release 0.0 (Jul 7, 2007)

- first public release

1.7.8 Todo list

Todo: Write documentation.

(The [original entry](#) is located in /home/docs/checkouts/readthedocs.org/user_builds/pyffi/checkouts/latest/pyffi/spells/cgf/__init__.py:doc of pyffi.spells.cgf, line 4.)

Todo: Limit the size of shapes (see operation optimization mod for Oblivion!)

(The [original entry](#) is located in /home/docs/checkouts/readthedocs.org/user_builds/pyffi/checkouts/latest/pyffi/spells/nif/optimize.py:doc of pyffi.spells.nif.optimize.SpellOptimizeGeometry.branchentry, line 8.)

Todo:

- Write dedicated utilities to optimize particular games (start with Oblivion, maybe eventually also do Fallout 3, Morrowind, etc.).
-

(The [original entry](#) is located in ../TODO.rst, line 3.)

Todo: Aion caf format (MtlNameChunk header?).

(The [original entry](#) is located in ../TODO.rst, line 9.)

Todo: refactoring plans

- what's up with get_global_child_names?
- common base classes for pyffi.object_models.xml.basic.BasicBase/StructBase and pyffi.object_models.xsd.SimpleType/ComplexType (e.g. pyffi.ObjectModel.SimpleType/ComplexType)
- derive object_models.array_type and object_models.StructType from common subclass pyffi.object_models.ComplexType, use these then as base classes for object_models.xml.array and object_models.xml.struct_.StructBase
- use pyffi.utils.graph for all object_models.XXX implementations
- upgrade QSkope and XML model to use GlobalNode instead of the current ad hoc system with Refs
- improve the abstract object_models.Delegate classes (i.e. naming, true abstract base classes, defining a common interface); also perhaps think about deriving these delegate classes from TreeLeaf (only leafs have editors!)?
- ditch version and user_version from the object_models interface, and instead use object_models.Data as a global root element that contains all file information with a minimal format independent interface; implementation plan (this is already partially implemented, namely in the nif format):
 - use abstract Data and Tree base classes fo the XSD parser, in subsequent 2.x.x releases
 - update the XML parser to follow the same scheme, when switching from 2.x.x to 3.0.0, and document the 2.x.x -> 3.0.0 migration strategy

- deprecate the old methods (XxxFormat.read, XxxFormat.write, XxxFormat.getVersion, and so on) in 3.x.x
 - remove old method in 4.x.x
 - one of the aims is that qskope no longer relies on any object_models.xml/object_models.xsd specific implementations; if it only relies on the abstract base classes in object_models.Graph and object_models.Data then future expansions are a lot easier to cope with; in particular, qskope should never have to import from object_models.XXX, or Formats.XXX
-

(The [original entry](#) is located in ../TODO.rst, line 13.)

Todo: Doctests for all spells.

(The [original entry](#) is located in ../TODO.rst, line 62.)

Todo: Improve overall documentation, for instance:

- add docstrings for all spells
 - add docstrings for all spell methods
-

(The [original entry](#) is located in ../TODO.rst, line 66.)

Todo:

- move all regression tests to the tests directory (but keep useful examples in the docstrings!)
 - add spell support for qskope directly using the pyffi.spells module
 - allow qskope to create new spells, from a user supplied spells module
 - custom spell module creation wizard (creates dir structure for user and stores it into the configuration)
 - custom spell creation wizard (adds new spell to user's spell module)
 - automatic templates for typical spells
 - pep8 conventions
 - resolve all complaints from cheesecake's pep8 checker
-

(The [original entry](#) is located in ../TODO.rst, line 73.)

Todo:

- Write dedicated utilities to optimize particular games (start with Oblivion, maybe eventually also do Fallout 3, Morrowind, etc.).
-

Todo: Aion caf format (MtlNameChunk header?).

Todo: refactoring plans

- what's up with get_global_child_names?
-

- common base classes for `pyffi.object_models.xml.basic.BasicBase/StructBase` and `pyffi.object_models.xsd.SimpleType/ComplexType` (e.g. `pyffi.ObjectModel.SimpleType/ComplexType`)
 - derive `object_models.array_type` and `object_models.StructType` from common subclass `pyffi.object_models.ComplexType`, use these then as base classes for `object_models.xml.array` and `object_models.xml.struct_.StructBase`
 - use `pyffi.utils.graph` for all `object_models.XXX` implementations
 - upgrade QScope and XML model to use `GlobalNode` instead of the current ad hoc system with `Refs`
 - improve the abstract `object_models.Delegate` classes (i.e. naming, true abstract base classes, defining a common interface); also perhaps think about deriving these delegate classes from `TreeLeaf` (only leafs have editors!)?
 - ditch `version` and `user_version` from the `object_models` interface, and instead use `object_models.Data` as a global root element that contains all file information with a minimal format independent interface; implementation plan (this is already partially implemented, namely in the `nif` format):
 - use abstract `Data` and `Tree` base classes for the XSD parser, in subsequent 2.x.x releases
 - update the XML parser to follow the same scheme, when switching from 2.x.x to 3.0.0, and document the 2.x.x -> 3.0.0 migration strategy
 - deprecate the old methods (`XxxFormat.read`, `XxxFormat.write`, `XxxFormat.getVersion`, and so on) in 3.x.x
 - remove old method in 4.x.x
 - one of the aims is that `qscope` no longer relies on any `object_models.xml/object_models.xsd` specific implementations; if it only relies on the abstract base classes in `object_models.Graph` and `object_models.Data` then future expansions are a lot easier to cope with; in particular, `qscope` should never have to import from `object_models.XXX`, or `Formats.XXX`
-

Todo: Doctests for all spells.

Todo: Improve overall documentation, for instance:

- add docstrings for all spells
 - add docstrings for all spell methods
-

Todo:

- move all regression tests to the `tests` directory (but keep useful examples in the docstrings!)
 - add spell support for `qscope` directly using the `pyffi.spells` module
 - allow `qscope` to create new spells, from a user supplied spells module
 - custom spell module creation wizard (creates dir structure for user and stores it into the configuration)
 - custom spell creation wizard (adds new spell to user's spell module)
 - automatic templates for typical spells
 - pep8 conventions
 - resolve all complaints from `cheesecake`'s pep8 checker
-

1.7.9 Thanks

Special thanks go in particular to:

- Guido van Rossum, and with him many others, for Python, and in particular for having metaclasses in Python: metaclasses make PyFFI's implementation very easy.
- m4444x for nifskope, which has been an inspiration for PyFFI's xml based design, and of course also an inspiration for QSkope.
- wz for his support, and for testing of the library, when the first version was being written.
- seith for design of the windows installer artwork.
- Crytek for releasing the Far Cry SDK and Crysis SDK, which contains much information about the cgf file format. This has saved many months of hard work.
- Crytek and Bethesda for the great games they make.
- Havok, for releasing their SDK without which custom mopp generation would not have been possible.
- Karl Norby and Michael Summers for pyxsd, which forms the basis of the xsd object model, used for instance to support Collada.

1.7.10 Glossary

PyFFI Python File Format Interface. Also see *file*, *interface*, and *pyffi*.

file A byte stream.

file format See *interface*.

file format interface See *interface*.

interface An interface provides a semantic translation of a byte stream to an organized collection of named Python objects, which are typically bundled into a classes.

spell A transformation that can be applied to a file.

toaster Applies one or more spells to all files of all subdirectories of a given directory.

1.8 Indices and tables

- [genindex](#)
- [modindex](#)
- [search](#)

PYTHON MODULE INDEX

p

- [pyffi](#), 7
- [pyffi.formats](#), 8
 - [pyffi.formats.bsa](#), 8
 - [pyffi.formats.cgf](#), 11
 - [pyffi.formats.dae](#), 22
 - [pyffi.formats.dds](#), 24
 - [pyffi.formats.egm](#), 27
 - [pyffi.formats.egt](#), 31
 - [pyffi.formats.esp](#), 34
 - [pyffi.formats.kfm](#), 38
 - [pyffi.formats.nif](#), 43
 - [pyffi.formats.tga](#), 196
 - [pyffi.formats.tri](#), 198
- [pyffi.object_models](#), 233
- [pyffi.spells](#), 205
 - [pyffi.spells.cgf](#), 205
 - [pyffi.spells.dds](#), 205
 - [pyffi.spells.kfm](#), 205
 - [pyffi.spells.nif](#), 205
 - [pyffi.spells.nif.check](#), 205
 - [pyffi.spells.nif.dump](#), 205
 - [pyffi.spells.nif.fix](#), 205
 - [pyffi.spells.nif.modify](#), 216
 - [pyffi.spells.nif.optimize](#), 214
 - [pyffi.spells.tga](#), 226

Symbols

`_SpellDelBranchClasses` (class in `pyffi.spells.nif.modify`), 224
`__init__()` (`pyffi.spells.Spell` method), 227
`_branchinspect()` (`pyffi.spells.Spell` method), 227
`_datainspect()` (`pyffi.spells.Spell` method), 227

A

`a()` (`pyffi.formats.nif.NifFormat.ByteColor4` property), 64
`a()` (`pyffi.formats.nif.NifFormat.Color4` property), 66
`access()` (`pyffi.formats.nif.NifFormat.NiDataStream` property), 99
`ACOUSTIC_SPACE` (`pyffi.formats.nif.NifFormat.Fallout3Layer` attribute), 75
`ACOUSTIC_SPACE` (`pyffi.formats.nif.NifFormat.SkyrimLayer` attribute), 166
`ACTION_DECREMENT` (`pyffi.formats.nif.NifFormat.StencilAction` attribute), 170
`ACTION_INCREMENT` (`pyffi.formats.nif.NifFormat.StencilAction` attribute), 170
`ACTION_INVERT` (`pyffi.formats.nif.NifFormat.StencilAction` attribute), 170
`ACTION_KEEP` (`pyffi.formats.nif.NifFormat.StencilAction` attribute), 170
`ACTION_REPLACE` (`pyffi.formats.nif.NifFormat.StencilAction` attribute), 170
`ACTION_ZERO` (`pyffi.formats.nif.NifFormat.StencilAction` attribute), 170
`active()` (`pyffi.formats.nif.NifFormat.NiPSysModifier` property), 129
`active_material()` (`pyffi.formats.nif.NifFormat.NiRenderObject` property), 141
`ACTIVESPELLCLASSES` (`pyffi.spells.SpellGroupBase` attribute), 229
`actor_descs()` (`pyffi.formats.nif.NifFormat.NiPhysXPPropDesc` property), 138
`ACTORZONE` (`pyffi.formats.nif.NifFormat.Fallout3Layer` attribute), 75
`ACTORZONE` (`pyffi.formats.nif.NifFormat.SkyrimLayer` attribute), 166

`add_asym_morph()` (`pyffi.formats.egm.EgmFormat.Data` method), 27
`add_bone()` (`pyffi.formats.nif.NifFormat.NiGeometry` method), 103
`add_child()` (`pyffi.formats.nif.NifFormat.NiNode` method), 111
`add_controlled_block()` (`pyffi.formats.nif.NifFormat.NiControllerSequence` method), 98
`add_controller()` (`pyffi.formats.nif.NifFormat.NiObjectNET` method), 113
`add_effect()` (`pyffi.formats.nif.NifFormat.NiNode` method), 111
`add_extra_data()` (`pyffi.formats.nif.NifFormat.NiObjectNET` method), 113
`add_integer_extra_data()` (`pyffi.formats.nif.NifFormat.NiObjectNET` method), 113
`add_modifier()` (`pyffi.formats.tri.TriFormat.Header` method), 200
`add_morph()` (`pyffi.formats.tri.TriFormat.Header` method), 200
`add_property()` (`pyffi.formats.nif.NifFormat.NiAVObject` method), 89
`add_shape()` (`pyffi.formats.nif.NifFormat.bhkListShape` method), 182
`add_shape()` (`pyffi.formats.nif.NifFormat.bhkPackedNiTriStripsShape` method), 183
`add_string()` (`pyffi.formats.nif.NifFormat.StringPalette` method), 171
`add_sym_morph()` (`pyffi.formats.egm.EgmFormat.Data` method), 27
`ADDONARM` (`pyffi.formats.nif.NifFormat.Fallout3Layer` attribute), 75
`ADDONCHEST` (`pyffi.formats.nif.NifFormat.Fallout3Layer` attribute), 75
`ADDONHEAD` (`pyffi.formats.nif.NifFormat.Fallout3Layer` attribute), 75
`ADDONLEG` (`pyffi.formats.nif.NifFormat.Fallout3Layer` attribute), 75
`affected_node_list_pointers()` (`pyffi.formats.nif.NifFormat.NiDynamicEffect` method), 27

property), 99
 affected_nodes() (pyffi.formats.nif.NifFormat.NiDynamicEffectproperty), 148
 property), 99
 ALIASDICT (pyffi.spells.Toaster attribute), 231
 Alpha (pyffi.formats.nif.NifFormat.LightingShaderControllerAttribute attribute), 82
 alpha() (pyffi.formats.nif.NifFormat.BSLightingShaderProperty property), 51
 ALPHA_BINARY (pyffi.formats.nif.NifFormat.AlphaFormat attribute), 44
 ALPHA_DEFAULT (pyffi.formats.nif.NifFormat.AlphaFormat attribute), 44
 alpha_format() (pyffi.formats.nif.NifFormat.NiSourceTexture property), 145
 ALPHA_NONE (pyffi.formats.nif.NifFormat.AlphaFormat attribute), 44
 ALPHA_SMOOTH (pyffi.formats.nif.NifFormat.AlphaFormat attribute), 44
 alpha_sort_bound() (pyffi.formats.nif.NifFormat.BSOrderedNode property), 53
 ALWAYS_FACE_CAMERA (pyffi.formats.nif.NifFormat.BillboardMode attribute), 62
 ALWAYS_FACE_CENTER (pyffi.formats.nif.NifFormat.BillboardMode attribute), 62
 always_update() (pyffi.formats.nif.NifFormat.NiGeomMorpherController property), 101
 ambient_color() (pyffi.formats.nif.NifFormat.NiLight property), 106
 ANIM_STATIC (pyffi.formats.nif.NifFormat.Fallout3Layer attribute), 75
 ANIM_STATIC (pyffi.formats.nif.NifFormat.OblivionLayer attribute), 155
 animation_type() (pyffi.formats.nif.NifFormat.FurniturePositionormethod), 78
 ANIMSTATIC (pyffi.formats.nif.NifFormat.SkyrimLayer attribute), 166
 append_comp_data() (pyffi.formats.nif.NifFormat.NiBSplineData method), 92
 append_float_data() (pyffi.formats.nif.NifFormat.NiBSplineData method), 92
 append_short_data() (pyffi.formats.nif.NifFormat.NiBSplineData method), 92
 APPLY_DECAL (pyffi.formats.nif.NifFormat.ApplyMode attribute), 44
 APPLY_HILIGHT (pyffi.formats.nif.NifFormat.ApplyMode attribute), 44
 APPLY_HILIGHT2 (pyffi.formats.nif.NifFormat.ApplyMode attribute), 45
 apply_mode() (pyffi.formats.nif.NifFormat.NiTexturingProperty attribute), 45
 APPLY_MODULATE (pyffi.formats.nif.NifFormat.ApplyMode attribute), 45
 APPLY_REPLACE (pyffi.formats.nif.NifFormat.ApplyMode attribute), 45
 apply_scale() (pyffi.formats.cgf.CgfFormat.Chunk method), 12
 apply_scale() (pyffi.formats.cgf.CgfFormat.DataStreamChunk method), 13
 apply_scale() (pyffi.formats.egm.EgmFormat.Data method), 27
 apply_scale() (pyffi.formats.egm.EgmFormat.MorphRecord method), 29
 apply_scale() (pyffi.formats.nif.NifFormat.bhkBoxShape method), 178
 apply_scale() (pyffi.formats.nif.NifFormat.bhkCapsuleShape method), 179
 apply_scale() (pyffi.formats.nif.NifFormat.bhkConvexVerticesShape method), 181
 apply_scale() (pyffi.formats.nif.NifFormat.bhkLimitedHingeConstraint method), 182
 apply_scale() (pyffi.formats.nif.NifFormat.bhkMalleableConstraint method), 182
 apply_scale() (pyffi.formats.nif.NifFormat.bhkRagdollConstraint method), 186
 apply_scale() (pyffi.formats.nif.NifFormat.bhkRigidBody method), 186
 apply_scale() (pyffi.formats.nif.NifFormat.bhkSphereShape method), 187
 apply_scale() (pyffi.formats.nif.NifFormat.bhkTransformShape method), 187
 apply_scale() (pyffi.formats.nif.NifFormat.BSBound method), 45
 apply_scale() (pyffi.formats.nif.NifFormat.hkPackedNiTriStripsData method), 189
 apply_scale() (pyffi.formats.nif.NifFormat.NiAVObject method), 89
 apply_scale() (pyffi.formats.nif.NifFormat.NiBSplineCompTransform method), 92
 apply_scale() (pyffi.formats.nif.NifFormat.NiBSplineTransformInterpo method), 93
 apply_scale() (pyffi.formats.nif.NifFormat.NiGeometryData method), 104
 apply_scale() (pyffi.formats.nif.NifFormat.NiKeyframeData method), 106
 apply_scale() (pyffi.formats.nif.NifFormat.NiMorphData method), 109
 apply_scale() (pyffi.formats.nif.NifFormat.NiObject method), 113
 apply_scale() (pyffi.formats.nif.NifFormat.NiSkinData method), 143
 apply_scale() (pyffi.formats.nif.NifFormat.NiTransformInterpolator method), 150

apply_scale() (pyffi.formats.tri.TriFormat.MorphReconstructable method), 200
 ARCHIVE_CLASSES (pyffi.formats.nif.NifFormat attribute), 43
 ARCHIVE_CLASSES (pyffi.object_models.FileFormat attribute), 234
 as_list() (pyffi.formats.cgf.CgfFormat.Matrix33 method), 15
 as_list() (pyffi.formats.cgf.CgfFormat.Matrix44 method), 15
 as_list() (pyffi.formats.nif.NifFormat.InertiaMatrix method), 80
 as_list() (pyffi.formats.nif.NifFormat.Matrix33 method), 83
 as_list() (pyffi.formats.nif.NifFormat.Matrix44 method), 84
 as_list() (pyffi.formats.nif.NifFormat.TexCoord method), 173
 as_list() (pyffi.formats.nif.NifFormat.Vector3 method), 176
 as_list() (pyffi.formats.nif.NifFormat.Vector4 method), 176
 as_tuple() (pyffi.formats.cgf.CgfFormat.Matrix33 method), 15
 as_tuple() (pyffi.formats.cgf.CgfFormat.Matrix44 method), 15
 as_tuple() (pyffi.formats.nif.NifFormat.InertiaMatrix method), 80
 as_tuple() (pyffi.formats.nif.NifFormat.Matrix33 method), 83
 as_tuple() (pyffi.formats.nif.NifFormat.Matrix44 method), 84
 as_tuple() (pyffi.formats.nif.NifFormat.Vector3 method), 176
 as_tuple() (pyffi.formats.nif.NifFormat.Vector4 method), 176
 aspect_ratio() (pyffi.formats.nif.NifFormat.NiPSysData property), 125
 assign() (pyffi.formats.nif.NifFormat.Vector3 method), 176
 atom_sizes() (pyffi.formats.nif.NifFormat.BSPackedAdditionalDataBlock property), 55
 attenuation() (pyffi.formats.nif.NifFormat.NiPSysFieldModifier property), 127
 av_object() (pyffi.formats.nif.NifFormat.AVObject property), 43
 AVOID_BOX (pyffi.formats.nif.NifFormat.OblivionLayer attribute), 155
 AVOIDBOX (pyffi.formats.nif.NifFormat.Fallout3Layer attribute), 75
 AVOIDBOX (pyffi.formats.nif.NifFormat.SkyrimLayer attribute), 166
 axis() (pyffi.formats.nif.NifFormat.BoxBV property), 63
 axis_a() (pyffi.formats.nif.NifFormat.HingeDescriptor property), 80
 axis_b() (pyffi.formats.nif.NifFormat.HingeDescriptor property), 80
B
 b() (pyffi.formats.nif.NifFormat.ByteColor3 property), 64
 b() (pyffi.formats.nif.NifFormat.ByteColor4 property), 64
 b() (pyffi.formats.nif.NifFormat.Color3 property), 66
 b() (pyffi.formats.nif.NifFormat.Color4 property), 66
 b() (pyffi.formats.nif.NifFormat.TBC property), 173
 BACK_WEAPON (pyffi.formats.nif.NifFormat.OblivionLayer attribute), 155
 BACK_WEAPON2 (pyffi.formats.nif.NifFormat.OblivionLayer attribute), 155
 backward() (pyffi.formats.nif.NifFormat.Key property), 81
 ball_and_socket() (pyffi.formats.nif.NifFormat.bhkBallAndSocketConstraint property), 177
 ball_and_socket() (pyffi.formats.nif.NifFormat.SubConstraint property), 172
 BallAndSocket (pyffi.formats.nif.NifFormat.hkConstraintType attribute), 188
 base() (pyffi.formats.nif.NifFormat.NiBSplineCompFloatInterpolator property), 91
 BASE_BV (pyffi.formats.nif.NifFormat.BoundVolumeType attribute), 63
 BASE_MAP (pyffi.formats.nif.NifFormat.TexType attribute), 175
 base_scale() (pyffi.formats.nif.NifFormat.NiPSysGrowFadeModifier property), 128
 base_texture() (pyffi.formats.nif.NifFormat.NiTexturingProperty property), 148
 behaviour_graph_file() (pyffi.formats.nif.NifFormat.BSBehaviorGraphExtraData property), 45
 behaviour_graph_file() (pyffi.formats.nif.NifFormat.FurnitureEntryPoints property), 78
 bezier_triangle() (pyffi.formats.nif.NifFormat.NiBezierMesh property), 94
 bias() (pyffi.formats.nif.NifFormat.NiBSplineCompFloatInterpolator property), 91
 big_tris() (pyffi.formats.nif.NifFormat.bhkCompressedMeshShapeData property), 180
 big_verts() (pyffi.formats.nif.NifFormat.bhkCompressedMeshShapeData property), 180
 billboard_mode() (pyffi.formats.nif.NifFormat.NiBillboardNode property), 94

binary_data() (pyffi.formats.nif.NifFormat.NiBinaryExtraData property), 45
 BIPED (pyffi.formats.nif.NifFormat.Fallout3Layer attribute), 75
 BIPED (pyffi.formats.nif.NifFormat.OblivionLayer attribute), 155
 BIPED (pyffi.formats.nif.NifFormat.SkyrimLayer attribute), 166
 BIPED_NO_CC (pyffi.formats.nif.NifFormat.SkyrimLayer attribute), 166
 bits_per_channel() (pyffi.formats.nif.NifFormat.ChannelData property), 65
 bits_per_index() (pyffi.formats.nif.NifFormat.bhkCompressedMeshShapeData property), 180
 bits_per_w_index() (pyffi.formats.nif.NifFormat.bhkCompressedMeshShapeData property), 180
 block_index() (pyffi.formats.nif.NifFormat.AdditionalDataInfo property), 44
 block_infos() (pyffi.formats.nif.NifFormat.BSPackedAdditionalGeometryData property), 55
 block_infos() (pyffi.formats.nif.NifFormat.NiAdditionalGeometryData property), 89
 block_offsets() (pyffi.formats.nif.NifFormat.AdditionalDataBlock property), 44
 block_offsets() (pyffi.formats.nif.NifFormat.BSPackedAdditionalDataBlock property), 55
 block_size() (pyffi.formats.nif.NifFormat.AdditionalDataBlock property), 44
 blocks() (pyffi.formats.nif.NifFormat.BSPackedAdditionalGeometryData property), 55
 blocks() (pyffi.formats.nif.NifFormat.NiAdditionalGeometryData property), 89
 BlockTypeIndex (pyffi.formats.nif.NifFormat attribute), 62
 BODY (pyffi.formats.nif.NifFormat.Fallout3Layer attribute), 75
 BODY (pyffi.formats.nif.NifFormat.OblivionLayer attribute), 155
 body() (pyffi.formats.nif.NifFormat.bhkNiCollisionObject property), 183
 body_part() (pyffi.formats.nif.NifFormat.BodyPartList property), 63
 bomb_axis() (pyffi.formats.nif.NifFormat.NiPSysBombModifier property), 124
 bomb_object() (pyffi.formats.nif.NifFormat.NiPSysBombModifier property), 124
 bone_bounds() (pyffi.formats.nif.NifFormat.NiSkinningMeshModifier property), 145
 bone_data() (pyffi.formats.nif.NifFormat.NiTriShapeSkinController property), 152
 bone_l_o_d_count() (pyffi.formats.nif.NifFormat.BSBoneLODExtraData property), 95
 bone_l_o_d_info() (pyffi.formats.nif.NifFormat.BSBoneLODExtraData property), 45
 bone_name() (pyffi.formats.nif.NifFormat.BoneLOD property), 63
 bone_transforms() (pyffi.formats.nif.NifFormat.NiSkinningMeshModifier property), 145
 bones() (pyffi.formats.nif.NifFormat.bhkRagdollTemplate property), 186
 bones() (pyffi.formats.nif.NifFormat.BSTreeNode property), 61
 bones() (pyffi.formats.nif.NifFormat.NiFurSpringController property), 101
 bones() (pyffi.formats.nif.NifFormat.NiSkinInstance property), 144
 bones() (pyffi.formats.nif.NifFormat.NiSkinningMeshModifier property), 145
 bones() (pyffi.formats.nif.NifFormat.NiTriShapeSkinController property), 145
 bones_1() (pyffi.formats.nif.NifFormat.BSTreeNode property), 61
 bones_2() (pyffi.formats.nif.NifFormat.NiFurSpringController property), 101
 bool (pyffi.formats.cgf.CgfFormat attribute), 19
 bool_value() (pyffi.formats.nif.NifFormat.NiBoolInterpolator property), 95
 bool_value() (pyffi.formats.nif.NifFormat.NiBoolInterpolator property), 96
 boolean_data() (pyffi.formats.nif.NifFormat.NiBooleanExtraData property), 97
 boolean_data() (pyffi.formats.nif.NifFormat.NiPSysCollider property), 125
 bound() (pyffi.formats.nif.NifFormat.NiMesh property), 107
 bound_center() (pyffi.formats.nif.NifFormat.NiScreenLODData property), 142
 bound_radius() (pyffi.formats.nif.NifFormat.NiScreenLODData property), 142
 bounding_volume() (pyffi.formats.nif.NifFormat.NiCollisionData property), 98
 bounding_volumes() (pyffi.formats.nif.NifFormat.UnionBV property), 175
 bounding_max() (pyffi.formats.nif.NifFormat.bhkCompressedMeshShapeData property), 180
 bounding_min() (pyffi.formats.nif.NifFormat.bhkCompressedMeshShapeData property), 180
 bounding_volume() (pyffi.formats.nif.NifFormat.BoundingVolume property), 63
 BOX_BV (pyffi.formats.nif.NifFormat.BoundVolumeType attribute), 63

BP_BRAIN (pyffi.formats.nif.NifFormat.BSDismemberBodyPartType attribute), 46

BP_HEAD (pyffi.formats.nif.NifFormat.BSDismemberBodyPartType attribute), 46

BP_HEAD2 (pyffi.formats.nif.NifFormat.BSDismemberBodyPartType attribute), 46

BP_LEFTARM (pyffi.formats.nif.NifFormat.BSDismemberBodyPartType attribute), 46

BP_LEFTARM2 (pyffi.formats.nif.NifFormat.BSDismemberBodyPartType attribute), 46

BP_LEFTLEG (pyffi.formats.nif.NifFormat.BSDismemberBodyPartType attribute), 46

BP_LEFTLEG2 (pyffi.formats.nif.NifFormat.BSDismemberBodyPartType attribute), 46

BP_LEFTLEG3 (pyffi.formats.nif.NifFormat.BSDismemberBodyPartType attribute), 46

BP_RIGHTARM (pyffi.formats.nif.NifFormat.BSDismemberBodyPartType attribute), 46

BP_RIGHTARM2 (pyffi.formats.nif.NifFormat.BSDismemberBodyPartType attribute), 46

BP_RIGHTLEG (pyffi.formats.nif.NifFormat.BSDismemberBodyPartType attribute), 46

BP_RIGHTLEG2 (pyffi.formats.nif.NifFormat.BSDismemberBodyPartType attribute), 46

BP_RIGHTLEG3 (pyffi.formats.nif.NifFormat.BSDismemberBodyPartType attribute), 46

BP_SECTIONCAP_BRAIN (pyffi.formats.nif.NifFormat.BSDismemberBodyPartType attribute), 46

BP_SECTIONCAP_HEAD (pyffi.formats.nif.NifFormat.BSDismemberBodyPartType attribute), 46

BP_SECTIONCAP_HEAD2 (pyffi.formats.nif.NifFormat.BSDismemberBodyPartType attribute), 46

BP_SECTIONCAP_LEFTARM (pyffi.formats.nif.NifFormat.BSDismemberBodyPartType attribute), 46

BP_SECTIONCAP_LEFTARM2 (pyffi.formats.nif.NifFormat.BSDismemberBodyPartType attribute), 46

BP_SECTIONCAP_LEFTLEG (pyffi.formats.nif.NifFormat.BSDismemberBodyPartType attribute), 47

BP_SECTIONCAP_LEFTLEG2 (pyffi.formats.nif.NifFormat.BSDismemberBodyPartType attribute), 47

BP_SECTIONCAP_LEFTLEG3 (pyffi.formats.nif.NifFormat.BSDismemberBodyPartType attribute), 47

BP_SECTIONCAP_RIGHTARM (pyffi.formats.nif.NifFormat.BSDismemberBodyPartType attribute), 47

BP_SECTIONCAP_RIGHTARM2 (pyffi.formats.nif.NifFormat.BSDismemberBodyPartType attribute), 47

BP_SECTIONCAP_RIGHTLEG (pyffi.formats.nif.NifFormat.BSDismemberBodyPartType attribute), 47

BP_SECTIONCAP_RIGHTLEG2 (pyffi.formats.nif.NifFormat.BSDismemberBodyPartType attribute), 47

BP_SECTIONCAP_RIGHTLEG3 (pyffi.formats.nif.NifFormat.BSDismemberBodyPartType attribute), 47

BP_TORSOCAP_BRAIN (pyffi.formats.nif.NifFormat.BSDismemberBodyPartType attribute), 47

BP_TORSOCAP_HEAD (pyffi.formats.nif.NifFormat.BSDismemberBodyPartType attribute), 47

BP_TORSOCAP_HEAD2 (pyffi.formats.nif.NifFormat.BSDismemberBodyPartType attribute), 47

BP_TORSOCAP_LEFTARM (pyffi.formats.nif.NifFormat.BSDismemberBodyPartType attribute), 47

BP_TORSOCAP_LEFTARM2 (pyffi.formats.nif.NifFormat.BSDismemberBodyPartType attribute), 47

BP_TORSOCAP_LEFTLEG (pyffi.formats.nif.NifFormat.BSDismemberBodyPartType attribute), 47

BP_TORSOCAP_LEFTLEG2 (pyffi.formats.nif.NifFormat.BSDismemberBodyPartType attribute), 47

BP_TORSOCAP_LEFTLEG3 (pyffi.formats.nif.NifFormat.BSDismemberBodyPartType attribute), 47

BP_TORSOCAP_RIGHTARM (pyffi.formats.nif.NifFormat.BSDismemberBodyPartType attribute), 47

BP_TORSOCAP_RIGHTARM2 (pyffi.formats.nif.NifFormat.BSDismemberBodyPartType attribute), 47

BP_TORSOCAP_RIGHTLEG (pyffi.formats.nif.NifFormat.BSDismemberBodyPartType attribute), 47

BP_TORSOCAP_RIGHTLEG2 (pyffi.formats.nif.NifFormat.BSDismemberBodyPartType attribute), 47

BP_TORSOCAP_RIGHTLEG3 (pyffi.formats.nif.NifFormat.BSDismemberBodyPartType attribute), 47

BP_TORSESECTION_BRAIN (pyffi.formats.nif.NifFormat.BSDismemberBodyPartType attribute), 47

BP_TORSOSECTION_HEAD *method*), 211
 (*pyffi.formats.nif.NifFormat.BSDismemberBodyPartType* *branchentry* () (*pyffi.spells.nif.fix.SpellMergeSkeletonRoots*
attribute), 47 *method*), 209
 BP_TORSOSECTION_HEAD2 *branchentry* () (*pyffi.spells.nif.fix.SpellScale*
 (*pyffi.formats.nif.NifFormat.BSDismemberBodyPartType* *method*), 210
attribute), 47 *branchentry* () (*pyffi.spells.nif.modify.SpellAddStencilProperty*
 BP_TORSOSECTION_LEFTARM *method*), 225
 (*pyffi.formats.nif.NifFormat.BSDismemberBodyPartType* *branchentry* () (*pyffi.spells.nif.modify.SpellChangeBonePriorities*
attribute), 47 *method*), 221
 BP_TORSOSECTION_LEFTARM2 *branchentry* () (*pyffi.spells.nif.modify.SpellCollisionMaterial*
 (*pyffi.formats.nif.NifFormat.BSDismemberBodyPartType* *method*), 219
attribute), 47 *branchentry* () (*pyffi.spells.nif.modify.SpellCollisionType*
 BP_TORSOSECTION_LEFTLEG *method*), 218
 (*pyffi.formats.nif.NifFormat.BSDismemberBodyPartType* *branchentry* () (*pyffi.spells.nif.modify.SpellDelBranches*
attribute), 47 *method*), 223
 BP_TORSOSECTION_LEFTLEG2 *branchentry* () (*pyffi.spells.nif.modify.SpellDelInterpolatorTransformData*
 (*pyffi.formats.nif.NifFormat.BSDismemberBodyPartType* *method*), 223
attribute), 47 *branchentry* () (*pyffi.spells.nif.modify.SpellDelVertexColor*
 BP_TORSOSECTION_LEFTLEG3 *method*), 225
 (*pyffi.formats.nif.NifFormat.BSDismemberBodyPartType* *branchentry* () (*pyffi.spells.nif.modify.SpellDisableParallax*
attribute), 47 *method*), 224
 BP_TORSOSECTION_RIGHTARM *branchentry* () (*pyffi.spells.nif.modify.SpellReverseAnimation*
 (*pyffi.formats.nif.NifFormat.BSDismemberBodyPartType* *method*), 220
attribute), 47 *branchentry* () (*pyffi.spells.nif.modify.SpellScaleAnimationTime*
 BP_TORSOSECTION_RIGHTARM2 *method*), 219
 (*pyffi.formats.nif.NifFormat.BSDismemberBodyPartType* *branchentry* () (*pyffi.spells.nif.modify.SpellSetInterpolatorTransRotScale*
attribute), 47 *method*), 222
 BP_TORSOSECTION_RIGHTLEG *branchentry* () (*pyffi.spells.nif.optimize.SpellCleanRefLists*
 (*pyffi.formats.nif.NifFormat.BSDismemberBodyPartType* *method*), 214
attribute), 47 *branchentry* () (*pyffi.spells.nif.optimize.SpellDelUnusedBones*
 BP_TORSOSECTION_RIGHTLEG2 *method*), 216
 (*pyffi.formats.nif.NifFormat.BSDismemberBodyPartType* *branchentry* () (*pyffi.spells.nif.optimize.SpellMergeDuplicates*
attribute), 47 *method*), 214
 BP_TORSOSECTION_RIGHTLEG3 *branchentry* () (*pyffi.spells.nif.optimize.SpellOptimizeGeometry*
 (*pyffi.formats.nif.NifFormat.BSDismemberBodyPartType* *method*), 215
attribute), 47 *branchentry* () (*pyffi.spells.Spell* *method*), 227
 BRANCH_CLASSES_TO_BE_DELETED *branchentry* () (*pyffi.spells.SpellGroupParallelBase*
 (*pyffi.spells.nif.modify._SpellDelBranchClasses* *method*), 230
attribute), 224 *branchentry* () (*pyffi.spells.SpellGroupSeriesBase*
branchentry () (*pyffi.spells.nif.fix.SpellAddTangentSpace* *method*), 230
method), 206 *branchexit* () (*pyffi.spells.Spell* *method*), 227
branchentry () (*pyffi.spells.nif.fix.SpellClampMaterialAlpha* *branchexit* () (*pyffi.spells.SpellGroupParallelBase*
method), 208 *method*), 230
branchentry () (*pyffi.spells.nif.fix.SpellCleanStringPalette* *branchinspect* () (*pyffi.spells.nif.fix.SpellAddTangentSpace*
method), 212 *method*), 206
branchentry () (*pyffi.spells.nif.fix.SpellDelTangentSpace* *branchinspect* () (*pyffi.spells.nif.fix.SpellClampMaterialAlpha*
method), 206 *method*), 209
branchentry () (*pyffi.spells.nif.fix.SpellDetachHavokTriStripsData* *branchinspect* () (*pyffi.spells.nif.fix.SpellCleanStringPalette*
method), 208 *method*), 212
branchentry () (*pyffi.spells.nif.fix.SpellFFVT3RSkinPartition* *branchinspect* () (*pyffi.spells.nif.fix.SpellDelTangentSpace*
method), 207 *method*), 206
branchentry () (*pyffi.spells.nif.fix.SpellFixEmptySkeletonRoots* *branchinspect* () (*pyffi.spells.nif.fix.SpellDetachHavokTriStripsData*
method), 213 *method*), 208
branchentry () (*pyffi.spells.nif.fix.SpellFixMopp* *branchinspect* () (*pyffi.spells.nif.fix.SpellFFVT3RSkinPartition*

method), 207
 branchinspect () (*pyffi.spells.nif.fix.SpellFixEmptySkeletonRoots* attribute), 164
method), 213
 branchinspect () (*pyffi.spells.nif.fix.SpellMergeSkeletonRoots* attribute), 164
method), 210
 branchinspect () (*pyffi.spells.nif.fix.SpellScale* attribute), 164
method), 210
 branchinspect () (*pyffi.spells.nif.modify.SpellAddStencilBuffer* attribute), 164
method), 225
 branchinspect () (*pyffi.spells.nif.modify.SpellChangeBonePriorities* attribute), 164
method), 221
 branchinspect () (*pyffi.spells.nif.modify.SpellCollisionMaterial* attribute), 164
method), 219
 branchinspect () (*pyffi.spells.nif.modify.SpellCollisionType* attribute), 164
method), 218
 branchinspect () (*pyffi.spells.nif.modify.SpellDelInterpolatorTransform* attribute), 85
method), 223
 branchinspect () (*pyffi.spells.nif.modify.SpellDelSkinShapes* attribute), 86
method), 224
 branchinspect () (*pyffi.spells.nif.modify.SpellDelVertexColor* attribute), 85
method), 225
 branchinspect () (*pyffi.spells.nif.modify.SpellDisableParallax* attribute), 85
method), 224
 branchinspect () (*pyffi.spells.nif.modify.SpellReverseAnimation* attribute), 175
method), 220
 branchinspect () (*pyffi.spells.nif.modify.SpellScaleAnimationTime* attribute), 148
method), 219
 branchinspect () (*pyffi.spells.nif.modify.SpellSetInterpolatorRotationScale* attribute), 148
method), 222
 branchinspect () (*pyffi.spells.nif.optimize.SpellCleanRefLists* attribute), 148
method), 214
 branchinspect () (*pyffi.spells.nif.optimize.SpellDelUnusedBones* attribute), 148
method), 216
 branchinspect () (*pyffi.spells.nif.optimize.SpellMergeDuplicatedTextures* attribute), 148
method), 215
 branchinspect () (*pyffi.spells.nif.optimize.SpellOptimizeGeometry* attribute), 148
method), 215
 branchinspect () (*pyffi.spells.Spell* method), 228
 branchinspect () (*pyffi.spells.SpellGroupParallelBase* method), 230
 branchinspect () (*pyffi.spells.SpellGroupSeriesBase* method), 230
 BsaFormat (class in *pyffi.formats.bsa*), 8
 BsaFormat.BZString (class in *pyffi.formats.bsa*), 8
 BsaFormat.FileVersion (class in *pyffi.formats.bsa*), 8
 BsaFormat.Hash (class in *pyffi.formats.bsa*), 9
 BsaFormat.Header (class in *pyffi.formats.bsa*), 9
 BsaFormat.ZString (class in *pyffi.formats.bsa*), 9
 BSROTATE_ABOUT_UP
 (*pyffi.formats.nif.NifFormat.BillboardMode* attribute), 62
 bsseg_water () (*pyffi.formats.nif.NifFormat.BSSegmentFlags* property), 57
 BSSM_SKY (*pyffi.formats.nif.NifFormat.SkyObjectType* attribute), 164
 BSSM_SKY_CLOUDS (*pyffi.formats.nif.NifFormat.SkyObjectType* attribute), 164
 BSSM_SKY_MOON_STARS_MASK
 (*pyffi.formats.nif.NifFormat.SkyObjectType* attribute), 164
 BSSM_SKY_STARS (*pyffi.formats.nif.NifFormat.SkyObjectType* attribute), 164
 BSSM_SKY_TEXTURE (*pyffi.formats.nif.NifFormat.SkyObjectType* attribute), 164
 BUILD_NOT_SET (*pyffi.formats.nif.NifFormat.MoppDataBuildType* attribute), 85
 BUILT_WITH_CHUNK_SUBDIVISION
 (*pyffi.formats.nif.NifFormat.MoppDataBuildType* attribute), 86
 BUILT_WITHOUT_CHUNK_SUBDIVISION
 (*pyffi.formats.nif.NifFormat.MoppDataBuildType* attribute), 85
 BUMP_MAP (*pyffi.formats.nif.NifFormat.TextureType* attribute), 175
 bump_map_luma_offset ()
 (*pyffi.formats.nif.NifFormat.NiTexturingProperty* property), 148
 bump_map_matrix ()
 (*pyffi.formats.nif.NifFormat.NiTexturingProperty* property), 148
 bump_map_texture ()
 (*pyffi.formats.nif.NifFormat.NiTexturingProperty* property), 148
 buttons () (*pyffi.formats.nif.NifFormat.FxRadioButton* property), 78
 byte (*pyffi.formats.cgf.CgfFormat* attribute), 19
 byte (*pyffi.formats.dds.DdsFormat* attribute), 25
 byte (*pyffi.formats.egm.EgmFormat* attribute), 29
 byte (*pyffi.formats.egt.EgtFormat* attribute), 33
 byte (*pyffi.formats.esp.EspFormat* attribute), 37
 byte (*pyffi.formats.kfm.KfmFormat* attribute), 41
 byte (*pyffi.formats.nif.NifFormat* attribute), 188
 byte (*pyffi.formats.tga.TgaFormat* attribute), 197
 byte (*pyffi.formats.tri.TriFormat* attribute), 201
 byte_1 () (*pyffi.formats.nif.NifFormat.BSProceduralLightningController* property), 56
 byte_2 () (*pyffi.formats.nif.NifFormat.BSProceduralLightningController* property), 56
 byte_3 () (*pyffi.formats.nif.NifFormat.BSProceduralLightningController* property), 56
 byte_set_to_0 () (*pyffi.formats.nif.NifFormat.bhkCMSDMaterial* property), 57

<i>property</i>), 179	<i>attribute</i>), 69
<code>bytes_remaining()</code> (<i>pyffi.formats.nif.NifFormat.NiStringExtraData</i> <i>property</i>), 146	<i>CgFormat</i> (class in <i>pyffi.formats.cgf</i>), 11
C	<i>CgFormat</i> .AbstractMtlChunk (class in <i>pyffi.formats.cgf</i>), 11
<code>c()</code> (<i>pyffi.formats.nif.NifFormat.TBC</i> <i>property</i>), 173	<i>CgFormat</i> .AbstractObjectChunk (class in <i>pyffi.formats.cgf</i>), 11
<code>CAMERA_PICK</code> (<i>pyffi.formats.nif.NifFormat.OblivionLayer</i> <i>attribute</i>), 155	<i>CgFormat</i> .BoneLink (class in <i>pyffi.formats.cgf</i>), 11
<code>CAMERAPICK</code> (<i>pyffi.formats.nif.NifFormat.Fallout3Layer</i> <i>attribute</i>), 75	<i>CgFormat</i> .CgError, 11
<code>CAMERAPICK</code> (<i>pyffi.formats.nif.NifFormat.SkyrimLayer</i> <i>attribute</i>), 166	<i>CgFormat</i> .Chunk (class in <i>pyffi.formats.cgf</i>), 12
<code>CAMERASHPERE</code> (<i>pyffi.formats.nif.NifFormat.SkyrimLayer</i> <i>attribute</i>), 166	<i>CgFormat</i> .ChunkHeader (class in <i>pyffi.formats.cgf</i>), 12
<code>CAMERASPHERE</code> (<i>pyffi.formats.nif.NifFormat.Fallout3Layer</i> <i>attribute</i>), 75	<i>CgFormat</i> .ChunkTable (class in <i>pyffi.formats.cgf</i>), 12
<code>capsule()</code> (<i>pyffi.formats.nif.NifFormat.BoundingBox</i> <i>property</i>), 63	<i>CgFormat</i> .ChunkType (class in <i>pyffi.formats.cgf</i>), 12
<code>CAPSULE_BV</code> (<i>pyffi.formats.nif.NifFormat.BoundsVolumeType</i> <i>attribute</i>), 63	<i>CgFormat</i> .ChunkVersion (class in <i>pyffi.formats.cgf</i>), 12
<code>CC_COMPRESSED</code> (<i>pyffi.formats.nif.NifFormat.ChannelConvention</i> <i>attribute</i>), 65	<i>CgFormat</i> .Data (class in <i>pyffi.formats.cgf</i>), 12
<code>CC_EMPTY</code> (<i>pyffi.formats.nif.NifFormat.ChannelConvention</i> <i>attribute</i>), 65	<i>CgFormat</i> .DataStreamChunk (class in <i>pyffi.formats.cgf</i>), 13
<code>CC_FIXED</code> (<i>pyffi.formats.nif.NifFormat.ChannelConvention</i> <i>attribute</i>), 65	<i>CgFormat</i> .ExportFlagsChunk (class in <i>pyffi.formats.cgf</i>), 13
<code>CC_INDEX</code> (<i>pyffi.formats.nif.NifFormat.ChannelConvention</i> <i>attribute</i>), 65	<i>CgFormat</i> .Face (class in <i>pyffi.formats.cgf</i>), 13
<code>center()</code> (<i>pyffi.formats.nif.NifFormat.BoxBV</i> <i>prop-</i> <i>erty</i>), 64	<i>CgFormat</i> .FileOffset (class in <i>pyffi.formats.cgf</i>), 14
<code>center()</code> (<i>pyffi.formats.nif.NifFormat.BSMultiBoundOBB</i> <i>property</i>), 53	<i>CgFormat</i> .FileSignature (class in <i>pyffi.formats.cgf</i>), 14
<code>center()</code> (<i>pyffi.formats.nif.NifFormat.CapsuleBV</i> <i>property</i>), 65	<i>CgFormat</i> .FileType (class in <i>pyffi.formats.cgf</i>), 14
<code>center()</code> (<i>pyffi.formats.nif.NifFormat.HalfSpaceBV</i> <i>property</i>), 79	<i>CgFormat</i> .FRGB (class in <i>pyffi.formats.cgf</i>), 13
<code>center()</code> (<i>pyffi.formats.nif.NifFormat.SphereBV</i> <i>prop-</i> <i>erty</i>), 170	<i>CgFormat</i> .GeomNameListChunk (class in <i>pyffi.formats.cgf</i>), 14
<code>center_offset()</code> (<i>pyffi.formats.nif.NifFormat.TextDesc</i> <i>property</i>), 174	<i>CgFormat</i> .Header (class in <i>pyffi.formats.cgf</i>), 14
<code>CG_DIFFUSE_CUBE_MAP</code> (<i>pyffi.formats.nif.NifFormat.CoordGenType</i> <i>attribute</i>), 69	<i>CgFormat</i> .InitialPosMatrix (class in <i>pyffi.formats.cgf</i>), 14
<code>CG_SPECULAR_CUBE_MAP</code> (<i>pyffi.formats.nif.NifFormat.CoordGenType</i> <i>attribute</i>), 69	<i>CgFormat</i> .IRGB (class in <i>pyffi.formats.cgf</i>), 14
<code>CG_SPHERE_MAP</code> (<i>pyffi.formats.nif.NifFormat.CoordGenType</i> <i>attribute</i>), 69	<i>CgFormat</i> .IRGBA (class in <i>pyffi.formats.cgf</i>), 14
<code>CG_WORLD_PARALLEL</code> (<i>pyffi.formats.nif.NifFormat.CoordGenType</i> <i>attribute</i>), 69	<i>CgFormat</i> .Matrix33 (class in <i>pyffi.formats.cgf</i>), 15
<code>CG_WORLD_PERSPECTIVE</code> (<i>pyffi.formats.nif.NifFormat.CoordGenType</i> <i>attribute</i>), 69	<i>CgFormat</i> .Matrix44 (class in <i>pyffi.formats.cgf</i>), 15
	<i>CgFormat</i> .MRMChunk (class in <i>pyffi.formats.cgf</i>), 15
	<i>CgFormat</i> .MtlListChunk (class in <i>pyffi.formats.cgf</i>), 16
	<i>CgFormat</i> .PatchMeshChunk (class in <i>pyffi.formats.cgf</i>), 16
	<i>CgFormat</i> .Ptr (class in <i>pyffi.formats.cgf</i>), 16
	<i>CgFormat</i> .Quat (class in <i>pyffi.formats.cgf</i>), 16
	<i>CgFormat</i> .Ref (class in <i>pyffi.formats.cgf</i>), 16
	<i>CgFormat</i> .ScenePropsChunk (class in <i>pyffi.formats.cgf</i>), 17
	<i>CgFormat</i> .SizedString (class in <i>pyffi.formats.cgf</i>), 17
	<i>CgFormat</i> .String128 (class in <i>pyffi.formats.cgf</i>), 18
	<i>CgFormat</i> .String16 (class in <i>pyffi.formats.cgf</i>), 18
	<i>CgFormat</i> .String256 (class in <i>pyffi.formats.cgf</i>),

- 18
- CgffFormat.String32 (class in pyffi.formats.cgf), 18
- CgffFormat.String64 (class in pyffi.formats.cgf), 18
- CgffFormat.Tangent (class in pyffi.formats.cgf), 18
- CgffFormat.UnknownAAFC0005Chunk (class in pyffi.formats.cgf), 18
- CgffFormat.UV (class in pyffi.formats.cgf), 18
- CgffFormat.UVFace (class in pyffi.formats.cgf), 18
- CgffFormat.Vector3 (class in pyffi.formats.cgf), 19
- CGFXMLPATH, 8
- CHAIN (pyffi.formats.nif.NifFormat.Fallout3Layer attribute), 75
- changed() (pyffi.spells.SpellGroupParallelBase property), 230
- changed() (pyffi.spells.SpellGroupSeriesBase property), 230
- channel_offset() (pyffi.formats.nif.NifFormat.AdditionalDataInfor attribute), 44
- char (pyffi.formats.cgf.CgffFormat attribute), 19
- char (pyffi.formats.dds.DdsFormat attribute), 25
- char (pyffi.formats.egm.EgmFormat attribute), 29
- char (pyffi.formats.egt.EgtFormat attribute), 33
- char (pyffi.formats.esp.EspFormat attribute), 37
- char (pyffi.formats.kfm.KfmFormat attribute), 41
- char (pyffi.formats.nif.NifFormat attribute), 188
- char (pyffi.formats.tga.TgaFormat attribute), 197
- char (pyffi.formats.tri.TriFormat attribute), 201
- CHAR_CONTROLLER (pyffi.formats.nif.NifFormat.OblivionLayer attribute), 155
- CHARCONTROLLER (pyffi.formats.nif.NifFormat.Fallout3Layer attribute), 75
- CHARCONTROLLER (pyffi.formats.nif.NifFormat.SkyrimLayer attribute), 166
- child() (pyffi.formats.nif.NifFormat.Ni3dsAnimationNode property), 87
- child_2() (pyffi.formats.nif.NifFormat.NiEnvMappedTriShape property), 100
- child_3() (pyffi.formats.nif.NifFormat.NiEnvMappedTriShape property), 100
- children() (pyffi.formats.nif.NifFormat.NiEnvMappedTriShape property), 100
- CHNL_ALPHA (pyffi.formats.nif.NifFormat.ChannelType attribute), 65
- CHNL_BLUE (pyffi.formats.nif.NifFormat.ChannelType attribute), 65
- CHNL_COMPRESSED (pyffi.formats.nif.NifFormat.ChannelType attribute), 65
- CHNL_EMPTY (pyffi.formats.nif.NifFormat.ChannelType attribute), 65
- CHNL_GREEN (pyffi.formats.nif.NifFormat.ChannelType attribute), 65
- CHNL_INDEX (pyffi.formats.nif.NifFormat.ChannelType attribute), 65
- CHNL_RED (pyffi.formats.nif.NifFormat.ChannelType attribute), 66
- chunk_materials() (pyffi.formats.nif.NifFormat.bhkCompressedMeshShapeData property), 180
- chunk_transforms() (pyffi.formats.nif.NifFormat.bhkCompressedMeshShapeData property), 180
- chunks() (pyffi.formats.nif.NifFormat.bhkCompressedMeshShapeData property), 180
- clamp() (pyffi.formats.nif.NifFormat.MultiTextureElement property), 87
- clamp_mode() (pyffi.formats.nif.NifFormat.TexDesc property), 174
- CLAMP_S_CLAMP_T (pyffi.formats.nif.NifFormat.TexClampMode attribute), 173
- CLAMP_S_WRAP_T (pyffi.formats.nif.NifFormat.TexClampMode attribute), 173
- cleanreflist() (pyffi.spells.nif.optimize.SpellCleanRefLists method), 214
- clear() (pyffi.formats.nif.NifFormat.StringPalette method), 171
- cli() (pyffi.spells.Toaster method), 231
- clipping_plane() (pyffi.formats.nif.NifFormat.NiTextureEffect property), 147
- cloning_behavior() (pyffi.formats.nif.NifFormat.NiDataStream property), 99
- CLONING_BLANK_COPY (pyffi.formats.nif.NifFormat.CloningBehavior attribute), 66
- CLONING_COPY (pyffi.formats.nif.NifFormat.CloningBehavior attribute), 66
- CLONING_SHARE (pyffi.formats.nif.NifFormat.CloningBehavior attribute), 66
- CLOUD_TRAP (pyffi.formats.nif.NifFormat.Fallout3Layer attribute), 75
- CLOUD_TRAP (pyffi.formats.nif.NifFormat.OblivionLayer attribute), 155
- CLOUD_TRAP (pyffi.formats.nif.NifFormat.SkyrimLayer attribute), 166
- CLUTTER (pyffi.formats.nif.NifFormat.Fallout3Layer attribute), 75
- CLUTTER (pyffi.formats.nif.NifFormat.OblivionLayer attribute), 155
- CLUTTER (pyffi.formats.nif.NifFormat.SkyrimLayer attribute), 166
- CM_NOTEST (pyffi.formats.nif.NifFormat.CollisionMode attribute), 66
- CM_USE_ABV (pyffi.formats.nif.NifFormat.CollisionMode attribute), 66
- CM_USE_NIBOUND (pyffi.formats.nif.NifFormat.CollisionMode attribute), 66
- CM_USE_OBB (pyffi.formats.nif.NifFormat.CollisionMode attribute), 66

CM_USE_TRI (pyffi.formats.nif.NifFormat.CollisionMode attribute), 66	maintain (pyffi.formats.nif.NifFormat.NiPointLight property), 140
collider () (pyffi.formats.nif.NifFormat.NiPSysColliderManager property), 125	maintain (pyffi.formats.nif.NifFormat.bhkRagdollTemplateData property), 186
collider_object () (pyffi.formats.nif.NifFormat.NiPSysCollider property), 125	controlled_blocks () (pyffi.formats.nif.NifFormat.NiSequence property), 142
colliders () (pyffi.formats.nif.NifFormat.NiPSSimulatorColliders property), 121	modify () (pyffi.formats.nif.NifFormat.NiParticleModifier property), 133
collision_mode () (pyffi.formats.nif.NifFormat.NiCollisionData property), 98	collider_data () (pyffi.formats.nif.NifFormat.NiFloatExtraDataController property), 101
collision_type () (pyffi.formats.nif.NifFormat.BoundingBox property), 63	controller_sequences () (pyffi.formats.nif.NifFormat.NiControllerManager property), 98
COLLISIONBOX (pyffi.formats.nif.NifFormat.Fallout3Layer attribute), 75	controls_base_skeleton () (pyffi.formats.nif.NifFormat.BSBehaviorGraphExtraData property), 45
COLLISIONBOX (pyffi.formats.nif.NifFormat.SkyrimLayer attribute), 166	creation () (pyffi.formats.nif.NifFormat.ChannelData property), 65
color_1_end_percent () (pyffi.formats.nif.NifFormat.BSPSysSimpleColorModifier property), 54	coordinate_generation_type () (pyffi.formats.nif.NifFormat.NiTextureEffect property), 147
color_1_start_percent () (pyffi.formats.nif.NifFormat.BSPSysSimpleColorModifier property), 54	count () (pyffi.formats.nif.NifFormat.Ni3dsAnimationNode property), 87
color_2_end_percent () (pyffi.formats.nif.NifFormat.BSPSysSimpleColorModifier property), 54	count_1 () (pyffi.formats.nif.NifFormat.NiBezierMesh property), 94
color_2_start_percent () (pyffi.formats.nif.NifFormat.BSPSysSimpleColorModifier property), 54	count_2 () (pyffi.formats.nif.NifFormat.NiBezierMesh property), 94
color_data () (pyffi.formats.nif.NifFormat.NiParticleColorModifier property), 132	create () (pyffi.formats.nif.NifFormat.DataStreamAccess property), 71
color_data () (pyffi.formats.nif.NifFormat.NiParticleSystemController property), 133	create_mutable () (pyffi.formats.nif.NifFormat.DataStreamAccess property), 71
color_keys () (pyffi.formats.nif.NifFormat.NiPSSimulatorGeneralStep property), 122	cpu_write_static () (pyffi.formats.nif.NifFormat.DataStreamAccess property), 71
color_loop_behavior () (pyffi.formats.nif.NifFormat.NiPSSimulatorGeneralStep property), 122	cpu_write_static_initialized () (pyffi.formats.nif.NifFormat.DataStreamAccess property), 71
colors () (pyffi.formats.nif.NifFormat.BSPSysSimpleColorModifier property), 54	cpu_write_volatile () (pyffi.formats.nif.NifFormat.DataStreamAccess property), 71
complete_points () (pyffi.formats.nif.NifFormat.NiMeshModifier property), 108	creator () (pyffi.formats.nif.NifFormat.ExportInfo property), 73
component_formats () (pyffi.formats.nif.NifFormat.NiDataStream property), 99	crossproduct () (pyffi.formats.nif.NifFormat.Vector3 method), 176
component_semantics () (pyffi.formats.nif.NifFormat.MeshData property), 85	CT_MUTABLE (pyffi.formats.nif.NifFormat.ConsistencyType attribute), 68
CONEPROJECTILE (pyffi.formats.nif.NifFormat.SkyrimLayer attribute), 166	CT_STATIC (pyffi.formats.nif.NifFormat.ConsistencyType attribute), 68
CONST_KEY (pyffi.formats.nif.NifFormat.KeyType attribute), 81	CT_VOLATILE (pyffi.formats.nif.NifFormat.ConsistencyType attribute), 68
constant_attenuation ()	cumulative () (pyffi.formats.nif.NifFormat.NiControllerManager

property), 98
 CUSTOM_PICK_1 (pyffi.formats.nif.NifFormat.OblivionLayer attribute), 155
 CUSTOM_PICK_2 (pyffi.formats.nif.NifFormat.OblivionLayer attribute), 155
 CUSTOMPICK1 (pyffi.formats.nif.NifFormat.Fallout3Layer attribute), 75
 CUSTOMPICK1 (pyffi.formats.nif.NifFormat.SkyrimLayer attribute), 166
 CUSTOMPICK2 (pyffi.formats.nif.NifFormat.Fallout3Layer attribute), 75
 CUSTOMPICK2 (pyffi.formats.nif.NifFormat.SkyrimLayer attribute), 166
 cutoff_angle() (pyffi.formats.nif.NifFormat.NiSpotLight property), 146
 CYCLE_CLAMP (pyffi.formats.nif.NifFormat.CycleType attribute), 69
 CYCLE_LOOP (pyffi.formats.nif.NifFormat.CycleType attribute), 70
 CYCLE_REVERSE (pyffi.formats.nif.NifFormat.CycleType attribute), 70
 CYLINDRICAL_SYMMETRY (pyffi.formats.nif.NifFormat.SymmetryType attribute), 172

D

DaeFormat (class in pyffi.formats.dae), 22
 DaeFormat.Data (class in pyffi.formats.dae), 22
 damping() (pyffi.formats.nif.NifFormat.bhkRDTMalleableConstraint property), 185
 damping() (pyffi.formats.nif.NifFormat.BSParentVelocityModifier property), 55
 DARK_MAP (pyffi.formats.nif.NifFormat.TextType attribute), 175
 dark_texture() (pyffi.formats.nif.NifFormat.NiTexturingProperty property), 148
 Data (pyffi.formats.bsa.BsaFormat attribute), 8
 Data (pyffi.formats.egt.EgtFormat attribute), 32
 Data (pyffi.formats.tri.TriFormat attribute), 199
 data (pyffi.spells.Spell attribute), 228
 data() (pyffi.formats.nif.NifFormat.AdditionalDataBlock property), 44
 data() (pyffi.formats.nif.NifFormat.bhkCompressedMeshShape property), 180
 data() (pyffi.formats.nif.NifFormat.BSMultiBound property), 52
 data() (pyffi.formats.nif.NifFormat.BSPackedAdditionalDataBlock property), 55
 data() (pyffi.formats.nif.NifFormat.BSPSysMultiTargetEmitterCtrl property), 54
 data() (pyffi.formats.nif.NifFormat.BSTreadTransfInterpolator property), 61
 data() (pyffi.formats.nif.NifFormat.NiAlphaController property), 89
 data() (pyffi.formats.nif.NifFormat.NiBinaryVoxelExtraData property), 95
 data() (pyffi.formats.nif.NifFormat.NiBoolData property), 96
 data() (pyffi.formats.nif.NifFormat.NiBoolInterpolator property), 96
 data() (pyffi.formats.nif.NifFormat.NiColorData property), 98
 data() (pyffi.formats.nif.NifFormat.NiColorExtraData property), 98
 data() (pyffi.formats.nif.NifFormat.NiDataStream property), 99
 data() (pyffi.formats.nif.NifFormat.NiFloatData property), 100
 data() (pyffi.formats.nif.NifFormat.NiFloatInterpolator property), 101
 data() (pyffi.formats.nif.NifFormat.NiFloatsExtraData property), 101
 data() (pyffi.formats.nif.NifFormat.NiGeomMorpherController property), 101
 data() (pyffi.formats.nif.NifFormat.NiIntegersExtraData property), 105
 data() (pyffi.formats.nif.NifFormat.NiKeyframeController property), 106
 data() (pyffi.formats.nif.NifFormat.NiMorpherController property), 109
 data() (pyffi.formats.nif.NifFormat.NiPoint3InterpController property), 139
 data() (pyffi.formats.nif.NifFormat.NiPoint3Interpolator property), 140
 data() (pyffi.formats.nif.NifFormat.NiPosData property), 140
 data() (pyffi.formats.nif.NifFormat.NiPSysColorModifier property), 125
 data() (pyffi.formats.nif.NifFormat.NiPSysEmitterCtrl property), 126
 data() (pyffi.formats.nif.NifFormat.NiPSysModifierActiveCtrl property), 129
 data() (pyffi.formats.nif.NifFormat.NiPSysModifierFloatCtrl property), 129
 data() (pyffi.formats.nif.NifFormat.NiRollController property), 141
 data() (pyffi.formats.nif.NifFormat.NiSkinInstance property), 144
 data() (pyffi.formats.nif.NifFormat.NiStringsExtraData property), 147
 data() (pyffi.formats.nif.NifFormat.NiTextureTransformController property), 148
 data() (pyffi.formats.nif.NifFormat.NiUVController property), 153
 data() (pyffi.formats.nif.NifFormat.NiVisController property), 153
 data_2() (pyffi.formats.nif.NifFormat.BSKeyframeController property), 50

`data_2()` (`pyffi.formats.nif.NifFormat.NiBezierMesh` property), 94
`data_sizes()` (`pyffi.formats.nif.NifFormat.AdditionalDataBlock` property), 44
`data_type()` (`pyffi.formats.nif.NifFormat.AdditionalDataBlock` property), 44
`dataentry()` (`pyffi.spells.nif.fix.SpellDelUnusedRoots` method), 212
`dataentry()` (`pyffi.spells.nif.fix.SpellDetachHavokTriStripsData` method), 208
`dataentry()` (`pyffi.spells.nif.fix.SpellFixEmptySkeletonRoots` method), 213
`dataentry()` (`pyffi.spells.nif.fix.SpellMergeSkeletonRoots` method), 210
`dataentry()` (`pyffi.spells.nif.fix.SpellScale` method), 211
`dataentry()` (`pyffi.spells.nif.optimize.SpellCleanRefLists` method), 214
`dataentry()` (`pyffi.spells.nif.optimize.SpellDelUnusedBones` method), 216
`dataentry()` (`pyffi.spells.Spell` method), 228
`dataentry()` (`pyffi.spells.SpellGroupParallelBase` method), 230
`dataentry()` (`pyffi.spells.SpellGroupSeriesBase` method), 231
`dataexit()` (`pyffi.spells.Spell` method), 228
`dataexit()` (`pyffi.spells.SpellGroupParallelBase` method), 230
`dataexit()` (`pyffi.spells.SpellGroupSeriesBase` method), 231
`datainspect()` (`pyffi.spells.nif.fix.SpellAddTangentSpace` method), 207
`datainspect()` (`pyffi.spells.nif.fix.SpellClampMaterialAlpha` method), 209
`datainspect()` (`pyffi.spells.nif.fix.SpellCleanStringPalette` method), 212
`datainspect()` (`pyffi.spells.nif.fix.SpellDelTangentSpace` method), 206
`datainspect()` (`pyffi.spells.nif.fix.SpellDelUnusedRoots` method), 213
`datainspect()` (`pyffi.spells.nif.fix.SpellDetachHavokTriStripsData` method), 208
`datainspect()` (`pyffi.spells.nif.fix.SpellFFVT3RSkinPartData` method), 207
`datainspect()` (`pyffi.spells.nif.fix.SpellFixEmptySkeletonRoots` method), 213
`datainspect()` (`pyffi.spells.nif.fix.SpellMergeSkeletonRoots` method), 210
`datainspect()` (`pyffi.spells.nif.modify._SpellDelBranchClasses` method), 224
`datainspect()` (`pyffi.spells.nif.modify.SpellAddStencilProperty` method), 225
`datainspect()` (`pyffi.spells.nif.modify.SpellChangeBonePriority` method), 221
`datainspect()` (`pyffi.spells.nif.modify.SpellCleanFarNif` method), 226
`datainspect()` (`pyffi.spells.nif.modify.SpellCollisionMaterial` method), 219
`datainspect()` (`pyffi.spells.nif.modify.SpellCollisionType` method), 218
`datainspect()` (`pyffi.spells.nif.modify.SpellDelInterpolatorTransformD` method), 223
`datainspect()` (`pyffi.spells.nif.modify.SpellDelVertexColor` method), 226
`datainspect()` (`pyffi.spells.nif.modify.SpellDisableParallax` method), 225
`datainspect()` (`pyffi.spells.nif.modify.SpellReverseAnimation` method), 220
`datainspect()` (`pyffi.spells.nif.modify.SpellScaleAnimationTime` method), 220
`datainspect()` (`pyffi.spells.nif.modify.SpellSetInterpolatorTransRotSca` method), 222
`datainspect()` (`pyffi.spells.nif.optimize.SpellCleanRefLists` method), 214
`datainspect()` (`pyffi.spells.nif.optimize.SpellDelUnusedBones` method), 216
`datainspect()` (`pyffi.spells.nif.optimize.SpellMergeDuplicates` method), 215
`datainspect()` (`pyffi.spells.nif.optimize.SpellOptimizeGeometry` method), 215
`datainspect()` (`pyffi.spells.Spell` method), 228
`datainspect()` (`pyffi.spells.SpellApplyPatch` method), 226
`datainspect()` (`pyffi.spells.SpellGroupBase` method), 229
`datas()` (`pyffi.formats.nif.NifFormat.NiMesh` property), 107
`DdsFormat` (class in `pyffi.formats.dds`), 24
`DdsFormat.Data` (class in `pyffi.formats.dds`), 24
`DdsFormat.FourCC` (class in `pyffi.formats.dds`), 25
`DdsFormat.HeaderString` (class in `pyffi.formats.dds`), 25
`DDSXMLPATH`, 8
`DEACTIVATOR_INVALID`
`DEACTIVATOR_NEVER`
`pyffi.formats.nif.NifFormat.DeactivatorType` (attribute), 72
`pyffi.formats.nif.NifFormat.DeactivatorType` (attribute), 72
`DEACTIVATOR_SPATIAL`
`pyffi.formats.nif.NifFormat.DeactivatorType` (attribute), 72
`CLASZIP` (`pyffi.formats.nif.NifFormat.SkyrimLayer` attribute), 166
`PROPERTY_LARGE` (`pyffi.formats.nif.NifFormat.Fallout3Layer` attribute), 75
`PRIORITY_LARGE` (`pyffi.formats.nif.NifFormat.SkyrimLayer` attribute), 166

DEBRIS_SMALL (*pyffi.formats.nif.NifFormat.Fallout3Layer attribute*), 75
 DEBRIS_SMALL (*pyffi.formats.nif.NifFormat.SkyrimLayer attribute*), 166
 DECAL_0_MAP (*pyffi.formats.nif.NifFormat.TextType attribute*), 175
 decal_0_texture() (*pyffi.formats.nif.NifFormat.NiTexturingProperty property*), 148
 DECAL_1_MAP (*pyffi.formats.nif.NifFormat.TextType attribute*), 175
 decal_1_texture() (*pyffi.formats.nif.NifFormat.NiTexturingProperty property*), 148
 DECAL_2_MAP (*pyffi.formats.nif.NifFormat.TextType attribute*), 175
 decal_2_texture() (*pyffi.formats.nif.NifFormat.NiTexturingProperty property*), 148
 DECAL_3_MAP (*pyffi.formats.nif.NifFormat.TextType attribute*), 175
 decal_3_texture() (*pyffi.formats.nif.NifFormat.NiTexturingProperty property*), 148
 decay() (*pyffi.formats.nif.NifFormat.NiParticleBomb property*), 132
 decay() (*pyffi.formats.nif.NifFormat.NiPSysBombModifier property*), 124
 decay() (*pyffi.formats.nif.NifFormat.NiPSysGravityModifier property*), 128
 DECAY_EXPONENTIAL (*pyffi.formats.nif.NifFormat.DecayType attribute*), 72
 DECAY_LINEAR (*pyffi.formats.nif.NifFormat.DecayType attribute*), 72
 DECAY_NONE (*pyffi.formats.nif.NifFormat.DecayType attribute*), 72
 decay_type() (*pyffi.formats.nif.NifFormat.NiParticleBomb property*), 132
 decay_type() (*pyffi.formats.nif.NifFormat.NiPSysBombModifier property*), 124
 declination() (*pyffi.formats.nif.NifFormat.NiPSysEmitter property*), 126
 declination_variation() (*pyffi.formats.nif.NifFormat.NiPSysEmitter property*), 126
 Default (*pyffi.formats.nif.NifFormat.BSLightingShaderProperty.ShaderType property*), 52
 DEFAULT_OPTIONS (*pyffi.spells.Toaster attribute*), 231
 delta() (*pyffi.formats.nif.NifFormat.NiFlipController property*), 100
 delta_v() (*pyffi.formats.nif.NifFormat.NiParticleBomb property*), 132
 delta_v() (*pyffi.formats.nif.NifFormat.NiPSysBombModifier property*), 124
 depth() (*pyffi.formats.nif.NifFormat.NiPSysBoxEmitter property*), 124
 DETAIL_MAP (*pyffi.formats.nif.NifFormat.TextType attribute*), 175
 detail_texture() (*pyffi.formats.nif.NifFormat.NiTexturingProperty property*), 148
 die_on_collide() (*pyffi.formats.nif.NifFormat.NiPSysCollider property*), 125
 diffuse_color() (*pyffi.formats.nif.NifFormat.NiLight property*), 106
 dimmer() (*pyffi.formats.nif.NifFormat.NiLight property*), 106
 direct_render() (*pyffi.formats.nif.NifFormat.NiSourceTexture property*), 145
 direction() (*pyffi.formats.nif.NifFormat.NiGravity property*), 105
 direction() (*pyffi.formats.nif.NifFormat.NiParticleBomb property*), 132
 direction() (*pyffi.formats.nif.NifFormat.NiPSysAirFieldModifier property*), 124
 direction() (*pyffi.formats.nif.NifFormat.NiPSysDragFieldModifier property*), 126
 direction() (*pyffi.formats.nif.NifFormat.NiPSysGravityFieldModifier property*), 128
 direction() (*pyffi.formats.nif.NifFormat.NiPSysVortexFieldModifier property*), 131
 distance() (*pyffi.formats.nif.NifFormat.BoneLOD property*), 63
 distance_weight() (*pyffi.formats.nif.NifFormat.BSProceduralLightningController property*), 56
 DOORDETECTION (*pyffi.formats.nif.NifFormat.Fallout3Layer attribute*), 75
 DOORDETECTION (*pyffi.formats.nif.NifFormat.SkyrimLayer attribute*), 166
 drag_axis() (*pyffi.formats.nif.NifFormat.NiPSysDragModifier property*), 126
 DRAW_BOTH (*pyffi.formats.nif.NifFormat.FaceDrawMode attribute*), 74
 DRAW_CCW (*pyffi.formats.nif.NifFormat.FaceDrawMode attribute*), 74
 DRAW_CCW_OR_BOTH (*pyffi.formats.nif.NifFormat.FaceDrawMode attribute*), 74
 DRAW_CW (*pyffi.formats.nif.NifFormat.FaceDrawMode attribute*), 74
 DrawType (*pyffi.formats.nif.NifFormat.NiStencilProperty property*), 146
 DROPPING_PICK (*pyffi.formats.nif.NifFormat.OblivionLayer attribute*), 155
 DROPPING_PICK (*pyffi.formats.nif.NifFormat.Fallout3Layer attribute*), 75
 DROPPING_PICK (*pyffi.formats.nif.NifFormat.SkyrimLayer attribute*), 166

`duration()` (`pyffi.formats.nif.NifFormat.NiParticleBomb` `EMIT_FROM_EDGE_SURFACE`
`property`), 132 (`pyffi.formats.nif.NifFormat.EmitFrom` `at-`
`tribute`), 73

E

`EFFECT_ENVIRONMENT_MAP` (`pyffi.formats.nif.NifFormat.EffectType` `at-`
`tribute`), 72 (`pyffi.formats.nif.NifFormat.EmitFrom` `at-`
`tribute`), 73

`EFFECT_FOG_MAP` (`pyffi.formats.nif.NifFormat.EffectType` `at-`
`tribute`), 72 (`pyffi.formats.nif.NifFormat.EmitFrom` `at-`
`tribute`), 73

`EFFECT_PROJECTED_LIGHT` (`pyffi.formats.nif.NifFormat.EffectType` `at-`
`tribute`), 72 (`pyffi.formats.nif.NifFormat.EmitFrom` `at-`
`tribute`), 73

`EFFECT_PROJECTED_SHADOW` (`pyffi.formats.nif.NifFormat.EffectType` `at-`
`tribute`), 72 (`pyffi.formats.nif.NifFormat.NiParticleSystemController`
`property`), 133

`EgmFormat` (`class` in `pyffi.formats.egm`), 27 (`pyffi.formats.nif.NifFormat.NiParticleSystemController`
`property`), 133

`EgmFormat.Data` (`class` in `pyffi.formats.egm`), 27 (`pyffi.formats.nif.NifFormat.NiParticleSystemController`
`property`), 133

`EgmFormat.FileSignature` (`class` in `pyffi.formats.egm`), 28 (`pyffi.formats.nif.NifFormat.NiParticleSystemController`
`property`), 133

`EgmFormat.FileVersion` (`class` in `pyffi.formats.egm`), 29 (`pyffi.formats.nif.NifFormat.NiParticleSystemController`
`property`), 133

`EgmFormat.MorphRecord` (`class` in `pyffi.formats.egm`), 29 (`pyffi.formats.nif.NifFormat.NiPSParticleSystem`
`property`), 120

`EgtFormat` (`class` in `pyffi.formats.egt`), 32 (`pyffi.formats.nif.NifFormat.NiPSysMeshEmitter`
`property`), 129

`EgtFormat.FileSignature` (`class` in `pyffi.formats.egt`), 32 (`pyffi.formats.nif.NifFormat.NiPSysVolumeEmitter`
`property`), 131

`EgtFormat.FileVersion` (`class` in `pyffi.formats.egt`), 32 (`pyffi.formats.nif.NifFormat.ExtraMeshDataEpicMickey2`
`property`), 73

`EgtFormat.Header` (`class` in `pyffi.formats.egt`), 32 (`pyffi.formats.nif.NifFormat.BSPSysSubTexModifier`
`property`), 55

`elements()` (`pyffi.formats.nif.NifFormat.NiMorphMeshModifier` `property`), 109 (`pyffi.formats.nif.NifFormat.EndianType`
`ENDIAN_BIG` `attribute`), 73

`emission_axis()` (`pyffi.formats.nif.NifFormat.NiPSysMeshEmitter` `property`), 129 (`pyffi.formats.nif.NifFormat.EndianType`
`ENDIAN_LITTLE` `attribute`), 73

`emission_type()` (`pyffi.formats.nif.NifFormat.NiPSysMeshEmitter` `property`), 129 (`pyffi.formats.nif.NifFormat.SubConstraint`
`property`), 172

`emissive_color()` (`pyffi.formats.nif.NifFormat.BSEffectShaderProperty` `property`), 49 (`pyffi.formats.nif.NifFormat.bhkRDTConstraint`
`property`), 185

`emissive_color()` (`pyffi.formats.nif.NifFormat.BSLightingShaderProperty` `property`), 51 (`pyffi.formats.nif.NifFormat.bhkRDTMalleableConstraint`
`property`), 185

`emissive_color()` (`pyffi.formats.nif.NifFormat.BSShaderProperty` `property`), 59 (`pyffi.formats.nif.NifFormat.bhkRDTMalleableConstraint`
`property`), 185

`emissive_multiple()` (`pyffi.formats.nif.NifFormat.BSEffectShaderProperty` `property`), 49 (`pyffi.formats.nif.NifFormat.bhkRDTConstraint`
`property`), 185

`emissive_multiple()` (`pyffi.formats.nif.NifFormat.BSLightingShaderProperty` `property`), 51 (`pyffi.formats.nif.NifFormat.bhkRDTMalleableConstraint`
`property`), 185

`EmissiveMultiple` (`pyffi.formats.nif.NifFormat.EffectShaderController` `property`), 78 (`pyffi.formats.nif.NifFormat.FurniturePosition`
`property`), 78

`emit_flags()` (`pyffi.formats.nif.NifFormat.NiParticleSystemController` `property`), 133 (`pyffi.formats.nif.NifFormat.EnvironmentVariable`
`CGXMLPATH`, 8
`DDXMLPATH`, 8
`KFMXMLPATH`, 8
`NIFXMLPATH`, 8
`TGXMLPATH`, 8

`EMIT_FROM_EDGE_CENTER` (`pyffi.formats.nif.NifFormat.EmitFrom` `at-`
`tribute`), 73

environment_map_scale() (pyffi.formats.nif.NifFormat.BSLightingShaderProperty attribute), 51
 environment_map_scale() (pyffi.formats.nif.NifFormat.BSShaderProperty attribute), 60
 EPSILON (pyffi.formats.nif.NifFormat attribute), 72
 error() (pyffi.formats.nif.NifFormat.bhkCompressedMeshShapeData attribute), 67
 error() (pyffi.formats.nif.NifFormat attribute), 180
 EspFormat (class in pyffi.formats.esp), 35
 EspFormat.Data (class in pyffi.formats.esp), 35
 EspFormat.GRUP (class in pyffi.formats.esp), 35
 EspFormat.Record (class in pyffi.formats.esp), 36
 EspFormat.RecordType (class in pyffi.formats.esp), 36
 EspFormat.SubRecord (class in pyffi.formats.esp), 36
 EspFormat.ZString (class in pyffi.formats.esp), 36
 EXAMPLES (pyffi.spells.Toaster attribute), 231
 exclude_types (pyffi.spells.Toaster attribute), 231
 exponent() (pyffi.formats.nif.NifFormat.NiSpotLight property), 146
 export_info_1() (pyffi.formats.nif.NifFormat.ExportInfo attribute), 73
 export_info_2() (pyffi.formats.nif.NifFormat.ExportInfo attribute), 73
 extent() (pyffi.formats.nif.NifFormat.BoxBV property), 64
 extent() (pyffi.formats.nif.NifFormat.BSMultiBoundAABB property), 52
 extra_flags() (pyffi.formats.nif.NifFormat.NiGeomMorphComponent attribute), 102
 extra_targets() (pyffi.formats.nif.NifFormat.NiMultiTargetTransformComponent attribute), 109
 eye_cubemap_scale() (pyffi.formats.nif.NifFormat.BSLightingShaderProperty attribute), 51
F
 F_FLOAT16_1 (pyffi.formats.nif.NifFormat.ComponentFormat attribute), 66
 F_FLOAT16_2 (pyffi.formats.nif.NifFormat.ComponentFormat attribute), 66
 F_FLOAT16_3 (pyffi.formats.nif.NifFormat.ComponentFormat attribute), 66
 F_FLOAT16_4 (pyffi.formats.nif.NifFormat.ComponentFormat attribute), 66
 F_FLOAT32_1 (pyffi.formats.nif.NifFormat.ComponentFormat attribute), 66
 F_FLOAT32_2 (pyffi.formats.nif.NifFormat.ComponentFormat attribute), 66
 F_FLOAT32_3 (pyffi.formats.nif.NifFormat.ComponentFormat attribute), 66
 F_FLOAT32_4 (pyffi.formats.nif.NifFormat.ComponentFormat attribute), 66
 F_INT16_1 (pyffi.formats.nif.NifFormat.ComponentFormat attribute), 67
 F_INT16_2 (pyffi.formats.nif.NifFormat.ComponentFormat attribute), 67
 F_INT16_3 (pyffi.formats.nif.NifFormat.ComponentFormat attribute), 67
 F_INT16_4 (pyffi.formats.nif.NifFormat.ComponentFormat attribute), 67
 F_INT32_1 (pyffi.formats.nif.NifFormat.ComponentFormat attribute), 67
 F_INT32_2 (pyffi.formats.nif.NifFormat.ComponentFormat attribute), 67
 F_INT32_3 (pyffi.formats.nif.NifFormat.ComponentFormat attribute), 67
 F_INT32_4 (pyffi.formats.nif.NifFormat.ComponentFormat attribute), 67
 F_INT8_1 (pyffi.formats.nif.NifFormat.ComponentFormat attribute), 67
 F_INT8_2 (pyffi.formats.nif.NifFormat.ComponentFormat attribute), 67
 F_INT8_3 (pyffi.formats.nif.NifFormat.ComponentFormat attribute), 67
 F_INT8_4 (pyffi.formats.nif.NifFormat.ComponentFormat attribute), 67
 F_NORMINT16_1 (pyffi.formats.nif.NifFormat.ComponentFormat attribute), 67
 F_NORMINT16_2 (pyffi.formats.nif.NifFormat.ComponentFormat attribute), 67
 F_NORMINT16_3 (pyffi.formats.nif.NifFormat.ComponentFormat attribute), 67
 F_NORMINT32_1 (pyffi.formats.nif.NifFormat.ComponentFormat attribute), 67
 F_NORMINT32_2 (pyffi.formats.nif.NifFormat.ComponentFormat attribute), 67
 F_NORMINT32_3 (pyffi.formats.nif.NifFormat.ComponentFormat attribute), 67
 F_NORMINT32_4 (pyffi.formats.nif.NifFormat.ComponentFormat attribute), 67
 F_NORMINT8_1 (pyffi.formats.nif.NifFormat.ComponentFormat attribute), 67
 F_NORMINT8_2 (pyffi.formats.nif.NifFormat.ComponentFormat attribute), 67
 F_NORMINT8_3 (pyffi.formats.nif.NifFormat.ComponentFormat attribute), 67
 F_NORMINT8_4 (pyffi.formats.nif.NifFormat.ComponentFormat attribute), 67
 F_NORMINT_10_10_10_2 (pyffi.formats.nif.NifFormat.ComponentFormat attribute), 67
 F_NORMINT_10_10_10_L1 (pyffi.formats.nif.NifFormat.ComponentFormat attribute), 67

<i>(pyffi.formats.nif.NifFormat.ComponentFormat attribute), 67</i>	<i>F_UINT8_4 (pyffi.formats.nif.NifFormat.ComponentFormat attribute), 68</i>
<i>F_NORMINT_11_11_10 (pyffi.formats.nif.NifFormat.ComponentFormat attribute), 67</i>	<i>F_UINT_10_10_10_2 (pyffi.formats.nif.NifFormat.ComponentFormat attribute), 68</i>
<i>F_NORMUINT16_1 (pyffi.formats.nif.NifFormat.ComponentFormat attribute), 67</i>	<i>F_UINT_10_10_10_L1 (pyffi.formats.nif.NifFormat.ComponentFormat attribute), 68</i>
<i>F_NORMUINT16_2 (pyffi.formats.nif.NifFormat.ComponentFormat attribute), 67</i>	<i>F_UNKNOWN (pyffi.formats.nif.NifFormat.ComponentFormat attribute), 68</i>
<i>F_NORMUINT16_3 (pyffi.formats.nif.NifFormat.ComponentFormat attribute), 67</i>	<i>fade () (pyffi.formats.nif.NifFormat.NiParticleGrowFade property), 132</i>
<i>F_NORMUINT16_4 (pyffi.formats.nif.NifFormat.ComponentFormat attribute), 67</i>	<i>fade_generation () (pyffi.formats.nif.NifFormat.NiPSysGrowFadeModifier property), 128</i>
<i>F_NORMUINT32_1 (pyffi.formats.nif.NifFormat.ComponentFormat attribute), 67</i>	<i>fade_in_percent () (pyffi.formats.nif.NifFormat.BSPSysSimpleColorModifier property), 54</i>
<i>F_NORMUINT32_2 (pyffi.formats.nif.NifFormat.ComponentFormat attribute), 67</i>	<i>fade_out_percent () (pyffi.formats.nif.NifFormat.BSPSysSimpleColorModifier property), 54</i>
<i>F_NORMUINT32_3 (pyffi.formats.nif.NifFormat.ComponentFormat attribute), 67</i>	<i>fade_time () (pyffi.formats.nif.NifFormat.NiPSysGrowFadeModifier property), 128</i>
<i>F_NORMUINT32_4 (pyffi.formats.nif.NifFormat.ComponentFormat attribute), 67</i>	<i>fade_action () (pyffi.formats.nif.NifFormat.NiStencilProperty property), 146</i>
<i>F_NORMUINT8_1 (pyffi.formats.nif.NifFormat.ComponentFormat attribute), 67</i>	<i>falloff_start_angle () (pyffi.formats.nif.NifFormat.BSEffectShaderProperty property), 49</i>
<i>F_NORMUINT8_2 (pyffi.formats.nif.NifFormat.ComponentFormat attribute), 68</i>	<i>falloff_start_angle () (pyffi.formats.nif.NifFormat.BSShaderNoLightingProperty property), 59</i>
<i>F_NORMUINT8_3 (pyffi.formats.nif.NifFormat.ComponentFormat attribute), 68</i>	<i>falloff_start_opacity () (pyffi.formats.nif.NifFormat.BSEffectShaderProperty property), 49</i>
<i>F_NORMUINT8_4 (pyffi.formats.nif.NifFormat.ComponentFormat attribute), 68</i>	<i>falloff_start_opacity () (pyffi.formats.nif.NifFormat.BSShaderNoLightingProperty property), 59</i>
<i>F_NORMUINT8_4_BGRA (pyffi.formats.nif.NifFormat.ComponentFormat attribute), 68</i>	<i>falloff_stop_angle () (pyffi.formats.nif.NifFormat.BSEffectShaderProperty property), 49</i>
<i>F_UINT16_1 (pyffi.formats.nif.NifFormat.ComponentFormat attribute), 68</i>	<i>falloff_stop_angle () (pyffi.formats.nif.NifFormat.BSShaderNoLightingProperty property), 59</i>
<i>F_UINT16_2 (pyffi.formats.nif.NifFormat.ComponentFormat attribute), 68</i>	<i>falloff_stop_opacity () (pyffi.formats.nif.NifFormat.BSEffectShaderProperty property), 49</i>
<i>F_UINT16_3 (pyffi.formats.nif.NifFormat.ComponentFormat attribute), 68</i>	<i>falloff_stop_opacity () (pyffi.formats.nif.NifFormat.BSShaderNoLightingProperty property), 59</i>
<i>F_UINT16_4 (pyffi.formats.nif.NifFormat.ComponentFormat attribute), 68</i>	<i>far_extent () (pyffi.formats.nif.NifFormat.LODRange property), 81</i>
<i>F_UINT32_1 (pyffi.formats.nif.NifFormat.ComponentFormat attribute), 68</i>	<i>field_object () (pyffi.formats.nif.NifFormat.NiPSysFieldModifier property), 127</i>
<i>F_UINT32_2 (pyffi.formats.nif.NifFormat.ComponentFormat attribute), 68</i>	<i>FIELD_POINT (pyffi.formats.nif.NifFormat.FieldType</i>
<i>F_UINT32_3 (pyffi.formats.nif.NifFormat.ComponentFormat attribute), 68</i>	
<i>F_UINT32_4 (pyffi.formats.nif.NifFormat.ComponentFormat attribute), 68</i>	
<i>F_UINT8_1 (pyffi.formats.nif.NifFormat.ComponentFormat attribute), 68</i>	
<i>F_UINT8_2 (pyffi.formats.nif.NifFormat.ComponentFormat attribute), 68</i>	
<i>F_UINT8_3 (pyffi.formats.nif.NifFormat.ComponentFormat attribute), 68</i>	

attribute), 77
FIELD_WIND (*pyffi.formats.nif.NifFormat.FieldType attribute*), 77
file, 267
file format, 267
file format interface, 267
file_name() (*pyffi.formats.nif.NifFormat.BSShaderNoLightingProperty*), 59
file_name() (*pyffi.formats.nif.NifFormat.NiImageProperty*), 105
file_name() (*pyffi.formats.nif.NifFormat.NiSourceTextureProperty*), 145
file_name() (*pyffi.formats.nif.NifFormat.SkyShaderProperty*), 164
file_name() (*pyffi.formats.nif.NifFormat.TallGrassShaderProperty*), 173
file_name() (*pyffi.formats.nif.NifFormat.TexSourceProperty*), 174
file_name() (*pyffi.formats.nif.NifFormat.TileShaderProperty*), 175
FileFormat (class in *pyffi.object_models*), 234
FILEFORMAT (*pyffi.spells.Toaster attribute*), 231
FileFormat.Data (class in *pyffi.object_models*), 234
filter() (*pyffi.formats.nif.NifFormat.MultiTextureElementProperty*), 87
FILTER_BILERP (*pyffi.formats.nif.NifFormat.TexFilterMode attribute*), 174
FILTER_BILERP_MIPNEAREST (*pyffi.formats.nif.NifFormat.TexFilterMode attribute*), 174
filter_mode() (*pyffi.formats.nif.NifFormat.TexDescProperty*), 174
FILTER_NEAREST (*pyffi.formats.nif.NifFormat.TexFilterMode attribute*), 174
FILTER_NEAREST_MIPLERP (*pyffi.formats.nif.NifFormat.TexFilterMode attribute*), 174
FILTER_NEAREST_MIPNEAREST (*pyffi.formats.nif.NifFormat.TexFilterMode attribute*), 174
FILTER_TRILERP (*pyffi.formats.nif.NifFormat.TexFilterMode attribute*), 174
find() (*pyffi.formats.nif.NifFormat.NiObject method*), 113
find_chain() (*pyffi.formats.nif.NifFormat.NiObject method*), 113
fix_links() (*pyffi.formats.cgf.CgfFormat.Ref method*), 16
fix_links() (*pyffi.formats.nif.NifFormat.Ref method*), 160
flag_or_num_constraints() (*pyffi.formats.nif.NifFormat.bhkRagdollTemplateDataProperty*), 186
flags() (*pyffi.formats.nif.NifFormat.bhkNiCollisionObjectProperty*), 183
flags() (*pyffi.formats.nif.NifFormat.BSSegmentProperty*), 57
flags() (*pyffi.formats.nif.NifFormat.NiAlphaPropertyProperty*), 90
flags() (*pyffi.formats.nif.NifFormat.NiDitherPropertyProperty*), 99
flags() (*pyffi.formats.nif.NifFormat.NiFogPropertyProperty*), 101
flags() (*pyffi.formats.nif.NifFormat.NiMorphMeshModifierProperty*), 109
flags() (*pyffi.formats.nif.NifFormat.NiMultiTexturePropertyProperty*), 110
flags() (*pyffi.formats.nif.NifFormat.NiShadePropertyProperty*), 143
flags() (*pyffi.formats.nif.NifFormat.NiSkinningMeshModifierProperty*), 145
flags() (*pyffi.formats.nif.NifFormat.NiSpecularPropertyProperty*), 145
flags() (*pyffi.formats.nif.NifFormat.NiStencilPropertyProperty*), 146
flags() (*pyffi.formats.nif.NifFormat.NiTexturePropertyProperty*), 148
flags() (*pyffi.formats.nif.NifFormat.NiTexturingPropertyProperty*), 148
flags() (*pyffi.formats.nif.NifFormat.NiTimeControllerProperty*), 149
flags() (*pyffi.formats.nif.NifFormat.NiVertexColorPropertyProperty*), 153
flags() (*pyffi.formats.nif.NifFormat.NiWireframePropertyProperty*), 154
flags() (*pyffi.formats.nif.NifFormat.NiZBufferPropertyProperty*), 154
flags() (*pyffi.formats.nif.NifFormat.TexDescProperty*), 174
flags_and_part_number() (*pyffi.formats.nif.NifFormat.HavokColFilterProperty*), 79
flatten_skin() (*pyffi.formats.nif.NifFormat.NiGeometry method*), 103
float (*pyffi.formats.cgf.CgfFormat attribute*), 19
float (*pyffi.formats.dds.DdsFormat attribute*), 25
float (*pyffi.formats.egm.EgmFormat attribute*), 30
float (*pyffi.formats.egt.EgtFormat attribute*), 33
float (*pyffi.formats.esp.EspFormat attribute*), 37
float (*pyffi.formats.kfm.KfmFormat attribute*), 41
float (*pyffi.formats.nif.NifFormat attribute*), 188
float (*pyffi.formats.tga.TgaFormat attribute*), 197
float (*pyffi.formats.tri.TriFormat attribute*), 201
float_2() (*pyffi.formats.nif.NifFormat.BSProceduralLightningControllerProperty*), 56
float_5() (*pyffi.formats.nif.NifFormat.BSProceduralLightningControllerProperty*), 56
float_data() (*pyffi.formats.nif.NifFormat.NiFloatExtraData*

property), 100
 float_data() (pyffi.formats.nif.NifFormat.NiPathController property), 97
 property), 135
 float_data() (pyffi.formats.nif.NifFormat.NiPathInterpolator property), 97
 property), 135
 float_keys() (pyffi.formats.nif.NifFormat.NiPSysEmitterCtrlData property), 97
 property), 126
 float_value() (pyffi.formats.nif.NifFormat.NiBlendFloatInterpolator property), 97
 property), 95
 float_value() (pyffi.formats.nif.NifFormat.NiFloatInterpolator property), 97
 property), 101
 floats() (pyffi.formats.nif.NifFormat.BSPSysScaleModifier property), 154
 property), 54
 floats_1() (pyffi.formats.nif.NifFormat.bhkBallSocketConstraintChain property), 177
 fog_color() (pyffi.formats.nif.NifFormat.NiFogProperty property), 101
 fog_depth() (pyffi.formats.nif.NifFormat.NiFogProperty property), 101
 force() (pyffi.formats.nif.NifFormat.NiGravity property), 105
 FORCE_PLANAR (pyffi.formats.nif.NifFormat.ForceType attribute), 78
 FORCE_SPHERICAL (pyffi.formats.nif.NifFormat.ForceType attribute), 78
 force_type() (pyffi.formats.nif.NifFormat.NiPSysGravityModifier property), 128
 FORCE_UNKNOWN (pyffi.formats.nif.NifFormat.ForceType attribute), 78
 forces() (pyffi.formats.nif.NifFormat.NiPSSimulatorForcesStep property), 122
 fork() (pyffi.formats.nif.NifFormat.BSPProceduralLightningController property), 56
 forward() (pyffi.formats.nif.NifFormat.Key property), 81
 frame_count() (pyffi.formats.nif.NifFormat.BSPSysSubTexModifier property), 55
 frame_count_fudge() (pyffi.formats.nif.NifFormat.BSPSysSubTexModifier property), 55
 frame_name() (pyffi.formats.nif.NifFormat.Morph property), 86
 frequency() (pyffi.formats.nif.NifFormat.NiPSysTurbulenceFieldModifier property), 131
 frequency() (pyffi.formats.nif.NifFormat.NiTimeController property), 149
 friction() (pyffi.formats.nif.NifFormat.bhkRagdollTemplateData property), 186
 friction() (pyffi.formats.nif.NifFormat.PrismaticDescriptor property), 158
 front() (pyffi.formats.nif.NifFormat.FurnitureEntryPoints property), 78
 frustum_bottom() (pyffi.formats.nif.NifFormat.NiCamera property), 97
 frustum_far() (pyffi.formats.nif.NifFormat.NiCamera property), 97
 frustum_left() (pyffi.formats.nif.NifFormat.NiCamera property), 97
 frustum_near() (pyffi.formats.nif.NifFormat.NiCamera property), 97
 frustum_right() (pyffi.formats.nif.NifFormat.NiCamera property), 97
 frustum_top() (pyffi.formats.nif.NifFormat.NiCamera property), 97
 function() (pyffi.formats.nif.NifFormat.NiZBufferProperty property), 154
 G
 g() (pyffi.formats.nif.NifFormat.ByteColor3 property), 64
 g() (pyffi.formats.nif.NifFormat.ByteColor4 property), 64
 g() (pyffi.formats.nif.NifFormat.Color3 property), 66
 g() (pyffi.formats.nif.NifFormat.Color4 property), 66
 games (pyffi.formats.nif.NifFormat attribute), 188
 GASTRAP (pyffi.formats.nif.NifFormat.Fallout3Layer attribute), 75
 GASTRAP (pyffi.formats.nif.NifFormat.SkyrimLayer attribute), 166
 generate() (pyffi.formats.nif.NifFormat.NiPSParticleSystem property), 120
 get_all_strings() (pyffi.formats.nif.NifFormat.StringPalette method), 171
 get_children() (pyffi.formats.nif.NifFormat.NiNode method), 111
 get_chunk_types() (pyffi.formats.cgf.CgfFormat.ChunkTable method), 12
 get_comp_data() (pyffi.formats.nif.NifFormat.NiBSplineData method), 93
 get_controller_type() (pyffi.formats.nif.NifFormat.ControllerLink method), 69
 get_controllers() (pyffi.formats.nif.NifFormat.NiObjectNET method), 113
 get_copy() (pyffi.formats.cgf.CgfFormat.Matrix33 method), 15
 get_copy() (pyffi.formats.cgf.CgfFormat.Matrix44 method), 15
 get_copy() (pyffi.formats.nif.NifFormat.InertiaMatrix method), 80
 get_copy() (pyffi.formats.nif.NifFormat.Matrix33 method), 83
 get_copy() (pyffi.formats.nif.NifFormat.Matrix44 method), 84

[get_copy\(\)](#) ([pyffi.formats.nif.NifFormat.Vector3 method](#)), 176
[get_copy\(\)](#) ([pyffi.formats.nif.NifFormat.Vector4 method](#)), 176
[get_detail_child_names\(\)](#) ([pyffi.formats.cgf.CgfFormat.Data method](#)), 12
[get_detail_child_names\(\)](#) ([pyffi.formats.dds.DdsFormat.Data method](#)), 24
[get_detail_child_names\(\)](#) ([pyffi.formats.egm.EgmFormat.Data method](#)), 28
[get_detail_child_names\(\)](#) ([pyffi.formats.esp.EspFormat.Data method](#)), 35
[get_detail_child_names\(\)](#) ([pyffi.formats.nif.NifFormat.Data method](#)), 70
[get_detail_child_names\(\)](#) ([pyffi.formats.tga.TgaFormat.Image method](#)), 197
[get_detail_child_nodes\(\)](#) ([pyffi.formats.cgf.CgfFormat.Data method](#)), 12
[get_detail_child_nodes\(\)](#) ([pyffi.formats.dds.DdsFormat.Data method](#)), 24
[get_detail_child_nodes\(\)](#) ([pyffi.formats.egm.EgmFormat.Data method](#)), 28
[get_detail_child_nodes\(\)](#) ([pyffi.formats.esp.EspFormat.Data method](#)), 35
[get_detail_child_nodes\(\)](#) ([pyffi.formats.nif.NifFormat.Data method](#)), 70
[get_detail_child_nodes\(\)](#) ([pyffi.formats.tga.TgaFormat.Image method](#)), 197
[get_detail_display\(\)](#) ([pyffi.formats.bsa.BsaFormat.Hash method](#)), 9
[get_detail_display\(\)](#) ([pyffi.formats.dds.DdsFormat.HeaderString method](#)), 25
[get_detail_display\(\)](#) ([pyffi.formats.egm.EgmFormat.FileSignature method](#)), 28
[get_detail_display\(\)](#) ([pyffi.formats.egm.EgmFormat.FileVersion method](#)), 29
[get_detail_display\(\)](#) ([pyffi.formats.egt.EgtFormat.FileSignature method](#)), 32
[get_detail_display\(\)](#) ([pyffi.formats.egt.EgtFormat.FileVersion method](#)), 32
[get_detail_display\(\)](#) ([pyffi.formats.kfm.KfmFormat.HeaderString method](#)), 39
[get_detail_display\(\)](#) ([pyffi.formats.nif.NifFormat.Data.VersionUInt method](#)), 70
[get_detail_display\(\)](#) ([pyffi.formats.nif.NifFormat.FileVersion method](#)), 77
[get_detail_display\(\)](#) ([pyffi.formats.nif.NifFormat.HeaderString method](#)), 79
[get_detail_display\(\)](#) ([pyffi.formats.nif.NifFormat.Ref method](#)), 160
[get_detail_display\(\)](#) ([pyffi.formats.tri.TriFormat.FileSignature method](#)), 199
[get_detail_display\(\)](#) ([pyffi.formats.tri.TriFormat.FileVersion method](#)), 199
[get_determinant\(\)](#) ([pyffi.formats.cgf.CgfFormat.Matrix33 method](#)), 15
[get_determinant\(\)](#) ([pyffi.formats.nif.NifFormat.Matrix33 method](#)), 83
[get_dismember_partitions\(\)](#) ([pyffi.formats.nif.NifFormat.BSDismemberSkinInstance method](#)), 49
[get_effects\(\)](#) ([pyffi.formats.nif.NifFormat.NiNode method](#)), 112
[get_extra_datas\(\)](#) ([pyffi.formats.nif.NifFormat.NiObjectNET method](#)), 113
[get_float_data\(\)](#) ([pyffi.formats.nif.NifFormat.NiBSplineData method](#)), 93
[get_global_child_nodes\(\)](#) ([pyffi.formats.cgf.CgfFormat.Data method](#)), 13
[get_global_child_nodes\(\)](#) ([pyffi.formats.egm.EgmFormat.Data method](#)), 28
[get_global_child_nodes\(\)](#) ([pyffi.formats.egt.EgtFormat.Header method](#)), 33
[get_global_child_nodes\(\)](#) ([pyffi.formats.esp.EspFormat.Data method](#)), 35
[get_global_child_nodes\(\)](#) ([pyffi.formats.esp.EspFormat.GRUP method](#)), 35
[get_global_child_nodes\(\)](#) ([pyffi.formats.esp.EspFormat.Record method](#)), 36
[get_global_child_nodes\(\)](#) ([pyffi.formats.kfm.KfmFormat.Data method](#)),

39	160
<code>get_global_child_nodes()</code>	<code>get_hash()</code> (<code>pyffi.formats.nif.NifFormat.ShortString</code>
(<code>pyffi.formats.nif.NifFormat.Data</code> <i>method</i>),	<i>method</i>), 162
70	<code>get_hash()</code> (<code>pyffi.formats.nif.NifFormat.SizedString</code>
<code>get_global_child_nodes()</code>	<i>method</i>), 162
(<code>pyffi.formats.tga.TgaFormat.Data</code> <i>method</i>),	<code>get_hash()</code> (<code>pyffi.formats.nif.NifFormat.string</code>
196	<i>method</i>), 189
<code>get_global_child_nodes()</code>	<code>get_hash()</code> (<code>pyffi.formats.tga.TgaFormat.FooterString</code>
(<code>pyffi.formats.tri.TriFormat.Header</code> <i>method</i>),	<i>method</i>), 196
200	<code>get_hash()</code> (<code>pyffi.formats.tri.TriFormat.FileSignature</code>
<code>get_global_display()</code>	<i>method</i>), 199
(<code>pyffi.formats.kfm.KfmFormat.Data</code> <i>method</i>),	<code>get_hash()</code> (<code>pyffi.formats.tri.TriFormat.FileVersion</code>
39	<i>method</i>), 199
<code>get_hash()</code> (<code>pyffi.formats.bsa.BsaFormat.ZString</code>	<code>get_interchangeable_packed_shape()</code>
<i>method</i>), 9	(<code>pyffi.formats.nif.NifFormat.bhkNiTriStripsShape</code>
<code>get_hash()</code> (<code>pyffi.formats.cgf.CgfFormat.FileSignature</code>	<i>method</i>), 183
<i>method</i>), 14	<code>get_interchangeable_tri_shape()</code>
<code>get_hash()</code> (<code>pyffi.formats.cgf.CgfFormat.Ref</code> <i>method</i>),	(<code>pyffi.formats.nif.NifFormat.NiTriBasedGeom</code>
16	<i>method</i>), 150
<code>get_hash()</code> (<code>pyffi.formats.cgf.CgfFormat.SizedString</code>	<code>get_interchangeable_tri_strips()</code>
<i>method</i>), 17	(<code>pyffi.formats.nif.NifFormat.NiTriBasedGeom</code>
<code>get_hash()</code> (<code>pyffi.formats.dds.DdsFormat.HeaderString</code>	<i>method</i>), 150
<i>method</i>), 25	<code>get_inverse()</code> (<code>pyffi.formats.cgf.CgfFormat.Matrix33</code>
<code>get_hash()</code> (<code>pyffi.formats.egm.EgmFormat.FileSignature</code>	<i>method</i>), 15
<i>method</i>), 28	<code>get_inverse()</code> (<code>pyffi.formats.cgf.CgfFormat.Matrix44</code>
<code>get_hash()</code> (<code>pyffi.formats.egm.EgmFormat.FileVersion</code>	<i>method</i>), 15
<i>method</i>), 29	<code>get_inverse()</code> (<code>pyffi.formats.nif.NifFormat.Matrix33</code>
<code>get_hash()</code> (<code>pyffi.formats.egt.EgtFormat.FileSignature</code>	<i>method</i>), 83
<i>method</i>), 32	<code>get_inverse()</code> (<code>pyffi.formats.nif.NifFormat.Matrix44</code>
<code>get_hash()</code> (<code>pyffi.formats.egt.EgtFormat.FileVersion</code>	<i>method</i>), 84
<i>method</i>), 32	<code>get_links()</code> (<code>pyffi.formats.cgf.CgfFormat.Ref</code>
<code>get_hash()</code> (<code>pyffi.formats.esp.EspFormat.ZString</code>	<i>method</i>), 16
<i>method</i>), 36	<code>get_links()</code> (<code>pyffi.formats.nif.NifFormat.Ref</code>
<code>get_hash()</code> (<code>pyffi.formats.kfm.KfmFormat.FilePath</code>	<i>method</i>), 160
<i>method</i>), 39	<code>get_mapped_triangles()</code>
<code>get_hash()</code> (<code>pyffi.formats.kfm.KfmFormat.HeaderString</code>	(<code>pyffi.formats.nif.NifFormat.SkinPartition</code>
<i>method</i>), 39	<i>method</i>), 163
<code>get_hash()</code> (<code>pyffi.formats.kfm.KfmFormat.SizedString</code>	<code>get_mass_center_inertia()</code>
<i>method</i>), 40	(<code>pyffi.formats.nif.NifFormat.bhkBoxShape</code>
<code>get_hash()</code> (<code>pyffi.formats.nif.NifFormat.bool</code> <i>method</i>),	<i>method</i>), 178
188	<code>get_mass_center_inertia()</code>
<code>get_hash()</code> (<code>pyffi.formats.nif.NifFormat.ByteArray</code>	(<code>pyffi.formats.nif.NifFormat.bhkCapsuleShape</code>
<i>method</i>), 64	<i>method</i>), 179
<code>get_hash()</code> (<code>pyffi.formats.nif.NifFormat.ByteMatrix</code>	<code>get_mass_center_inertia()</code>
<i>method</i>), 64	(<code>pyffi.formats.nif.NifFormat.bhkConvexVerticesShape</code>
<code>get_hash()</code> (<code>pyffi.formats.nif.NifFormat.FilePath</code>	<i>method</i>), 181
<i>method</i>), 77	<code>get_mass_center_inertia()</code>
<code>get_hash()</code> (<code>pyffi.formats.nif.NifFormat.HeaderString</code>	(<code>pyffi.formats.nif.NifFormat.bhkListShape</code>
<i>method</i>), 79	<i>method</i>), 182
<code>get_hash()</code> (<code>pyffi.formats.nif.NifFormat.LineString</code>	<code>get_mass_center_inertia()</code>
<i>method</i>), 82	(<code>pyffi.formats.nif.NifFormat.bhkMoppBvTreeShape</code>
<code>get_hash()</code> (<code>pyffi.formats.nif.NifFormat.Ptr</code> <i>method</i>),	<i>method</i>), 183
159	<code>get_mass_center_inertia()</code>
<code>get_hash()</code> (<code>pyffi.formats.nif.NifFormat.Ref</code> <i>method</i>),	(<code>pyffi.formats.nif.NifFormat.bhkMultiSphereShape</code>

`method)`, 183
`get_mass_center_inertia()` (`pyffi.formats.nif.NifFormat.bhkNiTriStripsShape` `method)`, 183
`get_mass_center_inertia()` (`pyffi.formats.nif.NifFormat.bhkPackedNiTriStripsShape` `method)`, 184
`get_mass_center_inertia()` (`pyffi.formats.nif.NifFormat.bhkSphereShape` `method)`, 187
`get_mass_center_inertia()` (`pyffi.formats.nif.NifFormat.bhkTransformShape` `method)`, 187
`get_matrix_33()` (`pyffi.formats.cgf.CgfFormat.Matrix44` `method)`, 16
`get_matrix_33()` (`pyffi.formats.nif.NifFormat.Matrix44` `method)`, 84
`get_node_name()` (`pyffi.formats.nif.NifFormat.ControllerLink` `method)`, 69
`get_properties()` (`pyffi.formats.nif.NifFormat.NiAVObject` `method)`, 89
`get_property_type()` (`pyffi.formats.nif.NifFormat.ControllerLink` `method)`, 69
`get_refs()` (`pyffi.formats.cgf.CgfFormat.Ptr` `method)`, 16
`get_refs()` (`pyffi.formats.cgf.CgfFormat.Ref` `method)`, 17
`get_refs()` (`pyffi.formats.nif.NifFormat.Ptr` `method)`, 159
`get_refs()` (`pyffi.formats.nif.NifFormat.Ref` `method)`, 160
`get_rotations()` (`pyffi.formats.nif.NifFormat.NiBSplineCompTransformInterpolator` `method)`, 92
`get_rotations()` (`pyffi.formats.nif.NifFormat.NiBSplineTransformInterpolator` `method)`, 93
`get_scale()` (`pyffi.formats.cgf.CgfFormat.Matrix33` `method)`, 15
`get_scale()` (`pyffi.formats.nif.NifFormat.Matrix33` `method)`, 83
`get_scale_quat()` (`pyffi.formats.cgf.CgfFormat.Matrix33` `method)`, 15
`get_scale_quat()` (`pyffi.formats.nif.NifFormat.Matrix33` `method)`, 83
`get_scale_quat_translation()` (`pyffi.formats.nif.NifFormat.Matrix44` `method)`, 84
`get_scale_rotation()` (`pyffi.formats.cgf.CgfFormat.Matrix33` `method)`, 15
`get_scale_rotation()` (`pyffi.formats.nif.NifFormat.Matrix33` `method)`, 84
`get_scale_rotation_translation()` (`pyffi.formats.nif.NifFormat.Matrix44` `method)`, 84
`get_scales()` (`pyffi.formats.nif.NifFormat.NiBSplineCompTransformInterpolator` `method)`, 92
`get_scales()` (`pyffi.formats.nif.NifFormat.NiBSplineTransformInterpolator` `method)`, 93
`get_shape_mass_center_inertia()` (`pyffi.formats.nif.NifFormat.bhkRefObject` `method)`, 186
`get_short_data()` (`pyffi.formats.nif.NifFormat.NiBSplineData` `method)`, 93
`get_size()` (`pyffi.formats.bsa.BsaFormat.BZString` `method)`, 8
`get_size()` (`pyffi.formats.bsa.BsaFormat.FileVersion` `method)`, 8
`get_size()` (`pyffi.formats.bsa.BsaFormat.ZString` `method)`, 9
`get_size()` (`pyffi.formats.cgf.CgfFormat.FileSignature` `method)`, 14
`get_size()` (`pyffi.formats.cgf.CgfFormat.Ref` `method)`, 17
`get_size()` (`pyffi.formats.cgf.CgfFormat.SizedString` `method)`, 18
`get_size()` (`pyffi.formats.dds.DdsFormat.HeaderString` `method)`, 25
`get_size()` (`pyffi.formats.egm.EgmFormat.FileSignature` `method)`, 28
`get_size()` (`pyffi.formats.egm.EgmFormat.FileVersion` `method)`, 29
`get_size()` (`pyffi.formats.egt.EgtFormat.FileSignature` `method)`, 32
`get_size()` (`pyffi.formats.egt.EgtFormat.FileVersion` `method)`, 32
`get_size()` (`pyffi.formats.esp.EspFormat.ZString` `method)`, 39
`get_size()` (`pyffi.formats.kfm.KfmFormat.HeaderString` `method)`, 39
`get_size()` (`pyffi.formats.kfm.KfmFormat.SizedString` `method)`, 40
`get_size()` (`pyffi.formats.nif.NifFormat.bool` `method)`, 188
`get_size()` (`pyffi.formats.nif.NifFormat.ByteArray` `method)`, 64
`get_size()` (`pyffi.formats.nif.NifFormat.ByteMatrix` `method)`, 64
`get_size()` (`pyffi.formats.nif.NifFormat.HeaderString` `method)`, 79
`get_size()` (`pyffi.formats.nif.NifFormat.LineString` `method)`, 82
`get_size()` (`pyffi.formats.nif.NifFormat.Ref` `method)`, 161
`get_size()` (`pyffi.formats.nif.NifFormat.ShortString` `method)`, 162
`get_size()` (`pyffi.formats.nif.NifFormat.SizedString` `method)`, 162

`method)`, 162
`get_size()` (`pyffi.formats.nif.NifFormat.string` `method)`, 189
`get_size()` (`pyffi.formats.tga.TgaFormat.FooterString` `method)`, 196
`get_size()` (`pyffi.formats.tri.TriFormat.FileSignature` `method)`, 199
`get_size()` (`pyffi.formats.tri.TriFormat.FileVersion` `method)`, 199
`get_size()` (`pyffi.formats.tri.TriFormat.SizedStringZ` `method)`, 201
`get_skin_deformation()` (`pyffi.formats.nif.NifFormat.NiGeometry` `method)`, 103
`get_skin_partition()` (`pyffi.formats.nif.NifFormat.NiGeometry` `method)`, 103
`get_skinned_geometries()` (`pyffi.formats.nif.NifFormat.NiNode` `method)`, 112
`get_string()` (`pyffi.formats.nif.NifFormat.StringPalette` `method)`, 172
`get_strings()` (`pyffi.formats.nif.NifFormat.string` `method)`, 189
`get_strips()` (`pyffi.formats.nif.NifFormat.NiTriShapeData` `method)`, 152
`get_strips()` (`pyffi.formats.nif.NifFormat.NiTriStripsData` `method)`, 152
`get_sub_record()` (`pyffi.formats.esp.EspFormat.Record` `method)`, 36
`get_sub_shapes()` (`pyffi.formats.nif.NifFormat.bhkPackedNiTriStripsData` `method)`, 184
`get_tangent_space()` (`pyffi.formats.nif.NifFormat.NiTriBasedGeom` `method)`, 150
`get_target_color()` (`pyffi.formats.nif.NifFormat.NiMaterialColorController` `method)`, 107
`get_times()` (`pyffi.formats.nif.NifFormat.NiBSplineInterpolator` `method)`, 93
`get_toast_head_root_ext()` (`pyffi.spells.Toaster` `method)`, 231
`get_toast_stream()` (`pyffi.spells.Toaster` `method)`, 232
`get_transform()` (`pyffi.formats.nif.NifFormat.NiAVObject` `method)`, 89
`get_transform()` (`pyffi.formats.nif.NifFormat.NiSkinData` `method)`, 144
`get_transform()` (`pyffi.formats.nif.NifFormat.SkinData` `method)`, 163
`get_transform()` (`pyffi.formats.nif.NifFormat.SkinTransform` `method)`, 163
`get_transform_a_b()` (`pyffi.formats.nif.NifFormat.bhkConstraint` `method)`, 181
`get_translation()` (`pyffi.formats.cgf.CgfFormat.Matrix44` `method)`, 16
`get_translation()` (`pyffi.formats.nif.NifFormat.Matrix44` `method)`, 84
`get_translations()` (`pyffi.formats.nif.NifFormat.NiBSplineCompTransformInterpolator` `method)`, 92
`get_translations()` (`pyffi.formats.nif.NifFormat.NiBSplineTransformInterpolator` `method)`, 94
`get_transpose()` (`pyffi.formats.cgf.CgfFormat.Matrix33` `method)`, 15
`get_transpose()` (`pyffi.formats.nif.NifFormat.Matrix33` `method)`, 84
`get_triangle_hash_generator()` (`pyffi.formats.nif.NifFormat.bhkPackedNiTriStripsShape` `method)`, 184
`get_triangle_indices()` (`pyffi.formats.nif.NifFormat.NiTriBasedGeomData` `method)`, 151
`get_triangles()` (`pyffi.formats.nif.NifFormat.NiTriShapeData` `method)`, 152
`get_triangles()` (`pyffi.formats.nif.NifFormat.NiTriStripsData` `method)`, 152
`get_triangles()` (`pyffi.formats.nif.NifFormat.SkinPartition` `method)`, 163
`get_value()` (`pyffi.formats.bsa.BsaFormat.ZString` `method)`, 40
`get_value()` (`pyffi.formats.cgf.CgfFormat.FileSignature` `method)`, 14
`get_value()` (`pyffi.formats.cgf.CgfFormat.Ref` `method)`, 17
`get_value()` (`pyffi.formats.cgf.CgfFormat.SizedString` `method)`, 18
`get_value()` (`pyffi.formats.egm.EgmFormat.FileVersion` `method)`, 29
`get_value()` (`pyffi.formats.egt.EgtFormat.FileVersion` `method)`, 32
`get_value()` (`pyffi.formats.esp.EspFormat.ZString` `method)`, 37
`get_value()` (`pyffi.formats.kfm.KfmFormat.HeaderString` `method)`, 39
`get_value()` (`pyffi.formats.kfm.KfmFormat.SizedString` `method)`, 40
`get_value()` (`pyffi.formats.nif.NifFormat.bool` `method)`, 188
`get_value()` (`pyffi.formats.nif.NifFormat.ByteArray` `method)`, 64
`get_value()` (`pyffi.formats.nif.NifFormat.ByteMatrix` `method)`, 64
`get_value()` (`pyffi.formats.nif.NifFormat.LineString` `method)`, 181

method), 82
 get_value() (pyffi.formats.nif.NifFormat.Ptr method), 159
 get_value() (pyffi.formats.nif.NifFormat.Ref method), 161
 get_value() (pyffi.formats.nif.NifFormat.ShortString method), 162
 get_value() (pyffi.formats.nif.NifFormat.SizedString method), 162
 get_value() (pyffi.formats.tga.TgaFormat.FooterString method), 196
 get_value() (pyffi.formats.tri.TriFormat.FileVersion method), 199
 get_variable_1() (pyffi.formats.nif.NifFormat.ControllerLink method), 69
 get_variable_2() (pyffi.formats.nif.NifFormat.ControllerLink method), 69
 get_vector_3() (pyffi.formats.nif.NifFormat.Vector4 method), 176
 get_vertex_hash_generator() (pyffi.formats.nif.NifFormat.bhkPackedNiTriStripsShape method), 184
 get_vertex_hash_generator() (pyffi.formats.nif.NifFormat.NiGeometryData method), 104
 get_vertex_weights() (pyffi.formats.nif.NifFormat.NiGeometry method), 103
 getVersion() (pyffi.formats.dae.DaeFormat.Data method), 22
 GLOSS_MAP (pyffi.formats.nif.NifFormat.Texture attribute), 175
 gloss_texture() (pyffi.formats.nif.NifFormat.NiTexturingProperty property), 149
 Glossiness (pyffi.formats.nif.NifFormat.LightingShaderControlledVariable attribute), 82
 glossiness() (pyffi.formats.nif.NifFormat.BSLightingShaderProperty property), 51
 GLOW_MAP (pyffi.formats.nif.NifFormat.Texture attribute), 175
 glow_texture() (pyffi.formats.nif.NifFormat.NiTexturingProperty property), 149
 gpu_read() (pyffi.formats.nif.NifFormat.DataStreamAccess property), 71
 gpu_write() (pyffi.formats.nif.NifFormat.DataStreamAccess property), 71
 gravity_axis() (pyffi.formats.nif.NifFormat.NiPSysGravityModifier property), 128
 gravity_object() (pyffi.formats.nif.NifFormat.NiPSysGravityModifier property), 128
 greyscale_texture() (pyffi.formats.nif.NifFormat.BSEffectShaderProperty property), 49
 GROUND (pyffi.formats.nif.NifFormat.Fallout3Layer attribute), 75
 GROUND (pyffi.formats.nif.NifFormat.OblivionLayer attribute), 155
 GROUND (pyffi.formats.nif.NifFormat.SkyrimLayer attribute), 166
 grow() (pyffi.formats.nif.NifFormat.NiParticleGrowFade property), 132
 grow_generation() (pyffi.formats.nif.NifFormat.NiPSSimulatorGeneralStep property), 122
 grow_generation() (pyffi.formats.nif.NifFormat.NiPSysGrowFadeModifier property), 128
 hair_tint_color() (pyffi.formats.nif.NifFormat.NiPSSimulatorGeneralStep property), 122
 hair_tint_color() (pyffi.formats.nif.NifFormat.NiPSysGrowFadeModifier property), 128
H
 hair_tint_color() (pyffi.formats.nif.NifFormat.BSLightingShaderProperty property), 51
 half_space() (pyffi.formats.nif.NifFormat.BoundingBox property), 63
 HALFSpace_BV (pyffi.formats.nif.NifFormat.BoundingBoxType attribute), 63
 has_base_texture() (pyffi.formats.nif.NifFormat.NiTexturingProperty property), 149
 has_block_type() (pyffi.formats.nif.NifFormat.Header method), 79
 has_bump_map_texture() (pyffi.formats.nif.NifFormat.NiTexturingProperty property), 149
 has_block_type() (pyffi.formats.nif.NifFormat.NiTexturingProperty property), 149
 has_data() (pyffi.formats.nif.NifFormat.AdditionalDataBlock property), 44
 has_data() (pyffi.formats.nif.NifFormat.BSPackedAdditionalDataBlock property), 55
 has_data() (pyffi.formats.nif.NifFormat.Ni3dsAnimationNode property), 87
 has_decals_0_texture() (pyffi.formats.nif.NifFormat.NiTexturingProperty property), 149
 has_decals_1_texture() (pyffi.formats.nif.NifFormat.NiTexturingProperty property), 149
 has_decals_2_texture() (pyffi.formats.nif.NifFormat.NiTexturingProperty property), 149
 has_decals_3_texture() (pyffi.formats.nif.NifFormat.NiTexturingProperty property), 149

`property)`, 149
`has_detail_texture()`
 (`pyffi.formats.nif.NifFormat.NiTexturingProperty`
 `property)`, 149
`has_gloss_texture()`
 (`pyffi.formats.nif.NifFormat.NiTexturingProperty`
 `property)`, 149
`has_glow_texture()`
 (`pyffi.formats.nif.NifFormat.NiTexturingProperty`
 `property)`, 149
`has_image()` (`pyffi.formats.nif.NifFormat.MultiTextureElement`
 `property)`, 87
`has_normal_texture()`
 (`pyffi.formats.nif.NifFormat.NiTexturingProperty`
 `property)`, 149
`has_radii()` (`pyffi.formats.nif.NifFormat.NiParticlesData`
 `property)`, 134
`has_rotation_angles()`
 (`pyffi.formats.nif.NifFormat.NiParticlesData`
 `property)`, 134
`has_rotation_axes()`
 (`pyffi.formats.nif.NifFormat.NiParticlesData`
 `property)`, 134
`has_rotations()` (`pyffi.formats.nif.NifFormat.NiParticlesData`
 `property)`, 135
`has_rotations_2()`
 (`pyffi.formats.nif.NifFormat.NiRotatingParticlesData`
 `property)`, 141
`has_sizes()` (`pyffi.formats.nif.NifFormat.NiParticlesData`
 `property)`, 135
`has_subtexture_offset_u_vs()`
 (`pyffi.formats.nif.NifFormat.NiPSysData`
 `property)`, 125
`has_texture_transform()`
 (`pyffi.formats.nif.NifFormat.TexDesc` `prop-`
 `erty)`, 174
`has_unknown_2_texture()`
 (`pyffi.formats.nif.NifFormat.NiTexturingProperty`
 `property)`, 149
`has_unknown_floats_3()`
 (`pyffi.formats.nif.NifFormat.NiPSysData`
 `property)`, 125
`has_uv_quadrants()`
 (`pyffi.formats.nif.NifFormat.NiParticlesData`
 `property)`, 135
`HAV_MAT_CHAIN` (`pyffi.formats.nif.NifFormat.OblivionHavokMaterial`
 `attribute)`, 154
`HAV_MAT_CHAIN_STAIRS`
 (`pyffi.formats.nif.NifFormat.OblivionHavokMaterial`
 `attribute)`, 154
`HAV_MAT_CLOTH` (`pyffi.formats.nif.NifFormat.OblivionHavokMaterial`
 `attribute)`, 154
`HAV_MAT_CLOTH_STAIRS`
 (`pyffi.formats.nif.NifFormat.OblivionHavokMaterial`
 `attribute)`, 154
 `attribute)`, 154
`HAV_MAT_DIRT` (`pyffi.formats.nif.NifFormat.OblivionHavokMaterial`
 `attribute)`, 154
`HAV_MAT_DIRT_STAIRS`
 (`pyffi.formats.nif.NifFormat.OblivionHavokMaterial`
 `attribute)`, 154
`HAV_MAT_ELEVATOR` (`pyffi.formats.nif.NifFormat.OblivionHavokMaterial`
 `attribute)`, 154
`HAV_MAT_GLASS` (`pyffi.formats.nif.NifFormat.OblivionHavokMaterial`
 `attribute)`, 154
`HAV_MAT_GLASS_STAIRS`
 (`pyffi.formats.nif.NifFormat.OblivionHavokMaterial`
 `attribute)`, 154
`HAV_MAT_GRASS` (`pyffi.formats.nif.NifFormat.OblivionHavokMaterial`
 `attribute)`, 154
`HAV_MAT_GRASS_STAIRS`
 (`pyffi.formats.nif.NifFormat.OblivionHavokMaterial`
 `attribute)`, 154
`HAV_MAT_HEAVY_METAL`
 (`pyffi.formats.nif.NifFormat.OblivionHavokMaterial`
 `attribute)`, 154
`HAV_MAT_HEAVY_METAL_STAIRS`
 (`pyffi.formats.nif.NifFormat.OblivionHavokMaterial`
 `attribute)`, 154
`HAV_MAT_HEAVY_STONE`
 (`pyffi.formats.nif.NifFormat.OblivionHavokMaterial`
 `attribute)`, 154
`HAV_MAT_HEAVY_STONE_STAIRS`
 (`pyffi.formats.nif.NifFormat.OblivionHavokMaterial`
 `attribute)`, 154
`HAV_MAT_HEAVY_WOOD`
 (`pyffi.formats.nif.NifFormat.OblivionHavokMaterial`
 `attribute)`, 154
`HAV_MAT_HEAVY_WOOD_STAIRS`
 (`pyffi.formats.nif.NifFormat.OblivionHavokMaterial`
 `attribute)`, 154
`HAV_MAT_METAL` (`pyffi.formats.nif.NifFormat.OblivionHavokMaterial`
 `attribute)`, 154
`HAV_MAT_METAL_STAIRS`
 (`pyffi.formats.nif.NifFormat.OblivionHavokMaterial`
 `attribute)`, 154
`HAV_MAT_ORGANIC` (`pyffi.formats.nif.NifFormat.OblivionHavokMaterial`
 `attribute)`, 154
`HAV_MAT_ORGANIC_STAIRS`
 (`pyffi.formats.nif.NifFormat.OblivionHavokMaterial`
 `attribute)`, 155
`HAV_MAT_RUBBER` (`pyffi.formats.nif.NifFormat.OblivionHavokMaterial`
 `attribute)`, 155
`HAV_MAT_SKIN` (`pyffi.formats.nif.NifFormat.OblivionHavokMaterial`
 `attribute)`, 155
`HAV_MAT_SKIN_STAIRS`
 (`pyffi.formats.nif.NifFormat.OblivionHavokMaterial`
 `attribute)`, 155
`HAV_MAT_SNOW` (`pyffi.formats.nif.NifFormat.OblivionHavokMaterial`

attribute), 155
 HAV_MAT_SNOW_STAIRS (*pyffi.formats.nif.NifFormat.OblivionHavokMaterial attribute*), 155
 HAV_MAT_STONE (*pyffi.formats.nif.NifFormat.OblivionHavokMaterial attribute*), 155
 HAV_MAT_STONE_STAIRS (*pyffi.formats.nif.NifFormat.OblivionHavokMaterial attribute*), 155
 HAV_MAT_WATER (*pyffi.formats.nif.NifFormat.OblivionHavokMaterial attribute*), 155
 HAV_MAT_WATER_STAIRS (*pyffi.formats.nif.NifFormat.OblivionHavokMaterial attribute*), 155
 HAV_MAT_WOOD (*pyffi.formats.nif.NifFormat.OblivionHavokMaterial attribute*), 155
 HAV_MAT_WOOD_STAIRS (*pyffi.formats.nif.NifFormat.OblivionHavokMaterial attribute*), 155
 havok_col_filter (*pyffi.formats.nif.NifFormat.bhkWorldObject property*), 188
 havok_col_filter (*pyffi.formats.nif.NifFormat.OblivionSubShape property*), 157
 HEAD (*pyffi.formats.nif.NifFormat.Fallout3Layer attribute*), 75
 HEAD (*pyffi.formats.nif.NifFormat.OblivionLayer attribute*), 155
 heading () (*pyffi.formats.nif.NifFormat.FurniturePosition property*), 78
 height () (*pyffi.formats.nif.NifFormat.MipMap property*), 85
 height () (*pyffi.formats.nif.NifFormat.NiPSysBoxEmitter property*), 124
 height () (*pyffi.formats.nif.NifFormat.NiPSysCylinderEmitter property*), 125
 height () (*pyffi.formats.nif.NifFormat.NiPSysPlanarCollider property*), 130
 height () (*pyffi.formats.nif.NifFormat.NiRawImageData property*), 140
 Heightmap (*pyffi.formats.nif.NifFormat.BSLightingShaderProperty ShaderType attribute*), 52
 Hinge (*pyffi.formats.nif.NifFormat.hkConstraintType attribute*), 188
 hinge () (*pyffi.formats.nif.NifFormat.bhkHingeConstraint property*), 181
 hinge () (*pyffi.formats.nif.NifFormat.bhkRDTMalleableConstraint property*), 185
 hinge () (*pyffi.formats.nif.NifFormat.SubConstraint property*), 172
 horizontal_angle () (*pyffi.formats.nif.NifFormat.NiParticleSystemController property*), 133
 horizontal_direction () (*pyffi.formats.nif.NifFormat.NiParticleSystemController property*), 133
 image () (*pyffi.formats.nif.NifFormat.MultiTextureElement property*), 87
 image () (*pyffi.formats.nif.NifFormat.NiTextureEffect property*), 147
 image () (*pyffi.formats.nif.NifFormat.NiTextureProperty property*), 148
 image_data () (*pyffi.formats.nif.NifFormat.NiImage property*), 105
 image_type () (*pyffi.formats.nif.NifFormat.NiRawImageData property*), 140
 images () (*pyffi.formats.nif.NifFormat.NiFlipController property*), 100
 importer_name () (*pyffi.formats.nif.NifFormat.NiArkImporterExtraData property*), 90
 in_portals () (*pyffi.formats.nif.NifFormat.NiRoom property*), 141
 include_types (*pyffi.spells.Toaster attribute*), 232
 indent (*pyffi.spells.Toaster attribute*), 232
 index () (*pyffi.formats.nif.NifFormat.SemanticData property*), 161
 index () (*pyffi.formats.nif.NifFormat.SkinWeight property*), 163
 indices () (*pyffi.formats.nif.NifFormat.bhkCMSDChunk property*), 179
 indices_2 () (*pyffi.formats.nif.NifFormat.bhkCMSDChunk property*), 179
 initial_axis () (*pyffi.formats.nif.NifFormat.NiParticleRotation property*), 133
 initial_axis () (*pyffi.formats.nif.NifFormat.NiPSysRotationModifier property*), 130
 initial_color () (*pyffi.formats.nif.NifFormat.NiPSysEmitter property*), 126
 initial_radius () (*pyffi.formats.nif.NifFormat.NiPSysEmitter property*), 126
 initial_rotation_angle () (*pyffi.formats.nif.NifFormat.NiPSysRotationModifier property*), 130
 initial_rotation_angle_variation () (*pyffi.formats.nif.NifFormat.NiPSysRotationModifier property*), 130
 initial_rotation_speed () (*pyffi.formats.nif.NifFormat.NiPSysRotationModifier property*), 130
 initial_rotation_speed_variation () (*pyffi.formats.nif.NifFormat.NiPSysRotationModifier property*), 130
 initial_velocity_type () (*pyffi.formats.nif.NifFormat.NiPSysMeshEmitter property*), 129

```

inspect () (pyffi.formats.bsa.BsaFormat.Header
            method), 9
inspect () (pyffi.formats.cgf.CgfFormat.Data method),
            13
inspect () (pyffi.formats.dae.DaeFormat.Data
            method), 22
inspect () (pyffi.formats.dds.DdsFormat.Data
            method), 24
inspect () (pyffi.formats.egm.EgmFormat.Data
            method), 28
inspect () (pyffi.formats.egt.EgtFormat.Header
            method), 33
inspect () (pyffi.formats.esp.EspFormat.Data
            method), 35
inspect () (pyffi.formats.kfm.KfmFormat.Data
            method), 39
inspect () (pyffi.formats.nif.NifFormat.Data method),
            70
inspect () (pyffi.formats.tga.TgaFormat.Data
            method), 196
inspect () (pyffi.formats.tri.TriFormat.Header
            method), 200
inspect () (pyffi.object_models.FileFormat.Data
            method), 234
inspect_filename () (pyffi.spells.Toaster method),
                    232
inspect_quick () (pyffi.formats.bsa.BsaFormat.Header
                  method), 9
inspect_quick () (pyffi.formats.dds.DdsFormat.Data
                  method), 25
inspect_quick () (pyffi.formats.egm.EgmFormat.Data
                  method), 28
inspect_quick () (pyffi.formats.egt.EgtFormat.Header
                  method), 33
inspect_quick () (pyffi.formats.esp.EspFormat.Data
                  method), 35
inspect_quick () (pyffi.formats.tri.TriFormat.Header
                  method), 200
inspect_version_only ()
    (pyffi.formats.cgf.CgfFormat.Data method), 13
inspect_version_only ()
    (pyffi.formats.nif.NifFormat.Data method),
    71
instancings_enabled ()
    (pyffi.formats.nif.NifFormat.NiMesh prop-
    erty), 107
int (pyffi.formats.cgf.CgfFormat attribute), 19
int (pyffi.formats.dds.DdsFormat attribute), 25
int (pyffi.formats.egm.EgmFormat attribute), 30
int (pyffi.formats.egt.EgtFormat attribute), 33
int (pyffi.formats.esp.EspFormat attribute), 37
int (pyffi.formats.kfm.KfmFormat attribute), 41
int (pyffi.formats.nif.NifFormat attribute), 189
int (pyffi.formats.tga.TgaFormat attribute), 197
int (pyffi.formats.tri.TriFormat attribute), 201
integer_data () (pyffi.formats.nif.NifFormat.NiIntegerExtraData
                property), 105
interface, 267
internal_index () (pyffi.formats.nif.NifFormat.BSSegment
                  property), 57
interpolation () (pyffi.formats.nif.NifFormat.KeyGroup
                  property), 81
interpolation () (pyffi.formats.nif.NifFormat.Morph
                  property), 86
interpolator () (pyffi.formats.nif.NifFormat.BSFrustumFOVController
                 property), 50
interpolator () (pyffi.formats.nif.NifFormat.BSRefractionFirePeriodC
                 property), 56
interpolator () (pyffi.formats.nif.NifFormat.MorphWeight
                 property), 86
interpolator () (pyffi.formats.nif.NifFormat.NiPSEmitterRadiusCtrlr
                 property), 117
interpolator () (pyffi.formats.nif.NifFormat.NiPSEmitterSpeedCtrlr
                 property), 117
interpolator () (pyffi.formats.nif.NifFormat.NiPSForceActiveCtrlr
                 property), 118
interpolator () (pyffi.formats.nif.NifFormat.NiSingleInterpController
                 property), 143
interpolator_10 ()
    (pyffi.formats.nif.NifFormat.BSProceduralLightningController
    property), 56
interpolator_2_mutation ()
    (pyffi.formats.nif.NifFormat.BSProceduralLightningController
    property), 56
interpolator_3 () (pyffi.formats.nif.NifFormat.BSProceduralLightning
                  property), 56
interpolator_4 () (pyffi.formats.nif.NifFormat.BSProceduralLightning
                  property), 56
interpolator_5 () (pyffi.formats.nif.NifFormat.BSProceduralLightning
                  property), 56
interpolator_6 () (pyffi.formats.nif.NifFormat.BSProceduralLightning
                  property), 56
interpolator_7 () (pyffi.formats.nif.NifFormat.BSProceduralLightning
                  property), 56
interpolator_8 () (pyffi.formats.nif.NifFormat.BSProceduralLightning
                  property), 56
interpolator_9_arc_offset ()
    (pyffi.formats.nif.NifFormat.BSProceduralLightningController
    property), 56
interpolator_weights ()
    (pyffi.formats.nif.NifFormat.NiGeomMorpherController
    property), 102
interpolators () (pyffi.formats.nif.NifFormat.NiGeomMorpherControl
                  property), 102
interpolators () (pyffi.formats.nif.NifFormat.NiMorphWeightsControl
                  property), 109
INVISIBLE_WALL (pyffi.formats.nif.NifFormat.Fallout3Layer
                 attribute), 75

```


INVISIBLE_WALL (*pyffi.formats.nif.NifFormat.SkyrimLayer* *property*), 145
attribute), 166
 is_admissible_branch_class() (*pyffi.spells.Toaster* *method*), 232
 is_branch_to_be_deleted() (*pyffi.spells.nif.modify._SpellDelBranchClasses* *method*), 224
 is_branch_to_be_deleted() (*pyffi.spells.nif.modify.SpellDelBranches* *method*), 223
 is_branch_to_be_deleted() (*pyffi.spells.nif.modify.SpellDelSkinShapes* *method*), 224
 is_branch_to_be_deleted() (*pyffi.spells.nif.modify.SpellDelVertexColor* *method*), 226
 is_identity() (*pyffi.formats.cgf.CgfFormat.Matrix33* *method*), 15
 is_identity() (*pyffi.formats.cgf.CgfFormat.Matrix44* *method*), 16
 is_identity() (*pyffi.formats.nif.NifFormat.InertiaMatrix* *method*), 81
 is_identity() (*pyffi.formats.nif.NifFormat.Matrix33* *method*), 84
 is_identity() (*pyffi.formats.nif.NifFormat.Matrix44* *method*), 84
 is_interchangeable() (*pyffi.formats.nif.NifFormat.NiMaterialProperty* *method*), 107
 is_interchangeable() (*pyffi.formats.nif.NifFormat.NiObject* *method*), 113
 is_interchangeable() (*pyffi.formats.nif.NifFormat.NiTriBasedGeomData* *method*), 151
 is_per_instance() (*pyffi.formats.nif.NifFormat.MeshData* *property*), 85
 is_rotation() (*pyffi.formats.cgf.CgfFormat.Matrix33* *method*), 15
 is_rotation() (*pyffi.formats.nif.NifFormat.Matrix33* *method*), 84
 is_scale_rotation() (*pyffi.formats.cgf.CgfFormat.Matrix33* *method*), 15
 is_scale_rotation() (*pyffi.formats.nif.NifFormat.Matrix33* *method*), 84
 is_scale_rotation_translation() (*pyffi.formats.nif.NifFormat.Matrix44* *method*), 84
 is_skin() (*pyffi.formats.nif.NifFormat.NiGeometry* *method*), 103
 is_static() (*pyffi.formats.nif.NifFormat.NiSourceTexture* *property*), 145
 is_static_bound() (*pyffi.formats.nif.NifFormat.BSOrderedNode* *property*), 53
 is_used() (*pyffi.formats.nif.NifFormat.ShaderTexDesc* *property*), 161
 ITEM_PICK (*pyffi.formats.nif.NifFormat.OblivionLayer* *attribute*), 155
 ITEM_PICK (*pyffi.formats.nif.NifFormat.Fallout3Layer* *attribute*), 76
 ITEM_PICK (*pyffi.formats.nif.NifFormat.SkyrimLayer* *attribute*), 166
 items() (*pyffi.formats.nif.NifFormat.BSArray* *property*), 61
 items() (*pyffi.formats.nif.NifFormat.NiRoom* *property*), 141
J
 joint_descs() (*pyffi.formats.nif.NifFormat.NiPhysXPropDesc* *property*), 138
K
 keys() (*pyffi.formats.nif.NifFormat.KeyGroup* *property*), 81
 keys() (*pyffi.formats.nif.NifFormat.Morph* *property*), 86
 keys() (*pyffi.formats.nif.NifFormat.NiVisData* *property*), 153
 KfmFormat (*class in pyffi.formats.kfm*), 39
 KfmFormat.Data (*class in pyffi.formats.kfm*), 39
 KfmFormat.FilePath (*class in pyffi.formats.kfm*), 39
 KfmFormat.HeaderString (*class in pyffi.formats.kfm*), 39
 KfmFormat.SizedString (*class in pyffi.formats.kfm*), 40
 KFMXMLPATH, 8
L
 L_CALF (*pyffi.formats.nif.NifFormat.Fallout3Layer* *attribute*), 76
 L_CALF (*pyffi.formats.nif.NifFormat.OblivionLayer* *attribute*), 155
 L_FOOT (*pyffi.formats.nif.NifFormat.Fallout3Layer* *attribute*), 76
 L_FOOT (*pyffi.formats.nif.NifFormat.OblivionLayer* *attribute*), 155
 L_FORE_ARM (*pyffi.formats.nif.NifFormat.Fallout3Layer* *attribute*), 76
 L_FOREARM (*pyffi.formats.nif.NifFormat.OblivionLayer* *attribute*), 155
 L_HAND (*pyffi.formats.nif.NifFormat.Fallout3Layer* *attribute*), 76
 L_HAND (*pyffi.formats.nif.NifFormat.OblivionLayer* *attribute*), 155

L_THIGH (*pyffi.formats.nif.NifFormat.Fallout3Layer attribute*), 76
 L_THIGH (*pyffi.formats.nif.NifFormat.OblivionLayer attribute*), 156
 L_UPPER_ARM (*pyffi.formats.nif.NifFormat.Fallout3Layer attribute*), 76
 L_UPPER_ARM (*pyffi.formats.nif.NifFormat.OblivionLayer attribute*), 156
 layer () (*pyffi.formats.nif.NifFormat.bhkCMSDMaterial property*), 179
 layer () (*pyffi.formats.nif.NifFormat.HavokColFilter property*), 79
 Lean (*pyffi.formats.nif.NifFormat.AnimationType attribute*), 44
 left () (*pyffi.formats.nif.NifFormat.FurnitureEntryPoints property*), 78
 left_eye_reflection_center ()
 (*pyffi.formats.nif.NifFormat.BSLightingShaderProperty property*), 51
 length () (*pyffi.formats.nif.NifFormat.StiffSpringDescriptor property*), 170
 level_0_size () (*pyffi.formats.nif.NifFormat.BSLODTriShape property*), 50
 level_1_size () (*pyffi.formats.nif.NifFormat.BSLODTriShape property*), 50
 level_2_size () (*pyffi.formats.nif.NifFormat.BSLODTriShape property*), 50
 life_span () (*pyffi.formats.nif.NifFormat.NiPSysEmitter property*), 126
 life_span () (*pyffi.formats.nif.NifFormat.NiPSysSpawnModifier property*), 130
 life_span_variation ()
 (*pyffi.formats.nif.NifFormat.NiPSysEmitter property*), 126
 life_span_variation ()
 (*pyffi.formats.nif.NifFormat.NiPSysSpawnModifier property*), 130
 lifespan () (*pyffi.formats.nif.NifFormat.Particle property*), 157
 lifetime () (*pyffi.formats.nif.NifFormat.NiParticleSystemController property*), 133
 lifetime () (*pyffi.formats.nif.NifFormat.Particle property*), 157
 lifetime_random ()
 (*pyffi.formats.nif.NifFormat.NiParticleSystemController property*), 133
 LIGHT_MODE_EMI_AMB_DIF
 (*pyffi.formats.nif.NifFormat.LightMode attribute*), 81
 LIGHT_MODE_EMISSIVE
 (*pyffi.formats.nif.NifFormat.LightMode attribute*), 81
 lighting_effect_1 ()
 (*pyffi.formats.nif.NifFormat.BSLightingShaderProperty property*), 51
 lighting_effect_2 ()
 (*pyffi.formats.nif.NifFormat.BSLightingShaderProperty property*), 51
 lighting_mode () (*pyffi.formats.nif.NifFormat.NiVertexColorProperty property*), 153
 limited_hinge () (*pyffi.formats.nif.NifFormat.bhkRDTMalleableConstraint property*), 185
 limited_hinge () (*pyffi.formats.nif.NifFormat.SubConstraint property*), 172
 LINE_OF_SIGHT (*pyffi.formats.nif.NifFormat.OblivionLayer attribute*), 155
 linear_attenuation ()
 (*pyffi.formats.nif.NifFormat.NiPointLight property*), 140
 LINEAR_KEY (*pyffi.formats.nif.NifFormat.KeyType attribute*), 81
 parent_near_rotation ()
 (*pyffi.formats.nif.NifFormat.BSLagBoneController property*), 50
 linear_velocity ()
 (*pyffi.formats.nif.NifFormat.BSLagBoneController property*), 50
 LINE_OF_SIGHT (*pyffi.formats.nif.NifFormat.Fallout3Layer attribute*), 76
 LINE_OF_SIGHT (*pyffi.formats.nif.NifFormat.SkyrimLayer attribute*), 166
 lines () (*pyffi.formats.nif.NifFormat.NiLinesData property*), 107
 link_pairs () (*pyffi.formats.nif.NifFormat.SkinShapeGroup property*), 163
 links () (*pyffi.formats.nif.NifFormat.bhkBallSocketConstraintChain property*), 177
 links_2 () (*pyffi.formats.nif.NifFormat.bhkBallSocketConstraintChain property*), 177
 lod_adjust () (*pyffi.formats.nif.NifFormat.NiCamera property*), 97
 lod_center () (*pyffi.formats.nif.NifFormat.NiLODNode property*), 106
 lod_center () (*pyffi.formats.nif.NifFormat.NiRangeLODData property*), 140
 lod_level_data () (*pyffi.formats.nif.NifFormat.NiLODNode property*), 106
 lod_levels () (*pyffi.formats.nif.NifFormat.NiLODNode property*), 106
 lod_levels () (*pyffi.formats.nif.NifFormat.NiRangeLODData property*), 140
 logger (*pyffi.spells.Toaster attribute*), 232
 look_at () (*pyffi.formats.nif.NifFormat.NiLookAtInterpolator property*), 107
 look_at_node () (*pyffi.formats.nif.NifFormat.NiLookAtController property*), 107
 loop_start_frame ()
 (*pyffi.formats.nif.NifFormat.BSPSysSubTexModifier*

property), 55

loop_start_frame_fudge() (pyffi.formats.nif.NifFormat.BSPSysSubTexModifier property), 55

M

m_11() (pyffi.formats.nif.NifFormat.Matrix22 property), 83

m_12() (pyffi.formats.nif.NifFormat.Matrix22 property), 83

m_21() (pyffi.formats.nif.NifFormat.Matrix22 property), 83

m_22() (pyffi.formats.nif.NifFormat.Matrix22 property), 83

magnitude() (pyffi.formats.nif.NifFormat.NiPSysFieldModifier property), 127

malleable_constraint() (pyffi.formats.nif.NifFormat.bhkRDTConstraint property), 185

map_index() (pyffi.formats.nif.NifFormat.ShaderTexDesc property), 162

mask_index() (pyffi.formats.nif.NifFormat.bhkCompressedMeshShapeData property), 180

mask_w_index() (pyffi.formats.nif.NifFormat.bhkCompressedMeshShapeData property), 180

mass() (pyffi.formats.nif.NifFormat.bhkRagdollTemplateData property), 186

MAT_BABY_RATTLE (pyffi.formats.nif.NifFormat.Fallout3HavokMaterial attribute), 74

MAT_BARREL (pyffi.formats.nif.NifFormat.Fallout3HavokMaterial attribute), 74

MAT_BARREL (pyffi.formats.nif.NifFormat.SkyrimHavokMaterial attribute), 164

MAT_BOTTLE (pyffi.formats.nif.NifFormat.Fallout3HavokMaterial attribute), 74

MAT_BOTTLE (pyffi.formats.nif.NifFormat.SkyrimHavokMaterial attribute), 164

MAT_BOTTLECAP (pyffi.formats.nif.NifFormat.Fallout3HavokMaterial attribute), 74

MAT_BROKEN_CONCRETE (pyffi.formats.nif.NifFormat.Fallout3HavokMaterial attribute), 74

MAT_BROKEN_STONE (pyffi.formats.nif.NifFormat.SkyrimHavokMaterial attribute), 164

MAT_CHAIN (pyffi.formats.nif.NifFormat.Fallout3HavokMaterial attribute), 74

MAT_CLOTH (pyffi.formats.nif.NifFormat.Fallout3HavokMaterial attribute), 74

MAT_CLOTH (pyffi.formats.nif.NifFormat.SkyrimHavokMaterial attribute), 164

MAT_DIRT (pyffi.formats.nif.NifFormat.Fallout3HavokMaterial attribute), 74

MAT_DIRT (pyffi.formats.nif.NifFormat.SkyrimHavokMaterial attribute), 164

MAT_DRAGON (pyffi.formats.nif.NifFormat.SkyrimHavokMaterial attribute), 164

MAT_ELEVATOR (pyffi.formats.nif.NifFormat.Fallout3HavokMaterial attribute), 74

MAT_GLASS (pyffi.formats.nif.NifFormat.Fallout3HavokMaterial attribute), 74

MAT_GLASS (pyffi.formats.nif.NifFormat.SkyrimHavokMaterial attribute), 164

MAT_GRASS (pyffi.formats.nif.NifFormat.Fallout3HavokMaterial attribute), 74

MAT_GRASS (pyffi.formats.nif.NifFormat.SkyrimHavokMaterial attribute), 164

MAT_GRAVEL (pyffi.formats.nif.NifFormat.SkyrimHavokMaterial attribute), 164

MAT_HEAVY_METAL (pyffi.formats.nif.NifFormat.Fallout3HavokMaterial attribute), 74

MAT_HEAVY_METAL (pyffi.formats.nif.NifFormat.SkyrimHavokMaterial attribute), 164

MAT_HEAVY_STONE (pyffi.formats.nif.NifFormat.Fallout3HavokMaterial attribute), 74

MAT_HEAVY_STONE (pyffi.formats.nif.NifFormat.SkyrimHavokMaterial attribute), 164

MAT_HEAVY_WOOD (pyffi.formats.nif.NifFormat.Fallout3HavokMaterial attribute), 74

MAT_HEAVY_WOOD (pyffi.formats.nif.NifFormat.SkyrimHavokMaterial attribute), 164

MAT_HOLLOW_METAL (pyffi.formats.nif.NifFormat.Fallout3HavokMaterial attribute), 74

MAT_ICE (pyffi.formats.nif.NifFormat.SkyrimHavokMaterial attribute), 164

MAT_LIGHT_WOOD (pyffi.formats.nif.NifFormat.SkyrimHavokMaterial attribute), 164

MAT_LUNCHBOX (pyffi.formats.nif.NifFormat.Fallout3HavokMaterial attribute), 74

MAT_MATERIAL_ARMOR_HEAVY (pyffi.formats.nif.NifFormat.SkyrimHavokMaterial attribute), 164

MAT_MATERIAL_ARMOR_LIGHT (pyffi.formats.nif.NifFormat.SkyrimHavokMaterial attribute), 164

MAT_MATERIAL_ARROW (pyffi.formats.nif.NifFormat.SkyrimHavokMaterial attribute), 164

MAT_MATERIAL_AXE_1HAND (pyffi.formats.nif.NifFormat.SkyrimHavokMaterial attribute), 164

MAT_MATERIAL_BASKET (pyffi.formats.nif.NifFormat.SkyrimHavokMaterial attribute), 164

MAT_MATERIAL_BLADE_1HAND (pyffi.formats.nif.NifFormat.SkyrimHavokMaterial attribute), 165

MAT_MATERIAL_BLADE_1HAND_SMALL (pyffi.formats.nif.NifFormat.SkyrimHavokMaterial attribute), 165

<i>attribute</i>), 165	<i>attribute</i>), 165
MAT_MATERIAL_BLADE_2HAND (<i>pyffi.formats.nif.NifFormat.SkyrimHavokMaterial</i> <i>attribute</i>), 165	MAT_MATERIAL_STONE_AS_STAIRS (<i>pyffi.formats.nif.NifFormat.SkyrimHavokMaterial</i> <i>attribute</i>), 165
MAT_MATERIAL_BLUNT_2HAND (<i>pyffi.formats.nif.NifFormat.SkyrimHavokMaterial</i> <i>attribute</i>), 165	MAT_MATERIAL_WOOD_AS_STAIRS (<i>pyffi.formats.nif.NifFormat.SkyrimHavokMaterial</i> <i>attribute</i>), 165
MAT_MATERIAL_BONE (<i>pyffi.formats.nif.NifFormat.SkyrimHavokMaterial</i> <i>attribute</i>), 165	MAT_METAL (<i>pyffi.formats.nif.NifFormat.Fallout3HavokMaterial</i> <i>attribute</i>), 74
MAT_MATERIAL_BOOK (<i>pyffi.formats.nif.NifFormat.SkyrimHavokMaterial</i> <i>attribute</i>), 165	MAT_MUD (<i>pyffi.formats.nif.NifFormat.SkyrimHavokMaterial</i> <i>attribute</i>), 165
MAT_MATERIAL_BOTTLE_SMALL (<i>pyffi.formats.nif.NifFormat.SkyrimHavokMaterial</i> <i>attribute</i>), 165	MAT_ORGANIC (<i>pyffi.formats.nif.NifFormat.Fallout3HavokMaterial</i> <i>attribute</i>), 74
MAT_MATERIAL_BOULDER_LARGE (<i>pyffi.formats.nif.NifFormat.SkyrimHavokMaterial</i> <i>attribute</i>), 165	MAT_ORGANIC (<i>pyffi.formats.nif.NifFormat.SkyrimHavokMaterial</i> <i>attribute</i>), 165
MAT_MATERIAL_BOULDER_MEDIUM (<i>pyffi.formats.nif.NifFormat.SkyrimHavokMaterial</i> <i>attribute</i>), 165	MAT_PISTOL (<i>pyffi.formats.nif.NifFormat.Fallout3HavokMaterial</i> <i>attribute</i>), 74
MAT_MATERIAL_BOULDER_SMALL (<i>pyffi.formats.nif.NifFormat.SkyrimHavokMaterial</i> <i>attribute</i>), 165	MAT_RIFLE (<i>pyffi.formats.nif.NifFormat.Fallout3HavokMaterial</i> <i>attribute</i>), 74
MAT_MATERIAL_BOWS_STAVES (<i>pyffi.formats.nif.NifFormat.SkyrimHavokMaterial</i> <i>attribute</i>), 165	MAT_RUBBER_BALL (<i>pyffi.formats.nif.NifFormat.Fallout3HavokMaterial</i> <i>attribute</i>), 74
MAT_MATERIAL_CARPET (<i>pyffi.formats.nif.NifFormat.SkyrimHavokMaterial</i> <i>attribute</i>), 165	MAT_SAND (<i>pyffi.formats.nif.NifFormat.Fallout3HavokMaterial</i> <i>attribute</i>), 74
MAT_MATERIAL_CERAMIC_MEDIUM (<i>pyffi.formats.nif.NifFormat.SkyrimHavokMaterial</i> <i>attribute</i>), 165	MAT_SAND (<i>pyffi.formats.nif.NifFormat.SkyrimHavokMaterial</i> <i>attribute</i>), 165
MAT_MATERIAL_CHAIN (<i>pyffi.formats.nif.NifFormat.SkyrimHavokMaterial</i> <i>attribute</i>), 165	MAT_SHEET_METAL (<i>pyffi.formats.nif.NifFormat.Fallout3HavokMaterial</i> <i>attribute</i>), 74
MAT_MATERIAL_CHAIN_METAL (<i>pyffi.formats.nif.NifFormat.SkyrimHavokMaterial</i> <i>attribute</i>), 165	MAT_SHOPPING_CART (<i>pyffi.formats.nif.NifFormat.Fallout3HavokMaterial</i> <i>attribute</i>), 74
MAT_MATERIAL_COIN (<i>pyffi.formats.nif.NifFormat.SkyrimHavokMaterial</i> <i>attribute</i>), 165	MAT_SKIN (<i>pyffi.formats.nif.NifFormat.Fallout3HavokMaterial</i> <i>attribute</i>), 74
MAT_MATERIAL_SHIELD_HEAVY (<i>pyffi.formats.nif.NifFormat.SkyrimHavokMaterial</i> <i>attribute</i>), 165	MAT_SKIN (<i>pyffi.formats.nif.NifFormat.SkyrimHavokMaterial</i> <i>attribute</i>), 165
MAT_MATERIAL_SHIELD_LIGHT (<i>pyffi.formats.nif.NifFormat.SkyrimHavokMaterial</i> <i>attribute</i>), 165	MAT_SNOW (<i>pyffi.formats.nif.NifFormat.SkyrimHavokMaterial</i> <i>attribute</i>), 165
MAT_MATERIAL_SKIN_LARGE (<i>pyffi.formats.nif.NifFormat.SkyrimHavokMaterial</i> <i>attribute</i>), 165	MAT_SODA_CAN (<i>pyffi.formats.nif.NifFormat.Fallout3HavokMaterial</i> <i>attribute</i>), 74
MAT_MATERIAL_SKIN_SMALL (<i>pyffi.formats.nif.NifFormat.SkyrimHavokMaterial</i> <i>attribute</i>), 165	MAT_SOLID_METAL (<i>pyffi.formats.nif.NifFormat.SkyrimHavokMaterial</i> <i>attribute</i>), 165
	MAT_STAIRS_BROKEN_STONE (<i>pyffi.formats.nif.NifFormat.SkyrimHavokMaterial</i> <i>attribute</i>), 165
	MAT_STAIRS_SNOW (<i>pyffi.formats.nif.NifFormat.SkyrimHavokMaterial</i> <i>attribute</i>), 165
	MAT_STAIRS_STONE (<i>pyffi.formats.nif.NifFormat.SkyrimHavokMaterial</i> <i>attribute</i>), 165
	MAT_STAIRS_WOOD (<i>pyffi.formats.nif.NifFormat.SkyrimHavokMaterial</i> <i>attribute</i>), 165
	MAT_STONE (<i>pyffi.formats.nif.NifFormat.Fallout3HavokMaterial</i> <i>attribute</i>), 75
	MAT_STONE (<i>pyffi.formats.nif.NifFormat.SkyrimHavokMaterial</i> <i>attribute</i>), 165
	MAT_UNKNOWN_1028101969

(pyffi.formats.nif.NifFormat.SkyrimHavokMaterial property), 94
 attribute), 165 max_distance() (pyffi.formats.nif.NifFormat.NiPSysFieldModifier
 MAT_UNKNOWN_1440721808 property), 127
 (pyffi.formats.nif.NifFormat.SkyrimHavokMaterial max_distance() (pyffi.formats.nif.NifFormat.PrismaticDescriptor
 attribute), 165 property), 158
 MAT_UNKNOWN_1574477864 max_emitter_objects()
 (pyffi.formats.nif.NifFormat.SkyrimHavokMaterial (pyffi.formats.nif.NifFormat.BSMasterParticleSystem
 attribute), 165 property), 52
 MAT_UNKNOWN_1591009235 max_num_to_spawn()
 (pyffi.formats.nif.NifFormat.SkyrimHavokMaterial (pyffi.formats.nif.NifFormat.NiPSysSpawnModifier
 attribute), 166 property), 130
 MAT_VEHICLE_BODY (pyffi.formats.nif.NifFormat.Fallout3HavokMaterial) (pyffi.formats.nif.NifFormat.BSLightingShaderProperty
 attribute), 75 property), 51
 MAT_VEHICLE_PART_HOLLOW max_polygons() (pyffi.formats.nif.NifFormat.NiScreenElementsData
 (pyffi.formats.nif.NifFormat.Fallout3HavokMaterial property), 142
 attribute), 75 maximum_distance()
 MAT_VEHICLE_PART_SOLID (pyffi.formats.nif.NifFormat.BSLagBoneController
 (pyffi.formats.nif.NifFormat.Fallout3HavokMaterial property), 50
 attribute), 75 merge_external_skeleton_root()
 MAT_WATER (pyffi.formats.nif.NifFormat.Fallout3HavokMaterial (pyffi.formats.nif.NifFormat.NiNode method),
 attribute), 75 112
 MAT_WATER (pyffi.formats.nif.NifFormat.SkyrimHavokMaterial merge_skeleton_roots()
 attribute), 166 (pyffi.formats.nif.NifFormat.NiNode method),
 MAT_WOOD (pyffi.formats.nif.NifFormat.Fallout3HavokMaterial 112
 attribute), 75 mesh_description()
 MAT_WOOD (pyffi.formats.nif.NifFormat.SkyrimHavokMaterial (pyffi.formats.nif.NifFormat.NiPhysXShapeDesc
 attribute), 166 property), 138
 material() (pyffi.formats.nif.NifFormat.bhkCMSDMaterial) MESH_PRIMITIVE_LINESTRIPS
 property), 179 (pyffi.formats.nif.NifFormat.MeshPrimitiveType
 material() (pyffi.formats.nif.NifFormat.bhkConvexListShape attribute), 85
 property), 181 MESH_PRIMITIVE_POINTS
 material() (pyffi.formats.nif.NifFormat.bhkSphereRepShape (pyffi.formats.nif.NifFormat.MeshPrimitiveType
 property), 187 attribute), 85
 material() (pyffi.formats.nif.NifFormat.HavokMaterial MESH_PRIMITIVE_QUADS
 property), 79 (pyffi.formats.nif.NifFormat.MeshPrimitiveType
 material() (pyffi.formats.nif.NifFormat.OblivionSubShape attribute), 85
 property), 157 MESH_PRIMITIVE_TRIANGLES
 material_data() (pyffi.formats.nif.NifFormat.NiRenderObject (pyffi.formats.nif.NifFormat.MeshPrimitiveType
 property), 141 attribute), 85
 material_desc() (pyffi.formats.nif.NifFormat.physXMaterialRep MESH_PRIMITIVE_TRISTRIPS
 property), 189 (pyffi.formats.nif.NifFormat.MeshPrimitiveType
 material_descs() (pyffi.formats.nif.NifFormat.NiPhysXPropDesc attribute), 85
 property), 138 meshes() (pyffi.formats.nif.NifFormat.NiPSysMeshUpdateModifier
 material_extra_data() property), 129
 (pyffi.formats.nif.NifFormat.MaterialData MetaFileFormat (class in pyffi.object_models), 233
 property), 83 min_distance() (pyffi.formats.nif.NifFormat.PrismaticDescriptor
 material_index() (pyffi.formats.nif.NifFormat.bhkCMSDChunk property), 158
 property), 179 min_num_to_spawn()
 material_name() (pyffi.formats.nif.NifFormat.MaterialData (pyffi.formats.nif.NifFormat.NiPSysSpawnModifier
 property), 83 property), 130
 material_needs_update_default() MIP_FMT_DEFAULT (pyffi.formats.nif.NifFormat.MipMapFormat
 (pyffi.formats.nif.NifFormat.NiRenderObject attribute), 85
 property), 141 MIP_FMT_NO (pyffi.formats.nif.NifFormat.MipMapFormat
 matrix() (pyffi.formats.nif.NifFormat.NiBezierTriangle4 attribute), 85

MIP_FMT_YES (*pyffi.formats.nif.NifFormat.MipMapFormat* *property*), 53
attribute), 85
 MO_QUAL_BULLET (*pyffi.formats.nif.NifFormat.MotionQuality* *property*), 129
attribute), 86
 MO_QUAL_CHARACTER (*pyffi.formats.nif.NifFormat.MotionQuality* *property*), 107
attribute), 86
 MO_QUAL_CRITICAL (*pyffi.formats.nif.NifFormat.MotionQuality* *property*), 133
attribute), 86
 MO_QUAL_DEBRIS (*pyffi.formats.nif.NifFormat.MotionQuality* *property*), 133
attribute), 86
 MO_QUAL_FIXED (*pyffi.formats.nif.NifFormat.MotionQuality* *property*), 133
attribute), 86
 MO_QUAL_INVALID (*pyffi.formats.nif.NifFormat.MotionQuality* *property*), 52
attribute), 86
 MO_QUAL_KEYFRAMED (*pyffi.formats.nif.NifFormat.MotionQuality* *property*), 91
attribute), 86
 MO_QUAL_KEYFRAMED_REPORT (*pyffi.formats.nif.NifFormat.MotionQuality* *property*), 43
attribute), 86
 MO_QUAL_MOVING (*pyffi.formats.nif.NifFormat.MotionQuality* *property*), 186
attribute), 86
 MO_QUAL_USER (*pyffi.formats.nif.NifFormat.MotionQuality* *property*), 186
attribute), 86
 MO_SYS_BOX (*pyffi.formats.nif.NifFormat.MotionSystem* *property*), 61
attribute), 86
 MO_SYS_BOX_STABILIZED (*pyffi.formats.nif.NifFormat.MotionSystem* *property*), 87
attribute), 86
 MO_SYS_CHARACTER (*pyffi.formats.nif.NifFormat.MotionSystem* *property*), 114
attribute), 86
 MO_SYS_DYNAMIC (*pyffi.formats.nif.NifFormat.MotionSystem* *property*), 115
attribute), 86
 MO_SYS_FIXED (*pyffi.formats.nif.NifFormat.MotionSystem* *property*), 119
attribute), 86
 MO_SYS_INVALID (*pyffi.formats.nif.NifFormat.MotionSystem* *property*), 121
attribute), 86
 MO_SYS_KEYFRAMED (*pyffi.formats.nif.NifFormat.MotionSystem* *property*), 122
attribute), 86
 MO_SYS_SPHERE (*pyffi.formats.nif.NifFormat.MotionSystem* *property*), 129
attribute), 87
 MO_SYS_SPHERE_INERTIA (*pyffi.formats.nif.NifFormat.MotionSystem* *property*), 142
attribute), 87
 MO_SYS_THIN_BOX (*pyffi.formats.nif.NifFormat.MotionSystem* *property*), 143
attribute), 87
 model_projection_matrix() (*pyffi.formats.nif.NifFormat.NiTextureEffect* *property*), 147
 model_projection_transform() (*pyffi.formats.nif.NifFormat.NiTextureEffect* *property*), 147
 modifier() (*pyffi.formats.nif.NifFormat.BSPSysHavokUpdateModifier* *property*), 235

N

name() (*pyffi.formats.nif.NifFormat.AVObject* *property*), 43
 name() (*pyffi.formats.nif.NifFormat.bhkRagdollTemplate* *property*), 186
 name() (*pyffi.formats.nif.NifFormat.bhkRagdollTemplateData* *property*), 186
 name() (*pyffi.formats.nif.NifFormat.BSTreadTransform* *property*), 61
 name() (*pyffi.formats.nif.NifFormat.Ni3dsAnimationNode* *property*), 87
 name() (*pyffi.formats.nif.NifFormat.NiExtraData* *property*), 100
 name() (*pyffi.formats.nif.NifFormat.NiPSBombForce* *property*), 114
 name() (*pyffi.formats.nif.NifFormat.NiPSBoxEmitter* *property*), 115
 name() (*pyffi.formats.nif.NifFormat.NiPSMeshEmitter* *property*), 119
 name() (*pyffi.formats.nif.NifFormat.NiPSPlanarCollider* *property*), 121
 name() (*pyffi.formats.nif.NifFormat.NiPSSphereEmitter* *property*), 122
 name() (*pyffi.formats.nif.NifFormat.NiPSysModifier* *property*), 129
 name() (*pyffi.formats.nif.NifFormat.NiSequence* *property*), 142
 name() (*pyffi.formats.nif.NifFormat.NiShadowGenerator* *property*), 143
 name() (*pyffi.formats.nif.NifFormat.SemanticData* *property*), 161
 name_attribute() (*pyffi.object_models.FileFormat* *class method*), 234
 name_class() (*pyffi.object_models.FileFormat* *class method*), 234
 name_parts() (*pyffi.object_models.FileFormat* *class method*), 235

near_extent() (pyffi.formats.nif.NifFormat.LODRange NifFormat.bhkCMSDChunk (class in
 property), 81 pyffi.formats.nif), 179
 next_collider() (pyffi.formats.nif.NifFormat.NiPSysCollider NifFormat.bhkCMSDMaterial (class in
 property), 125 pyffi.formats.nif), 179
 next_controller() NifFormat.bhkCMSDTransform (class in
 (pyffi.formats.nif.NifFormat.NiTimeController pyffi.formats.nif), 179
 property), 149 NifFormat.bhkCollisionObject (class in
 next_extra_data() pyffi.formats.nif.NifFormat.NiExtraData NifFormat.bhkCompressedMeshShape (class in
 property), 100 pyffi.formats.nif), 179
 next_modifier() (pyffi.formats.nif.NifFormat.NiParticleModifier NifFormat.bhkCompressedMeshShapeData
 property), 133 (class in pyffi.formats.nif), 180
 NifFormat (class in pyffi.formats.nif), 43 NifFormat.bhkConstraint (class in
 NifFormat.AbstractAdditionalGeometryData pyffi.formats.nif), 181
 (class in pyffi.formats.nif), 44 NifFormat.bhkConvexListShape (class in
 NifFormat.AdditionalDataBlock (class in pyffi.formats.nif), 181
 pyffi.formats.nif), 44 NifFormat.bhkConvexShape (class in
 NifFormat.AdditionalDataInfo (class in pyffi.formats.nif), 181
 pyffi.formats.nif), 44 NifFormat.bhkConvexTransformShape (class
 NifFormat.AlphaFormat (class in in pyffi.formats.nif), 181
 pyffi.formats.nif), 44 NifFormat.bhkConvexVerticesShape (class in
 NifFormat.AnimationType (class in pyffi.formats.nif), 181
 pyffi.formats.nif), 44 NifFormat.bhkEntity (class in pyffi.formats.nif),
 NifFormat.ApplyMode (class in pyffi.formats.nif), 181
 44 NifFormat.bhkHingeConstraint (class in
 NifFormat.ArkTexture (class in pyffi.formats.nif), pyffi.formats.nif), 181
 45 NifFormat.bhkLimitedHingeConstraint
 NifFormat.ATextureRenderData (class in (class in pyffi.formats.nif), 182
 pyffi.formats.nif), 43 NifFormat.bhkLiquidAction (class in
 NifFormat.AVObject (class in pyffi.formats.nif), 43 pyffi.formats.nif), 182
 NifFormat.AvoidNode (class in pyffi.formats.nif), NifFormat.bhkListShape (class in
 45 pyffi.formats.nif), 182
 NifFormat.BallAndSocketDescriptor (class NifFormat.bhkMalleableConstraint (class in
 in pyffi.formats.nif), 62 pyffi.formats.nif), 182
 NifFormat.bhkAabbPhantom (class in NifFormat.bhkMeshShape (class in
 pyffi.formats.nif), 177 pyffi.formats.nif), 182
 NifFormat.bhkBallAndSocketConstraint NifFormat.bhkMoppBvTreeShape (class in
 (class in pyffi.formats.nif), 177 pyffi.formats.nif), 182
 NifFormat.bhkBallSocketConstraintChain NifFormat.bhkMultiSphereShape (class in
 (class in pyffi.formats.nif), 177 pyffi.formats.nif), 183
 NifFormat.bhkBlendCollisionObject (class NifFormat.bhkNiCollisionObject (class in
 in pyffi.formats.nif), 178 pyffi.formats.nif), 183
 NifFormat.bhkBlendController (class in NifFormat.bhkNiTriStripsShape (class in
 pyffi.formats.nif), 178 pyffi.formats.nif), 183
 NifFormat.bhkBoxShape (class in NifFormat.bhkOrientHingedBodyAction
 pyffi.formats.nif), 178 (class in pyffi.formats.nif), 183
 NifFormat.bhkBreakableConstraint (class in NifFormat.bhkPackedNiTriStripsShape
 pyffi.formats.nif), 178 (class in pyffi.formats.nif), 183
 NifFormat.bhkBvTreeShape (class in NifFormat.bhkPCollisionObject (class in
 pyffi.formats.nif), 178 pyffi.formats.nif), 183
 NifFormat.bhkCapsuleShape (class in NifFormat.bhkPhantom (class in pyffi.formats.nif),
 pyffi.formats.nif), 179 185
 NifFormat.bhkCMSDBigTris (class in NifFormat.bhkPrismaticConstraint (class in
 pyffi.formats.nif), 178 pyffi.formats.nif), 185

NifFormat.bhkRagdollConstraint (class in pyffi.formats.nif), 186	NifFormat.BSBlastNode (class in pyffi.formats.nif), 45
NifFormat.bhkRagdollTemplate (class in pyffi.formats.nif), 186	NifFormat.BSBoneLODExtraData (class in pyffi.formats.nif), 45
NifFormat.bhkRagdollTemplateData (class in pyffi.formats.nif), 186	NifFormat.BSBound (class in pyffi.formats.nif), 45
NifFormat.bhkRDTConstraint (class in pyffi.formats.nif), 185	NifFormat.BSDamageStage (class in pyffi.formats.nif), 45
NifFormat.bhkRDTMalleableConstraint (class in pyffi.formats.nif), 185	NifFormat.BSDebrisNode (class in pyffi.formats.nif), 46
NifFormat.bhkRefObject (class in pyffi.formats.nif), 186	NifFormat.BSDecalPlacementVectorExtraData (class in pyffi.formats.nif), 46
NifFormat.bhkRigidBody (class in pyffi.formats.nif), 186	NifFormat.BSDismemberBodyPartType (class in pyffi.formats.nif), 46
NifFormat.bhkRigidBodyT (class in pyffi.formats.nif), 186	NifFormat.BSDismemberSkinInstance (class in pyffi.formats.nif), 49
NifFormat.bhkSerializable (class in pyffi.formats.nif), 187	NifFormat.BSDistantTreeShaderProperty (class in pyffi.formats.nif), 49
NifFormat.bhkShape (class in pyffi.formats.nif), 187	NifFormat.BSEffectShaderProperty (class in pyffi.formats.nif), 49
NifFormat.bhkShapeCollection (class in pyffi.formats.nif), 187	NifFormat.BSEffectShaderPropertyColorController (class in pyffi.formats.nif), 49
NifFormat.bhkShapePhantom (class in pyffi.formats.nif), 187	NifFormat.BSEffectShaderPropertyFloatController (class in pyffi.formats.nif), 49
NifFormat.bhkSimpleShapePhantom (class in pyffi.formats.nif), 187	NifFormat.BSFadeNode (class in pyffi.formats.nif), 49
NifFormat.bhkSPCollisionObject (class in pyffi.formats.nif), 186	NifFormat.BSFrustumFOVController (class in pyffi.formats.nif), 50
NifFormat.bhkSphereRepShape (class in pyffi.formats.nif), 187	NifFormat.BSFurnitureMarker (class in pyffi.formats.nif), 50
NifFormat.bhkSphereShape (class in pyffi.formats.nif), 187	NifFormat.BSFurnitureMarkerNode (class in pyffi.formats.nif), 50
NifFormat.bhkStiffSpringConstraint (class in pyffi.formats.nif), 187	NifFormat.BSInvMarker (class in pyffi.formats.nif), 50
NifFormat.bhkTransformShape (class in pyffi.formats.nif), 187	NifFormat.BSKeyframeController (class in pyffi.formats.nif), 50
NifFormat.bhkWorldObject (class in pyffi.formats.nif), 188	NifFormat.BSLagBoneController (class in pyffi.formats.nif), 50
NifFormat.BillboardMode (class in pyffi.formats.nif), 62	NifFormat.BSLeafAnimNode (class in pyffi.formats.nif), 50
NifFormat.BodyPartList (class in pyffi.formats.nif), 63	NifFormat.BSLightingShaderProperty (class in pyffi.formats.nif), 51
NifFormat.BoneLOD (class in pyffi.formats.nif), 63	NifFormat.BSLightingShaderPropertyColorController (class in pyffi.formats.nif), 51
NifFormat.bool (class in pyffi.formats.nif), 188	NifFormat.BSLightingShaderPropertyFloatController (class in pyffi.formats.nif), 52
NifFormat.BoundingBox (class in pyffi.formats.nif), 63	NifFormat.BSLightingShaderPropertyShaderType (class in pyffi.formats.nif), 52
NifFormat.BoundingBoxVolume (class in pyffi.formats.nif), 63	NifFormat.BSLODTriShape (class in pyffi.formats.nif), 50
NifFormat.BoundsVolumeType (class in pyffi.formats.nif), 63	NifFormat.BSMasterParticleSystem (class in pyffi.formats.nif), 52
NifFormat.BoxBV (class in pyffi.formats.nif), 63	NifFormat.BSMaterialEmittanceMultController (class in pyffi.formats.nif), 52
NifFormat.BSAnimNotes (class in pyffi.formats.nif), 45	
NifFormat.BSBehaviorGraphExtraData (class in pyffi.formats.nif), 45	

NifFormat.BSMultiBound	(class in pyffi.formats.nif), 52	NifFormat.BSSegmentedTriShape	(class in pyffi.formats.nif), 57
NifFormat.BSMultiBoundAABB	(class in pyffi.formats.nif), 52	NifFormat.BSSegmentFlags	(class in pyffi.formats.nif), 57
NifFormat.BSMultiBoundData	(class in pyffi.formats.nif), 52	NifFormat.BSShaderFlags	(class in pyffi.formats.nif), 57
NifFormat.BSMultiBoundNode	(class in pyffi.formats.nif), 52	NifFormat.BSShaderFlags2	(class in pyffi.formats.nif), 58
NifFormat.BSMultiBoundOBB	(class in pyffi.formats.nif), 53	NifFormat.BSShaderLightingProperty	(class in pyffi.formats.nif), 59
NifFormat.BSMultiBoundSphere	(class in pyffi.formats.nif), 53	NifFormat.BSShaderNoLightingProperty	(class in pyffi.formats.nif), 59
NifFormat.BSNiAlphaPropertyTestRefControl	(class in pyffi.formats.nif), 53	NifFormat.BSShaderPPLightingProperty	(class in pyffi.formats.nif), 59
NifFormat.BSOrderedNode	(class in pyffi.formats.nif), 53	NifFormat.BSShaderProperty	(class in pyffi.formats.nif), 59
NifFormat.BSPackedAdditionalDataBlock	(class in pyffi.formats.nif), 55	NifFormat.BSShaderTextureSet	(class in pyffi.formats.nif), 60
NifFormat.BSPackedAdditionalGeometryData	(class in pyffi.formats.nif), 55	NifFormat.BSShaderType	(class in pyffi.formats.nif), 60
NifFormat.BSParentVelocityModifier	(class in pyffi.formats.nif), 55	NifFormat.BSSkyShaderProperty	(class in pyffi.formats.nif), 60
NifFormat.BSPartFlag	(class in pyffi.formats.nif), 55	NifFormat.BSStripParticleSystem	(class in pyffi.formats.nif), 61
NifFormat.BSProceduralLightningController	(class in pyffi.formats.nif), 56	NifFormat.BSStripPSysData	(class in pyffi.formats.nif), 60
NifFormat.BSPSysArrayEmitter	(class in pyffi.formats.nif), 53	NifFormat.BSTreadTransfInterpolator	(class in pyffi.formats.nif), 61
NifFormat.BSPSysHavokUpdateModifier	(class in pyffi.formats.nif), 53	NifFormat.BSTreadTransform	(class in pyffi.formats.nif), 61
NifFormat.BSPSysInheritVelocityModifier	(class in pyffi.formats.nif), 53	NifFormat.BSTreadTransformData	(class in pyffi.formats.nif), 61
NifFormat.BSPSysLODModifier	(class in pyffi.formats.nif), 53	NifFormat.BSTreeNode	(class in pyffi.formats.nif), 61
NifFormat.BSPSysMultiTargetEmitterCtrlr	(class in pyffi.formats.nif), 54	NifFormat.BSValueNode	(class in pyffi.formats.nif), 61
NifFormat.BSPSysRecycleBoundModifier	(class in pyffi.formats.nif), 54	NifFormat.BSWArray	(class in pyffi.formats.nif), 61
NifFormat.BSPSysScaleModifier	(class in pyffi.formats.nif), 54	NifFormat.BSWaterShaderProperty	(class in pyffi.formats.nif), 62
NifFormat.BSPSysSimpleColorModifier	(class in pyffi.formats.nif), 54	NifFormat.BSWindModifier	(class in pyffi.formats.nif), 62
NifFormat.BSPSysStripUpdateModifier	(class in pyffi.formats.nif), 54	NifFormat.BSXFlags	(class in pyffi.formats.nif), 62
NifFormat.BSPSysSubTexModifier	(class in pyffi.formats.nif), 55	NifFormat.ByteArray	(class in pyffi.formats.nif), 64
NifFormat.BSRefractionFirePeriodController	(class in pyffi.formats.nif), 56	NifFormat.ByteColor3	(class in pyffi.formats.nif), 64
NifFormat.BSRefractionStrengthController	(class in pyffi.formats.nif), 56	NifFormat.ByteColor4	(class in pyffi.formats.nif), 64
NifFormat.BSRotAccumTransfInterpolator	(class in pyffi.formats.nif), 56	NifFormat.ByteMatrix	(class in pyffi.formats.nif), 64
NifFormat.BSSegment	(class in pyffi.formats.nif), 57	NifFormat.CapsuleBV	(class in pyffi.formats.nif), 65
		NifFormat.ChannelConvention	(class in pyffi.formats.nif), 65

NifFormat.ChannelData (class in pyffi.formats.nif), 65	NifFormat.ExtraVectorsFlags (class in pyffi.formats.nif), 73
NifFormat.ChannelType (class in pyffi.formats.nif), 65	NifFormat.FaceDrawMode (class in pyffi.formats.nif), 73
NifFormat.CloningBehavior (class in pyffi.formats.nif), 66	NifFormat.Fallout3HavokMaterial (class in pyffi.formats.nif), 74
NifFormat.CollisionMode (class in pyffi.formats.nif), 66	NifFormat.Fallout3Layer (class in pyffi.formats.nif), 75
NifFormat.Color3 (class in pyffi.formats.nif), 66	NifFormat.FieldType (class in pyffi.formats.nif), 77
NifFormat.Color4 (class in pyffi.formats.nif), 66	NifFormat.FilePath (class in pyffi.formats.nif), 77
NifFormat.ComponentFormat (class in pyffi.formats.nif), 66	NifFormat.FileVersion (class in pyffi.formats.nif), 77
NifFormat.ConsistencyType (class in pyffi.formats.nif), 68	NifFormat.Flags (class in pyffi.formats.nif), 77
NifFormat.ControllerLink (class in pyffi.formats.nif), 68	NifFormat.Footer (class in pyffi.formats.nif), 77
NifFormat.CoordGenType (class in pyffi.formats.nif), 69	NifFormat.ForceType (class in pyffi.formats.nif), 77
NifFormat.CStreamableAssetData (class in pyffi.formats.nif), 65	NifFormat.FurnitureEntryPoints (class in pyffi.formats.nif), 78
NifFormat.CycleType (class in pyffi.formats.nif), 69	NifFormat.FurniturePosition (class in pyffi.formats.nif), 78
NifFormat.Data (class in pyffi.formats.nif), 70	NifFormat.FxButton (class in pyffi.formats.nif), 78
NifFormat.Data.VersionUInt (class in pyffi.formats.nif), 70	NifFormat.FxRadioButton (class in pyffi.formats.nif), 78
NifFormat.DataStreamAccess (class in pyffi.formats.nif), 71	NifFormat.FxWidget (class in pyffi.formats.nif), 78
NifFormat.DataStreamUsage (class in pyffi.formats.nif), 71	NifFormat.HairShaderProperty (class in pyffi.formats.nif), 79
NifFormat.DeactivatorType (class in pyffi.formats.nif), 71	NifFormat.HalfSpaceBV (class in pyffi.formats.nif), 79
NifFormat.DecalVectorArray (class in pyffi.formats.nif), 72	NifFormat.HavokColFilter (class in pyffi.formats.nif), 79
NifFormat.DecayType (class in pyffi.formats.nif), 72	NifFormat.HavokMaterial (class in pyffi.formats.nif), 79
NifFormat.DistantLODShaderProperty (class in pyffi.formats.nif), 72	NifFormat.Header (class in pyffi.formats.nif), 79
NifFormat.EffectShaderControlledColor (class in pyffi.formats.nif), 72	NifFormat.HeaderString (class in pyffi.formats.nif), 79
NifFormat.EffectShaderControlledVariable (class in pyffi.formats.nif), 72	NifFormat.HingeDescriptor (class in pyffi.formats.nif), 80
NifFormat.EffectType (class in pyffi.formats.nif), 72	NifFormat.hkConstraintType (class in pyffi.formats.nif), 188
NifFormat.ElementReference (class in pyffi.formats.nif), 72	NifFormat.hkPackedNiTriStripsData (class in pyffi.formats.nif), 189
NifFormat.EmitFrom (class in pyffi.formats.nif), 72	NifFormat.hkResponseType (class in pyffi.formats.nif), 189
NifFormat.EndianType (class in pyffi.formats.nif), 73	NifFormat.hkTriangle (class in pyffi.formats.nif), 189
NifFormat.ExportInfo (class in pyffi.formats.nif), 73	NifFormat.ImageType (class in pyffi.formats.nif), 80
NifFormat.ExtraMeshDataEpicMickey (class in pyffi.formats.nif), 73	NifFormat.InertiaMatrix (class in pyffi.formats.nif), 80
NifFormat.ExtraMeshDataEpicMickey2 (class in pyffi.formats.nif), 73	NifFormat.Key (class in pyffi.formats.nif), 81
	NifFormat.KeyGroup (class in pyffi.formats.nif), 81
	NifFormat.KeyType (class in pyffi.formats.nif), 81
	NifFormat.Lighting30ShaderProperty (class in pyffi.formats.nif), 81

in pyffi.formats.nif), 81
 NifFormat.LightingShaderControlledColor (class in *pyffi.formats.nif*), 82
 NifFormat.LightingShaderControlledVariable (class in *pyffi.formats.nif*), 82
 NifFormat.LightMode (class in *pyffi.formats.nif*), 81
 NifFormat.LimitedHingeDescriptor (class in *pyffi.formats.nif*), 82
 NifFormat.LineString (class in *pyffi.formats.nif*), 82
 NifFormat.LODRange (class in *pyffi.formats.nif*), 81
 NifFormat.MatchGroup (class in *pyffi.formats.nif*), 83
 NifFormat.MaterialData (class in *pyffi.formats.nif*), 83
 NifFormat.Matrix22 (class in *pyffi.formats.nif*), 83
 NifFormat.Matrix33 (class in *pyffi.formats.nif*), 83
 NifFormat.Matrix44 (class in *pyffi.formats.nif*), 84
 NifFormat.MeshData (class in *pyffi.formats.nif*), 85
 NifFormat.MeshPrimitiveType (class in *pyffi.formats.nif*), 85
 NifFormat.MipMap (class in *pyffi.formats.nif*), 85
 NifFormat.MipMapFormat (class in *pyffi.formats.nif*), 85
 NifFormat.MoppDataBuildType (class in *pyffi.formats.nif*), 85
 NifFormat.Morph (class in *pyffi.formats.nif*), 86
 NifFormat.MorphWeight (class in *pyffi.formats.nif*), 86
 NifFormat.MotionQuality (class in *pyffi.formats.nif*), 86
 NifFormat.MotionSystem (class in *pyffi.formats.nif*), 86
 NifFormat.MotorDescriptor (class in *pyffi.formats.nif*), 87
 NifFormat.MTransform (class in *pyffi.formats.nif*), 83
 NifFormat.MultiTextureElement (class in *pyffi.formats.nif*), 87
 NifFormat.Ni3dsAlphaAnimator (class in *pyffi.formats.nif*), 87
 NifFormat.Ni3dsAnimationNode (class in *pyffi.formats.nif*), 87
 NifFormat.Ni3dsColorAnimator (class in *pyffi.formats.nif*), 88
 NifFormat.Ni3dsMorphShape (class in *pyffi.formats.nif*), 88
 NifFormat.Ni3dsParticleSystem (class in *pyffi.formats.nif*), 88
 NifFormat.Ni3dsPathController (class in *pyffi.formats.nif*), 88
 NifFormat.NiAdditionalGeometryData (class in *pyffi.formats.nif*), 89
 NifFormat.NiAlphaController (class in *pyffi.formats.nif*), 89
 NifFormat.NiAlphaProperty (class in *pyffi.formats.nif*), 90
 NifFormat.NiAmbientLight (class in *pyffi.formats.nif*), 90
 NifFormat.NiArkAnimationExtraData (class in *pyffi.formats.nif*), 90
 NifFormat.NiArkImporterExtraData (class in *pyffi.formats.nif*), 90
 NifFormat.NiArkShaderExtraData (class in *pyffi.formats.nif*), 90
 NifFormat.NiArkTextureExtraData (class in *pyffi.formats.nif*), 90
 NifFormat.NiArkViewportInfoExtraData (class in *pyffi.formats.nif*), 90
 NifFormat.NiAutoNormalParticles (class in *pyffi.formats.nif*), 91
 NifFormat.NiAutoNormalParticlesData (class in *pyffi.formats.nif*), 91
 NifFormat.NiAVObject (class in *pyffi.formats.nif*), 88
 NifFormat.NiAVObjectPalette (class in *pyffi.formats.nif*), 89
 NifFormat.NiBezierMesh (class in *pyffi.formats.nif*), 94
 NifFormat.NiBezierTriangle4 (class in *pyffi.formats.nif*), 94
 NifFormat.NiBillboardNode (class in *pyffi.formats.nif*), 94
 NifFormat.NiBinaryExtraData (class in *pyffi.formats.nif*), 95
 NifFormat.NiBinaryVoxelData (class in *pyffi.formats.nif*), 95
 NifFormat.NiBinaryVoxelExtraData (class in *pyffi.formats.nif*), 95
 NifFormat.NiBlendBoolInterpolator (class in *pyffi.formats.nif*), 95
 NifFormat.NiBlendFloatInterpolator (class in *pyffi.formats.nif*), 95
 NifFormat.NiBlendInterpolator (class in *pyffi.formats.nif*), 95
 NifFormat.NiBlendPoint3Interpolator (class in *pyffi.formats.nif*), 95
 NifFormat.NiBlendTransformInterpolator (class in *pyffi.formats.nif*), 96
 NifFormat.NiBone (class in *pyffi.formats.nif*), 96
 NifFormat.NiBoneLODController (class in *pyffi.formats.nif*), 96
 NifFormat.NiBoolData (class in *pyffi.formats.nif*), 96
 NifFormat.NiBooleanExtraData (class in *pyffi.formats.nif*), 96
 NifFormat.NiBoolInterpController (class in

<i>pyffi.formats.nif</i>), 96		<i>in pyffi.formats.nif</i>), 99	
NifFormat.NiBoolInterpolator (class in <i>pyffi.formats.nif</i>), 96		NifFormat.NiDirectionalLight (class in <i>pyffi.formats.nif</i>), 99	
NifFormat.NiBoolTimelineInterpolator (class in <i>pyffi.formats.nif</i>), 96		NifFormat.NiDitherProperty (class in <i>pyffi.formats.nif</i>), 99	
NifFormat.NiBSAnimationNode (class in <i>pyffi.formats.nif</i>), 91		NifFormat.NiDynamicEffect (class in <i>pyffi.formats.nif</i>), 99	
NifFormat.NiBSBoneLODController (class in <i>pyffi.formats.nif</i>), 91		NifFormat.NiEnvMappedTriShape (class in <i>pyffi.formats.nif</i>), 100	
NifFormat.NiBSPArrayController (class in <i>pyffi.formats.nif</i>), 91		NifFormat.NiEnvMappedTriShapeData (class in <i>pyffi.formats.nif</i>), 100	
NifFormat.NiBSParticleNode (class in <i>pyffi.formats.nif</i>), 91		NifFormat.NiExtraData (class in <i>pyffi.formats.nif</i>), 100	
NifFormat.NiBSplineBasisData (class in <i>pyffi.formats.nif</i>), 91		NifFormat.NiExtraDataController (class in <i>pyffi.formats.nif</i>), 100	
NifFormat.NiBSplineCompFloatInterpolator (class in <i>pyffi.formats.nif</i>), 91		NifFormat.NiError, 154	
NifFormat.NiBSplineCompPoint3Interpolator (class in <i>pyffi.formats.nif</i>), 91		NifFormat.NiFlipController (class in <i>pyffi.formats.nif</i>), 100	
NifFormat.NiBSplineCompTransformEvaluator (class in <i>pyffi.formats.nif</i>), 91		NifFormat.NiFloatData (class in <i>pyffi.formats.nif</i>), 100	
NifFormat.NiBSplineCompTransformInterpolator (class in <i>pyffi.formats.nif</i>), 91		NifFormat.NiFloatExtraData (class in <i>pyffi.formats.nif</i>), 100	
NifFormat.NiBSplineData (class in <i>pyffi.formats.nif</i>), 92		NifFormat.NiFloatExtraDataController (class in <i>pyffi.formats.nif</i>), 100	
NifFormat.NiBSplineFloatInterpolator (class in <i>pyffi.formats.nif</i>), 93		NifFormat.NiFloatInterpController (class in <i>pyffi.formats.nif</i>), 101	
NifFormat.NiBSplineInterpolator (class in <i>pyffi.formats.nif</i>), 93		NifFormat.NiFloatInterpolator (class in <i>pyffi.formats.nif</i>), 101	
NifFormat.NiBSplinePoint3Interpolator (class in <i>pyffi.formats.nif</i>), 93		NifFormat.NiFloatsExtraData (class in <i>pyffi.formats.nif</i>), 101	
NifFormat.NiBSplineTransformInterpolator (class in <i>pyffi.formats.nif</i>), 93		NifFormat.NiFogProperty (class in <i>pyffi.formats.nif</i>), 101	
NifFormat.NiCamera (class in <i>pyffi.formats.nif</i>), 97		NifFormat.NiFurSpringController (class in <i>pyffi.formats.nif</i>), 101	
NifFormat.NiClod (class in <i>pyffi.formats.nif</i>), 97		NifFormat.NiGeometry (class in <i>pyffi.formats.nif</i>), 102	
NifFormat.NiClodData (class in <i>pyffi.formats.nif</i>), 97		NifFormat.NiGeometryData (class in <i>pyffi.formats.nif</i>), 103	
NifFormat.NiClodSkinInstance (class in <i>pyffi.formats.nif</i>), 98		NifFormat.NiGeomMorpherController (class in <i>pyffi.formats.nif</i>), 101	
NifFormat.NiCollisionData (class in <i>pyffi.formats.nif</i>), 98		NifFormat.NiGravity (class in <i>pyffi.formats.nif</i>), 105	
NifFormat.NiCollisionObject (class in <i>pyffi.formats.nif</i>), 98		NifFormat.NiImage (class in <i>pyffi.formats.nif</i>), 105	
NifFormat.NiColorData (class in <i>pyffi.formats.nif</i>), 98		NifFormat.NiInstancingMeshModifier (class in <i>pyffi.formats.nif</i>), 105	
NifFormat.NiColorExtraData (class in <i>pyffi.formats.nif</i>), 98		NifFormat.NiIntegerExtraData (class in <i>pyffi.formats.nif</i>), 105	
NifFormat.NiControllerManager (class in <i>pyffi.formats.nif</i>), 98		NifFormat.NiIntegersExtraData (class in <i>pyffi.formats.nif</i>), 105	
NifFormat.NiControllerSequence (class in <i>pyffi.formats.nif</i>), 98		NifFormat.NiInterpController (class in <i>pyffi.formats.nif</i>), 105	
NifFormat.NiDataStream (class in <i>pyffi.formats.nif</i>), 99		NifFormat.NiInterpolator (class in <i>pyffi.formats.nif</i>), 105	
NifFormat.NiDefaultAVObjectPalette (class in <i>pyffi.formats.nif</i>), 99		NifFormat.NiKeyBasedInterpolator (class in <i>pyffi.formats.nif</i>), 105	

pyffi.formats.nif), 105
 NifFormat.NiKeyframeController (class in *pyffi.formats.nif*), 106
 NifFormat.NiKeyframeData (class in *pyffi.formats.nif*), 106
 NifFormat.NiLight (class in *pyffi.formats.nif*), 106
 NifFormat.NiLightColorController (class in *pyffi.formats.nif*), 106
 NifFormat.NiLightDimmerController (class in *pyffi.formats.nif*), 106
 NifFormat.NiLightIntensityController (class in *pyffi.formats.nif*), 106
 NifFormat.NiLines (class in *pyffi.formats.nif*), 106
 NifFormat.NiLinesData (class in *pyffi.formats.nif*), 107
 NifFormat.NiLODData (class in *pyffi.formats.nif*), 106
 NifFormat.NiLODNode (class in *pyffi.formats.nif*), 106
 NifFormat.NiLookAtController (class in *pyffi.formats.nif*), 107
 NifFormat.NiLookAtInterpolator (class in *pyffi.formats.nif*), 107
 NifFormat.NiMaterialColorController (class in *pyffi.formats.nif*), 107
 NifFormat.NiMaterialProperty (class in *pyffi.formats.nif*), 107
 NifFormat.NiMesh (class in *pyffi.formats.nif*), 107
 NifFormat.NiMeshHWInstance (class in *pyffi.formats.nif*), 108
 NifFormat.NiMeshModifier (class in *pyffi.formats.nif*), 108
 NifFormat.NiMeshParticleSystem (class in *pyffi.formats.nif*), 109
 NifFormat.NiMeshPSysData (class in *pyffi.formats.nif*), 108
 NifFormat.NiMorphController (class in *pyffi.formats.nif*), 109
 NifFormat.NiMorphData (class in *pyffi.formats.nif*), 109
 NifFormat.NiMorpherController (class in *pyffi.formats.nif*), 109
 NifFormat.NiMorphMeshModifier (class in *pyffi.formats.nif*), 109
 NifFormat.NiMorphWeightsController (class in *pyffi.formats.nif*), 109
 NifFormat.NiMultiTargetTransformController (class in *pyffi.formats.nif*), 109
 NifFormat.NiMultiTextureProperty (class in *pyffi.formats.nif*), 110
 NifFormat.NiNode (class in *pyffi.formats.nif*), 110
 NifFormat.NiObject (class in *pyffi.formats.nif*), 113
 NifFormat.NiObjectNET (class in *pyffi.formats.nif*), 113
 NifFormat.NiPalette (class in *pyffi.formats.nif*), 131
 NifFormat.NiParticleBomb (class in *pyffi.formats.nif*), 132
 NifFormat.NiParticleColorModifier (class in *pyffi.formats.nif*), 132
 NifFormat.NiParticleGrowFade (class in *pyffi.formats.nif*), 132
 NifFormat.NiParticleMeshes (class in *pyffi.formats.nif*), 132
 NifFormat.NiParticleMeshesData (class in *pyffi.formats.nif*), 132
 NifFormat.NiParticleMeshModifier (class in *pyffi.formats.nif*), 132
 NifFormat.NiParticleModifier (class in *pyffi.formats.nif*), 133
 NifFormat.NiParticleRotation (class in *pyffi.formats.nif*), 133
 NifFormat.NiParticles (class in *pyffi.formats.nif*), 134
 NifFormat.NiParticlesData (class in *pyffi.formats.nif*), 134
 NifFormat.NiParticleSystem (class in *pyffi.formats.nif*), 133
 NifFormat.NiParticleSystemController (class in *pyffi.formats.nif*), 133
 NifFormat.NiPathController (class in *pyffi.formats.nif*), 135
 NifFormat.NiPathInterpolator (class in *pyffi.formats.nif*), 135
 NifFormat.NiPersistentSrcTextureRendererData (class in *pyffi.formats.nif*), 135
 NifFormat.NiPhysXActorDesc (class in *pyffi.formats.nif*), 136
 NifFormat.NiPhysXBodyDesc (class in *pyffi.formats.nif*), 136
 NifFormat.NiPhysXD6JointDesc (class in *pyffi.formats.nif*), 136
 NifFormat.NiPhysXKinematicSrc (class in *pyffi.formats.nif*), 136
 NifFormat.NiPhysXMaterialDesc (class in *pyffi.formats.nif*), 136
 NifFormat.NiPhysXMeshDesc (class in *pyffi.formats.nif*), 137
 NifFormat.NiPhysXProp (class in *pyffi.formats.nif*), 137
 NifFormat.NiPhysXPropDesc (class in *pyffi.formats.nif*), 137
 NifFormat.NiPhysXShapeDesc (class in *pyffi.formats.nif*), 138
 NifFormat.NiPhysXTransformDest (class in *pyffi.formats.nif*), 138
 NifFormat.NiPixelData (class in

[pyffi.formats.nif](#)), 139

NifFormat.NiPlanarCollider (class in [pyffi.formats.nif](#)), 139

NifFormat.NiPoint3InterpController (class in [pyffi.formats.nif](#)), 139

NifFormat.NiPoint3Interpolator (class in [pyffi.formats.nif](#)), 139

NifFormat.NiPointLight (class in [pyffi.formats.nif](#)), 140

NifFormat.NiPortal (class in [pyffi.formats.nif](#)), 140

NifFormat.NiPosData (class in [pyffi.formats.nif](#)), 140

NifFormat.NiProperty (class in [pyffi.formats.nif](#)), 140

NifFormat.NiPSBombForce (class in [pyffi.formats.nif](#)), 114

NifFormat.NiPSBoundUpdater (class in [pyffi.formats.nif](#)), 115

NifFormat.NiPSBoxEmitter (class in [pyffi.formats.nif](#)), 115

NifFormat.NiPSCylinderEmitter (class in [pyffi.formats.nif](#)), 116

NifFormat.NiPSDragForce (class in [pyffi.formats.nif](#)), 116

NifFormat.NiPSEmitParticlesCtrlr (class in [pyffi.formats.nif](#)), 117

NifFormat.NiPSEmitterDeclinationCtrlr (class in [pyffi.formats.nif](#)), 117

NifFormat.NiPSEmitterDeclinationVarCtrlr (class in [pyffi.formats.nif](#)), 117

NifFormat.NiPSEmitterLifeSpanCtrlr (class in [pyffi.formats.nif](#)), 117

NifFormat.NiPSEmitterPlanarAngleCtrlr (class in [pyffi.formats.nif](#)), 117

NifFormat.NiPSEmitterPlanarAngleVarCtrlr (class in [pyffi.formats.nif](#)), 117

NifFormat.NiPSEmitterRadiusCtrlr (class in [pyffi.formats.nif](#)), 117

NifFormat.NiPSEmitterRotAngleCtrlr (class in [pyffi.formats.nif](#)), 117

NifFormat.NiPSEmitterRotAngleVarCtrlr (class in [pyffi.formats.nif](#)), 117

NifFormat.NiPSEmitterRotSpeedCtrlr (class in [pyffi.formats.nif](#)), 117

NifFormat.NiPSEmitterRotSpeedVarCtrlr (class in [pyffi.formats.nif](#)), 117

NifFormat.NiPSEmitterSpeedCtrlr (class in [pyffi.formats.nif](#)), 117

NifFormat.NiPSFacingQuadGenerator (class in [pyffi.formats.nif](#)), 117

NifFormat.NiPSForceActiveCtrlr (class in [pyffi.formats.nif](#)), 118

NifFormat.NiPSGravityForce (class in [pyffi.formats.nif](#)), 118

NifFormat.NiPSGravityStrengthCtrlr (class in [pyffi.formats.nif](#)), 119

NifFormat.NiPSMeshEmitter (class in [pyffi.formats.nif](#)), 119

NifFormat.NiPSMeshParticleSystem (class in [pyffi.formats.nif](#)), 120

NifFormat.NiPSParticleSystem (class in [pyffi.formats.nif](#)), 120

NifFormat.NiPSPlanarCollider (class in [pyffi.formats.nif](#)), 121

NifFormat.NiPSResetOnLoopCtrlr (class in [pyffi.formats.nif](#)), 121

NifFormat.NiPSSimulator (class in [pyffi.formats.nif](#)), 121

NifFormat.NiPSSimulatorCollidersStep (class in [pyffi.formats.nif](#)), 121

NifFormat.NiPSSimulatorFinalStep (class in [pyffi.formats.nif](#)), 121

NifFormat.NiPSSimulatorForcesStep (class in [pyffi.formats.nif](#)), 121

NifFormat.NiPSSimulatorGeneralStep (class in [pyffi.formats.nif](#)), 122

NifFormat.NiPSSimulatorMeshAlignStep (class in [pyffi.formats.nif](#)), 122

NifFormat.NiPSSimulatorStep (class in [pyffi.formats.nif](#)), 122

NifFormat.NiPSSpawner (class in [pyffi.formats.nif](#)), 122

NifFormat.NiPSSphereEmitter (class in [pyffi.formats.nif](#)), 122

NifFormat.NiPSSphericalCollider (class in [pyffi.formats.nif](#)), 123

NifFormat.NiPSysAgeDeathModifier (class in [pyffi.formats.nif](#)), 123

NifFormat.NiPSysAirFieldAirFrictionCtrlr (class in [pyffi.formats.nif](#)), 123

NifFormat.NiPSysAirFieldInheritVelocityCtrlr (class in [pyffi.formats.nif](#)), 124

NifFormat.NiPSysAirFieldModifier (class in [pyffi.formats.nif](#)), 124

NifFormat.NiPSysAirFieldSpreadCtrlr (class in [pyffi.formats.nif](#)), 124

NifFormat.NiPSysBombModifier (class in [pyffi.formats.nif](#)), 124

NifFormat.NiPSysBoundUpdateModifier (class in [pyffi.formats.nif](#)), 124

NifFormat.NiPSysBoxEmitter (class in [pyffi.formats.nif](#)), 124

NifFormat.NiPSysCollider (class in [pyffi.formats.nif](#)), 124

NifFormat.NiPSysColliderManager (class in [pyffi.formats.nif](#)), 125

NifFormat.NiPSysColorModifier (class in

<i>pyffi.formats.nif</i>), 125	<i>pyffi.formats.nif</i>), 129
NifFormat.NiPSysCylinderEmitter (class in <i>pyffi.formats.nif</i>), 125	NifFormat.NiPSysMeshUpdateModifier (class in <i>pyffi.formats.nif</i>), 129
NifFormat.NiPSysData (class in <i>pyffi.formats.nif</i>), 125	NifFormat.NiPSysModifier (class in <i>pyffi.formats.nif</i>), 129
NifFormat.NiPSysDragFieldModifier (class in <i>pyffi.formats.nif</i>), 126	NifFormat.NiPSysModifierActiveCtrlr (class in <i>pyffi.formats.nif</i>), 129
NifFormat.NiPSysDragModifier (class in <i>pyffi.formats.nif</i>), 126	NifFormat.NiPSysModifierBoolCtrlr (class in <i>pyffi.formats.nif</i>), 129
NifFormat.NiPSysEmitter (class in <i>pyffi.formats.nif</i>), 126	NifFormat.NiPSysModifierCtrlr (class in <i>pyffi.formats.nif</i>), 129
NifFormat.NiPSysEmitterCtrlr (class in <i>pyffi.formats.nif</i>), 126	NifFormat.NiPSysModifierFloatCtrlr (class in <i>pyffi.formats.nif</i>), 129
NifFormat.NiPSysEmitterCtrlrData (class in <i>pyffi.formats.nif</i>), 126	NifFormat.NiPSysPlanarCollider (class in <i>pyffi.formats.nif</i>), 129
NifFormat.NiPSysEmitterDeclinationCtrlr (class in <i>pyffi.formats.nif</i>), 127	NifFormat.NiPSysPositionModifier (class in <i>pyffi.formats.nif</i>), 130
NifFormat.NiPSysEmitterDeclinationVarCtrlr (class in <i>pyffi.formats.nif</i>), 127	NifFormat.NiPSysRadialFieldModifier (class in <i>pyffi.formats.nif</i>), 130
NifFormat.NiPSysEmitterInitialRadiusCtrlr (class in <i>pyffi.formats.nif</i>), 127	NifFormat.NiPSysResetOnLoopCtrlr (class in <i>pyffi.formats.nif</i>), 130
NifFormat.NiPSysEmitterLifeSpanCtrlr (class in <i>pyffi.formats.nif</i>), 127	NifFormat.NiPSysRotationModifier (class in <i>pyffi.formats.nif</i>), 130
NifFormat.NiPSysEmitterPlanarAngleCtrlr (class in <i>pyffi.formats.nif</i>), 127	NifFormat.NiPSysSpawnModifier (class in <i>pyffi.formats.nif</i>), 130
NifFormat.NiPSysEmitterPlanarAngleVarCtrlr (class in <i>pyffi.formats.nif</i>), 127	NifFormat.NiPSysSphereEmitter (class in <i>pyffi.formats.nif</i>), 131
NifFormat.NiPSysEmitterSpeedCtrlr (class in <i>pyffi.formats.nif</i>), 127	NifFormat.NiPSysSphericalCollider (class in <i>pyffi.formats.nif</i>), 131
NifFormat.NiPSysFieldAttenuationCtrlr (class in <i>pyffi.formats.nif</i>), 127	NifFormat.NiPSysTrailEmitter (class in <i>pyffi.formats.nif</i>), 131
NifFormat.NiPSysFieldMagnitudeCtrlr (class in <i>pyffi.formats.nif</i>), 127	NifFormat.NiPSysTurbulenceFieldModifier (class in <i>pyffi.formats.nif</i>), 131
NifFormat.NiPSysFieldMaxDistanceCtrlr (class in <i>pyffi.formats.nif</i>), 127	NifFormat.NiPSysUpdateCtrlr (class in <i>pyffi.formats.nif</i>), 131
NifFormat.NiPSysFieldModifier (class in <i>pyffi.formats.nif</i>), 127	NifFormat.NiPSysVolumeEmitter (class in <i>pyffi.formats.nif</i>), 131
NifFormat.NiPSysGravityFieldModifier (class in <i>pyffi.formats.nif</i>), 128	NifFormat.NiPSysVortexFieldModifier (class in <i>pyffi.formats.nif</i>), 131
NifFormat.NiPSysGravityModifier (class in <i>pyffi.formats.nif</i>), 128	NifFormat.NiRangeLODData (class in <i>pyffi.formats.nif</i>), 140
NifFormat.NiPSysGravityStrengthCtrlr (class in <i>pyffi.formats.nif</i>), 128	NifFormat.NiRawImageData (class in <i>pyffi.formats.nif</i>), 140
NifFormat.NiPSysGrowFadeModifier (class in <i>pyffi.formats.nif</i>), 128	NifFormat.NiRenderObject (class in <i>pyffi.formats.nif</i>), 141
NifFormat.NiPSysInitialRotAngleCtrlr (class in <i>pyffi.formats.nif</i>), 128	NifFormat.NiRollController (class in <i>pyffi.formats.nif</i>), 141
NifFormat.NiPSysInitialRotAngleVarCtrlr (class in <i>pyffi.formats.nif</i>), 128	NifFormat.NiRoom (class in <i>pyffi.formats.nif</i>), 141
NifFormat.NiPSysInitialRotSpeedCtrlr (class in <i>pyffi.formats.nif</i>), 128	NifFormat.NiRoomGroup (class in <i>pyffi.formats.nif</i>), 141
NifFormat.NiPSysInitialRotSpeedVarCtrlr (class in <i>pyffi.formats.nif</i>), 128	NifFormat.NiRotatingParticles (class in <i>pyffi.formats.nif</i>), 141
NifFormat.NiPSysMeshEmitter (class in <i>pyffi.formats.nif</i>), 129	NifFormat.NiRotatingParticlesData (class in <i>pyffi.formats.nif</i>), 141

NifFormat.NiScreenElements (class in pyffi.formats.nif), 141	NifFormat.NiTextureEffect (class in pyffi.formats.nif), 147
NifFormat.NiScreenElementsData (class in pyffi.formats.nif), 142	NifFormat.NiTextureModeProperty (class in pyffi.formats.nif), 148
NifFormat.NiScreenLODData (class in pyffi.formats.nif), 142	NifFormat.NiTextureProperty (class in pyffi.formats.nif), 148
NifFormat.NiSequence (class in pyffi.formats.nif), 142	NifFormat.NiTextureTransformController (class in pyffi.formats.nif), 148
NifFormat.NiSequenceData (class in pyffi.formats.nif), 142	NifFormat.NiTexturingProperty (class in pyffi.formats.nif), 148
NifFormat.NiSequenceStreamHelper (class in pyffi.formats.nif), 142	NifFormat.NiTimeController (class in pyffi.formats.nif), 149
NifFormat.NiShadeProperty (class in pyffi.formats.nif), 143	NifFormat.NiTransformController (class in pyffi.formats.nif), 149
NifFormat.NiShadowGenerator (class in pyffi.formats.nif), 143	NifFormat.NiTransformData (class in pyffi.formats.nif), 149
NifFormat.NiSingleInterpController (class in pyffi.formats.nif), 143	NifFormat.NiTransformEvaluator (class in pyffi.formats.nif), 150
NifFormat.NiSkinData (class in pyffi.formats.nif), 143	NifFormat.NiTransformInterpolator (class in pyffi.formats.nif), 150
NifFormat.NiSkinInstance (class in pyffi.formats.nif), 144	NifFormat.NiTransparentProperty (class in pyffi.formats.nif), 150
NifFormat.NiSkinningLODController (class in pyffi.formats.nif), 145	NifFormat.NiTriBasedGeom (class in pyffi.formats.nif), 150
NifFormat.NiSkinningMeshModifier (class in pyffi.formats.nif), 145	NifFormat.NiTriBasedGeomData (class in pyffi.formats.nif), 151
NifFormat.NiSkinPartition (class in pyffi.formats.nif), 144	NifFormat.NiTriShape (class in pyffi.formats.nif), 151
NifFormat.NiSortAdjustNode (class in pyffi.formats.nif), 145	NifFormat.NiTriShapeData (class in pyffi.formats.nif), 151
NifFormat.NiSourceCubeMap (class in pyffi.formats.nif), 145	NifFormat.NiTriShapeSkinController (class in pyffi.formats.nif), 152
NifFormat.NiSourceTexture (class in pyffi.formats.nif), 145	NifFormat.NiTriStrips (class in pyffi.formats.nif), 152
NifFormat.NiSpecularProperty (class in pyffi.formats.nif), 145	NifFormat.NiTriStripsData (class in pyffi.formats.nif), 152
NifFormat.NiSphericalCollider (class in pyffi.formats.nif), 145	NifFormat.NiUVController (class in pyffi.formats.nif), 153
NifFormat.NiSpotLight (class in pyffi.formats.nif), 146	NifFormat.NiUVData (class in pyffi.formats.nif), 153
NifFormat.NiStencilProperty (class in pyffi.formats.nif), 146	NifFormat.NiVectorExtraData (class in pyffi.formats.nif), 153
NifFormat.NiStringExtraData (class in pyffi.formats.nif), 146	NifFormat.NiVertexColorProperty (class in pyffi.formats.nif), 153
NifFormat.NiStringPalette (class in pyffi.formats.nif), 146	NifFormat.NiVertWeightsExtraData (class in pyffi.formats.nif), 153
NifFormat.NiStringsExtraData (class in pyffi.formats.nif), 147	NifFormat.NiVisController (class in pyffi.formats.nif), 153
NifFormat.NiSwitchNode (class in pyffi.formats.nif), 147	NifFormat.NiVisData (class in pyffi.formats.nif), 153
NifFormat.NiTextKeyExtraData (class in pyffi.formats.nif), 147	NifFormat.NiWireframeProperty (class in pyffi.formats.nif), 153
NifFormat.NiTexture (class in pyffi.formats.nif), 147	NifFormat.NiZBufferProperty (class in pyffi.formats.nif), 154

NifFormat.NodeGroup (class in <i>pyffi.formats.nif</i>), 154	163
NifFormat.OblivionHavokMaterial (class in <i>pyffi.formats.nif</i>), 154	NifFormat.SkinShapeGroup (class in <i>pyffi.formats.nif</i>), 163
NifFormat.OblivionLayer (class in <i>pyffi.formats.nif</i>), 155	NifFormat.SkinTransform (class in <i>pyffi.formats.nif</i>), 163
NifFormat.OblivionSubShape (class in <i>pyffi.formats.nif</i>), 157	NifFormat.SkinWeight (class in <i>pyffi.formats.nif</i>), 163
NifFormat.OldSkinData (class in <i>pyffi.formats.nif</i>), 157	NifFormat.SkyObjectType (class in <i>pyffi.formats.nif</i>), 164
NifFormat.Particle (class in <i>pyffi.formats.nif</i>), 157	NifFormat.SkyrimHavokMaterial (class in <i>pyffi.formats.nif</i>), 164
NifFormat.ParticleDesc (class in <i>pyffi.formats.nif</i>), 157	NifFormat.SkyrimLayer (class in <i>pyffi.formats.nif</i>), 166
NifFormat.physXMaterialRef (class in <i>pyffi.formats.nif</i>), 189	NifFormat.SkyrimShaderPropertyFlags1 (class in <i>pyffi.formats.nif</i>), 167
NifFormat.PixelFormat (class in <i>pyffi.formats.nif</i>), 158	NifFormat.SkyrimShaderPropertyFlags2 (class in <i>pyffi.formats.nif</i>), 168
NifFormat.PixelLayout (class in <i>pyffi.formats.nif</i>), 158	NifFormat.SkyrimWaterShaderFlags (class in <i>pyffi.formats.nif</i>), 169
NifFormat.Polygon (class in <i>pyffi.formats.nif</i>), 158	NifFormat.SkyShaderProperty (class in <i>pyffi.formats.nif</i>), 164
NifFormat.PrismaticDescriptor (class in <i>pyffi.formats.nif</i>), 158	NifFormat.SolverDeactivation (class in <i>pyffi.formats.nif</i>), 169
NifFormat.PropagationMode (class in <i>pyffi.formats.nif</i>), 159	NifFormat.SortingMode (class in <i>pyffi.formats.nif</i>), 170
NifFormat.PSLoopBehavior (class in <i>pyffi.formats.nif</i>), 157	NifFormat.SphereBV (class in <i>pyffi.formats.nif</i>), 170
NifFormat.Ptr (class in <i>pyffi.formats.nif</i>), 159	NifFormat.StencilAction (class in <i>pyffi.formats.nif</i>), 170
NifFormat.QTransform (class in <i>pyffi.formats.nif</i>), 159	NifFormat.StencilCompareMode (class in <i>pyffi.formats.nif</i>), 170
NifFormat.Quaternion (class in <i>pyffi.formats.nif</i>), 160	NifFormat.StiffSpringDescriptor (class in <i>pyffi.formats.nif</i>), 170
NifFormat.QuaternionXYZW (class in <i>pyffi.formats.nif</i>), 160	NifFormat.string (class in <i>pyffi.formats.nif</i>), 189
NifFormat.QuatKey (class in <i>pyffi.formats.nif</i>), 160	NifFormat.StringOffset (class in <i>pyffi.formats.nif</i>), 171
NifFormat.RagdollDescriptor (class in <i>pyffi.formats.nif</i>), 160	NifFormat.StringPalette (class in <i>pyffi.formats.nif</i>), 171
NifFormat.Ref (class in <i>pyffi.formats.nif</i>), 160	NifFormat.SubConstraint (class in <i>pyffi.formats.nif</i>), 172
NifFormat.Region (class in <i>pyffi.formats.nif</i>), 161	NifFormat.SymmetryType (class in <i>pyffi.formats.nif</i>), 172
NifFormat.RootCollisionNode (class in <i>pyffi.formats.nif</i>), 161	NifFormat.SyncPoint (class in <i>pyffi.formats.nif</i>), 172
NifFormat.SemanticData (class in <i>pyffi.formats.nif</i>), 161	NifFormat.TallGrassShaderProperty (class in <i>pyffi.formats.nif</i>), 173
NifFormat.ShaderTexDesc (class in <i>pyffi.formats.nif</i>), 161	NifFormat.TargetColor (class in <i>pyffi.formats.nif</i>), 173
NifFormat.ShortString (class in <i>pyffi.formats.nif</i>), 162	NifFormat.TBC (class in <i>pyffi.formats.nif</i>), 173
NifFormat.SizedString (class in <i>pyffi.formats.nif</i>), 162	NifFormat.TexClampMode (class in <i>pyffi.formats.nif</i>), 173
NifFormat.SkinData (class in <i>pyffi.formats.nif</i>), 163	NifFormat.TexCoord (class in <i>pyffi.formats.nif</i>), 173
NifFormat.SkinPartition (class in <i>pyffi.formats.nif</i>), 163	NifFormat.TexDesc (class in <i>pyffi.formats.nif</i>), 173
NifFormat.SkinShape (class in <i>pyffi.formats.nif</i>),	

NifFormat.TexFilterMode (class in [pyffi.formats.nif](#)), 174
 NifFormat.TexSource (class in [pyffi.formats.nif](#)), 174
 NifFormat.TexTransform (class in [pyffi.formats.nif](#)), 174
 NifFormat.TexType (class in [pyffi.formats.nif](#)), 175
 NifFormat.TileShaderProperty (class in [pyffi.formats.nif](#)), 175
 NifFormat.Triangle (class in [pyffi.formats.nif](#)), 175
 NifFormat.UnionBV (class in [pyffi.formats.nif](#)), 175
 NifFormat.Vector3 (class in [pyffi.formats.nif](#)), 175
 NifFormat.Vector4 (class in [pyffi.formats.nif](#)), 176
 NifFormat.VelocityType (class in [pyffi.formats.nif](#)), 176
 NifFormat.VertMode (class in [pyffi.formats.nif](#)), 177
 NifFormat.VolumetricFogShaderProperty (class in [pyffi.formats.nif](#)), 177
 NifFormat.WaterShaderProperty (class in [pyffi.formats.nif](#)), 177
 NifFormat.ZCompareMode (class in [pyffi.formats.nif](#)), 177
 NIFXMLPATH, 8
 node() ([pyffi.formats.nif.NifFormat.NiPhysXTransformDestination](#) property), 138
 node_groups() ([pyffi.formats.nif.NifFormat.NiBoneLODController](#) property), 96
 nodes() ([pyffi.formats.nif.NifFormat.BSPSysHavokUpdateModifier](#) property), 53
 nodes() ([pyffi.formats.nif.NifFormat.NodeGroup](#) property), 154
 NONCOLLIDABLE ([pyffi.formats.nif.NifFormat.Fallout3Layer](#) attribute), 76
 NONCOLLIDABLE ([pyffi.formats.nif.NifFormat.OblivionLayer](#) attribute), 156
 NONCOLLIDABLE ([pyffi.formats.nif.NifFormat.SkyrimLayer](#) attribute), 166
 None ([pyffi.formats.nif.NifFormat.ExtraVectorsFlags](#) attribute), 73
 norm() ([pyffi.formats.nif.NifFormat.Vector3](#) method), 176
 normal() ([pyffi.formats.nif.NifFormat.HalfSpaceBV](#) property), 79
 normal() ([pyffi.formats.nif.NifFormat.hkTriangle](#) property), 189
 NORMAL_MAP ([pyffi.formats.nif.NifFormat.TexType](#) attribute), 175
 normal_texture() ([pyffi.formats.nif.NifFormat.NiTexturingProperty](#) property), 149
 normalize() ([pyffi.formats.nif.NifFormat.TexCoord](#) method), 173
 normalize() ([pyffi.formats.nif.NifFormat.Vector3](#) method), 176
 normalize_flag() ([pyffi.formats.nif.NifFormat.ElementReference](#) property), 72
 normalized() ([pyffi.formats.nif.NifFormat.Vector3](#) method), 176
 normals() ([pyffi.formats.nif.NifFormat.DecalVertexArray](#) property), 72
 NULL ([pyffi.formats.nif.NifFormat.Fallout3Layer](#) attribute), 76
 NULL ([pyffi.formats.nif.NifFormat.OblivionLayer](#) attribute), 156
 NULL ([pyffi.formats.nif.NifFormat.SkyrimLayer](#) attribute), 166
 num_1() ([pyffi.formats.nif.NifFormat.Ni3dsAlphaAnimator](#) property), 87
 num_2() ([pyffi.formats.nif.NifFormat.Ni3dsAlphaAnimator](#) property), 87
 num_active() ([pyffi.formats.nif.NifFormat.NiParticlesData](#) property), 135
 num_affected_node_list_pointers() ([pyffi.formats.nif.NifFormat.NiDynamicEffect](#) property), 99
 num_affected_nodes() ([pyffi.formats.nif.NifFormat.NiDynamicEffect](#) property), 99
 num_atoms() ([pyffi.formats.nif.NifFormat.BSPackedAdditionalDataBlock](#) property), 55
 num_bezier_triangles() ([pyffi.formats.nif.NifFormat.NiBezierMesh](#) property), 94
 num_big_tris() ([pyffi.formats.nif.NifFormat.bhkCompressedMeshShape](#) property), 180
 num_big_verts() ([pyffi.formats.nif.NifFormat.bhkCompressedMeshShape](#) property), 180
 num_block_infos() ([pyffi.formats.nif.NifFormat.BSPackedAdditionalGeometryData](#) property), 55
 num_block_infos() ([pyffi.formats.nif.NifFormat.NiAdditionalGeometryData](#) property), 89
 num_blocks() ([pyffi.formats.nif.NifFormat.AdditionalDataBlock](#) property), 44
 num_blocks() ([pyffi.formats.nif.NifFormat.BSPackedAdditionalDataBlock](#) property), 55
 num_blocks() ([pyffi.formats.nif.NifFormat.BSPackedAdditionalGeometryData](#) property), 55
 num_blocks() ([pyffi.formats.nif.NifFormat.NiAdditionalGeometryData](#) property), 89
 num_bones() ([pyffi.formats.nif.NifFormat.bhkRagdollTemplate](#) property), 186
 num_bones() ([pyffi.formats.nif.NifFormat.NiFurSpringController](#) property), 101
 num_bones() ([pyffi.formats.nif.NifFormat.NiSkinInstance](#) property), 144

num_bones () (pyffi.formats.nif.NifFormat.NiSkinningMeshModifier property), 145
 num_bones () (pyffi.formats.nif.NifFormat.NiTriShapeSkinController property), 152
 num_bones_1 () (pyffi.formats.nif.NifFormat.BSTreeNode property), 61
 num_bones_2 () (pyffi.formats.nif.NifFormat.BSTreeNode property), 61
 num_bones_2 () (pyffi.formats.nif.NifFormat.NiFurSpringController property), 101
 num_buttons () (pyffi.formats.nif.NifFormat.FxRadioButton property), 78
 num_bv () (pyffi.formats.nif.NifFormat.UnionBV property), 175
 num_bytes () (pyffi.formats.nif.NifFormat.NiDataStream property), 99
 num_bytes () (pyffi.formats.nif.NifFormat.NiVertWeightsExtraData property), 153
 num_channel_bytes () (pyffi.formats.nif.NifFormat.AdditionalDataInfo property), 44
 num_channel_bytes_per_element () (pyffi.formats.nif.NifFormat.AdditionalDataInfo property), 44
 num_children () (pyffi.formats.nif.NifFormat.NiEnvMappedTriShape property), 100
 num_chunks () (pyffi.formats.nif.NifFormat.bhkCompressedMeshShapeData property), 180
 num_colliders () (pyffi.formats.nif.NifFormat.NiPSSimulatorCollisionStep property), 121
 num_color_keys () (pyffi.formats.nif.NifFormat.NiPSSimulatorGeneralStep property), 122
 num_complete_points () (pyffi.formats.nif.NifFormat.NiMeshModifier property), 108
 num_components () (pyffi.formats.nif.NifFormat.MeshData property), 85
 num_components () (pyffi.formats.nif.NifFormat.NiDataStream property), 99
 num_control_points () (pyffi.formats.nif.NifFormat.NiBSplineBasisData property), 91
 num_controlled_blocks () (pyffi.formats.nif.NifFormat.NiSequence property), 142
 num_controller_sequences () (pyffi.formats.nif.NifFormat.NiControllerManager property), 98
 num_data () (pyffi.formats.nif.NifFormat.AdditionalDataBlock property), 44
 num_datas () (pyffi.formats.nif.NifFormat.NiMesh property), 108
 num_dests () (pyffi.formats.nif.NifFormat.NiPhysXProp property), 137
 num_emitter_meshes () (pyffi.formats.nif.NifFormat.NiPSysMeshEmitter property), 129
 num_entities () (pyffi.formats.nif.NifFormat.SubConstraint property), 172
 num_entries () (pyffi.formats.nif.NifFormat.NiPalette property), 132
 num_extra_bytes () (pyffi.formats.nif.NifFormat.NiFloatExtraDataController property), 101
 num_extra_targets () (pyffi.formats.nif.NifFormat.NiMultiTargetTransformController property), 110
 num_faces () (pyffi.formats.nif.NifFormat.NiPersistentSrcTextureRender property), 136
 num_faces () (pyffi.formats.nif.NifFormat.NiPixelData property), 139
 num_floats () (pyffi.formats.nif.NifFormat.bhkBallSocketConstraintCha property), 177
 num_floats () (pyffi.formats.nif.NifFormat.BSPSysScaleModifier property), 54
 num_floats () (pyffi.formats.nif.NifFormat.NiFloatsExtraData property), 101
 num_forces () (pyffi.formats.nif.NifFormat.NiPSSimulatorForcesStep property), 122
 num_in_portals () (pyffi.formats.nif.NifFormat.NiRoom property), 141
 num_indices () (pyffi.formats.nif.NifFormat.bhkCMSDChunk property), 179
 num_indices () (pyffi.formats.nif.NifFormat.Region property), 161
 num_indices_2 () (pyffi.formats.nif.NifFormat.bhkCMSDChunk property), 179
 num_integers () (pyffi.formats.nif.NifFormat.NiIntegersExtraData property), 105
 num_interpolators () (pyffi.formats.nif.NifFormat.NiGeomMorpherController property), 102
 num_interpolators () (pyffi.formats.nif.NifFormat.NiMorphWeightsController property), 109
 num_items () (pyffi.formats.nif.NifFormat.BSWArray property), 62
 num_items () (pyffi.formats.nif.NifFormat.NiRoom property), 141
 num_joints () (pyffi.formats.nif.NifFormat.NiPhysXPropDesc property), 138
 num_keys () (pyffi.formats.nif.NifFormat.KeyGroup property), 81
 num_keys () (pyffi.formats.nif.NifFormat.Morph property), 138

erty), 86
 num_keys() (pyffi.formats.nif.NifFormat.NiVisData property), 153
 num_link_pairs() (pyffi.formats.nif.NifFormat.SkinShapeGroup property), 163
 num_links() (pyffi.formats.nif.NifFormat.bhkBallSocketConstraintChain property), 178
 num_links_2() (pyffi.formats.nif.NifFormat.bhkBallSocketConstraintChain property), 178
 num_lod_levels() (pyffi.formats.nif.NifFormat.NiLODNode property), 106
 num_lod_levels() (pyffi.formats.nif.NifFormat.NiRangeLODData property), 140
 num_materials() (pyffi.formats.nif.NifFormat.bhkCompressedMeshShapeData property), 180
 num_materials() (pyffi.formats.nif.NifFormat.NiPhysXPromData property), 138
 num_materials() (pyffi.formats.nif.NifFormat.NiRenderObject property), 141
 num_meshes() (pyffi.formats.nif.NifFormat.NiPSysMeshUpdateModifier property), 129
 num_modifiers() (pyffi.formats.nif.NifFormat.NiMesh property), 108
 num_modifiers() (pyffi.formats.nif.NifFormat.NiParticleSystem property), 133
 num_node_groups() (pyffi.formats.nif.NifFormat.NiBoneLODController property), 96
 num_node_groups_2() (pyffi.formats.nif.NifFormat.NiBoneLODController property), 96
 num_nodes() (pyffi.formats.nif.NifFormat.BSPSysHavokUpdateModifier property), 53
 num_nodes() (pyffi.formats.nif.NifFormat.NodeGroup property), 154
 num_objs() (pyffi.formats.nif.NifFormat.NiDefaultAVObjectPalette property), 99
 num_particle_meshes() (pyffi.formats.nif.NifFormat.NiParticleMeshModifier property), 132
 num_particle_systems() (pyffi.formats.nif.NifFormat.BSMasterParticleSystem property), 52
 num_particles() (pyffi.formats.nif.NifFormat.NiParticlesData property), 135
 num_particles() (pyffi.formats.nif.NifFormat.NiParticleSystemComponent property), 133
 num_pixels() (pyffi.formats.nif.NifFormat.NiPersistentSrcTextureResourceData property), 136
 num_pixels() (pyffi.formats.nif.NifFormat.NiPixelData property), 139
 num_polygons() (pyffi.formats.nif.NifFormat.NiScreenElementsData property), 142
 num_portals_2() (pyffi.formats.nif.NifFormat.NiRoom property), 141
 num_positions() (pyffi.formats.nif.NifFormat.BSFurnitureMarker property), 50
 num_regions() (pyffi.formats.nif.NifFormat.NiDataStream property), 99
 num_rooms() (pyffi.formats.nif.NifFormat.NiRoomGroup property), 141
 num_rotations_keys() (pyffi.formats.nif.NifFormat.NiPSSimulatorGeneralStep property), 122
 num_rotation_keys() (pyffi.formats.nif.NifFormat.NiPSSimulatorMeshAlignStep property), 122
 num_shape_data() (pyffi.formats.nif.NifFormat.BSSegmentedTriShape property), 57
 num_shape_groups() (pyffi.formats.nif.NifFormat.NiTexturingProperty property), 149
 num_shape_groups_2() (pyffi.formats.nif.NifFormat.NiBoneLODController property), 96
 num_shape_groups_2() (pyffi.formats.nif.NifFormat.NiBoneLODController property), 96
 num_simulation_steps() (pyffi.formats.nif.NifFormat.NiPSSimulator property), 121
 num_size_keys() (pyffi.formats.nif.NifFormat.NiPSSimulatorGeneralStep property), 122
 num_skin_partition_blocks() (pyffi.formats.nif.NifFormat.NiSkinPartition property), 144
 num_sources() (pyffi.formats.nif.NifFormat.NiFlipController property), 100
 num_spawn_generations() (pyffi.formats.nif.NifFormat.NiPSysSpawnModifier property), 130
 num_strings() (pyffi.formats.nif.NifFormat.NiStringsExtraData property), 147
 num_strips() (pyffi.formats.nif.NifFormat.bhkCMSDChunk property), 179
 num_strips_data() (pyffi.formats.nif.NifFormat.bhkMeshShape property), 182
 num_sub_shapes() (pyffi.formats.nif.NifFormat.bhkConvexListShape property), 181
 num_submeshes() (pyffi.formats.nif.NifFormat.MeshData property), 108
 num_submeshes() (pyffi.formats.nif.NifFormat.NiMesh property), 108
 num_submit_points() (pyffi.formats.nif.NifFormat.NiMeshModifier property), 108
 num_subtexture_offset_u_vs()

(pyffi.formats.nif.NifFormat.NiPSysData
 property), 125
 num_targets() (pyffi.formats.nif.NifFormat.NiMorphMeshModifier
 property), 109
 num_targets() (pyffi.formats.nif.NifFormat.NiMorphWeightsController
 property), 109
 num_text_keys() (pyffi.formats.nif.NifFormat.NiTextKeyExtraData
 property), 147
 num_textures() (pyffi.formats.nif.NifFormat.BSShaderTextureSet
 property), 60
 num_textures() (pyffi.formats.nif.NifFormat.NiArkTextureExtraData
 property), 90
 num_total_bytes() (pyffi.formats.nif.NifFormat.BSPackedAdditionalDataBlock
 property), 55
 num_total_bytes_per_element() (pyffi.formats.nif.NifFormat.AdditionalDataInfo
 property), 44
 num_total_bytes_per_element() (pyffi.formats.nif.NifFormat.BSPackedAdditionalDataBlock
 property), 55
 num_transforms() (pyffi.formats.nif.NifFormat.bhkCompressedMeshShapeData
 property), 180
 num_tread_transforms() (pyffi.formats.nif.NifFormat.BSTreadTransfInterpolator
 property), 61
 num_triangles() (pyffi.formats.nif.NifFormat.Polygon
 property), 158
 num_unknown_bytes_2() (pyffi.formats.nif.NifFormat.NiBinaryVoxelData
 property), 95
 num_unknown_floats() (pyffi.formats.nif.NifFormat.bhkMeshShape
 property), 182
 num_unknown_ints() (pyffi.formats.nif.NifFormat.NiGeomMorpherController
 property), 102
 num_unknown_ints_1() (pyffi.formats.nif.NifFormat.NiMeshPSysData
 property), 109
 num_unknown_links_1() (pyffi.formats.nif.NifFormat.NiShadowGenerator
 property), 143
 num_unknown_vectors() (pyffi.formats.nif.NifFormat.NiBinaryVoxelData
 property), 95
 num_uv_quadrants() (pyffi.formats.nif.NifFormat.NiParticlesData
 property), 135
 num_valid() (pyffi.formats.nif.NifFormat.NiParticleSystemController
 property), 133
 num_vector_blocks() (pyffi.formats.nif.NifFormat.BSDecalPlacementVectorExtraData
 property), 46
 num_vectors() (pyffi.formats.nif.NifFormat.DecalVectorArray
 property), 72
 num_vertices() (pyffi.formats.nif.NifFormat.bhkCMSDChunk
 property), 179
 num_vertices() (pyffi.formats.nif.NifFormat.BSPackedAdditionalGeom
 property), 55
 num_vertices() (pyffi.formats.nif.NifFormat.MatchGroup
 property), 83
 num_vertices() (pyffi.formats.nif.NifFormat.NiAdditionalGeometryDa
 property), 89
 num_vertices() (pyffi.formats.nif.NifFormat.NiPhysXMeshDesc
 property), 137
 num_vertices() (pyffi.formats.nif.NifFormat.NiPortal
 property), 140
 num_vertices() (pyffi.formats.nif.NifFormat.NiVertWeightsExtraData
 property), 153
 num_vertices() (pyffi.formats.nif.NifFormat.OblivionSubShape
 property), 157
 num_vertices() (pyffi.formats.nif.NifFormat.Polygon
 property), 158
 num_visibility_keys() (pyffi.formats.nif.NifFormat.NiPSysEmitterCtrlData
 property), 126
 num_walls() (pyffi.formats.nif.NifFormat.NiRoom
 property), 141
 number() (pyffi.formats.nif.NifFormat.physXMaterialRef
 property), 189

O

object_palette() (pyffi.formats.nif.NifFormat.NiControllerManager
 property), 98
 objs() (pyffi.formats.nif.NifFormat.NiDefaultAVObjectPalette
 property), 99
 offset() (pyffi.formats.nif.NifFormat.FurniturePosition
 property), 78
 offset() (pyffi.formats.nif.NifFormat.MipMap prop-
 erty), 85
 offset() (pyffi.formats.nif.NifFormat.NiBSplineCompFloatInterpolator
 property), 91
 old_emit_rate() (pyffi.formats.nif.NifFormat.NiParticleSystemControl
 property), 133
 old_speed() (pyffi.formats.nif.NifFormat.NiParticleSystemController
 property), 133
 only_regexs (pyffi.spells.Toaster attribute), 233
 openfile() (pyffi.object_models.MetaFileFormat
 static method), 233
 operation() (pyffi.formats.nif.NifFormat.NiTextureTransformController
 property), 148
 options (pyffi.spells.Toaster attribute), 233
 operation() (pyffi.formats.nif.NifFormat.NiPSysModifier
 property), 129
 orientation() (pyffi.formats.nif.NifFormat.FurniturePosition
 property), 78

origin() (pyffi.formats.nif.NifFormat.CapsuleBV property), 65
 OTHER (pyffi.formats.nif.NifFormat.Fallout3Layer attribute), 76
 OTHER (pyffi.formats.nif.NifFormat.OblivionLayer attribute), 156
P
 PACK (pyffi.formats.nif.NifFormat.Fallout3Layer attribute), 76
 palette() (pyffi.formats.nif.NifFormat.NiPalette property), 132
 palette() (pyffi.formats.nif.NifFormat.NiStringPalette property), 146
 parallax_envmap_strength() (pyffi.formats.nif.NifFormat.BSLightingShaderProperty property), 51
 parallax_inner_layer_texture_scale() (pyffi.formats.nif.NifFormat.BSLightingShaderProperty property), 51
 parallax_inner_layer_thickness() (pyffi.formats.nif.NifFormat.BSLightingShaderProperty property), 51
 parallax_refraction_scale() (pyffi.formats.nif.NifFormat.BSLightingShaderProperty property), 51
 parent() (pyffi.formats.nif.NifFormat.Ni3dsAlphaAnimator property), 87
 parent() (pyffi.formats.nif.NifFormat.NiPSysCollider property), 125
 parent() (pyffi.formats.nif.NifFormat.NiPSysDragModifier property), 126
 parse_inifile() (pyffi.spells.Toaster static method), 233
 parse_mopp() (pyffi.formats.nif.NifFormat.bhkMoppBvTreeShape method), 183
 part_flag() (pyffi.formats.nif.NifFormat.BodyPartList property), 63
 particle_descriptions() (pyffi.formats.nif.NifFormat.NiPSysData property), 125
 particle_extra() (pyffi.formats.nif.NifFormat.NiParticleSystemController property), 134
 particle_lifetime() (pyffi.formats.nif.NifFormat.NiParticleSystemController property), 134
 particle_link() (pyffi.formats.nif.NifFormat.NiParticleSystemController property), 134
 particle_meshes() (pyffi.formats.nif.NifFormat.NiParticleMeshModifier property), 132
 particle_radius() (pyffi.formats.nif.NifFormat.NiParticlesData property), 135
 particle_systems() (pyffi.formats.nif.NifFormat.BSMasterParticleSystem property), 52
 particle_timestamp() (pyffi.formats.nif.NifFormat.NiParticleSystemController property), 134
 particle_unknown_short() (pyffi.formats.nif.NifFormat.NiParticleSystemController property), 134
 particle_unknown_vector() (pyffi.formats.nif.NifFormat.NiParticleSystemController property), 134
 particle_velocity() (pyffi.formats.nif.NifFormat.NiParticleSystemController property), 134
 particle_vertex_id() (pyffi.formats.nif.NifFormat.NiParticleSystemController property), 134
 particles() (pyffi.formats.nif.NifFormat.NiParticleSystemController property), 134
 pass_action() (pyffi.formats.nif.NifFormat.NiStencilProperty property), 146
 PATH_PICK (pyffi.formats.nif.NifFormat.OblivionLayer attribute), 156
 PATH_PICK (pyffi.formats.nif.NifFormat.Fallout3Layer attribute), 76
 PATH_PICK (pyffi.formats.nif.NifFormat.SkyrimLayer attribute), 166
 percentage() (pyffi.formats.nif.NifFormat.NiPSysDragModifier property), 126
 percentage_spawned() (pyffi.formats.nif.NifFormat.NiPSysSpawnModifier property), 130
 perp_2_axle_in_a_1() (pyffi.formats.nif.NifFormat.HingeDescriptor property), 80
 perp_2_axle_in_a_2() (pyffi.formats.nif.NifFormat.HingeDescriptor property), 80
 perp_2_axle_in_b_1() (pyffi.formats.nif.NifFormat.HingeDescriptor property), 80
 perp_2_axle_in_b_2() (pyffi.formats.nif.NifFormat.HingeDescriptor property), 80
 persist_render_data() (pyffi.formats.nif.NifFormat.NiSourceTexture property), 145
 pf_editor_visible() (pyffi.formats.nif.NifFormat.BSPartFlag property), 56
 pf_start_net_boneset() (pyffi.formats.nif.NifFormat.BSPartFlag property), 56

phase() (pyffi.formats.nif.NifFormat.NiTimeController property), 158
 plane_b() (pyffi.formats.nif.NifFormat.PrismaticDescriptor property), 158
 pivot_a() (pyffi.formats.nif.NifFormat.HingeDescriptor property), 80
 pivot_a() (pyffi.formats.nif.NifFormat.PrismaticDescriptor property), 158
 pivot_a() (pyffi.formats.nif.NifFormat.StiffSpringDescriptor property), 96
 pivot_b() (pyffi.formats.nif.NifFormat.HingeDescriptor property), 80
 pivot_b() (pyffi.formats.nif.NifFormat.PrismaticDescriptor property), 158
 pivot_b() (pyffi.formats.nif.NifFormat.StiffSpringDescriptor property), 170
 PIX_LAY_BUMPMAP (pyffi.formats.nif.NifFormat.PixelLayout attribute), 158
 PIX_LAY_COMPRESSED (pyffi.formats.nif.NifFormat.PixelLayout attribute), 158
 PIX_LAY_DEFAULT (pyffi.formats.nif.NifFormat.PixelLayout attribute), 158
 PIX_LAY_HIGH_COLOR_16 (pyffi.formats.nif.NifFormat.PixelLayout attribute), 158
 PIX_LAY_PALETTISED (pyffi.formats.nif.NifFormat.PixelLayout attribute), 158
 PIX_LAY_PALETTISED_4 (pyffi.formats.nif.NifFormat.PixelLayout attribute), 158
 PIX_LAY_TRUE_COLOR_32 (pyffi.formats.nif.NifFormat.PixelLayout attribute), 158
 pixel_data() (pyffi.formats.nif.NifFormat.NiPersistentSrcTextureData property), 136
 pixel_data() (pyffi.formats.nif.NifFormat.NiPixelData property), 139
 pixel_data() (pyffi.formats.nif.NifFormat.NiSourceTexture property), 145
 pixel_data() (pyffi.formats.nif.NifFormat.TexSource property), 174
 pixel_layout() (pyffi.formats.nif.NifFormat.NiSourceTexture property), 145
 PixelData (pyffi.formats.dds.DdsFormat attribute), 25
 PixelData (pyffi.formats.tga.TgaFormat attribute), 197
 planar_angle() (pyffi.formats.nif.NifFormat.NiPSysEmitter property), 126
 planar_angle_variation() (pyffi.formats.nif.NifFormat.NiPSysEmitter property), 126
 PLANAR_SYMMETRY (pyffi.formats.nif.NifFormat.SymmetryType attribute), 172
 plane_a() (pyffi.formats.nif.NifFormat.PrismaticDescriptor property), 158
 plane_b() (pyffi.formats.nif.NifFormat.PrismaticDescriptor property), 158
 point_3_value() (pyffi.formats.nif.NifFormat.NiPoint3Interpolator property), 140
 point_value() (pyffi.formats.nif.NifFormat.NiBlendPoint3Interpolator property), 96
 points() (pyffi.formats.nif.NifFormat.DecalVectorArray property), 72
 points_1() (pyffi.formats.nif.NifFormat.NiBezierMesh property), 94
 points_2() (pyffi.formats.nif.NifFormat.NiBezierMesh property), 94
 polygon_indices() (pyffi.formats.nif.NifFormat.NiScreenElementsData property), 142
 polygons() (pyffi.formats.nif.NifFormat.NiScreenElementsData property), 142
 PONYTAIL (pyffi.formats.nif.NifFormat.Fallout3Layer attribute), 76
 PONYTAIL (pyffi.formats.nif.NifFormat.OblivionLayer attribute), 156
 PORTAL (pyffi.formats.nif.NifFormat.Fallout3Layer attribute), 76
 PORTAL (pyffi.formats.nif.NifFormat.OblivionLayer attribute), 156
 PORTAL (pyffi.formats.nif.NifFormat.SkyrimLayer attribute), 166
 portals_2() (pyffi.formats.nif.NifFormat.NiRoom property), 141
 pos_data() (pyffi.formats.nif.NifFormat.NiPathController property), 135
 pos_data() (pyffi.formats.nif.NifFormat.NiPathInterpolator property), 135
 position() (pyffi.formats.nif.NifFormat.BSMultiBoundAABB property), 52
 position() (pyffi.formats.nif.NifFormat.NiGravity property), 105
 position() (pyffi.formats.nif.NifFormat.NiParticleBomb property), 132
 position_ref_1() (pyffi.formats.nif.NifFormat.FurniturePosition property), 78
 position_ref_2() (pyffi.formats.nif.NifFormat.FurniturePosition property), 78
 positions() (pyffi.formats.nif.NifFormat.BSFurnitureMarker property), 50
 primitive_type() (pyffi.formats.nif.NifFormat.NiMesh property), 108
 priority() (pyffi.formats.nif.NifFormat.bhkRDTConstraint property), 185
 priority() (pyffi.formats.nif.NifFormat.bhkRDTMalleableConstraint property), 185
 priority() (pyffi.formats.nif.NifFormat.SubConstraint property), 172

Prismatic (*pyffi.formats.nif.NifFormat.hkConstraintType* *ps_2_1* () (*pyffi.formats.nif.NifFormat.NiTextureEffect* *attribute*), 188 (*property*), 147
 prismatic () (*pyffi.formats.nif.NifFormat.bhkPrismaticConstraint* (*pyffi.formats.nif.NifFormat.NiTextureModeProperty* *property*), 185 (*property*), 148
 prismatic () (*pyffi.formats.nif.NifFormat.SubConstraint* *ps_2_1* () (*pyffi.formats.nif.NifFormat.TexDesc* *property*), 172 (*property*), 174
 PROJECTILE (*pyffi.formats.nif.NifFormat.Fallout3Layer* *PS_LOOP_AGESCALE* (*pyffi.formats.nif.NifFormat.PSLoopBehavior* *attribute*), 76 (*attribute*), 157
 PROJECTILE (*pyffi.formats.nif.NifFormat.OblivionLayer* *PS_LOOP_CLAMP_BIRTH* (*pyffi.formats.nif.NifFormat.PSLoopBehavior* *attribute*), 156 (*attribute*), 157
 PROJECTILE (*pyffi.formats.nif.NifFormat.SkyrimLayer* *PS_LOOP_CLAMP_DEATH* (*pyffi.formats.nif.NifFormat.PSLoopBehavior* *attribute*), 166 (*attribute*), 157
 PROJECTILEZONE (*pyffi.formats.nif.NifFormat.Fallout3Layer* (*pyffi.formats.nif.NifFormat.PSLoopBehavior* *attribute*), 76 (*attribute*), 157
 PROJECTILEZONE (*pyffi.formats.nif.NifFormat.SkyrimLayer* *PS_LOOP_LOOP* (*pyffi.formats.nif.NifFormat.PSLoopBehavior* *attribute*), 167 (*attribute*), 157
 prop_description () *PS_LOOP_REFLECT* (*pyffi.formats.nif.NifFormat.PSLoopBehavior* *attribute*), 157
 (*pyffi.formats.nif.NifFormat.NiPhysXProp* *PX_FMT_DXT1* (*pyffi.formats.nif.NifFormat.PixelFormat* *attribute*), 137 (*attribute*), 158
 PROPAGATE_ALWAYS (*pyffi.formats.nif.NifFormat.PropagationMode* *PX_FMT_DXT5* (*pyffi.formats.nif.NifFormat.PixelFormat* *attribute*), 159 (*attribute*), 158
 PROPAGATE_NEVER (*pyffi.formats.nif.NifFormat.PropagationMode* *attribute*), 159 (*attribute*), 158
 PROPAGATE_ON_FAILURE *PX_FMT_DXT5_ALT* (*pyffi.formats.nif.NifFormat.PixelFormat* *attribute*), 158
 (*pyffi.formats.nif.NifFormat.PropagationMode* *PX_FMT_PAL8* (*pyffi.formats.nif.NifFormat.PixelFormat* *attribute*), 159 (*attribute*), 158
 PROPAGATE_ON_SUCCESS *PX_FMT_RGB8* (*pyffi.formats.nif.NifFormat.PixelFormat* *attribute*), 158
 (*pyffi.formats.nif.NifFormat.PropagationMode* *attribute*), 159 (*attribute*), 158
 propagation_mode () *PX_FMT_RGBA8* (*pyffi.formats.nif.NifFormat.PixelFormat* *attribute*), 158
 (*pyffi.formats.nif.NifFormat.NiCollisionData* *PyFFI*, 267
property), 98 *pyffi* (*module*), 7
 proportion_count () *pyffi.formats* (*module*), 8
 (*pyffi.formats.nif.NifFormat.NiScreenLODData* *pyffi.formats.bsa* (*module*), 8
property), 142 *pyffi.formats.cgf* (*module*), 11
 proportion_levels () *pyffi.formats.dae* (*module*), 22
 (*pyffi.formats.nif.NifFormat.NiScreenLODData* *pyffi.formats.dds* (*module*), 24
property), 142 *pyffi.formats.egm* (*module*), 27
 PROPS (*pyffi.formats.nif.NifFormat.Fallout3Layer* *pyffi.formats.egt* (*module*), 31
attribute), 76 *pyffi.formats.esp* (*module*), 34
 PROPS (*pyffi.formats.nif.NifFormat.OblivionLayer* *pyffi.formats.kfm* (*module*), 38
attribute), 156 *pyffi.formats.nif* (*module*), 43
 PROPS (*pyffi.formats.nif.NifFormat.SkyrimLayer* *pyffi.formats.tga* (*module*), 196
attribute), 167 *pyffi.formats.tri* (*module*), 198
 ps_2_k () (*pyffi.formats.nif.NifFormat.MultiTextureElement* *pyffi.object_models* (*module*), 233
property), 87 *pyffi.spells* (*module*), 205
 ps_2_k () (*pyffi.formats.nif.NifFormat.NiTextureEffect* *pyffi.spells.cgf* (*module*), 205
property), 147 *pyffi.spells.dds* (*module*), 205
 ps_2_k () (*pyffi.formats.nif.NifFormat.NiTextureModeProperty* *pyffi.spells.kfm* (*module*), 205
property), 148 *pyffi.spells.nif* (*module*), 205
 ps_2_k () (*pyffi.formats.nif.NifFormat.TexDesc* *property*), 174 *pyffi.spells.nif.check* (*module*), 205
property), 174 *pyffi.spells.nif.dump* (*module*), 205
 ps_2_1 () (*pyffi.formats.nif.NifFormat.MultiTextureElement* *pyffi.spells.nif.fix* (*module*), 205
property), 87 *pyffi.spells.nif.modify* (*module*), 216

`pyffi.spells.nif.optimize (module)`, 214
`pyffi.spells.tga (module)`, 226

Q

`quadratic_attenuation()`
 (`pyffi.formats.nif.NifFormat.NiPointLight`
 property), 140
`QUADRATIC_KEY` (`pyffi.formats.nif.NifFormat.KeyType`
 attribute), 81
`QUIVER` (`pyffi.formats.nif.NifFormat.Fallout3Layer` at-
 tribute), 76
`QUIVER` (`pyffi.formats.nif.NifFormat.OblivionLayer` at-
 tribute), 156

R

`r()` (`pyffi.formats.nif.NifFormat.ByteColor3` property),
 64
`r()` (`pyffi.formats.nif.NifFormat.ByteColor4` property),
 64
`r()` (`pyffi.formats.nif.NifFormat.Color3` property), 66
`r()` (`pyffi.formats.nif.NifFormat.Color4` property), 66
`R_CALF` (`pyffi.formats.nif.NifFormat.Fallout3Layer` at-
 tribute), 76
`R_CALF` (`pyffi.formats.nif.NifFormat.OblivionLayer` at-
 tribute), 156
`R_FOOT` (`pyffi.formats.nif.NifFormat.Fallout3Layer` at-
 tribute), 76
`R_FOOT` (`pyffi.formats.nif.NifFormat.OblivionLayer` at-
 tribute), 156
`R_FORE_ARM` (`pyffi.formats.nif.NifFormat.Fallout3Layer`
 attribute), 76
`R_FOREARM` (`pyffi.formats.nif.NifFormat.OblivionLayer`
 attribute), 156
`R_HAND` (`pyffi.formats.nif.NifFormat.Fallout3Layer` at-
 tribute), 76
`R_HAND` (`pyffi.formats.nif.NifFormat.OblivionLayer` at-
 tribute), 156
`R_THIGH` (`pyffi.formats.nif.NifFormat.Fallout3Layer` at-
 tribute), 76
`R_THIGH` (`pyffi.formats.nif.NifFormat.OblivionLayer` at-
 tribute), 156
`R_UPPER_ARM` (`pyffi.formats.nif.NifFormat.Fallout3Layer`
 attribute), 76
`R_UPPER_ARM` (`pyffi.formats.nif.NifFormat.OblivionLayer`
 attribute), 156
`radial_type()` (`pyffi.formats.nif.NifFormat.NiPSysRadialFieldModifier`
 property), 130
`radii()` (`pyffi.formats.nif.NifFormat.NiParticlesData`
 property), 135
`radius()` (`pyffi.formats.nif.NifFormat.bhkCompressedMeshShape`
 property), 180
`radius()` (`pyffi.formats.nif.NifFormat.bhkRagdollTemplateData`
 property), 186

`radius()` (`pyffi.formats.nif.NifFormat.bhkSphereRepShape`
 property), 187
`radius()` (`pyffi.formats.nif.NifFormat.BoundingBox`
 property), 63
`radius()` (`pyffi.formats.nif.NifFormat.BSMultiBoundSphere`
 property), 53
`radius()` (`pyffi.formats.nif.NifFormat.NiPSysCylinderEmitter`
 property), 125
`radius()` (`pyffi.formats.nif.NifFormat.NiPSysSphereEmitter`
 property), 131
`radius()` (`pyffi.formats.nif.NifFormat.NiPSysSphericalCollider`
 property), 131
`radius()` (`pyffi.formats.nif.NifFormat.SphereBV` prop-
 erty), 170
`radius_variation()`
 (`pyffi.formats.nif.NifFormat.NiPSysEmitter`
 property), 126
`Ragdoll` (`pyffi.formats.nif.NifFormat.hkConstraintType`
 attribute), 188
`ragdoll()` (`pyffi.formats.nif.NifFormat.bhkRDTConstraint`
 property), 185
`ragdoll()` (`pyffi.formats.nif.NifFormat.bhkRDTMalleableConstraint`
 property), 185
`ragdoll()` (`pyffi.formats.nif.NifFormat.SubConstraint`
 property), 172
`random_initial_axis()`
 (`pyffi.formats.nif.NifFormat.NiParticleRotation`
 property), 133
`random_initial_axis()`
 (`pyffi.formats.nif.NifFormat.NiPSysRotationModifier`
 property), 130
`random_rot_speed_sign()`
 (`pyffi.formats.nif.NifFormat.NiPSysRotationModifier`
 property), 130
`range()` (`pyffi.formats.nif.NifFormat.NiPSysDragModifier`
 property), 126
`range_falloff()` (`pyffi.formats.nif.NifFormat.NiPSysDragModifier`
 property), 126
`RE_FILENAME` (`pyffi.formats.nif.NifFormat` attribute),
 160
`RE_FILENAME` (`pyffi.object_models.FileFormat` at-
 tribute), 234
`read()` (`pyffi.formats.bsa.BsaFormat.BZString`
 method), 8
`read()` (`pyffi.formats.bsa.BsaFormat.FileVersion`
 method), 8
`read()` (`pyffi.formats.bsa.BsaFormat.Header` method),
 9
`read()` (`pyffi.formats.bsa.BsaFormat.ZString` method),
 10
`read()` (`pyffi.formats.cgf.CgfFormat.Data` method), 13
`read()` (`pyffi.formats.cgf.CgfFormat.FileSignature`
 method), 14
`read()` (`pyffi.formats.cgf.CgfFormat.Ref` method), 17

`read()` (`pyffi.formats.cgf.CgfFormat.SizedString` method), 18
`read()` (`pyffi.formats.dae.DaeFormat.Data` method), 23
`read()` (`pyffi.formats.dds.DdsFormat.Data` method), 25
`read()` (`pyffi.formats.dds.DdsFormat.HeaderString` method), 25
`read()` (`pyffi.formats.egm.EgmFormat.Data` method), 28
`read()` (`pyffi.formats.egm.EgmFormat.FileSignature` method), 28
`read()` (`pyffi.formats.egm.EgmFormat.FileVersion` method), 29
`read()` (`pyffi.formats.egt.EgtFormat.FileSignature` method), 32
`read()` (`pyffi.formats.egt.EgtFormat.FileVersion` method), 32
`read()` (`pyffi.formats.egt.EgtFormat.Header` method), 33
`read()` (`pyffi.formats.esp.EspFormat.Data` method), 35
`read()` (`pyffi.formats.esp.EspFormat.GRUP` method), 36
`read()` (`pyffi.formats.esp.EspFormat.Record` method), 36
`read()` (`pyffi.formats.esp.EspFormat.ZString` method), 37
`read()` (`pyffi.formats.kfm.KfmFormat.Data` method), 39
`read()` (`pyffi.formats.kfm.KfmFormat.HeaderString` method), 39
`read()` (`pyffi.formats.kfm.KfmFormat.SizedString` method), 41
`read()` (`pyffi.formats.nif.NifFormat.bool` method), 188
`read()` (`pyffi.formats.nif.NifFormat.ByteArray` method), 64
`read()` (`pyffi.formats.nif.NifFormat.ByteMatrix` method), 65
`read()` (`pyffi.formats.nif.NifFormat.Data` method), 71
`read()` (`pyffi.formats.nif.NifFormat.FileVersion` method), 77
`read()` (`pyffi.formats.nif.NifFormat.Footer` method), 77
`read()` (`pyffi.formats.nif.NifFormat.HeaderString` method), 79
`read()` (`pyffi.formats.nif.NifFormat.LineString` method), 82
`read()` (`pyffi.formats.nif.NifFormat.Ref` method), 161
`read()` (`pyffi.formats.nif.NifFormat.ShortString` method), 162
`read()` (`pyffi.formats.nif.NifFormat.SizedString` method), 163
`read()` (`pyffi.formats.nif.NifFormat.string` method), 189
`read()` (`pyffi.formats.tga.TgaFormat.Data` method), 196
`read()` (`pyffi.formats.tga.TgaFormat.FooterString` method), 196
`read()` (`pyffi.formats.tri.TriFormat.FileSignature` method), 199
`read()` (`pyffi.formats.tri.TriFormat.FileVersion` method), 199
`read()` (`pyffi.formats.tri.TriFormat.Header` method), 200
`read()` (`pyffi.formats.tri.TriFormat.SizedStringZ` method), 201
`read()` (`pyffi.object_models.FileFormat.Data` method), 234
`READONLY` (`pyffi.spells.Spell` attribute), 227
`recurse()` (`pyffi.spells.Spell` method), 228
`recurse()` (`pyffi.spells.SpellGroupSeriesBase` method), 231
`refraction_fire_period()` (`pyffi.formats.nif.NifFormat.BSShaderPPLightingProperty` property), 59
`refraction_strength()` (`pyffi.formats.nif.NifFormat.BSLightingShaderProperty` property), 51
`refraction_strength()` (`pyffi.formats.nif.NifFormat.BSShaderPPLightingProperty` property), 59
`regions()` (`pyffi.formats.nif.NifFormat.NiDataStream` property), 99
`remove_child()` (`pyffi.formats.nif.NifFormat.NiNode` method), 112
`remove_effect()` (`pyffi.formats.nif.NifFormat.NiNode` method), 112
`remove_extra_data()` (`pyffi.formats.nif.NifFormat.NiObjectNET` method), 113
`remove_if_broken()` (`pyffi.formats.nif.NifFormat.bhkBreakableConstraint` property), 178
`remove_property()` (`pyffi.formats.nif.NifFormat.NiAVObject` method), 89
`remove_shape()` (`pyffi.formats.nif.NifFormat.bhkListShape` method), 182
`replace_global_node()` (`pyffi.formats.cgf.CgfFormat.Data` method), 13
`replace_global_node()` (`pyffi.formats.nif.NifFormat.Data` method), 71
`replace_global_node()` (`pyffi.formats.nif.NifFormat.Ptr` method), 159
`replace_global_node()` (`pyffi.formats.nif.NifFormat.Ref` method), 161
`reserved_bits_0()` (`pyffi.formats.nif.NifFormat.BSSegmentFlags` property), 57
`reserved_bits_1()`

(pyffi.formats.nif.NifFormat.BSPartFlag property), 56
 RESPONSE_INVALID (pyffi.formats.nif.NifFormat.hkResponseType attribute), 189
 RESPONSE_NONE (pyffi.formats.nif.NifFormat.hkResponseType attribute), 189
 RESPONSE_REPORTING (pyffi.formats.nif.NifFormat.hkResponseType attribute), 189
 RESPONSE_SIMPLE_CONTACT (pyffi.formats.nif.NifFormat.hkResponseType attribute), 189
 restitution() (pyffi.formats.nif.NifFormat.bhkRagdollTemplateData property), 186
 RGB (pyffi.formats.nif.NifFormat.ImageType attribute), 80
 rgb_image_data() (pyffi.formats.nif.NifFormat.NiRawImageData property), 140
 RGBA (pyffi.formats.nif.NifFormat.ImageType attribute), 80
 rgba_image_data() (pyffi.formats.nif.NifFormat.NiRawImageData property), 140
 right() (pyffi.formats.nif.NifFormat.FurnitureEntryPoints property), 78
 right_eye_reflection_center() (pyffi.formats.nif.NifFormat.BSLightingShaderProperty property), 51
 RIGID_FACE_CAMERA (pyffi.formats.nif.NifFormat.BillboardMode attribute), 62
 RIGID_FACE_CENTER (pyffi.formats.nif.NifFormat.BillboardMode attribute), 62
 rooms() (pyffi.formats.nif.NifFormat.NiRoomGroup property), 141
 root() (pyffi.formats.nif.NifFormat.CStreamableAssetData property), 65
 ROTATE_ABOUT_UP (pyffi.formats.nif.NifFormat.BillboardMode attribute), 62
 ROTATE_ABOUT_UP2 (pyffi.formats.nif.NifFormat.BillboardMode attribute), 62
 rotation() (pyffi.formats.nif.NifFormat.bhkCMSDTransform property), 179
 rotation() (pyffi.formats.nif.NifFormat.BoundingBox property), 63
 rotation() (pyffi.formats.nif.NifFormat.BSMultiBoundOBBS property), 53
 rotation() (pyffi.formats.nif.NifFormat.BSTreadTransformData property), 61
 rotation() (pyffi.formats.nif.NifFormat.MTransform property), 83
 rotation() (pyffi.formats.nif.NifFormat.NiLookAtInterpolator property), 107
 rotation() (pyffi.formats.nif.NifFormat.QTransform property), 159
 rotation_a() (pyffi.formats.nif.NifFormat.PrismaticDescriptor property), 158
 rotation_angles() (pyffi.formats.nif.NifFormat.NiParticlesData property), 135
 rotation_axes() (pyffi.formats.nif.NifFormat.NiParticlesData property), 135
 rotation_b() (pyffi.formats.nif.NifFormat.PrismaticDescriptor property), 159
 rotation_keys() (pyffi.formats.nif.NifFormat.NiPSSimulatorGeneralStep property), 122
 rotation_keys() (pyffi.formats.nif.NifFormat.NiPSSimulatorMeshAlignStep property), 122
 rotation_loop_behavior() (pyffi.formats.nif.NifFormat.NiPSSimulatorGeneralStep property), 122
 rotation_loop_behavior() (pyffi.formats.nif.NifFormat.NiPSSimulatorMeshAlignStep property), 122
 rotation_matrix_a() (pyffi.formats.nif.NifFormat.PrismaticDescriptor property), 159
 rotation_speed() (pyffi.formats.nif.NifFormat.NiParticleRotation property), 133
 rotation_x() (pyffi.formats.nif.NifFormat.BSInvMarker property), 50
 rotation_y() (pyffi.formats.nif.NifFormat.BSInvMarker property), 50
 rotation_z() (pyffi.formats.nif.NifFormat.BSInvMarker property), 50
 rotations() (pyffi.formats.nif.NifFormat.NiParticlesData property), 135
 rotations_2() (pyffi.formats.nif.NifFormat.NiRotatingParticlesData property), 141
 save_as_dds() (pyffi.formats.nif.NifFormat.ATextureRenderData method), 43
 SBP_130_HEAD (pyffi.formats.nif.NifFormat.BSDismemberBodyPartType attribute), 47
 SBP_131_HAIR (pyffi.formats.nif.NifFormat.BSDismemberBodyPartType attribute), 48
 SBP_141_LONGHAIR (pyffi.formats.nif.NifFormat.BSDismemberBodyPartType attribute), 48
 SBP_142_CIRCLET (pyffi.formats.nif.NifFormat.BSDismemberBodyPartType attribute), 48
 SBP_143_EARS (pyffi.formats.nif.NifFormat.BSDismemberBodyPartType attribute), 48
 SBP_150_DECAPITATEDHEAD (pyffi.formats.nif.NifFormat.BSDismemberBodyPartType attribute), 48

112

`set_children()` (`pyffi.formats.nif.NifFormat.NiNode` method), 112

`set_controller_type()` (`pyffi.formats.nif.NifFormat.ControllerLink` method), 69

`set_effects()` (`pyffi.formats.nif.NifFormat.NiNode` method), 113

`set_extra_datas()` (`pyffi.formats.nif.NifFormat.NiObjectNET` method), 114

`set_identity()` (`pyffi.formats.cgf.CgfFormat.Matrix33` method), 15

`set_identity()` (`pyffi.formats.cgf.CgfFormat.Matrix44` method), 16

`set_identity()` (`pyffi.formats.nif.NifFormat.InertiaMatrix` method), 81

`set_identity()` (`pyffi.formats.nif.NifFormat.Matrix33` method), 84

`set_identity()` (`pyffi.formats.nif.NifFormat.Matrix44` method), 84

`set_matrix_33()` (`pyffi.formats.cgf.CgfFormat.Matrix44` method), 16

`set_matrix_33()` (`pyffi.formats.nif.NifFormat.Matrix44` method), 84

`set_node_name()` (`pyffi.formats.nif.NifFormat.ControllerLink` method), 69

`set_properties()` (`pyffi.formats.nif.NifFormat.NiAVObject` method), 89

`set_property_type()` (`pyffi.formats.nif.NifFormat.ControllerLink` method), 69

`set_rows()` (`pyffi.formats.cgf.CgfFormat.Matrix44` method), 16

`set_rows()` (`pyffi.formats.nif.NifFormat.Matrix44` method), 84

`set_scale_rotation()` (`pyffi.formats.cgf.CgfFormat.Matrix33` method), 15

`set_scale_rotation()` (`pyffi.formats.nif.NifFormat.Matrix33` method), 84

`set_scale_rotation_translation()` (`pyffi.formats.nif.NifFormat.Matrix44` method), 85

`set_skin_partition()` (`pyffi.formats.nif.NifFormat.NiGeometry` method), 103

`set_strips()` (`pyffi.formats.nif.NifFormat.NiTriShapeData` method), 152

`set_strips()` (`pyffi.formats.nif.NifFormat.NiTriStripsData` method), 152

`set_target_color()` (`pyffi.formats.nif.NifFormat.NiMaterialColorControl` method), 107

`set_transform()` (`pyffi.formats.nif.NifFormat.NiAVObject` method), 89

`set_transform()` (`pyffi.formats.nif.NifFormat.NiSkinData` method), 144

`set_transform()` (`pyffi.formats.nif.NifFormat.SkinData` method), 163

`set_transform()` (`pyffi.formats.nif.NifFormat.SkinTransform` method), 163

`set_translation()` (`pyffi.formats.cgf.CgfFormat.Matrix44` method), 16

`set_translation()` (`pyffi.formats.nif.NifFormat.Matrix44` method), 85

`set_triangles()` (`pyffi.formats.nif.NifFormat.NiTriShapeData` method), 152

`set_triangles()` (`pyffi.formats.nif.NifFormat.NiTriStripsData` method), 152

`set_value()` (`pyffi.formats.bsa.BsaFormat.ZString` method), 10

`set_value()` (`pyffi.formats.cgf.CgfFormat.FileSignature` method), 14

`set_value()` (`pyffi.formats.cgf.CgfFormat.Ref` method), 17

`set_value()` (`pyffi.formats.cgf.CgfFormat.SizedString` method), 18

`set_value()` (`pyffi.formats.egm.EgmFormat.FileVersion` method), 29

`set_value()` (`pyffi.formats.egt.EgtFormat.FileVersion` method), 32

`set_value()` (`pyffi.formats.esp.EspFormat.ZString` method), 37

`set_value()` (`pyffi.formats.kfm.KfmFormat.HeaderString` method), 40

`set_value()` (`pyffi.formats.kfm.KfmFormat.SizedString` method), 41

`set_value()` (`pyffi.formats.nif.NifFormat.bool` method), 188

`set_value()` (`pyffi.formats.nif.NifFormat.ByteArray` method), 64

`set_value()` (`pyffi.formats.nif.NifFormat.ByteMatrix` method), 65

`set_value()` (`pyffi.formats.nif.NifFormat.Data.VersionUInt` method), 70

`set_value()` (`pyffi.formats.nif.NifFormat.FileVersion` method), 77

`set_value()` (`pyffi.formats.nif.NifFormat.LineString` method), 82

`set_value()` (`pyffi.formats.nif.NifFormat.Ptr` method), 159

`set_value()` (`pyffi.formats.nif.NifFormat.Ref` method), 161

`set_value()` (`pyffi.formats.nif.NifFormat.ShortString` method), 161

```

        method), 162
set_value() (pyffi.formats.nif.NifFormat.SizedString
    method), 163
set_value() (pyffi.formats.tga.TgaFormat.FooterString
    method), 197
set_value() (pyffi.formats.tri.TriFormat.FileVersion
    method), 199
set_variable_1() (pyffi.formats.nif.NifFormat.ControllerLink
    method), 69
set_variable_2() (pyffi.formats.nif.NifFormat.ControllerLink
    method), 69
sf_2_1_st_light_is_point_light()
    (pyffi.formats.nif.NifFormat.BSShaderFlags2
    property), 58
sf_2_2_nd_light()
    (pyffi.formats.nif.NifFormat.BSShaderFlags2
    property), 58
sf_2_3_rd_light()
    (pyffi.formats.nif.NifFormat.BSShaderFlags2
    property), 58
sf_2_alpha_decals()
    (pyffi.formats.nif.NifFormat.BSShaderFlags2
    property), 58
sf_2_billboard_and_envmap_light_fade()
    (pyffi.formats.nif.NifFormat.BSShaderFlags2
    property), 58
sf_2_envmap_light_fade()
    (pyffi.formats.nif.NifFormat.BSShaderFlags2
    property), 58
sf_2_fit_slope() (pyffi.formats.nif.NifFormat.BSShaderFlags2
    property), 58
sf_2_lod_building()
    (pyffi.formats.nif.NifFormat.BSShaderFlags2
    property), 58
sf_2_lod_landscape()
    (pyffi.formats.nif.NifFormat.BSShaderFlags2
    property), 58
sf_2_no_fade() (pyffi.formats.nif.NifFormat.BSShaderFlags2
    property), 58
sf_2_no_lod_land_blend()
    (pyffi.formats.nif.NifFormat.BSShaderFlags2
    property), 58
sf_2_no_transparency_multisampling()
    (pyffi.formats.nif.NifFormat.BSShaderFlags2
    property), 58
sf_2_premult_alpha()
    (pyffi.formats.nif.NifFormat.BSShaderFlags2
    property), 58
sf_2_refraction_tint()
    (pyffi.formats.nif.NifFormat.BSShaderFlags2
    property), 58
sf_2_show_in_local_map()
    (pyffi.formats.nif.NifFormat.BSShaderFlags2
    property), 58
sf_2_skip_normal_maps()
    (pyffi.formats.nif.NifFormat.BSShaderFlags2
    property), 58
sf_2_uniform_scale()
    (pyffi.formats.nif.NifFormat.BSShaderFlags2
    property), 58
sf_2_unknown_1() (pyffi.formats.nif.NifFormat.BSShaderFlags2
    property), 58
sf_2_unknown_10()
    (pyffi.formats.nif.NifFormat.BSShaderFlags2
    property), 58
sf_2_unknown_2() (pyffi.formats.nif.NifFormat.BSShaderFlags2
    property), 59
sf_2_unknown_3() (pyffi.formats.nif.NifFormat.BSShaderFlags2
    property), 59
sf_2_unknown_4() (pyffi.formats.nif.NifFormat.BSShaderFlags2
    property), 59
sf_2_unknown_5() (pyffi.formats.nif.NifFormat.BSShaderFlags2
    property), 59
sf_2_unknown_6() (pyffi.formats.nif.NifFormat.BSShaderFlags2
    property), 59
sf_2_unknown_7() (pyffi.formats.nif.NifFormat.BSShaderFlags2
    property), 59
sf_2_unknown_8() (pyffi.formats.nif.NifFormat.BSShaderFlags2
    property), 59
sf_2_unknown_9() (pyffi.formats.nif.NifFormat.BSShaderFlags2
    property), 59
sf_2_vats_selection()
    (pyffi.formats.nif.NifFormat.BSShaderFlags2
    property), 59
sf_2_vertex_colors()
    (pyffi.formats.nif.NifFormat.BSShaderFlags2
    property), 59
sf_2_vertex_lighting()
    (pyffi.formats.nif.NifFormat.BSShaderFlags2
    property), 59
sf_2_wireframe() (pyffi.formats.nif.NifFormat.BSShaderFlags2
    property), 59
sf_2_z_buffer_write()
    (pyffi.formats.nif.NifFormat.BSShaderFlags2
    property), 59
sf_alpha_texture()
    (pyffi.formats.nif.NifFormat.BSShaderFlags
    property), 57
sf_decals_single_pass()
    (pyffi.formats.nif.NifFormat.BSShaderFlags
    property), 57
sf_dynamic_alpha()
    (pyffi.formats.nif.NifFormat.BSShaderFlags
    property), 57
sf_dynamic_decals_single_pass()
    (pyffi.formats.nif.NifFormat.BSShaderFlags
    property), 57
sf_empty() (pyffi.formats.nif.NifFormat.BSShaderFlags

```

property), 57
 sf_environment_mapping() (*pyffi.formats.nif.NifFormat.BSShaderFlags property*), 57
 sf_external_emittance() (*pyffi.formats.nif.NifFormat.BSShaderFlags property*), 57
 sf_eye_environment_mapping() (*pyffi.formats.nif.NifFormat.BSShaderFlags property*), 57
 sf_face_gen() (*pyffi.formats.nif.NifFormat.BSShaderFlags property*), 57
 sf_fire_refraction() (*pyffi.formats.nif.NifFormat.BSShaderFlags property*), 57
 sf_hair() (*pyffi.formats.nif.NifFormat.BSShaderFlags property*), 57
 sf_localmap_hide_secret() (*pyffi.formats.nif.NifFormat.BSShaderFlags property*), 57
 sf_low_detail() (*pyffi.formats.nif.NifFormat.BSShaderFlags property*), 57
 sf_multiple_textures() (*pyffi.formats.nif.NifFormat.BSShaderFlags property*), 57
 sf_non_projective_shadows() (*pyffi.formats.nif.NifFormat.BSShaderFlags property*), 57
 sf_parallax_occlusion() (*pyffi.formats.nif.NifFormat.BSShaderFlags property*), 57
 sf_parallax_shader_index_15() (*pyffi.formats.nif.NifFormat.BSShaderFlags property*), 57
 sf_refraction() (*pyffi.formats.nif.NifFormat.BSShaderFlags property*), 57
 sf_remappable_textures() (*pyffi.formats.nif.NifFormat.BSShaderFlags property*), 58
 sf_shadow_frustum() (*pyffi.formats.nif.NifFormat.BSShaderFlags property*), 58
 sf_shadow_map() (*pyffi.formats.nif.NifFormat.BSShaderFlags property*), 58
 sf_single_pass() (*pyffi.formats.nif.NifFormat.BSShaderFlags property*), 58
 sf_skinned() (*pyffi.formats.nif.NifFormat.BSShaderFlags property*), 58
 sf_specular() (*pyffi.formats.nif.NifFormat.BSShaderFlags property*), 58
 sf_tree_billboard() (*pyffi.formats.nif.NifFormat.BSShaderFlags property*), 58
 sf_unknown_1() (*pyffi.formats.nif.NifFormat.BSShaderFlags property*), 58
 sf_unknown_2() (*pyffi.formats.nif.NifFormat.BSShaderFlags property*), 58
 sf_unknown_3() (*pyffi.formats.nif.NifFormat.BSShaderFlags property*), 58
 sf_unknown_4() (*pyffi.formats.nif.NifFormat.BSShaderFlags property*), 58
 sf_vertex_alpha() (*pyffi.formats.nif.NifFormat.BSShaderFlags property*), 58
 sf_window_environment_mapping() (*pyffi.formats.nif.NifFormat.BSShaderFlags property*), 58
 sf_z_buffer_test() (*pyffi.formats.nif.NifFormat.BSShaderFlags property*), 58
 SHADER_DEFAULT (*pyffi.formats.nif.NifFormat.BSShaderType attribute*), 60
 shader_flags() (*pyffi.formats.nif.NifFormat.BSShaderProperty property*), 60
 shader_flags_1() (*pyffi.formats.nif.NifFormat.BSEffectShaderProperty property*), 49
 shader_flags_1() (*pyffi.formats.nif.NifFormat.BSLightingShaderProperty property*), 51
 shader_flags_1() (*pyffi.formats.nif.NifFormat.BSSkyShaderProperty property*), 60
 shader_flags_1() (*pyffi.formats.nif.NifFormat.BSWaterShaderProperty property*), 62
 shader_flags_2() (*pyffi.formats.nif.NifFormat.BSEffectShaderProperty property*), 49
 shader_flags_2() (*pyffi.formats.nif.NifFormat.BSLightingShaderProperty property*), 51
 shader_flags_2() (*pyffi.formats.nif.NifFormat.BSShaderProperty property*), 60
 shader_flags_2() (*pyffi.formats.nif.NifFormat.BSSkyShaderProperty property*), 60
 shader_flags_2() (*pyffi.formats.nif.NifFormat.BSWaterShaderProperty property*), 62
 SHADER_LIGHTING30 (*pyffi.formats.nif.NifFormat.BSShaderType attribute*), 60
 SHADER_NOLIGHTING (*pyffi.formats.nif.NifFormat.BSShaderType attribute*), 60
 SHADER_SKIN (*pyffi.formats.nif.NifFormat.BSShaderType attribute*), 60
 SHADER_SKY (*pyffi.formats.nif.NifFormat.BSShaderType attribute*), 60
 SHADER_TALL_GRASS (*pyffi.formats.nif.NifFormat.BSShaderType attribute*), 60
 shader_textures() (*pyffi.formats.nif.NifFormat.NiTexturingProperty property*), 149

SHADER_TILE (*pyffi.formats.nif.NifFormat.BSShaderType* size () (*pyffi.formats.nif.NifFormat.NiParticleSystemController* attribute), 60 property), 134
 shader_type () (*pyffi.formats.nif.NifFormat.BSShaderProperty* property keys () (*pyffi.formats.nif.NifFormat.NiPSSimulatorGeneralStep* property), 60 property), 122
 SHADER_WATER (*pyffi.formats.nif.NifFormat.BSShaderType* size_loop_behavior () (*pyffi.formats.nif.NifFormat.NiPSSimulatorGeneralStep* attribute), 60 property), 122
 shape () (*pyffi.formats.nif.NifFormat.bhkWorldObject* property), 188 sizes () (*pyffi.formats.nif.NifFormat.NiParticlesData* property), 135
 shape () (*pyffi.formats.nif.NifFormat.SkinShape* property), 163 skeleton_root () (*pyffi.formats.nif.NifFormat.NiSkinInstance* property), 144
 shape_description () (*pyffi.formats.nif.NifFormat.NiPhysXActorDesc* skeleton_root () (*pyffi.formats.nif.NifFormat.NiSkinningMeshModifier* property), 136 property), 145
 shape_groups_1 () (*pyffi.formats.nif.NifFormat.NiBoneLODController* transform () (*pyffi.formats.nif.NifFormat.NiSkinningMeshModifier* property), 96 property), 145
 shape_groups_2 () (*pyffi.formats.nif.NifFormat.NiBoneLODController* property), 96 skelrootentry () (*pyffi.spells.nif.fix.SpellSendBonesToBindPosition* method), 209
 shell_link () (*pyffi.formats.nif.NifFormat.NiRoomGroup* property), 141 skelrootentry () (*pyffi.spells.nif.fix.SpellSendDetachedGeometriesToBindPosition* method), 209
 SHELLCASING (*pyffi.formats.nif.NifFormat.Fallout3Layer* attribute), 76 skelrootentry () (*pyffi.spells.nif.fix.SpellSendGeometriesToBindPosition* method), 209
 SHELLCASING (*pyffi.formats.nif.NifFormat.SkyrimLayer* attribute), 167 skin_instance () (*pyffi.formats.nif.NifFormat.SkinShape* property), 163
 SHIELD (*pyffi.formats.nif.NifFormat.Fallout3Layer* attribute), 76 skin_partition () (*pyffi.formats.nif.NifFormat.NiSkinInstance* property), 144
 SHIELD (*pyffi.formats.nif.NifFormat.OblivionLayer* attribute), 156 skin_partition_blocks () (*pyffi.formats.nif.NifFormat.NiSkinPartition* property), 144
 short (*pyffi.formats.cgf.CgfFormat* attribute), 19 skin_tint_color () (*pyffi.formats.nif.NifFormat.BSLightingShaderProperty* property), 51
 short (*pyffi.formats.dds.DdsFormat* attribute), 26 skip_regexs (*pyffi.spells.Toaster* attribute), 233
 short (*pyffi.formats.egm.EgmFormat* attribute), 30 sky_object_type () (*pyffi.formats.nif.NifFormat.BSSkyShaderProperty* property), 60
 short (*pyffi.formats.egt.EgtFormat* attribute), 33 sky_object_type () (*pyffi.formats.nif.NifFormat.SkyShaderProperty* property), 164
 short (*pyffi.formats.esp.EspFormat* attribute), 37 sliding_b () (*pyffi.formats.nif.NifFormat.PrismaticDescriptor* property), 159
 short (*pyffi.formats.kfm.KfmFormat* attribute), 41 simulation_steps () (*pyffi.formats.nif.NifFormat.NiPSSimulator* property), 121
 short (*pyffi.formats.nif.NifFormat* attribute), 189 simulator () (*pyffi.formats.nif.NifFormat.NiPSParticleSystem* property), 120
 short (*pyffi.formats.tga.TgaFormat* attribute), 197 Sit (*pyffi.formats.nif.NifFormat.AnimationType* attribute), 44
 short (*pyffi.formats.tri.TriFormat* attribute), 201 size () (*pyffi.formats.nif.NifFormat.BSMultiBoundOBB* property), 53
 short_set_to_0 () (*pyffi.formats.nif.NifFormat.bhkCMSDMatrix* property), 179
 shrink_generation () (*pyffi.formats.nif.NifFormat.NiPSSimulatorGeneralStep* property), 122
 shrink_time () (*pyffi.formats.nif.NifFormat.NiPSSimulatorGeneralStep* property), 122
 SIDE_WEAPON (*pyffi.formats.nif.NifFormat.OblivionLayer* attribute), 156
 simulation_steps () (*pyffi.formats.nif.NifFormat.NiPSSimulator* property), 121
 simulator () (*pyffi.formats.nif.NifFormat.NiPSParticleSystem* property), 120
 Sit (*pyffi.formats.nif.NifFormat.AnimationType* attribute), 44
 size () (*pyffi.formats.nif.NifFormat.BSMultiBoundOBB* property), 53


```

        (pyffi.formats.nif.NifFormat.SkyrimShaderPropertyFlags1 property), 167
        (pyffi.formats.nif.NifFormat.SkyrimShaderPropertyFlags1 property), 168
slsf_1_external_emittance()                slsf_1_refraction()
        (pyffi.formats.nif.NifFormat.SkyrimShaderPropertyFlags1 property), 167
        (pyffi.formats.nif.NifFormat.SkyrimShaderPropertyFlags1 property), 168
slsf_1_eye_environment_mapping()           slsf_1_remappable_textures()
        (pyffi.formats.nif.NifFormat.SkyrimShaderPropertyFlags1 property), 167
        (pyffi.formats.nif.NifFormat.SkyrimShaderPropertyFlags1 property), 168
slsf_1_face_gen_rgb_tint()                 slsf_1_screendoor_alpha_fade()
        (pyffi.formats.nif.NifFormat.SkyrimShaderPropertyFlags1 property), 167
        (pyffi.formats.nif.NifFormat.SkyrimShaderPropertyFlags1 property), 168
slsf_1_facegen_detail_map()               slsf_1_skinned() (pyffi.formats.nif.NifFormat.SkyrimShaderProperty
        (pyffi.formats.nif.NifFormat.SkyrimShaderPropertyFlags1 property), 168
        (pyffi.formats.nif.NifFormat.SkyrimShaderPropertyFlags1 property), 167
slsf_1_fire_refraction()                  (pyffi.formats.nif.NifFormat.SkyrimShaderPropertyFlags1
        (pyffi.formats.nif.NifFormat.SkyrimShaderPropertyFlags1 property), 168
        (pyffi.formats.nif.NifFormat.SkyrimShaderPropertyFlags1 property), 167
slsf_1_greyscale_to_palette_alpha()        slsf_1_specular()
        (pyffi.formats.nif.NifFormat.SkyrimShaderPropertyFlags1 property), 167
        (pyffi.formats.nif.NifFormat.SkyrimShaderPropertyFlags1 property), 168
slsf_1_greyscale_to_palette_color()        slsf_1_temp_refraction()
        (pyffi.formats.nif.NifFormat.SkyrimShaderPropertyFlags1 property), 167
        (pyffi.formats.nif.NifFormat.SkyrimShaderPropertyFlags1 property), 168
slsf_1_hair_soft_lighting()               slsf_1_use_falloff()
        (pyffi.formats.nif.NifFormat.SkyrimShaderPropertyFlags1 property), 167
        (pyffi.formats.nif.NifFormat.SkyrimShaderPropertyFlags1 property), 168
slsf_1_landscape()                       slsf_1_vertex_alpha()
        (pyffi.formats.nif.NifFormat.SkyrimShaderPropertyFlags1 property), 167
        (pyffi.formats.nif.NifFormat.SkyrimShaderPropertyFlags1 property), 168
slsf_1_localmap_hide_secret()             slsf_1_z_buffer_test()
        (pyffi.formats.nif.NifFormat.SkyrimShaderPropertyFlags1 property), 167
        (pyffi.formats.nif.NifFormat.SkyrimShaderPropertyFlags1 property), 168
slsf_1_model_space_normals()              slsf_2_anisotropic_lighting()
        (pyffi.formats.nif.NifFormat.SkyrimShaderPropertyFlags1 property), 167
        (pyffi.formats.nif.NifFormat.SkyrimShaderPropertyFlags2 property), 168
slsf_1_multiple_textures()                slsf_2_assume_shadowmask()
        (pyffi.formats.nif.NifFormat.SkyrimShaderPropertyFlags1 property), 168
        (pyffi.formats.nif.NifFormat.SkyrimShaderPropertyFlags2 property), 168
slsf_1_non_projective_shadows()           slsf_2_back_lighting()
        (pyffi.formats.nif.NifFormat.SkyrimShaderPropertyFlags1 property), 168
        (pyffi.formats.nif.NifFormat.SkyrimShaderPropertyFlags2 property), 168
slsf_1_own_emit()                        slsf_2_billboard()
        (pyffi.formats.nif.NifFormat.SkyrimShaderPropertyFlags1 property), 168
        (pyffi.formats.nif.NifFormat.SkyrimShaderPropertyFlags2 property), 168
slsf_1_parallax()                        slsf_2_cloud_lod()
        (pyffi.formats.nif.NifFormat.SkyrimShaderPropertyFlags1 property), 168
        (pyffi.formats.nif.NifFormat.SkyrimShaderPropertyFlags2 property), 168
slsf_1_parallax_occlusion()               slsf_2_double_sided()
        (pyffi.formats.nif.NifFormat.SkyrimShaderPropertyFlags1 property), 168
        (pyffi.formats.nif.NifFormat.SkyrimShaderPropertyFlags2 property), 168
slsf_1_projected_uv()                   slsf_2_effect_lighting()
        (pyffi.formats.nif.NifFormat.SkyrimShaderPropertyFlags1 property), 168
        (pyffi.formats.nif.NifFormat.SkyrimShaderPropertyFlags2 property), 168
slsf_1_recieve_shadows()                 slsf_2_env_map_light_fade()
        (pyffi.formats.nif.NifFormat.SkyrimShaderPropertyFlags2

```

```

        property), 168
slsf_2_fit_slope()
    (pyffi.formats.nif.NifFormat.SkyrimShaderPropertyFlags2
    property), 168
slsf_2_glow_map()
    (pyffi.formats.nif.NifFormat.SkyrimShaderPropertyFlags2
    property), 168
slsf_2_hd_lod_objects()
    (pyffi.formats.nif.NifFormat.SkyrimShaderPropertyFlags2
    property), 168
slsf_2_hide_on_local_map()
    (pyffi.formats.nif.NifFormat.SkyrimShaderPropertyFlags2
    property), 168
slsf_2_lod_landscape()
    (pyffi.formats.nif.NifFormat.SkyrimShaderPropertyFlags2
    property), 168
slsf_2_lod_objects()
    (pyffi.formats.nif.NifFormat.SkyrimShaderPropertyFlags2
    property), 168
slsf_2_multi_index_snow()
    (pyffi.formats.nif.NifFormat.SkyrimShaderPropertyFlags2
    property), 168
slsf_2_multi_layer_parallax()
    (pyffi.formats.nif.NifFormat.SkyrimShaderPropertyFlags2
    property), 168
slsf_2_no_fade()
    (pyffi.formats.nif.NifFormat.SkyrimShaderPropertyFlags2
    property), 169
slsf_2_no_lod_land_blend()
    (pyffi.formats.nif.NifFormat.SkyrimShaderPropertyFlags2
    property), 169
slsf_2_no_transparency_multisampling()
    (pyffi.formats.nif.NifFormat.SkyrimShaderPropertyFlags2
    property), 169
slsf_2_packed_tangent()
    (pyffi.formats.nif.NifFormat.SkyrimShaderPropertyFlags2
    property), 169
slsf_2_premult_alpha()
    (pyffi.formats.nif.NifFormat.SkyrimShaderPropertyFlags2
    property), 169
slsf_2_rim_lighting()
    (pyffi.formats.nif.NifFormat.SkyrimShaderPropertyFlags2
    property), 169
slsf_2_soft_lighting()
    (pyffi.formats.nif.NifFormat.SkyrimShaderPropertyFlags2
    property), 169
slsf_2_tree_anim()
    (pyffi.formats.nif.NifFormat.SkyrimShaderPropertyFlags2
    property), 169
slsf_2_uniform_scale()
    (pyffi.formats.nif.NifFormat.SkyrimShaderPropertyFlags2
    property), 169
slsf_2_unused_01()
    (pyffi.formats.nif.NifFormat.SkyrimShaderPropertyFlags2
    property), 169
    slsf_2_unused_02()
        (pyffi.formats.nif.NifFormat.SkyrimShaderPropertyFlags2
        property), 169
    slsf_2_vertex_colors()
        (pyffi.formats.nif.NifFormat.SkyrimShaderPropertyFlags2
        property), 169
    slsf_2_vertex_lighting()
        (pyffi.formats.nif.NifFormat.SkyrimShaderPropertyFlags2
        property), 169
    slsf_2_weapon_blood()
        (pyffi.formats.nif.NifFormat.SkyrimShaderPropertyFlags2
        property), 169
    slsf_2_wireframe()
        (pyffi.formats.nif.NifFormat.SkyrimShaderPropertyFlags2
        property), 169
    slsf_2_z_buffer_write()
        (pyffi.formats.nif.NifFormat.SkyrimShaderPropertyFlags2
        property), 169
    smooth()
        (pyffi.formats.nif.NifFormat.BSShaderProperty
        property), 60
    falloff_depth()
        (pyffi.formats.nif.NifFormat.BSEffectShaderProperty
        property), 49
    SOLVER_DEACTIVATION_HIGH
        (pyffi.formats.nif.NifFormat.SolverDeactivation
        attribute), 169
    SOLVER_DEACTIVATION_INVALID
        (pyffi.formats.nif.NifFormat.SolverDeactivation
        attribute), 169
    SOLVER_DEACTIVATION_LOW
        (pyffi.formats.nif.NifFormat.SolverDeactivation
        attribute), 169
    SOLVER_DEACTIVATION_MAX
        (pyffi.formats.nif.NifFormat.SolverDeactivation
        attribute), 169
    SOLVER_DEACTIVATION_MEDIUM
        (pyffi.formats.nif.NifFormat.SolverDeactivation
        attribute), 169
    SOLVER_DEACTIVATION_OFF
        (pyffi.formats.nif.NifFormat.SolverDeactivation
        attribute), 170
    SORTING_INHERIT
        (pyffi.formats.nif.NifFormat.SortingMode
        attribute), 170
    slsf_2_sorting_mode()
        (pyffi.formats.nif.NifFormat.NiSortAdjustNode
        property), 145
    SORTING_OFF
        (pyffi.formats.nif.NifFormat.SortingMode
        attribute), 170
    source()
        (pyffi.formats.nif.NifFormat.TexDesc
        property), 174
    slsf_2_texture()
        (pyffi.formats.nif.NifFormat.BSEffectShaderProperty
        property), 49
    source_texture()
        (pyffi.formats.nif.NifFormat.BSSkyShaderProperty
        property), 60
    source_texture()
        (pyffi.formats.nif.NifFormat.NiTextureEffect

```

property), 147
 sources() (pyffi.formats.nif.NifFormat.NiFlipController
 property), 100
 sparkle_parameters() (pyffi.formats.nif.NifFormat.BSLightingShaderProperty
 property), 51
 spawn_dir_chaos() (pyffi.formats.nif.NifFormat.NiPSysSpawnModifier
 property), 130
 spawn_modifier() (pyffi.formats.nif.NifFormat.NiPSysAgeDeathModifier
 property), 123
 spawn_modifier() (pyffi.formats.nif.NifFormat.NiPSysCollider
 property), 125
 spawn_on_collide() (pyffi.formats.nif.NifFormat.NiPSysCollider
 property), 125
 spawn_on_death() (pyffi.formats.nif.NifFormat.NiPSysAgeDeathModifier
 property), 123
 spawn_speed_chaos() (pyffi.formats.nif.NifFormat.NiPSysSpawnModifier
 property), 130
 specular_color() (pyffi.formats.nif.NifFormat.BSLightingShaderProperty
 property), 51
 specular_color() (pyffi.formats.nif.NifFormat.NiLight
 property), 106
 specular_strength() (pyffi.formats.nif.NifFormat.BSLightingShaderProperty
 property), 51
 speed() (pyffi.formats.nif.NifFormat.NiParticleSystemController
 property), 134
 speed() (pyffi.formats.nif.NifFormat.NiPSysEmitter
 property), 126
 speed_random() (pyffi.formats.nif.NifFormat.NiParticleSystemController
 property), 134
 speed_variation() (pyffi.formats.nif.NifFormat.NiPSysEmitter
 property), 126
 spell, 267
 Spell (class in pyffi.spells), 227
 SPELL (pyffi.formats.nif.NifFormat.Fallout3Layer
 attribute), 76
 SPELL (pyffi.formats.nif.NifFormat.OblivionLayer
 attribute), 156
 SPELL (pyffi.formats.nif.NifFormat.SkyrimLayer
 attribute), 167
 SPELL_EXPLOSION (pyffi.formats.nif.NifFormat.OblivionLayer
 attribute), 156
 SpellAddStencilProperty (class in
 pyffi.spells.nif.modify), 225
 SpellAddTangentSpace (class in pyffi.spells.nif.fix),
 206
 SpellApplyPatch (class in pyffi.spells), 226
 SpellApplySkinDeformation (class in
 pyffi.spells.nif.fix), 210
 SpellChangeBonePriorities (class in
 pyffi.spells.nif.modify), 221
 SpellClampMaterialAlpha (class in
 pyffi.spells.nif.fix), 208
 SPELLCLASSES (pyffi.spells.SpellGroupBase attribute),
 229
 SpellCleanFarNif (class in pyffi.spells.nif.modify),
 226
 SpellCleanRefLists (class in
 pyffi.spells.nif.optimize), 214
 SpellCleanStringPalette (class in
 pyffi.spells.nif.fix), 211
 SpellCollisionMaterial (class in
 pyffi.spells.nif.modify), 218
 SpellCollisionType (class in
 pyffi.spells.nif.modify), 218
 SpellDeleteMeshes (class in pyffi.spells.nif.modify),
 223
 SpellDelInterpolatorTransformData (class
 in pyffi.spells.nif.modify), 223
 SpellDelSkinShapes (class in
 pyffi.spells.nif.modify), 224
 SpellDelTangentSpace (class in pyffi.spells.nif.fix),
 206
 SpellDelUnusedBones (class in
 pyffi.spells.nif.optimize), 216
 SpellDelUnusedRoots (class in pyffi.spells.nif.fix),
 212
 SpellDelVertexColor (class in
 pyffi.spells.nif.modify), 225
 SpellDetachHavokTriStripsData (class in
 pyffi.spells.nif.fix), 208
 SpellDisableParallax (class in
 pyffi.spells.nif.modify), 224
 SPELLEXPLOSION (pyffi.formats.nif.NifFormat.Fallout3Layer
 attribute), 76
 SPELLEXPLOSION (pyffi.formats.nif.NifFormat.SkyrimLayer
 attribute), 167
 SpellFFVT3RSkinPartition (class in
 pyffi.spells.nif.fix), 207
 SpellFixCenterRadius (class in pyffi.spells.nif.fix),
 211
 SpellFixEmptySkeletonRoots (class in
 pyffi.spells.nif.fix), 213
 SpellFixMopp (class in pyffi.spells.nif.fix), 211
 SpellFixSkinCenterRadius (class in
 pyffi.spells.nif.fix), 211
 SpellFixTexturePath (class in pyffi.spells.nif.fix),
 207
 SpellGroupBase (class in pyffi.spells), 229
 SpellGroupParallel() (in module pyffi.spells),
 229
 SpellGroupParallelBase (class in pyffi.spells),
 230

SpellGroupSeries() (in module *pyffi.spells*), 229
 SpellGroupSeriesBase (class in *pyffi.spells*), 230
 SpellLowResTexturePath (class in *pyffi.spells.nif.modify*), 217
 SpellMakeFarNif (class in *pyffi.spells.nif.modify*), 226
 SpellMakeSkinlessNif (class in *pyffi.spells.nif.modify*), 226
 SpellMergeDuplicates (class in *pyffi.spells.nif.optimize*), 214
 SpellMergeSkeletonRoots (class in *pyffi.spells.nif.fix*), 209
 SPELLNAME (*pyffi.spells.Spell* attribute), 227
 spellnames (*pyffi.spells.Toaster* attribute), 233
 SpellOptimize (class in *pyffi.spells.nif.optimize*), 216
 SpellOptimizeGeometry (class in *pyffi.spells.nif.optimize*), 215
 SpellReverseAnimation (class in *pyffi.spells.nif.modify*), 220
 spells (*pyffi.spells.SpellGroupBase* attribute), 229
 SPELLS (*pyffi.spells.Toaster* attribute), 231
 SpellScale (class in *pyffi.spells.nif.fix*), 210
 SpellScaleAnimationTime (class in *pyffi.spells.nif.modify*), 219
 SpellSendBonesToBindPosition (class in *pyffi.spells.nif.fix*), 209
 SpellSendDetachedGeometriesToNodePosition (class in *pyffi.spells.nif.fix*), 209
 SpellSendGeometriesToBindPosition (class in *pyffi.spells.nif.fix*), 209
 SpellSetInterpolatorTransRotScale (class in *pyffi.spells.nif.modify*), 222
 SpellSubstituteStringPalette (class in *pyffi.spells.nif.modify*), 221
 SpellSubstituteTexturePath (class in *pyffi.spells.nif.modify*), 217
 SpellTexturePath (class in *pyffi.spells.nif.modify*), 217
 sphere() (*pyffi.formats.nif.NifFormat.BoundingBox* property), 63
 SPHERE_BV (*pyffi.formats.nif.NifFormat.BoundsVolumeType* attribute), 63
 SPHERICAL_SYMMETRY (*pyffi.formats.nif.NifFormat.SymmetryType* attribute), 172
 SPINE1 (*pyffi.formats.nif.NifFormat.Fallout3Layer* attribute), 76
 SPINE1 (*pyffi.formats.nif.NifFormat.OblivionLayer* attribute), 156
 SPINE2 (*pyffi.formats.nif.NifFormat.Fallout3Layer* attribute), 76
 SPINE2 (*pyffi.formats.nif.NifFormat.OblivionLayer* attribute), 156
 split_triangles() (*pyffi.formats.nif.NifFormat.bhkMoppBvTreeShape* method), 183
 STAIRHELPER (*pyffi.formats.nif.NifFormat.SkyrimLayer* attribute), 167
 STAIRS (*pyffi.formats.nif.NifFormat.OblivionLayer* attribute), 156
 start() (*pyffi.formats.nif.NifFormat.ExtraMeshDataEpicMickey2* property), 73
 start() (*pyffi.formats.nif.NifFormat.NiParticleBomb* property), 132
 start_frame() (*pyffi.formats.nif.NifFormat.BSPSysSubTexModifier* property), 55
 start_frame_fudge() (*pyffi.formats.nif.NifFormat.BSPSysSubTexModifier* property), 55
 start_index() (*pyffi.formats.nif.NifFormat.Region* property), 161
 start_random() (*pyffi.formats.nif.NifFormat.NiParticleSystemController* property), 134
 start_time() (*pyffi.formats.nif.NifFormat.NiTimeController* property), 149
 STATIC (*pyffi.formats.nif.NifFormat.Fallout3Layer* attribute), 76
 STATIC (*pyffi.formats.nif.NifFormat.OblivionLayer* attribute), 156
 STATIC (*pyffi.formats.nif.NifFormat.SkyrimLayer* attribute), 167
 stencil_enabled() (*pyffi.formats.nif.NifFormat.NiStencilProperty* property), 146
 stencil_function() (*pyffi.formats.nif.NifFormat.NiStencilProperty* property), 146
 stencil_mask() (*pyffi.formats.nif.NifFormat.NiStencilProperty* property), 146
 stencil_ref() (*pyffi.formats.nif.NifFormat.NiStencilProperty* property), 146
 stiff_spring() (*pyffi.formats.nif.NifFormat.bhkStiffSpringConstraint* property), 187
 stiff_spring() (*pyffi.formats.nif.NifFormat.SubConstraint* property), 172
 StiffSpring (*pyffi.formats.nif.NifFormat.hkConstraintType* attribute), 189
 stop_time() (*pyffi.formats.nif.NifFormat.NiTimeController* property), 149
 stream (*pyffi.spells.Spell* attribute), 228
 stream() (*pyffi.formats.nif.NifFormat.MeshData* property), 85
 streamable() (*pyffi.formats.nif.NifFormat.NiDataStream* property), 99
 strength() (*pyffi.formats.nif.NifFormat.BSWindModifier* property), 62
 strength() (*pyffi.formats.nif.NifFormat.NiPSysGravityModifier* property), 128

String (pyffi.formats.cgf.CgfFormat attribute), 18
string_data() (pyffi.formats.nif.NifFormat.NiStringExtraData property), 146
StringIndex (pyffi.formats.nif.NifFormat attribute), 170
strip_width() (pyffi.formats.nif.NifFormat.BSPProceduralLightningControl property), 56
strips() (pyffi.formats.nif.NifFormat.bhkCMSDChunk property), 179
strips_data() (pyffi.formats.nif.NifFormat.bhkMeshShape property), 182
sub_constraint() (pyffi.formats.nif.NifFormat.bhkBreakableComponent property), 178
sub_shapes() (pyffi.formats.nif.NifFormat.bhkConvexListShape property), 181
submesh_to_region_map() (pyffi.formats.nif.NifFormat.MeshData property), 85
submit_points() (pyffi.formats.nif.NifFormat.NiMeshModifier property), 108
substitute() (pyffi.spells.nif.fix.SpellCleanStringPalette method), 212
substitute() (pyffi.spells.nif.fix.SpellFixTexturePath method), 207
substitute() (pyffi.spells.nif.modify.SpellLowResTexturePath method), 217
substitute() (pyffi.spells.nif.modify.SpellSubstituteStringPalette method), 221
substitute() (pyffi.spells.nif.modify.SpellSubstituteTexturePath method), 217
substitute() (pyffi.spells.nif.modify.SpellTexturePath method), 217
subtexture_offset_u_vs() (pyffi.formats.nif.NifFormat.NiPSysData property), 125
sup_norm() (pyffi.formats.cgf.CgfFormat.Matrix44 method), 16
sup_norm() (pyffi.formats.nif.NifFormat.Matrix33 method), 84
sup_norm() (pyffi.formats.nif.NifFormat.Matrix44 method), 85
switch_state() (pyffi.formats.nif.NifFormat.NiDynamicEffect property), 99
swsf_1_bypass_refraction_map() (pyffi.formats.nif.NifFormat.SkyrimWaterShaderFlags property), 169
swsf_1_enabled() (pyffi.formats.nif.NifFormat.SkyrimWaterShaderFlags property), 169
swsf_1_highlight_layer_toggle() (pyffi.formats.nif.NifFormat.SkyrimWaterShaderFlags property), 169
swsf_1_unknown_0() (pyffi.formats.nif.NifFormat.SkyrimWaterShaderFlags property), 169
swsf_1_unknown_3() (pyffi.formats.nif.NifFormat.SkyrimWaterShaderFlags property), 169
swsf_1_unknown_4() (pyffi.formats.nif.NifFormat.SkyrimWaterShaderFlags property), 169
swsf_1_unknown_5() (pyffi.formats.nif.NifFormat.SkyrimWaterShaderFlags property), 169
swsf_1_water_toggle() (pyffi.formats.nif.NifFormat.SkyrimWaterShaderFlags property), 169
symmetry_type() (pyffi.formats.nif.NifFormat.NiParticleBomb property), 132
symmetry_type() (pyffi.formats.nif.NifFormat.NiPSysBombModifier property), 124
SYNC_ANY (pyffi.formats.nif.NifFormat.SyncPoint attribute), 172
PHYSICS_COMPLETED (pyffi.formats.nif.NifFormat.SyncPoint attribute), 173
SYNC_PHYSICS_SIMULATE (pyffi.formats.nif.NifFormat.SyncPoint attribute), 173
SYNC_POST_UPDATE (pyffi.formats.nif.NifFormat.SyncPoint attribute), 173
REFLECTIONS (pyffi.formats.nif.NifFormat.SyncPoint attribute), 173
RENDER (pyffi.formats.nif.NifFormat.SyncPoint attribute), 173
SYNC_UPDATE (pyffi.formats.nif.NifFormat.SyncPoint attribute), 173
SYNC_VISIBLE (pyffi.formats.nif.NifFormat.SyncPoint attribute), 173
T
t() (pyffi.formats.nif.NifFormat.TBC property), 173
TAIL (pyffi.formats.nif.NifFormat.Fallout3Layer attribute), 76
TAIL (pyffi.formats.nif.NifFormat.OblivionLayer attribute), 156
tangents_Bitangents (pyffi.formats.nif.NifFormat.ExtraVectorsFlags attribute), 73
target() (pyffi.formats.nif.NifFormat.bhkCompressedMeshShape property), 180
target() (pyffi.formats.nif.NifFormat.NiCollisionObject property), 98
target() (pyffi.formats.nif.NifFormat.NiLookAtInterpolator property), 107
target() (pyffi.formats.nif.NifFormat.NiPortal property), 140
target() (pyffi.formats.nif.NifFormat.NiPSysModifier property), 129

target () (pyffi.formats.nif.NifFormat.NiShadowGenerator property), 49
 target () (pyffi.formats.nif.NifFormat.NiTimeController property), 143
 target_color () (pyffi.formats.nif.NifFormat.NiPoint3Interpolator property), 149
 target_names () (pyffi.formats.nif.NifFormat.NiMorphWeightsController property), 139
 target_names () (pyffi.formats.nif.NifFormat.NiMorphWeightsController property), 109
 tbc () (pyffi.formats.nif.NifFormat.Key property), 81
 tbc () (pyffi.formats.nif.NifFormat.QuatKey property), 160
 TBC_KEY (pyffi.formats.nif.NifFormat.KeyType attribute), 81
 TC_AMBIENT (pyffi.formats.nif.NifFormat.TargetColor attribute), 173
 TC_DIFFUSE (pyffi.formats.nif.NifFormat.TargetColor attribute), 173
 TC_SELF_ILLUM (pyffi.formats.nif.NifFormat.TargetColor attribute), 173
 TC_SPECULAR (pyffi.formats.nif.NifFormat.TargetColor attribute), 173
 TERRAIN (pyffi.formats.nif.NifFormat.Fallout3Layer attribute), 76
 TERRAIN (pyffi.formats.nif.NifFormat.OblivionLayer attribute), 156
 TERRAIN (pyffi.formats.nif.NifFormat.SkyrimLayer attribute), 167
 TEST_ALWAYS (pyffi.formats.nif.NifFormat.StencilCompareMode attribute), 170
 TEST_EQUAL (pyffi.formats.nif.NifFormat.StencilCompareMode attribute), 170
 TEST_GREATER (pyffi.formats.nif.NifFormat.StencilCompareMode attribute), 170
 TEST_GREATER_EQUAL (pyffi.formats.nif.NifFormat.StencilCompareMode attribute), 170
 TEST_LESS (pyffi.formats.nif.NifFormat.StencilCompareMode attribute), 170
 TEST_LESS_EQUAL (pyffi.formats.nif.NifFormat.StencilCompareMode attribute), 170
 TEST_NEVER (pyffi.formats.nif.NifFormat.StencilCompareMode attribute), 170
 TEST_NOT_EQUAL (pyffi.formats.nif.NifFormat.StencilCompareMode attribute), 170
 text_keys () (pyffi.formats.nif.NifFormat.NiSequence property), 142
 text_keys () (pyffi.formats.nif.NifFormat.NiTextKeyExtraData property), 147
 text_keys_name () (pyffi.formats.nif.NifFormat.NiSequence property), 142
 TextString (pyffi.formats.kfm.KfmFormat attribute), 41
 texture_clamp_mode () (pyffi.formats.nif.NifFormat.BSEffectShaderProperty property), 49
 texture_clamp_mode () (pyffi.formats.nif.NifFormat.BSLightingShaderProperty property), 51
 texture_clamp_mode () (pyffi.formats.nif.NifFormat.BSShaderLightingProperty property), 59
 texture_clamping () (pyffi.formats.nif.NifFormat.NiTextureEffect property), 147
 texture_count () (pyffi.formats.nif.NifFormat.NiTexturingProperty property), 149
 texture_data () (pyffi.formats.nif.NifFormat.ShaderTexDesc property), 162
 texture_elements () (pyffi.formats.nif.NifFormat.NiMultiTextureProperty property), 110
 texture_filtering () (pyffi.formats.nif.NifFormat.NiTextureEffect property), 147
 texture_name () (pyffi.formats.nif.NifFormat.ArkTexture property), 45
 texture_set () (pyffi.formats.nif.NifFormat.BSLightingShaderProperty property), 51
 texture_set () (pyffi.formats.nif.NifFormat.BSShaderPPLightingProperty property), 59
 texture_slot () (pyffi.formats.nif.NifFormat.NiFlipController property), 100
 texture_slot () (pyffi.formats.nif.NifFormat.NiTextureTransformController property), 148
 texture_type () (pyffi.formats.nif.NifFormat.NiTextureEffect property), 147
 textures () (pyffi.formats.nif.NifFormat.BSShaderTextureSet property), 60
 textures () (pyffi.formats.nif.NifFormat.NiArkTextureExtraData property), 90
 texturing_property () (pyffi.formats.nif.NifFormat.ArkTexture property), 45
 TgaFormat (class in pyffi.formats.tga), 196
 TgaFormat.ColorMapType (class in pyffi.formats.tga), 196
 TgaFormat.Data (class in pyffi.formats.tga), 196
 TgaFormat.FooterString (class in pyffi.formats.tga), 196
 TgaFormat.Image (class in pyffi.formats.tga), 197
 TgaFormat.ImageType (class in pyffi.formats.tga), 197
 TgaXMLPATH, 8
 threshold () (pyffi.formats.nif.NifFormat.bhkBreakableConstraint property), 178
 threshold () (pyffi.formats.nif.NifFormat.NiAlphaProperty property), 90
 tiling () (pyffi.formats.nif.NifFormat.TexDesc property), 147

erty), 174

time() (pyffi.formats.nif.NifFormat.Key property), 81

time() (pyffi.formats.nif.NifFormat.QuatKey property), 160

timestamp() (pyffi.formats.nif.NifFormat.Particle property), 157

toast() (pyffi.spells.Toaster method), 233

toast_archives() (pyffi.spells.Toaster method), 233

toastentry() (pyffi.spells.nif.fix.SpellScale class method), 211

toastentry() (pyffi.spells.nif.modify.SpellChangeBonePriorities class method), 221

toastentry() (pyffi.spells.nif.modify.SpellCollisionMaterial class method), 219

toastentry() (pyffi.spells.nif.modify.SpellCollisionType class method), 218

toastentry() (pyffi.spells.nif.modify.SpellDelInterpolatorTransformData class method), 223

toastentry() (pyffi.spells.nif.modify.SpellLowResTexturePath class method), 217

toastentry() (pyffi.spells.nif.modify.SpellScaleAnimationTime class method), 220

toastentry() (pyffi.spells.nif.modify.SpellSetInterpolatorTransformScale class method), 222

toastentry() (pyffi.spells.nif.modify.SpellSubstituteStringPalette class method), 221

toastentry() (pyffi.spells.nif.modify.SpellSubstituteTexturePath class method), 217

toastentry() (pyffi.spells.nif.modify.SpellTexturePath class method), 217

toastentry() (pyffi.spells.Spell class method), 228

toastentry() (pyffi.spells.SpellGroupBase class method), 229

toaster, 267

Toaster (class in pyffi.spells), 231

toaster (pyffi.spells.Spell attribute), 229

toastexit() (pyffi.spells.Spell class method), 229

toastexit() (pyffi.spells.SpellGroupBase class method), 229

top (pyffi.spells.Toaster attribute), 233

trailer() (pyffi.formats.nif.NifFormat.NiParticleSystemController property), 134

transform_1() (pyffi.formats.nif.NifFormat.BSTreadTransform property), 61

transform_2() (pyffi.formats.nif.NifFormat.BSTreadTransform property), 61

transform_dests() (pyffi.formats.nif.NifFormat.NiPhysXProp property), 137

transform_index() (pyffi.formats.nif.NifFormat.bhkCMSDChunk property), 179

transform_type() (pyffi.formats.nif.NifFormat.TexDesc property), 174

translation() (pyffi.formats.nif.NifFormat.bhkCMSDChunk property), 179

translation() (pyffi.formats.nif.NifFormat.bhkCMSDTransform property), 179

translation() (pyffi.formats.nif.NifFormat.BoundingBox property), 63

translation() (pyffi.formats.nif.NifFormat.BSTreadTransformData property), 61

translation() (pyffi.formats.nif.NifFormat.MTransform property), 83

translation() (pyffi.formats.nif.NifFormat.NiLookAtInterpolator property), 107

translation() (pyffi.formats.nif.NifFormat.ParticleDesc property), 157

translation() (pyffi.formats.nif.NifFormat.QTransform property), 160

translation_data() (pyffi.formats.nif.NifFormat.TexDesc property), 174

TRANSPARENT (pyffi.formats.nif.NifFormat.Fallout3Layer attribute), 76

TRANSPARENT (pyffi.formats.nif.NifFormat.OblivionLayer attribute), 156

TRANSPARENT (pyffi.formats.nif.NifFormat.SkyrimLayer attribute), 167

TRANSPARENT_SMALL (pyffi.formats.nif.NifFormat.Fallout3Layer attribute), 76

TRANSPARENT_SMALL (pyffi.formats.nif.NifFormat.SkyrimLayer attribute), 167

TRANSPARENT_SMALL_ANIM (pyffi.formats.nif.NifFormat.Fallout3Layer attribute), 77

TRANSPARENT_SMALL_ANIM (pyffi.formats.nif.NifFormat.SkyrimLayer attribute), 167

TRAP (pyffi.formats.nif.NifFormat.Fallout3Layer attribute), 77

TRAP (pyffi.formats.nif.NifFormat.OblivionLayer attribute), 156

TRAP (pyffi.formats.nif.NifFormat.SkyrimLayer attribute), 167

transforms() (pyffi.formats.nif.NifFormat.BSTreadTransformInterpolator property), 61

tree() (pyffi.formats.cgf.CgfFormat.Chunk method), 12

tree() (pyffi.formats.nif.NifFormat.NiObject method), 113

TREES (pyffi.formats.nif.NifFormat.Fallout3Layer attribute), 77

TREES (pyffi.formats.nif.NifFormat.OblivionLayer attribute), 156

TREES (pyffi.formats.nif.NifFormat.SkyrimLayer attribute), 167

tribute), 167
 triangle() (pyffi.formats.nif.NifFormat.hkTriangle property), 189
 triangle_1() (pyffi.formats.nif.NifFormat.bhkCMSDBigTris property), 178
 triangle_2() (pyffi.formats.nif.NifFormat.bhkCMSDBigTris property), 178
 triangle_3() (pyffi.formats.nif.NifFormat.bhkCMSDBigTris property), 178
 triangle_offset() (pyffi.formats.nif.NifFormat.Polygon property), 158
 TriFormat (class in pyffi.formats.tri), 199
 TriFormat.FileSignature (class in pyffi.formats.tri), 199
 TriFormat.FileVersion (class in pyffi.formats.tri), 199
 TriFormat.Header (class in pyffi.formats.tri), 199
 TriFormat.ModifierRecord (class in pyffi.formats.tri), 200
 TriFormat.MorphRecord (class in pyffi.formats.tri), 200
 TriFormat.QuadFace (class in pyffi.formats.tri), 201
 TriFormat.SizedStringZ (class in pyffi.formats.tri), 201
 TRIGGER (pyffi.formats.nif.NifFormat.Fallout3Layer attribute), 77
 TRIGGER (pyffi.formats.nif.NifFormat.OblivionLayer attribute), 156
 TRIGGER (pyffi.formats.nif.NifFormat.SkyrimLayer attribute), 167
 TT_ROTATE (pyffi.formats.nif.NifFormat.TexTransform attribute), 175
 TT_SCALE_U (pyffi.formats.nif.NifFormat.TexTransform attribute), 175
 TT_SCALE_V (pyffi.formats.nif.NifFormat.TexTransform attribute), 175
 TT_TRANSLATE_U (pyffi.formats.nif.NifFormat.TexTransform attribute), 175
 TT_TRANSLATE_V (pyffi.formats.nif.NifFormat.TexTransform attribute), 175
 turbulence() (pyffi.formats.nif.NifFormat.NiPSysGravityModifier property), 128
 turbulence_scale() (pyffi.formats.nif.NifFormat.NiPSysGravityModifier property), 128
 type() (pyffi.formats.nif.NifFormat.bhkRDTConstraint property), 185
 type() (pyffi.formats.nif.NifFormat.bhkRDTMalleableConstraint property), 185
 type() (pyffi.formats.nif.NifFormat.ChannelData property), 65
 type() (pyffi.formats.nif.NifFormat.NiGravity property), 105
 type() (pyffi.formats.nif.NifFormat.SubConstraint property), 172
 type_of_controlled_color() (pyffi.formats.nif.NifFormat.BSEffectShaderPropertyColorControl property), 49
 type_of_controlled_color() (pyffi.formats.nif.NifFormat.BSLightingShaderPropertyColorControl property), 51
 type_of_controlled_variable() (pyffi.formats.nif.NifFormat.BSEffectShaderPropertyFloatControl property), 49
 type_of_controlled_variable() (pyffi.formats.nif.NifFormat.BSLightingShaderPropertyFloatControl property), 52
U
 ubyte (pyffi.formats.cgf.CgfFormat attribute), 19
 ubyte (pyffi.formats.dds.DdsFormat attribute), 26
 ubyte (pyffi.formats.egm.EgmFormat attribute), 30
 ubyte (pyffi.formats.egt.EgtFormat attribute), 33
 ubyte (pyffi.formats.esp.EspFormat attribute), 37
 ubyte (pyffi.formats.tga.TgaFormat attribute), 197
 ubyte (pyffi.formats.tri.TriFormat attribute), 201
 uint (pyffi.formats.cgf.CgfFormat attribute), 19
 uint (pyffi.formats.dds.DdsFormat attribute), 26
 uint (pyffi.formats.egm.EgmFormat attribute), 30
 uint (pyffi.formats.egt.EgtFormat attribute), 33
 uint (pyffi.formats.esp.EspFormat attribute), 37
 uint (pyffi.formats.kfm.KfmFormat attribute), 41
 uint (pyffi.formats.nif.NifFormat attribute), 190
 uint (pyffi.formats.tga.TgaFormat attribute), 197
 uint (pyffi.formats.tri.TriFormat attribute), 201
 UInt32 (pyffi.formats.bsa.BsaFormat attribute), 9
 uint64 (pyffi.formats.esp.EspFormat attribute), 37
 unknown_float_1() (pyffi.formats.nif.NifFormat.BSPSysLODModifier property), 54
 unknown_float_2() (pyffi.formats.nif.NifFormat.BSPSysLODModifier property), 54
 unknown_float_3() (pyffi.formats.nif.NifFormat.BSPSysLODModifier property), 54
 unknown_float_4() (pyffi.formats.nif.NifFormat.BSPSysLODModifier property), 54
 ulittle32 (pyffi.formats.nif.NifFormat attribute), 190
 UNIDENTIFIED (pyffi.formats.nif.NifFormat.Fallout3Layer attribute), 77
 UNIDENTIFIED (pyffi.formats.nif.NifFormat.OblivionLayer attribute), 156
 UNIDENTIFIED (pyffi.formats.nif.NifFormat.SkyrimLayer attribute), 167
 union() (pyffi.formats.nif.NifFormat.BoundingVolume property), 63
 UNION_BV (pyffi.formats.nif.NifFormat.BoundsVolumeType attribute), 63

unknown () (pyffi.formats.nif.NifFormat.ExportInfo property), 73
 unknown () (pyffi.formats.nif.NifFormat.NiTextureEffect property), 147
 unknown () (pyffi.formats.nif.NifFormat.NiTransparentProperty property), 150
 UNKNOWN1 (pyffi.formats.nif.NifFormat.OblivionLayer attribute), 156
 UNKNOWN2 (pyffi.formats.nif.NifFormat.OblivionLayer attribute), 156
 UNKNOWN2_MAP (pyffi.formats.nif.NifFormat.TextType attribute), 175
 unknown_1 () (pyffi.formats.nif.NifFormat.bhkMeshShape property), 182
 unknown_1 () (pyffi.formats.nif.NifFormat.Ni3dsAlphaAnimator property), 87
 unknown_1 () (pyffi.formats.nif.NifFormat.Ni3dsColorAnimator property), 88
 unknown_1 () (pyffi.formats.nif.NifFormat.Ni3dsMorphShape property), 88
 unknown_1 () (pyffi.formats.nif.NifFormat.Ni3dsParticleSystem property), 88
 unknown_1 () (pyffi.formats.nif.NifFormat.Ni3dsPathController property), 88
 unknown_1 () (pyffi.formats.nif.NifFormat.NiBezierTriangle property), 94
 unknown_1 () (pyffi.formats.nif.NifFormat.NiEnvMappedTriShape property), 100
 unknown_1 () (pyffi.formats.nif.NifFormat.NiLookAtController property), 107
 unknown_1 () (pyffi.formats.nif.NifFormat.NiPSBombForce property), 114
 unknown_1 () (pyffi.formats.nif.NifFormat.NiPSBoundUpdater property), 115
 unknown_1 () (pyffi.formats.nif.NifFormat.NiPSBoxEmitter property), 115
 unknown_1 () (pyffi.formats.nif.NifFormat.NiPSDragForce property), 116
 unknown_1 () (pyffi.formats.nif.NifFormat.NiPSFacingQuadGenerator property), 117
 unknown_1 () (pyffi.formats.nif.NifFormat.NiPSGravityForce property), 118
 unknown_1 () (pyffi.formats.nif.NifFormat.NiPSMeshEmitter property), 119
 unknown_1 () (pyffi.formats.nif.NifFormat.NiPSSimulatorGeneralStep property), 122
 unknown_1 () (pyffi.formats.nif.NifFormat.NiPSSphericalCollider property), 123
 unknown_1 () (pyffi.formats.nif.NifFormat.TextDesc property), 174
 unknown_10 () (pyffi.formats.nif.NifFormat.NiPSBombForce property), 114
 unknown_10 () (pyffi.formats.nif.NifFormat.NiPSBoxEmitter property), 115
 unknown_10 () (pyffi.formats.nif.NifFormat.NiPSDragForce property), 116
 unknown_10 () (pyffi.formats.nif.NifFormat.NiPSFacingQuadGenerator property), 117
 unknown_10 () (pyffi.formats.nif.NifFormat.NiPSGravityForce property), 118
 unknown_10 () (pyffi.formats.nif.NifFormat.NiPSMeshEmitter property), 119
 unknown_10 () (pyffi.formats.nif.NifFormat.NiPSParticleSystem property), 120
 unknown_10 () (pyffi.formats.nif.NifFormat.NiPSSphereEmitter property), 122
 unknown_100 () (pyffi.formats.nif.NifFormat.NiMesh property), 108
 unknown_101 () (pyffi.formats.nif.NifFormat.NiMesh property), 108
 unknown_102 () (pyffi.formats.nif.NifFormat.NiMesh property), 108
 unknown_103 () (pyffi.formats.nif.NifFormat.NiMesh property), 108
 unknown_11 () (pyffi.formats.nif.NifFormat.NiPSBoxEmitter property), 115
 unknown_11 () (pyffi.formats.nif.NifFormat.NiPSFacingQuadGenerator property), 117
 unknown_11 () (pyffi.formats.nif.NifFormat.NiPSGravityForce property), 118
 unknown_11 () (pyffi.formats.nif.NifFormat.NiPSMeshEmitter property), 119
 unknown_11 () (pyffi.formats.nif.NifFormat.NiPSParticleSystem property), 120
 unknown_11 () (pyffi.formats.nif.NifFormat.NiPSSphereEmitter property), 123
 unknown_12 () (pyffi.formats.nif.NifFormat.NiPSBoxEmitter property), 115
 unknown_12 () (pyffi.formats.nif.NifFormat.NiPSFacingQuadGenerator property), 117
 unknown_12 () (pyffi.formats.nif.NifFormat.NiPSGravityForce property), 118
 unknown_12 () (pyffi.formats.nif.NifFormat.NiPSMeshEmitter property), 119
 unknown_12 () (pyffi.formats.nif.NifFormat.NiPSParticleSystem property), 120
 unknown_12 () (pyffi.formats.nif.NifFormat.NiPSSphereEmitter property), 123
 unknown_13 () (pyffi.formats.nif.NifFormat.NiPSBoxEmitter property), 115
 unknown_13 () (pyffi.formats.nif.NifFormat.NiPSGravityForce property), 118
 unknown_13 () (pyffi.formats.nif.NifFormat.NiPSMeshEmitter property), 119
 unknown_13 () (pyffi.formats.nif.NifFormat.NiPSSphereEmitter property), 123
 unknown_14 () (pyffi.formats.nif.NifFormat.NiPSBoxEmitter property), 115

unknown_14 () (pyffi.formats.nif.NifFormat.NiPSGravityForceController property), 118
 unknown_14 () (pyffi.formats.nif.NifFormat.NiPSMeshEmitter property), 119
 unknown_14 () (pyffi.formats.nif.NifFormat.NiPSSphereEmitter property), 123
 unknown_15 () (pyffi.formats.nif.NifFormat.NiPSBoxEmitter property), 115
 unknown_15 () (pyffi.formats.nif.NifFormat.NiPSGravityForceController property), 118
 unknown_15 () (pyffi.formats.nif.NifFormat.NiPSMeshEmitter property), 119
 unknown_15 () (pyffi.formats.nif.NifFormat.NiPSParticleSystem property), 120
 unknown_15 () (pyffi.formats.nif.NifFormat.NiPSSphereEmitter property), 123
 unknown_16 () (pyffi.formats.nif.NifFormat.NiPSBoxEmitter property), 115
 unknown_16 () (pyffi.formats.nif.NifFormat.NiPSGravityForceController property), 118
 unknown_16 () (pyffi.formats.nif.NifFormat.NiPSMeshEmitter property), 119
 unknown_16 () (pyffi.formats.nif.NifFormat.NiPSParticleSystem property), 120
 unknown_16 () (pyffi.formats.nif.NifFormat.NiPSSphereEmitter property), 123
 unknown_17 () (pyffi.formats.nif.NifFormat.NiPSBoxEmitter property), 115
 unknown_17 () (pyffi.formats.nif.NifFormat.NiPSGravityForceController property), 118
 unknown_17 () (pyffi.formats.nif.NifFormat.NiPSMeshEmitter property), 119
 unknown_17 () (pyffi.formats.nif.NifFormat.NiPSParticleSystem property), 120
 unknown_17 () (pyffi.formats.nif.NifFormat.NiPSSphereEmitter property), 123
 unknown_18 () (pyffi.formats.nif.NifFormat.NiPSBoxEmitter property), 115
 unknown_18 () (pyffi.formats.nif.NifFormat.NiPSGravityForceController property), 118
 unknown_18 () (pyffi.formats.nif.NifFormat.NiPSMeshEmitter property), 119
 unknown_18 () (pyffi.formats.nif.NifFormat.NiPSSphereEmitter property), 123
 unknown_19 () (pyffi.formats.nif.NifFormat.NiPSBoxEmitter property), 115
 unknown_19 () (pyffi.formats.nif.NifFormat.NiPSGravityForceController property), 118
 unknown_19 () (pyffi.formats.nif.NifFormat.NiPSMeshEmitter property), 119
 unknown_19 () (pyffi.formats.nif.NifFormat.NiPSParticleSystem property), 120
 unknown_19 () (pyffi.formats.nif.NifFormat.NiPSSphereEmitter property), 123
 unknown_20 () (pyffi.formats.nif.NifFormat.NiPSBoxEmitter property), 115
 unknown_20 () (pyffi.formats.nif.NifFormat.NiPSGravityForceController property), 118
 unknown_20 () (pyffi.formats.nif.NifFormat.NiPSMeshEmitter property), 119
 unknown_20 () (pyffi.formats.nif.NifFormat.NiPSParticleSystem property), 120
 unknown_20 () (pyffi.formats.nif.NifFormat.NiPSSphereEmitter property), 123
 unknown_200 () (pyffi.formats.nif.NifFormat.NiMesh property), 108
 unknown_201 () (pyffi.formats.nif.NifFormat.NiMesh property), 108
 unknown_21 () (pyffi.formats.nif.NifFormat.NiPSBoxEmitter property), 115

unknown_21 () (pyffi.formats.nif.NifFormat.NiPSGravityForce property), 118
 unknown_21 () (pyffi.formats.nif.NifFormat.NiPSMeshEmitter property), 119
 unknown_21 () (pyffi.formats.nif.NifFormat.NiPSParticleSystem property), 120
 unknown_21 () (pyffi.formats.nif.NifFormat.NiPSSphereEmitter property), 123
 unknown_22 () (pyffi.formats.nif.NifFormat.NiPSBoxEmitter property), 115
 unknown_22 () (pyffi.formats.nif.NifFormat.NiPSGravityForce property), 118
 unknown_22 () (pyffi.formats.nif.NifFormat.NiPSMeshEmitter property), 119
 unknown_22 () (pyffi.formats.nif.NifFormat.NiPSParticleSystem property), 120
 unknown_22 () (pyffi.formats.nif.NifFormat.NiPSSphereEmitter property), 123
 unknown_23 () (pyffi.formats.nif.NifFormat.NiPSBoxEmitter property), 115
 unknown_23 () (pyffi.formats.nif.NifFormat.NiPSCylinderEmitter property), 116
 unknown_23 () (pyffi.formats.nif.NifFormat.NiPSGravityForce property), 118
 unknown_23 () (pyffi.formats.nif.NifFormat.NiPSMeshEmitter property), 119
 unknown_23 () (pyffi.formats.nif.NifFormat.NiPSMeshParticleSystem property), 120
 unknown_24 () (pyffi.formats.nif.NifFormat.NiPSBoxEmitter property), 115
 unknown_24 () (pyffi.formats.nif.NifFormat.NiPSGravityForce property), 118
 unknown_24 () (pyffi.formats.nif.NifFormat.NiPSMeshEmitter property), 119
 unknown_24 () (pyffi.formats.nif.NifFormat.NiPSMeshParticleSystem property), 120
 unknown_25 () (pyffi.formats.nif.NifFormat.NiPSBoxEmitter property), 115
 unknown_25 () (pyffi.formats.nif.NifFormat.NiPSGravityForce property), 118
 unknown_25 () (pyffi.formats.nif.NifFormat.NiPSMeshEmitter property), 119
 unknown_25 () (pyffi.formats.nif.NifFormat.NiPSMeshParticleSystem property), 120
 unknown_250 () (pyffi.formats.nif.NifFormat.NiMesh property), 108
 unknown_251 () (pyffi.formats.nif.NifFormat.NiMesh property), 108
 unknown_26 () (pyffi.formats.nif.NifFormat.NiPSBoxEmitter property), 115
 unknown_26 () (pyffi.formats.nif.NifFormat.NiPSGravityForce property), 118
 unknown_26 () (pyffi.formats.nif.NifFormat.NiPSMeshEmitter property), 119
 unknown_26 () (pyffi.formats.nif.NifFormat.NiPSMeshParticleSystem property), 120
 unknown_27 () (pyffi.formats.nif.NifFormat.NiPSBoxEmitter property), 115
 unknown_27 () (pyffi.formats.nif.NifFormat.NiPSGravityForce property), 118
 unknown_27 () (pyffi.formats.nif.NifFormat.NiPSMeshEmitter property), 119
 unknown_27 () (pyffi.formats.nif.NifFormat.NiPSParticleSystem property), 120
 unknown_28 () (pyffi.formats.nif.NifFormat.NiPSBoxEmitter property), 115
 unknown_28 () (pyffi.formats.nif.NifFormat.NiPSGravityForce property), 118
 unknown_28 () (pyffi.formats.nif.NifFormat.NiPSMeshEmitter property), 120
 unknown_28 () (pyffi.formats.nif.NifFormat.NiPSParticleSystem property), 120
 unknown_29 () (pyffi.formats.nif.NifFormat.NiPSBoxEmitter property), 115
 unknown_29 () (pyffi.formats.nif.NifFormat.NiPSGravityForce property), 118
 unknown_29 () (pyffi.formats.nif.NifFormat.NiPSParticleSystem property), 120
 unknown_292_bytes () (pyffi.formats.nif.NifFormat.FxWidget property), 79
 unknown_2_float () (pyffi.formats.nif.NifFormat.NiTexturingProperty property), 149
 unknown_2_texture () (pyffi.formats.nif.NifFormat.NiTexturingProperty property), 149
 unknown_3 () (pyffi.formats.nif.NifFormat.FxWidget property), 79
 unknown_3 () (pyffi.formats.nif.NifFormat.NiBezierMesh property), 94
 unknown_3 () (pyffi.formats.nif.NifFormat.NiBezierTriangle4 property), 94
 unknown_3 () (pyffi.formats.nif.NifFormat.NiPSBombForce property), 115
 unknown_3 () (pyffi.formats.nif.NifFormat.NiPSBoxEmitter property), 115
 unknown_3 () (pyffi.formats.nif.NifFormat.NiPSDragForce property), 116
 unknown_3 () (pyffi.formats.nif.NifFormat.NiPSEmitterSpeedCtrl property), 117
 unknown_3 () (pyffi.formats.nif.NifFormat.NiPSFacingQuadGenerator property), 117
 unknown_3 () (pyffi.formats.nif.NifFormat.NiPSGravityForce property), 118
 unknown_3 () (pyffi.formats.nif.NifFormat.NiPSGravityStrengthCtrl property), 119
 unknown_3 () (pyffi.formats.nif.NifFormat.NiPSMeshEmitter property), 119

property), 116
 unknown_45 () (pyffi.formats.nif.NifFormat.NiPSBoxEmitter
 property), 116
 unknown_46 () (pyffi.formats.nif.NifFormat.NiPSBoxEmitter
 property), 116
 unknown_47 () (pyffi.formats.nif.NifFormat.NiPSBoxEmitter
 property), 116
 unknown_48 () (pyffi.formats.nif.NifFormat.NiPSBoxEmitter
 property), 116
 unknown_4_bytes ()
 (pyffi.formats.nif.NifFormat.BallAndSocketDescriptor
 property), 62
 unknown_4_bytes ()
 (pyffi.formats.nif.NifFormat.bhkCompressedMeshShape
 property), 180
 unknown_5 () (pyffi.formats.nif.NifFormat.NiBezierMesh
 property), 94
 unknown_5 () (pyffi.formats.nif.NifFormat.NiBezierTriangle4
 property), 94
 unknown_5 () (pyffi.formats.nif.NifFormat.NiPSBombForce
 property), 115
 unknown_5 () (pyffi.formats.nif.NifFormat.NiPSBoxEmitter
 property), 116
 unknown_5 () (pyffi.formats.nif.NifFormat.NiPSDragForce
 property), 116
 unknown_5 () (pyffi.formats.nif.NifFormat.NiPSFacingQuadGenerator
 property), 118
 unknown_5 () (pyffi.formats.nif.NifFormat.NiPSGravityForce
 property), 119
 unknown_5 () (pyffi.formats.nif.NifFormat.NiPSMeshEmitter
 property), 120
 unknown_5 () (pyffi.formats.nif.NifFormat.NiPSParticleSystem
 property), 121
 unknown_5 () (pyffi.formats.nif.NifFormat.NiPSSphereEmitter
 property), 123
 unknown_5 () (pyffi.formats.nif.NifFormat.NiPSSphericalCollider
 property), 123
 unknown_51 () (pyffi.formats.nif.NifFormat.NiMesh
 property), 108
 unknown_52 () (pyffi.formats.nif.NifFormat.NiMesh
 property), 108
 unknown_53 () (pyffi.formats.nif.NifFormat.NiMesh
 property), 108
 unknown_54 () (pyffi.formats.nif.NifFormat.NiMesh
 property), 108
 unknown_55 () (pyffi.formats.nif.NifFormat.NiMesh
 property), 108
 unknown_56 () (pyffi.formats.nif.NifFormat.NiMesh
 property), 108
 unknown_5_ints () (pyffi.formats.nif.NifFormat.NiBinaryVoxelData
 property), 95
 unknown_6 () (pyffi.formats.nif.NifFormat.NiBezierMesh
 property), 94
 unknown_6 () (pyffi.formats.nif.NifFormat.NiBezierTriangle4
 property), 94
 unknown_6 () (pyffi.formats.nif.NifFormat.NiPSBombForce
 property), 115
 unknown_6 () (pyffi.formats.nif.NifFormat.NiPSBoxEmitter
 property), 116
 unknown_6 () (pyffi.formats.nif.NifFormat.NiPSDragForce
 property), 116
 unknown_6 () (pyffi.formats.nif.NifFormat.NiPSFacingQuadGenerator
 property), 118
 unknown_6 () (pyffi.formats.nif.NifFormat.NiPSGravityForce
 property), 119
 unknown_6 () (pyffi.formats.nif.NifFormat.NiPSMeshEmitter
 property), 120
 unknown_6 () (pyffi.formats.nif.NifFormat.NiPSParticleSystem
 property), 121
 unknown_6 () (pyffi.formats.nif.NifFormat.NiPSSphereEmitter
 property), 123
 unknown_6 () (pyffi.formats.nif.NifFormat.NiPSSphericalCollider
 property), 123
 unknown_7 () (pyffi.formats.nif.NifFormat.NiPSBombForce
 property), 115
 unknown_7 () (pyffi.formats.nif.NifFormat.NiPSBoxEmitter
 property), 116
 unknown_7 () (pyffi.formats.nif.NifFormat.NiPSDragForce
 property), 117
 unknown_7 () (pyffi.formats.nif.NifFormat.NiPSFacingQuadGenerator
 property), 118
 unknown_7 () (pyffi.formats.nif.NifFormat.NiPSGravityForce
 property), 119
 unknown_7 () (pyffi.formats.nif.NifFormat.NiPSMeshEmitter
 property), 120
 unknown_7 () (pyffi.formats.nif.NifFormat.NiPSParticleSystem
 property), 121
 unknown_7 () (pyffi.formats.nif.NifFormat.NiPSSphereEmitter
 property), 123
 unknown_7 () (pyffi.formats.nif.NifFormat.NiPSSphericalCollider
 property), 123
 unknown_7_floats ()
 (pyffi.formats.nif.NifFormat.NiBinaryVoxelData
 property), 95
 unknown_8 () (pyffi.formats.nif.NifFormat.NiPSBombForce
 property), 115
 unknown_8 () (pyffi.formats.nif.NifFormat.NiPSBoxEmitter
 property), 116
 unknown_8 () (pyffi.formats.nif.NifFormat.NiPSDragForce
 property), 117
 unknown_8 () (pyffi.formats.nif.NifFormat.NiPSFacingQuadGenerator
 property), 118
 unknown_8 () (pyffi.formats.nif.NifFormat.NiPSGravityForce
 property), 119
 unknown_8 () (pyffi.formats.nif.NifFormat.NiPSMeshEmitter
 property), 120
 unknown_8 () (pyffi.formats.nif.NifFormat.NiPSParticleSystem
 property), 121

unknown_8 () (pyffi.formats.nif.NifFormat.NiPSSphereEmitter property), 46
 property), 123
 unknown_9 () (pyffi.formats.nif.NifFormat.NiPSBombForce property), 46
 property), 115
 unknown_9 () (pyffi.formats.nif.NifFormat.NiPSBoxEmitter property), 57
 property), 116
 unknown_9 () (pyffi.formats.nif.NifFormat.NiPSDragForce property), 65
 property), 117
 unknown_9 () (pyffi.formats.nif.NifFormat.NiPSFacingQuadGenerator property), 87
 property), 118
 unknown_9 () (pyffi.formats.nif.NifFormat.NiPSGravityForce property), 135
 property), 119
 unknown_9 () (pyffi.formats.nif.NifFormat.NiPSMeshEmitter property), 136
 property), 120
 unknown_9 () (pyffi.formats.nif.NifFormat.NiPSParticleSystem property), 137
 property), 121
 unknown_9 () (pyffi.formats.nif.NifFormat.NiPSSphereEmitter property), 137
 property), 123
 unknown_array () (pyffi.formats.nif.NifFormat.Ni3dsAnimationNode property), 139
 property), 88
 unknown_boolean_1 ()
 (pyffi.formats.nif.NifFormat.NiPSysAirFieldModifier property), 124
 unknown_boolean_2 ()
 (pyffi.formats.nif.NifFormat.NiPSysAirFieldModifier property), 124
 unknown_boolean_3 ()
 (pyffi.formats.nif.NifFormat.NiPSysAirFieldModifier property), 124
 unknown_byte () (pyffi.formats.nif.NifFormat.BSValueNode property), 61
 unknown_byte () (pyffi.formats.nif.NifFormat.NiArkTextureExtraData property), 90
 unknown_byte () (pyffi.formats.nif.NifFormat.NiPalette property), 132
 unknown_byte () (pyffi.formats.nif.NifFormat.NiParticleSystemController property), 134
 unknown_byte () (pyffi.formats.nif.NifFormat.NiPhysXProp property), 137
 unknown_byte () (pyffi.formats.nif.NifFormat.NiPSysGravityModifier property), 128
 unknown_byte () (pyffi.formats.nif.NifFormat.NiSourceTexture property), 145
 unknown_byte () (pyffi.formats.nif.NifFormat.TexSource property), 174
 unknown_byte_1 () (pyffi.formats.nif.NifFormat.AdditionalDataInfo property), 44
 unknown_byte_1 () (pyffi.formats.nif.NifFormat.bhkCompressedMeshShapeData property), 180
 unknown_byte_1 () (pyffi.formats.nif.NifFormat.bhkCompressedMeshShapeData property), 181
 unknown_byte_1 () (pyffi.formats.nif.NifFormat.BSBlastNode property), 45
 unknown_byte_1 () (pyffi.formats.nif.NifFormat.BSDamageStage property), 46
 unknown_byte_1 () (pyffi.formats.nif.NifFormat.BSDebrisNode property), 46
 unknown_byte_1 () (pyffi.formats.nif.NifFormat.BSSegment property), 57
 unknown_byte_1 () (pyffi.formats.nif.NifFormat.ChannelData property), 65
 unknown_byte_1 () (pyffi.formats.nif.NifFormat.MotorDescriptor property), 87
 unknown_byte_1 () (pyffi.formats.nif.NifFormat.NiParticlesData property), 135
 unknown_byte_1 () (pyffi.formats.nif.NifFormat.NiPhysXActorDesc property), 136
 unknown_byte_1 () (pyffi.formats.nif.NifFormat.NiPhysXMaterialDesc property), 137
 unknown_byte_1 () (pyffi.formats.nif.NifFormat.NiPhysXMeshDesc property), 137
 unknown_byte_1 () (pyffi.formats.nif.NifFormat.NiPhysXTransformDesc property), 139
 unknown_byte_1 () (pyffi.formats.nif.NifFormat.physXMaterialRef property), 189
 unknown_byte_1 () (pyffi.formats.nif.NifFormat.PrismaticDescriptor property), 159
 unknown_byte_2 () (pyffi.formats.nif.NifFormat.bhkCompressedMeshShapeData property), 180
 unknown_byte_2 () (pyffi.formats.nif.NifFormat.NiParticlesData property), 135
 unknown_byte_2 () (pyffi.formats.nif.NifFormat.NiPhysXActorDesc property), 136
 unknown_byte_2 () (pyffi.formats.nif.NifFormat.NiPhysXMaterialDesc property), 137
 unknown_byte_2 () (pyffi.formats.nif.NifFormat.NiPhysXMeshDesc property), 137
 unknown_byte_2 () (pyffi.formats.nif.NifFormat.NiPhysXTransformDesc property), 139
 unknown_byte_3 () (pyffi.formats.nif.NifFormat.NiMeshPSysData property), 109
 unknown_byte_4 () (pyffi.formats.nif.NifFormat.NiPSPlanarCollider property), 121
 unknown_byte_4 () (pyffi.formats.nif.NifFormat.NiPSysData property), 125
 unknown_byte_6 () (pyffi.formats.nif.NifFormat.BSStripPSysData property), 60
 unknown_byte_6 () (pyffi.formats.nif.NifFormat.NiPhysXPropDesc property), 138
 unknown_bytes () (pyffi.formats.nif.NifFormat.ArkTexture property), 45
 unknown_bytes () (pyffi.formats.nif.NifFormat.CStreamableAssetData property), 65
 unknown_bytes () (pyffi.formats.nif.NifFormat.NiArkAnimationExtraData property), 90
 unknown_bytes () (pyffi.formats.nif.NifFormat.NiArkImporterExtraData property), 90
 unknown_bytes () (pyffi.formats.nif.NifFormat.NiArkViewportInfoExtraData property), 90

property), 91
 unknown_bytes() (pyffi.formats.nif.NifFormat.NiFloatExtraDataController
 property), 101
 unknown_bytes() (pyffi.formats.nif.NifFormat.NiPhysXBodyDesc
 property), 136
 unknown_bytes() (pyffi.formats.nif.NifFormat.NiPhysXJointDesc
 property), 136
 unknown_bytes() (pyffi.formats.nif.NifFormat.NiPhysXKinematicSrc
 property), 136
 unknown_bytes_0()
 (pyffi.formats.nif.NifFormat.NiPhysXMeshDesc
 property), 137
 unknown_bytes_1()
 (pyffi.formats.nif.NifFormat.NiBinaryVoxelData
 property), 95
 unknown_bytes_1()
 (pyffi.formats.nif.NifFormat.NiPhysXMeshDesc
 property), 137
 unknown_bytes_2()
 (pyffi.formats.nif.NifFormat.NiBinaryVoxelData
 property), 95
 unknown_bytes_2()
 (pyffi.formats.nif.NifFormat.NiPhysXMeshDesc
 property), 137
 unknown_bytes_3()
 (pyffi.formats.nif.NifFormat.NiPhysXMeshDesc
 property), 137
 unknown_clod_shorts_1()
 (pyffi.formats.nif.NifFormat.NiClodData
 property), 97
 unknown_clod_shorts_2()
 (pyffi.formats.nif.NifFormat.NiClodData
 property), 97
 unknown_clod_shorts_3()
 (pyffi.formats.nif.NifFormat.NiClodData
 property), 97
 unknown_color() (pyffi.formats.nif.NifFormat.NiParticleSystemController
 property), 134
 unknown_count_1()
 (pyffi.formats.nif.NifFormat.NiClodData
 property), 97
 unknown_count_2()
 (pyffi.formats.nif.NifFormat.NiClodData
 property), 97
 unknown_count_3()
 (pyffi.formats.nif.NifFormat.NiClodData
 property), 97
 unknown_extra_bytes()
 (pyffi.formats.nif.NifFormat.NiFloatExtraDataController
 property), 101
 unknown_flags() (pyffi.formats.nif.NifFormat.NiPortal
 property), 140
 unknown_flags() (pyffi.formats.nif.NifFormat.NiShadowGenerator
 property), 143
 unknown_flags_1()
 (pyffi.formats.nif.NifFormat.NiSwitchNode
 property), 147
 unknown_float() (pyffi.formats.nif.NifFormat.bhkSimpleShapePhantom
 property), 187
 unknown_float() (pyffi.formats.nif.NifFormat.NiClodData
 property), 97
 unknown_float() (pyffi.formats.nif.NifFormat.NiFurSpringController
 property), 101
 unknown_float() (pyffi.formats.nif.NifFormat.NiImage
 property), 105
 unknown_float() (pyffi.formats.nif.NifFormat.NiSpotLight
 property), 146
 unknown_float() (pyffi.formats.nif.NifFormat.NiTextureEffect
 property), 147
 unknown_float() (pyffi.formats.nif.NifFormat.NiVectorExtraData
 property), 153
 unknown_float_1()
 (pyffi.formats.nif.NifFormat.bhkBallSocketConstraintChain
 property), 178
 unknown_float_1()
 (pyffi.formats.nif.NifFormat.bhkBlendCollisionObject
 property), 178
 unknown_float_1()
 (pyffi.formats.nif.NifFormat.bhkCompressedMeshShape
 property), 180
 unknown_float_1()
 (pyffi.formats.nif.NifFormat.bhkConvexListShape
 property), 181
 unknown_float_1()
 (pyffi.formats.nif.NifFormat.bhkLiquidAction
 property), 182
 unknown_float_1()
 (pyffi.formats.nif.NifFormat.BSDecalPlacementVectorExtraData
 property), 46
 unknown_float_1()
 (pyffi.formats.nif.NifFormat.BSPSysInheritVelocityModifier
 property), 53
 unknown_float_1()
 (pyffi.formats.nif.NifFormat.BSPSysRecycleBoundModifier
 property), 54
 unknown_float_1()
 (pyffi.formats.nif.NifFormat.CapsuleBV
 property), 65
 unknown_float_1()
 (pyffi.formats.nif.NifFormat.HalfSpaceBV
 property), 79
 unknown_float_1()
 (pyffi.formats.nif.NifFormat.MotorDescriptor
 property), 87
 unknown_float_1()
 (pyffi.formats.nif.NifFormat.NiGravity
 property), 105
 unknown_float_1()

<i>(pyffi.formats.nif.NifFormat.NiParticleSystemController property)</i> , 134	<i>(pyffi.formats.nif.NifFormat.bhkBlendCollisionObject property)</i> , 178
unknown_float_1 ()	unknown_float_2 ()
<i>(pyffi.formats.nif.NifFormat.NiPathInterpolator property)</i> , 135	<i>(pyffi.formats.nif.NifFormat.bhkLiquidAction property)</i> , 182
unknown_float_1 ()	unknown_float_2 ()
<i>(pyffi.formats.nif.NifFormat.NiPhysXMeshDesc property)</i> , 137	<i>(pyffi.formats.nif.NifFormat.BSPSysInheritVelocityModifier property)</i> , 53
unknown_float_1 ()	unknown_float_2 ()
<i>(pyffi.formats.nif.NifFormat.NiPhysXProp property)</i> , 137	<i>(pyffi.formats.nif.NifFormat.BSPSysRecycleBoundModifier property)</i> , 54
unknown_float_1 ()	unknown_float_2 ()
<i>(pyffi.formats.nif.NifFormat.NiPhysXShapeDesc property)</i> , 138	<i>(pyffi.formats.nif.NifFormat.CapsuleBV property)</i> , 65
unknown_float_1 ()	unknown_float_2 ()
<i>(pyffi.formats.nif.NifFormat.NiPlanarCollider property)</i> , 139	<i>(pyffi.formats.nif.NifFormat.MotorDescriptor property)</i> , 87
unknown_float_1 ()	unknown_float_2 ()
<i>(pyffi.formats.nif.NifFormat.NiPSysTrailEmitter property)</i> , 131	<i>(pyffi.formats.nif.NifFormat.NiFurSpringController property)</i> , 101
unknown_float_1 ()	unknown_float_2 ()
<i>(pyffi.formats.nif.NifFormat.NiSphericalCollider property)</i> , 146	<i>(pyffi.formats.nif.NifFormat.NiPathController property)</i> , 135
unknown_float_1 ()	unknown_float_2 ()
<i>(pyffi.formats.nif.NifFormat.ParticleDesc property)</i> , 157	<i>(pyffi.formats.nif.NifFormat.NiPathInterpolator property)</i> , 135
unknown_float_10 ()	unknown_float_2 ()
<i>(pyffi.formats.nif.NifFormat.NiPlanarCollider property)</i> , 139	<i>(pyffi.formats.nif.NifFormat.NiPhysXMeshDesc property)</i> , 137
unknown_float_11 ()	unknown_float_2 ()
<i>(pyffi.formats.nif.NifFormat.NiPlanarCollider property)</i> , 139	<i>(pyffi.formats.nif.NifFormat.NiPhysXShapeDesc property)</i> , 138
unknown_float_12 ()	unknown_float_2 ()
<i>(pyffi.formats.nif.NifFormat.NiPlanarCollider property)</i> , 139	<i>(pyffi.formats.nif.NifFormat.NiPlanarCollider property)</i> , 139
unknown_float_13 ()	unknown_float_2 ()
<i>(pyffi.formats.nif.NifFormat.NiParticleSystemController property)</i> , 134	<i>(pyffi.formats.nif.NifFormat.NiPSysAirFieldModifier property)</i> , 124
unknown_float_13 ()	unknown_float_2 ()
<i>(pyffi.formats.nif.NifFormat.NiPlanarCollider property)</i> , 139	<i>(pyffi.formats.nif.NifFormat.NiPSysTrailEmitter property)</i> , 131
unknown_float_14 ()	unknown_float_2 ()
<i>(pyffi.formats.nif.NifFormat.NiPlanarCollider property)</i> , 139	<i>(pyffi.formats.nif.NifFormat.NiSphericalCollider property)</i> , 146
unknown_float_15 ()	unknown_float_2 ()
<i>(pyffi.formats.nif.NifFormat.NiPlanarCollider property)</i> , 139	<i>(pyffi.formats.nif.NifFormat.ParticleDesc property)</i> , 157
unknown_float_16 ()	unknown_float_3 ()
<i>(pyffi.formats.nif.NifFormat.NiPlanarCollider property)</i> , 139	<i>(pyffi.formats.nif.NifFormat.bhkCompressedMeshShape property)</i> , 180
unknown_float_2 ()	unknown_float_3 ()
<i>(pyffi.formats.nif.NifFormat.bhkBallSocketConstraintChain property)</i> , 178	<i>(pyffi.formats.nif.NifFormat.bhkLiquidAction property)</i> , 182
unknown_float_2 ()	unknown_float_3 ()

<i>(pyffi.formats.nif.NifFormat.BSPSysInheritVelocityModifier</i>	<i>(pyffi.formats.nif.NifFormat.NiSphericalCollider</i>
<i>property), 53</i>	<i>property), 146</i>
unknown_float_3()	unknown_float_5()
<i>(pyffi.formats.nif.NifFormat.BSPSysRecycleBoundModifier</i>	<i>(pyffi.formats.nif.NifFormat.bhkCompressedMeshShape</i>
<i>property), 54</i>	<i>property), 180</i>
unknown_float_3()	unknown_float_5()
<i>(pyffi.formats.nif.NifFormat.MotorDescriptor</i>	<i>(pyffi.formats.nif.NifFormat.BSPSysRecycleBoundModifier</i>
<i>property), 87</i>	<i>property), 54</i>
unknown_float_3()	unknown_float_5()
<i>(pyffi.formats.nif.NifFormat.NiPathController</i>	<i>(pyffi.formats.nif.NifFormat.BSShaderPPLightingProperty</i>
<i>property), 135</i>	<i>property), 59</i>
unknown_float_3()	unknown_float_5()
<i>(pyffi.formats.nif.NifFormat.NiPhysXShapeDesc</i>	<i>(pyffi.formats.nif.NifFormat.MotorDescriptor</i>
<i>property), 138</i>	<i>property), 87</i>
unknown_float_3()	unknown_float_5()
<i>(pyffi.formats.nif.NifFormat.NiPlanarCollider</i>	<i>(pyffi.formats.nif.NifFormat.NiPlanarCollider</i>
<i>property), 139</i>	<i>property), 139</i>
unknown_float_3()	unknown_float_5()
<i>(pyffi.formats.nif.NifFormat.NiPSysAirFieldModifier</i>	<i>(pyffi.formats.nif.NifFormat.NiPSysTrailEmitter</i>
<i>property), 124</i>	<i>property), 131</i>
unknown_float_3()	unknown_float_5()
<i>(pyffi.formats.nif.NifFormat.NiPSysTrailEmitter</i>	<i>(pyffi.formats.nif.NifFormat.NiSphericalCollider</i>
<i>property), 131</i>	<i>property), 146</i>
unknown_float_3()	unknown_float_6()
<i>(pyffi.formats.nif.NifFormat.NiSphericalCollider</i>	<i>(pyffi.formats.nif.NifFormat.BSPSysRecycleBoundModifier</i>
<i>property), 146</i>	<i>property), 54</i>
unknown_float_3()	unknown_float_6()
<i>(pyffi.formats.nif.NifFormat.ParticleDesc</i>	<i>(pyffi.formats.nif.NifFormat.MotorDescriptor</i>
<i>property), 157</i>	<i>property), 87</i>
unknown_float_4()	unknown_float_6()
<i>(pyffi.formats.nif.NifFormat.bhkCompressedMeshShape</i>	<i>(pyffi.formats.nif.NifFormat.NiPlanarCollider</i>
<i>property), 180</i>	<i>property), 139</i>
unknown_float_4()	unknown_float_6()
<i>(pyffi.formats.nif.NifFormat.bhkLiquidAction</i>	<i>(pyffi.formats.nif.NifFormat.NiPSysTrailEmitter</i>
<i>property), 182</i>	<i>property), 131</i>
unknown_float_4()	unknown_float_7()
<i>(pyffi.formats.nif.NifFormat.BSPSysRecycleBoundModifier</i>	<i>(pyffi.formats.nif.NifFormat.NiPlanarCollider</i>
<i>property), 54</i>	<i>property), 139</i>
unknown_float_4()	unknown_float_7()
<i>(pyffi.formats.nif.NifFormat.BSShaderPPLightingProperty</i>	<i>(pyffi.formats.nif.NifFormat.NiPSysTrailEmitter</i>
<i>property), 59</i>	<i>property), 131</i>
unknown_float_4()	unknown_float_8()
<i>(pyffi.formats.nif.NifFormat.MotorDescriptor</i>	<i>(pyffi.formats.nif.NifFormat.BSStripPSysData</i>
<i>property), 87</i>	<i>property), 60</i>
unknown_float_4()	unknown_float_8()
<i>(pyffi.formats.nif.NifFormat.NiPlanarCollider</i>	<i>(pyffi.formats.nif.NifFormat.NiPlanarCollider</i>
<i>property), 139</i>	<i>property), 139</i>
unknown_float_4()	unknown_float_9()
<i>(pyffi.formats.nif.NifFormat.NiPSysAirFieldModifier</i>	<i>(pyffi.formats.nif.NifFormat.NiPlanarCollider</i>
<i>property), 124</i>	<i>property), 139</i>
unknown_float_4()	unknown_floats()
<i>(pyffi.formats.nif.NifFormat.NiPSysTrailEmitter</i>	<i>(pyffi.formats.nif.NifFormat.bhkConvexListShape</i>
<i>property), 131</i>	<i>property), 181</i>
unknown_float_4()	unknown_floats()
	<i>(pyffi.formats.nif.NifFormat.bhkMeshShape</i>
	<i>property), 182</i>


```

unknown_floats() (pyffi.formats.nif.NifFormat.NiArkImporterExtraData (pyffi.formats.nif.NifFormat.NiCamera
    property), 90
    property), 97
unknown_floats() (pyffi.formats.nif.NifFormat.NiBSplinePrimitiveInterpolator (pyffi.formats.nif.NifFormat.NiDefaultAVObjectPalette
    property), 93
    property), 99
unknown_floats_1() (pyffi.formats.nif.NifFormat.BallAndSocketDescriptor (pyffi.formats.nif.NifFormat.NiImage
    property), 62
    property), 105
unknown_floats_1() (pyffi.formats.nif.NifFormat.bhkCompressedMeshShape (pyffi.formats.nif.NifFormat.NiMultiTextureProperty
    property), 180
    property), 110
unknown_floats_1() (pyffi.formats.nif.NifFormat.Ni3dsAnimationNode (pyffi.formats.nif.NifFormat.NiPathInterpolator
    property), 88
    property), 135
unknown_floats_1() (pyffi.formats.nif.NifFormat.ParticleDesc (pyffi.formats.nif.NifFormat.NiPhysXMaterialDesc
    property), 157
    property), 137
unknown_floats_1() (pyffi.formats.nif.NifFormat.Ni3dsAnimationNode (pyffi.formats.nif.NifFormat.NiPhysXProp
    property), 88
    property), 137
unknown_floats_1() (pyffi.formats.nif.NifFormat.NiPSysData (pyffi.formats.nif.NifFormat.NiPSysSpawnModifier
    property), 125
    property), 130
unknown_floats_2() (pyffi.formats.nif.NifFormat.BallAndSocketDescriptor (pyffi.formats.nif.NifFormat.BallAndSocketDescriptor
    property), 62
    property), 62
unknown_floats_2() (pyffi.formats.nif.NifFormat.bhkSimpleShapePhantom (pyffi.formats.nif.NifFormat.bhkBallSocketConstraint
    property), 187
    property), 178
unknown_floats_2() (pyffi.formats.nif.NifFormat.Ni3dsAnimationNode (pyffi.formats.nif.NifFormat.bhkCMSDBigTris
    property), 88
    property), 178
unknown_floats_2() (pyffi.formats.nif.NifFormat.NiParticleSystemController (pyffi.formats.nif.NifFormat.bhkCompressedMeshShape
    property), 134
    property), 180
unknown_floats_2() (pyffi.formats.nif.NifFormat.NiPSPlanarCollider (pyffi.formats.nif.NifFormat.bhkLiquidAction
    property), 121
    property), 182
unknown_floats_3() (pyffi.formats.nif.NifFormat.NiPSysData (pyffi.formats.nif.NifFormat.BSMultiBoundSphere
    property), 125
    property), 53
unknown_floats_5() (pyffi.formats.nif.NifFormat.NiPSPlanarCollider (pyffi.formats.nif.NifFormat.BSPackedAdditionalData
    property), 121
    property), 55
unknown_int() (pyffi.formats.nif.NifFormat.bhkBlendController (pyffi.formats.nif.NifFormat.BSPSysInheritVelocityM
    property), 178
    property), 53
unknown_int() (pyffi.formats.nif.NifFormat.bhkRagdollTemplateData (pyffi.formats.nif.NifFormat.BSPSysMultiTargetEmitt
    property), 186
    property), 54
unknown_int() (pyffi.formats.nif.NifFormat.bhkRDTConstraint (pyffi.formats.nif.NifFormat.BSPSysRecycleBoundMe
    property), 185
    property), 54
unknown_int() (pyffi.formats.nif.NifFormat.bhkRDTMalleableConstraint (pyffi.formats.nif.NifFormat.ExtraMeshDataEpicMic
    property), 186
    property), 73
unknown_int() (pyffi.formats.nif.NifFormat.BoundingBox (pyffi.formats.nif.NifFormat.FxRadioButton
    property), 63
    property), 78
unknown_int() (pyffi.formats.nif.NifFormat.BSMultiBoundNode (pyffi.formats.nif.NifFormat.NiArkImporterExtraData
    property), 53
    property), 90
unknown_int() (pyffi.formats.nif.NifFormat.Morph (pyffi.formats.nif.NifFormat.NiBoneLODController
    property), 86
    property), 96
unknown_int() (pyffi.formats.nif.NifFormat.NiArkShadeExtraData (pyffi.formats.nif.NifFormat.NiParticleSystem
    property), 90
    property), 133
unknown_int() (pyffi.formats.nif.NifFormat.NiBinaryVoxelExtraData (pyffi.formats.nif.NifFormat.NiParticleSystemControll
    property), 95
    property), 134
unknown_int() (pyffi.formats.nif.NifFormat.NiBlendInterpolator (pyffi.formats.nif.NifFormat.NiPathController
    property), 95
    property), 135
unknown_int() (pyffi.formats.nif.NifFormat.NiBlendInterpolator (pyffi.formats.nif.NifFormat.NiPhysXActorDesc
    property), 95
    property), 136
unknown_int() (pyffi.formats.nif.NifFormat.NiBlendInterpolator (pyffi.formats.nif.NifFormat.NiPhysXMeshDesc
    property), 95
    property), 137

```

unknown_int_1 () (pyffi.formats.nif.NifFormat.NiPhysXPropDesc_int_2 () (pyffi.formats.nif.NifFormat.NiPSPlanarCollider property), 137
property), 121
unknown_int_1 () (pyffi.formats.nif.NifFormat.NiPhysXPropDesc_int_2 () (pyffi.formats.nif.NifFormat.NiPSysTrailEmitter property), 138
property), 131
unknown_int_1 () (pyffi.formats.nif.NifFormat.NiPhysXShapeDesc_int_2 () (pyffi.formats.nif.NifFormat.NiSortAdjustNode property), 138
property), 145
unknown_int_1 () (pyffi.formats.nif.NifFormat.NiPSPlanarCollider_int_3 () (pyffi.formats.nif.NifFormat.ArkTexture property), 121
property), 45
unknown_int_1 () (pyffi.formats.nif.NifFormat.NiPSysTrailEmitter_int_3 () (pyffi.formats.nif.NifFormat.bhkBallSocketConstraint property), 131
property), 178
unknown_int_1 () (pyffi.formats.nif.NifFormat.NiSequence_int_3 () (pyffi.formats.nif.NifFormat.bhkCompressedMeshShape property), 142
property), 180
unknown_int_1 () (pyffi.formats.nif.NifFormat.NiSwitchNode_int_3 () (pyffi.formats.nif.NifFormat.bhkLiquidAction property), 147
property), 182
unknown_int_1 () (pyffi.formats.nif.NifFormat.NiTextKeyExtraData_int_3 () (pyffi.formats.nif.NifFormat.BSMultiBoundSphere property), 147
property), 53
unknown_int_1 () (pyffi.formats.nif.NifFormat.ParticleDesc_int_3 () (pyffi.formats.nif.NifFormat.ExtraMeshDataEpicMic property), 158
property), 73
unknown_int_12 () (pyffi.formats.nif.NifFormat.bhkCompressedMeshShapeData_int_3 () (pyffi.formats.nif.NifFormat.FxRadioButton property), 180
property), 78
unknown_int_2 () (pyffi.formats.nif.NifFormat.bhkBallSocketConstraintChain_int_3 () (pyffi.formats.nif.NifFormat.NiBoneLODController property), 178
property), 96
unknown_int_2 () (pyffi.formats.nif.NifFormat.bhkLiquidAction_int_3 () (pyffi.formats.nif.NifFormat.NiCamera property), 182
property), 97
unknown_int_2 () (pyffi.formats.nif.NifFormat.BSMultiBoundSphere_int_3 () (pyffi.formats.nif.NifFormat.NiPhysXPropDesc property), 53
property), 138
unknown_int_2 () (pyffi.formats.nif.NifFormat.ExtraMeshDataEpicMic_int_3 () (pyffi.formats.nif.NifFormat.NiPhysXShapeDesc property), 73
property), 138
unknown_int_2 () (pyffi.formats.nif.NifFormat.FxRadioButton_int_3 () (pyffi.formats.nif.NifFormat.NiPSysTrailEmitter property), 78
property), 131
unknown_int_2 () (pyffi.formats.nif.NifFormat.NiAlphaProperty_int_4 () (pyffi.formats.nif.NifFormat.ArkTexture property), 90
property), 45
unknown_int_2 () (pyffi.formats.nif.NifFormat.NiArkImporterExtraData_int_4 () (pyffi.formats.nif.NifFormat.bhkCompressedMeshShape property), 90
property), 181
unknown_int_2 () (pyffi.formats.nif.NifFormat.NiArkTextureExtraData_int_4 () (pyffi.formats.nif.NifFormat.ExtraMeshDataEpicMic property), 90
property), 73
unknown_int_2 () (pyffi.formats.nif.NifFormat.NiBoneLODController_int_4 () (pyffi.formats.nif.NifFormat.NiPhysXActorDesc property), 96
property), 136
unknown_int_2 () (pyffi.formats.nif.NifFormat.NiCamera_int_4 () (pyffi.formats.nif.NifFormat.NiPhysXMeshDesc property), 97
property), 137
unknown_int_2 () (pyffi.formats.nif.NifFormat.NiFlipController_int_4 () (pyffi.formats.nif.NifFormat.NiPhysXShapeDesc property), 100
property), 138
unknown_int_2 () (pyffi.formats.nif.NifFormat.NiMeshPSysData_int_4 () (pyffi.formats.nif.NifFormat.NiPSysData property), 109
property), 125
unknown_int_2 () (pyffi.formats.nif.NifFormat.NiParticleSystemController_int_4 () (pyffi.formats.nif.NifFormat.NiPSysTrailEmitter property), 134
property), 131
unknown_int_2 () (pyffi.formats.nif.NifFormat.NiPhysXActorDesc_int_4 () (pyffi.formats.nif.NifFormat.NiSequence property), 136
property), 142
unknown_int_2 () (pyffi.formats.nif.NifFormat.NiPhysXMeshDesc_int_5 () (pyffi.formats.nif.NifFormat.bhkCompressedMeshShape property), 137
property), 181
unknown_int_2 () (pyffi.formats.nif.NifFormat.NiPhysXPropDesc_int_5 () (pyffi.formats.nif.NifFormat.ExtraMeshDataEpicMic property), 138
property), 73
unknown_int_2 () (pyffi.formats.nif.NifFormat.NiPhysXShapeDesc_int_5 () (pyffi.formats.nif.NifFormat.NiPhysXActorDesc property), 138
property), 136

unknown_int_5 () (pyffi.formats.nif.NifFormat.NiPhysXPropDescproperty), 135
 property), 138
 unknown_int_5 () (pyffi.formats.nif.NifFormat.NiPhysXShapeDescproperty), 134
 property), 138
 unknown_int_5 () (pyffi.formats.nif.NifFormat.NiPSysDataproperty), 145
 property), 125
 unknown_int_5 () (pyffi.formats.nif.NifFormat.NiSequenceproperty), 174
 property), 142
 unknown_int_6 () (pyffi.formats.nif.NifFormat.bhkCompressedMeshShapeData), 107
 property), 181
 unknown_int_6 () (pyffi.formats.nif.NifFormat.ExtraMeshDataEpinkiekey), 107
 property), 73
 unknown_int_6 () (pyffi.formats.nif.NifFormat.NiPersistentSrcTextureRemote3DData), 136
 property), 136
 unknown_int_6 () (pyffi.formats.nif.NifFormat.NiPhysXActorDescproperty), 134
 property), 136
 unknown_int_6 () (pyffi.formats.nif.NifFormat.NiPSysDataproperty), 107
 property), 125
 unknown_int_7 () (pyffi.formats.nif.NifFormat.BSStripPSysDataproperty), 121
 property), 60
 unknown_int_7 () (pyffi.formats.nif.NifFormat.NiPersistentSrcTextureRemote3DData), 136
 property), 136
 unknown_int_7 () (pyffi.formats.nif.NifFormat.NiPhysXShapeDescmatrix () (pyffi.formats.nif.NifFormat.NiEnvMappedTriShapeproperty), 138
 property), 100
 unknown_int_8 () (pyffi.formats.nif.NifFormat.NiPhysXShapeDescnode () (pyffi.formats.nif.NifFormat.NiMeshPSysDataproperty), 138
 property), 109
 unknown_integer () (pyffi.formats.nif.NifFormat.NiTimeControllerproperty), 149
 property), 134
 unknown_ints () (pyffi.formats.nif.NifFormat.LODRangeproperty), 81
 property), 136
 unknown_ints () (pyffi.formats.nif.NifFormat.NiArkAnimationExtraDataproperty), 138
 property), 90
 unknown_ints () (pyffi.formats.nif.NifFormat.NiGeomMorpherComplementproperty), 136
 property), 102
 unknown_ints () (pyffi.formats.nif.NifFormat.NiTextureModePropertyproperty), 138
 property), 148
 unknown_ints_1 () (pyffi.formats.nif.NifFormat.bhkAabbPhantomproperty), 136
 property), 177
 unknown_ints_1 () (pyffi.formats.nif.NifFormat.bhkOrientHingedBodyActionproperty), 138
 property), 183
 unknown_ints_1 () (pyffi.formats.nif.NifFormat.NiArkTextureExtraDataproperty), 136
 property), 90
 unknown_ints_1 () (pyffi.formats.nif.NifFormat.NiMeshPSysDataproperty), 136
 property), 109
 unknown_ints_1 () (pyffi.formats.nif.NifFormat.NiPhysXMeshDescproperty), 136
 property), 137
 unknown_ints_1 () (pyffi.formats.nif.NifFormat.NiTexturePropertyproperty), 137
 property), 148
 unknown_ints_2 () (pyffi.formats.nif.NifFormat.NiTexturePropertyproperty), 136
 property), 148
 unknown_link () (pyffi.formats.nif.NifFormat.NiCameraproperty), 79
 property), 97
 unknown_link () (pyffi.formats.nif.NifFormat.NiParticlesDataproperty), 88

unknown_short () (pyffi.formats.nif.NifFormat.NiBlendInterpolator property), 95
 unknown_short () (pyffi.formats.nif.NifFormat.NiCamera property), 146
 unknown_short () (pyffi.formats.nif.NifFormat.NiClodData property), 98
 unknown_short () (pyffi.formats.nif.NifFormat.NiLookAtInterpolator property), 107
 unknown_short () (pyffi.formats.nif.NifFormat.NiPathController property), 135
 unknown_short () (pyffi.formats.nif.NifFormat.NiPathInterpolator property), 135
 unknown_short () (pyffi.formats.nif.NifFormat.NiPlanarCollider property), 139
 unknown_short () (pyffi.formats.nif.NifFormat.NiTextureEffect property), 147
 unknown_short () (pyffi.formats.nif.NifFormat.NiTextureModeProperty property), 148
 unknown_short () (pyffi.formats.nif.NifFormat.NiUVController property), 153
 unknown_short () (pyffi.formats.nif.NifFormat.Particle property), 157
 unknown_short () (pyffi.formats.nif.NifFormat.TexDesc property), 174
 unknown_short_1 () (pyffi.formats.nif.NifFormat.bhkCMSDBigTris property), 178
 unknown_short_1 () (pyffi.formats.nif.NifFormat.bhkCMSDChunk property), 179
 unknown_short_1 () (pyffi.formats.nif.NifFormat.BSAnimNotes property), 45
 unknown_short_1 () (pyffi.formats.nif.NifFormat.BSProceduralLightningController property), 56
 unknown_short_1 () (pyffi.formats.nif.NifFormat.BSPSysMultiTargetEmitterCtrlr property), 54
 unknown_short_1 () (pyffi.formats.nif.NifFormat.NiAlphaProperty property), 90
 unknown_short_1 () (pyffi.formats.nif.NifFormat.NiBinaryVoxelData property), 95
 unknown_short_1 () (pyffi.formats.nif.NifFormat.NiPhysXMeshDesc property), 137
 unknown_short_1 () (pyffi.formats.nif.NifFormat.NiPhysXShapeDesc property), 138
 unknown_short_1 () (pyffi.formats.nif.NifFormat.NiPSysData property), 125
 unknown_short_2 () (pyffi.formats.nif.NifFormat.NiSphericalCollider property), 146
 unknown_short_2 () (pyffi.formats.nif.NifFormat.BSBlastNode property), 45
 unknown_short_2 () (pyffi.formats.nif.NifFormat.BSDamageStage property), 46
 unknown_short_2 () (pyffi.formats.nif.NifFormat.BSDebrisNode property), 46
 unknown_short_2 () (pyffi.formats.nif.NifFormat.BSProceduralLightningController property), 56
 unknown_short_2 () (pyffi.formats.nif.NifFormat.NiBinaryVoxelData property), 95
 unknown_short_2 () (pyffi.formats.nif.NifFormat.NiParticleSystem property), 133
 unknown_short_2 () (pyffi.formats.nif.NifFormat.NiParticleSystemController property), 134
 unknown_short_2 () (pyffi.formats.nif.NifFormat.NiPathController property), 135
 unknown_short_2 () (pyffi.formats.nif.NifFormat.NiPathInterpolator property), 135
 unknown_short_2 () (pyffi.formats.nif.NifFormat.NiPhysXMeshDesc property), 137
 unknown_short_2 () (pyffi.formats.nif.NifFormat.NiPhysXShapeDesc property), 138
 unknown_short_2 () (pyffi.formats.nif.NifFormat.NiPlanarCollider property), 139
 unknown_short_2 () (pyffi.formats.nif.NifFormat.NiPortal property), 140
 unknown_short_2 () (pyffi.formats.nif.NifFormat.NiPSysData property), 125
 unknown_short_2 () (pyffi.formats.nif.NifFormat.NiSphericalCollider property), 146
 unknown_short_3 () (pyffi.formats.nif.NifFormat.BSProceduralLightningController property), 56
 unknown_short_3 () (pyffi.formats.nif.NifFormat.BSWaterShaderProperty property), 62

unknown_short_3 () (pyffi.formats.nif.NifFormat.MultiTextureElement property), 87
 unknown_short_3 () (pyffi.formats.nif.NifFormat.NiBinaryVoxelData property), 95
 unknown_short_3 () (pyffi.formats.nif.NifFormat.NiParticleSystem property), 133
 unknown_short_3 () (pyffi.formats.nif.NifFormat.NiParticleSystemController property), 134
 unknown_short_3 () (pyffi.formats.nif.NifFormat.NiPSPlanarCollider property), 121
 unknown_short_3 () (pyffi.formats.nif.NifFormat.NiPSysData property), 126
 unknown_short_5 () (pyffi.formats.nif.NifFormat.BSStripPSysData property), 61
 unknown_shorts () (pyffi.formats.nif.NifFormat.ExtraMeshDataEpicMick (pyffi.formats.nif.NifFormat.LimitedHingeDescriptor property), 73
 unknown_shorts () (pyffi.formats.nif.NifFormat.NiCloudData update_a_b () (pyffi.formats.nif.NifFormat.RagdollDescriptor method), 98
 unknown_shorts_1 () (pyffi.formats.nif.NifFormat.NiPhysXMeshDesc update_bind_position () (pyffi.formats.nif.NifFormat.NiGeometry method), 137
 unknown_string () (pyffi.formats.nif.NifFormat.NiArkShaderExtraData update_extra_data_radius () (pyffi.formats.nif.NifFormat.NiGeometryData method), 90
 unknown_string_4 () (pyffi.formats.nif.NifFormat.NiPhysXPropDesc update_delta_time () (pyffi.formats.nif.NifFormat.BSPSysStripUpdateModifier property), 138
 unknown_u_short_1 () (pyffi.formats.nif.NifFormat.NiScreenElementsData update_mass_center_inertia () (pyffi.formats.nif.NifFormat.bhkRigidBody method), 142
 unknown_u_short_2 () (pyffi.formats.nif.NifFormat.NiScreenElementsData update_mopp () (pyffi.formats.nif.NifFormat.bhkMoppBvTreeShape method), 142
 unknown_u_short_3 () (pyffi.formats.nif.NifFormat.NiScreenElementsData update_mopp_welding () (pyffi.formats.nif.NifFormat.bhkMoppBvTreeShape method), 142
 unknown_vector () (pyffi.formats.nif.NifFormat.NiTextureEffect update_effect_origin_scale () (pyffi.formats.nif.NifFormat.bhkMoppBvTreeShape method), 148
 unknown_vector () (pyffi.formats.nif.NifFormat.OldSkinData update_skin_center_radius () (pyffi.formats.nif.NifFormat.NiTriBasedGeom method), 157
 unknown_vector () (pyffi.formats.nif.NifFormat.Particle update_skin_partition () (pyffi.formats.nif.NifFormat.NiTriBasedGeom method), 157
 unknown_vectors () (pyffi.formats.nif.NifFormat.NiBinaryVoxelData update_skin_partition () (pyffi.formats.nif.NifFormat.NiTriBasedGeom method), 95
 unkown_byte_5 () (pyffi.formats.nif.NifFormat.NiShadowGenerator skip () (pyffi.formats.nif.NifFormat.NiPSysBoundUpdateModifier property), 143
 unkown_byte_9 () (pyffi.formats.nif.NifFormat.NiShadowGenerator tangent_space () (pyffi.formats.nif.NifFormat.NiShadowGenerator property), 143
 unkown_float_4 () (pyffi.formats.nif.NifFormat.NiShadowGenerator property), 143
 unkown_floats () (pyffi.formats.nif.NifFormat.bhkSimpleShapePhantom property), 187
 unkown_int_2 () (pyffi.formats.nif.NifFormat.NiShadowGenerator property), 143
 unkown_int_6 () (pyffi.formats.nif.NifFormat.NiShadowGenerator property), 143
 unkown_int_7 () (pyffi.formats.nif.NifFormat.NiShadowGenerator property), 143
 unkown_int_8 () (pyffi.formats.nif.NifFormat.NiShadowGenerator property), 143
 up () (pyffi.formats.nif.NifFormat.FurnitureEntryPoints property), 78
 update_a_b () (pyffi.formats.nif.NifFormat.bhkLimitedHingeConstraint method), 182
 update_a_b () (pyffi.formats.nif.NifFormat.bhkMalleableConstraint method), 182
 update_a_b () (pyffi.formats.nif.NifFormat.bhkRagdollConstraint method), 186
 update_a_b () (pyffi.formats.nif.NifFormat.LimitedHingeDescriptor method), 82
 update_a_b () (pyffi.formats.nif.NifFormat.RagdollDescriptor method), 160
 update_bind_position () (pyffi.formats.nif.NifFormat.NiGeometry method), 103
 update_extra_data_radius () (pyffi.formats.nif.NifFormat.NiGeometryData method), 105
 update_delta_time () (pyffi.formats.nif.NifFormat.BSPSysStripUpdateModifier property), 55
 update_mass_center_inertia () (pyffi.formats.nif.NifFormat.bhkRigidBody method), 186
 update_mopp () (pyffi.formats.nif.NifFormat.bhkMoppBvTreeShape method), 183
 update_mopp_welding () (pyffi.formats.nif.NifFormat.bhkMoppBvTreeShape method), 183
 update_effect_origin_scale () (pyffi.formats.nif.NifFormat.bhkMoppBvTreeShape method), 183
 update_skin_center_radius () (pyffi.formats.nif.NifFormat.NiTriBasedGeom method), 150
 update_skin_partition () (pyffi.formats.nif.NifFormat.NiTriBasedGeom method), 150
 skip () (pyffi.formats.nif.NifFormat.NiPSysBoundUpdateModifier property), 124
 tangent_space () (pyffi.formats.nif.NifFormat.NiShadowGenerator property), 143

(pyffi.formats.nif.NifFormat.NiTriBasedGeom
 method), 151
 update_versions() (pyffi.formats.cgf.CgfFormat.Data method), 13
 usage() (pyffi.formats.nif.NifFormat.NiDataStream
 property), 99
 USAGE_SHADER_CONSTANT
 (pyffi.formats.nif.NifFormat.DataStreamUsage
 attribute), 71
 USAGE_USER (pyffi.formats.nif.NifFormat.DataStreamUsage
 attribute), 71
 USAGE_VERTEX (pyffi.formats.nif.NifFormat.DataStreamUsage
 attribute), 71
 USAGE_VERTEX_INDEX
 (pyffi.formats.nif.NifFormat.DataStreamUsage
 attribute), 71
 use_abv() (pyffi.formats.nif.NifFormat.NiCollisionData
 property), 98
 use_direction() (pyffi.formats.nif.NifFormat.NiPSysDragFieldModifier
 property), 126
 use_external() (pyffi.formats.nif.NifFormat.NiImage
 property), 105
 use_external() (pyffi.formats.nif.NifFormat.NiSourceTexture
 property), 145
 use_external() (pyffi.formats.nif.NifFormat.TexSource
 property), 174
 use_max_distance() (pyffi.formats.nif.NifFormat.NiPSysFieldModifier
 property), 127
 use_mipmaps() (pyffi.formats.nif.NifFormat.NiSourceTexture
 property), 145
 use_orthographic_projection() (pyffi.formats.nif.NifFormat.NiCamera prop-
 erty), 97
 used_triangle_points() (pyffi.formats.nif.NifFormat.NiScreenElementsData
 property), 142
 used_vertices() (pyffi.formats.nif.NifFormat.NiScreenElementsData
 property), 142
 user_version (pyffi.object_models.FileFormat.Data
 attribute), 234
 user_version() (pyffi.formats.nif.NifFormat.Data
 property), 71
 user_version_2() (pyffi.formats.nif.NifFormat.Data
 property), 71
 ushort (pyffi.formats.cgf.CgfFormat attribute), 19
 ushort (pyffi.formats.dds.DdsFormat attribute), 26
 ushort (pyffi.formats.egm.EgmFormat attribute), 30
 ushort (pyffi.formats.egt.EgtFormat attribute), 33
 ushort (pyffi.formats.esp.EspFormat attribute), 37
 ushort (pyffi.formats.kfm.KfmFormat attribute), 41
 ushort (pyffi.formats.nif.NifFormat attribute), 190
 ushort (pyffi.formats.tga.TgaFormat attribute), 197
 ushort (pyffi.formats.tri.TriFormat attribute), 201
 uv_groups() (pyffi.formats.nif.NifFormat.NiUVData
 property), 153
 uv_offset() (pyffi.formats.nif.NifFormat.BSEffectShaderProperty
 property), 49
 uv_offset() (pyffi.formats.nif.NifFormat.BSLightingShaderProperty
 property), 51
 uv_offset() (pyffi.formats.nif.NifFormat.BSSkyShaderProperty
 property), 60
 uv_offset() (pyffi.formats.nif.NifFormat.BSWaterShaderProperty
 property), 62
 uv_quadrants() (pyffi.formats.nif.NifFormat.NiParticlesData
 property), 135
 uv_scale() (pyffi.formats.nif.NifFormat.BSEffectShaderProperty
 property), 49
 uv_scale() (pyffi.formats.nif.NifFormat.BSLightingShaderProperty
 property), 51
 uv_scale() (pyffi.formats.nif.NifFormat.BSSkyShaderProperty
 property), 60
 uv_scale() (pyffi.formats.nif.NifFormat.BSWaterShaderProperty
 property), 62
 uv_set() (pyffi.formats.nif.NifFormat.MultiTextureElement
 property), 87
 uv_set() (pyffi.formats.nif.NifFormat.TexDesc prop-
 erty), 174
V
 v_1() (pyffi.formats.nif.NifFormat.Triangle property),
 175
 v_2() (pyffi.formats.nif.NifFormat.Triangle property),
 175
 v_3() (pyffi.formats.nif.NifFormat.Triangle property),
 175
 value() (pyffi.formats.nif.NifFormat.BSValueNode
 property), 61
 value() (pyffi.formats.nif.NifFormat.Key property), 81
 value() (pyffi.formats.nif.NifFormat.QuatKey prop-
 erty), 160
 vectors() (pyffi.formats.nif.NifFormat.NiBezierTriangle4
 property), 94
 vector_2() (pyffi.formats.nif.NifFormat.NiBezierTriangle4
 property), 94
 vector_blocks() (pyffi.formats.nif.NifFormat.BSDecalPlacementVector
 property), 46
 vector_data() (pyffi.formats.nif.NifFormat.NiVectorExtraData
 property), 153
 vectors() (pyffi.formats.nif.NifFormat.Morph prop-
 erty), 86
 velocity() (pyffi.formats.nif.NifFormat.Particle prop-
 erty), 157
 VELOCITY_USE_DIRECTION
 (pyffi.formats.nif.NifFormat.VelocityType
 attribute), 176
 VELOCITY_USE_NORMALS
 (pyffi.formats.nif.NifFormat.VelocityType

attribute), 176
 VELOCITY_USE_RANDOM
 (pyffi.formats.nif.NifFormat.VelocityType
 attribute), 176
 version (pyffi.object_models.FileFormat.Data at-
 tribute), 234
 version() (pyffi.formats.nif.NifFormat.Data prop-
 erty), 71
 version_number() (pyffi.formats.bsa.BsaFormat
 static method), 10
 version_number() (pyffi.formats.cgf.CgfFormat
 static method), 19
 version_number() (pyffi.formats.dds.DdsFormat
 static method), 26
 version_number() (pyffi.formats.egm.EgmFormat
 static method), 30
 version_number() (pyffi.formats.egt.EgtFormat
 static method), 33
 version_number() (pyffi.formats.esp.EspFormat
 static method), 37
 version_number() (pyffi.formats.kfm.KfmFormat
 static method), 41
 version_number() (pyffi.formats.nif.NifFormat
 static method), 190
 version_number() (pyffi.formats.tri.TriFormat
 static method), 201
 version_number() (pyffi.object_models.FileFormat
 static method), 235
 version_string() (pyffi.formats.kfm.KfmFormat.HeaderString
 static method), 40
 version_string() (pyffi.formats.nif.NifFormat.HeaderString
 static method), 79
 versions (pyffi.formats.nif.NifFormat attribute), 190
 VERT_MODE_SRC_AMB_DIF
 (pyffi.formats.nif.NifFormat.VertMode at-
 tribute), 177
 VERT_MODE_SRC_EMISSIVE
 (pyffi.formats.nif.NifFormat.VertMode at-
 tribute), 177
 VERT_MODE_SRC_IGNORE
 (pyffi.formats.nif.NifFormat.VertMode at-
 tribute), 177
 vertex_counts() (pyffi.formats.nif.NifFormat.NiTriShapeSkinController
 property), 152
 vertex_id() (pyffi.formats.nif.NifFormat.Particle
 property), 157
 vertex_index() (pyffi.formats.nif.NifFormat.OldSkinData
 property), 157
 vertex_indices() (pyffi.formats.nif.NifFormat.MatchGroup
 property), 83
 vertex_mode() (pyffi.formats.nif.NifFormat.NiVertexColorProperty
 property), 153
 vertex_offset() (pyffi.formats.nif.NifFormat.Polygon
 property), 158
 vertex_weight() (pyffi.formats.nif.NifFormat.OldSkinData
 property), 157
 vertical_angle() (pyffi.formats.nif.NifFormat.NiParticleSystemContr
 property), 134
 vertical_direction()
 (pyffi.formats.nif.NifFormat.NiParticleSystemController
 property), 134
 vertices() (pyffi.formats.nif.NifFormat.bhkCMSDChunk
 property), 179
 vertices() (pyffi.formats.nif.NifFormat.NiPhysXMeshDesc
 property), 137
 vertices() (pyffi.formats.nif.NifFormat.NiPortal
 property), 140
 viewport_bottom()
 (pyffi.formats.nif.NifFormat.NiCamera prop-
 erty), 97
 viewport_left() (pyffi.formats.nif.NifFormat.NiCamera
 property), 97
 viewport_right() (pyffi.formats.nif.NifFormat.NiCamera
 property), 97
 viewport_top() (pyffi.formats.nif.NifFormat.NiCamera
 property), 97
 visibility_interpolator()
 (pyffi.formats.nif.NifFormat.BSPSysMultiTargetEmitterCtrlr
 property), 54
 visibility_interpolator()
 (pyffi.formats.nif.NifFormat.NiPSysEmitterCtrlr
 property), 126
 visibility_keys()
 (pyffi.formats.nif.NifFormat.NiPSysEmitterCtrlrData
 property), 127

W

w() (pyffi.formats.nif.NifFormat.Quaternion property),
 160
 w() (pyffi.formats.nif.NifFormat.QuaternionXYZW prop-
 erty), 160
 w_rotation() (pyffi.formats.nif.NifFormat.TexDesc
 property), 174
 walk() (pyffi.object_models.FileFormat class method),
 235
 walkData() (pyffi.object_models.FileFormat class
 method), 235
 wall_plane() (pyffi.formats.nif.NifFormat.NiRoom
 property), 141
 WARD (pyffi.formats.nif.NifFormat.SkyrimLayer at-
 tribute), 167
 WATER (pyffi.formats.nif.NifFormat.Fallout3Layer
 attribute), 77
 WATER (pyffi.formats.nif.NifFormat.OblivionLayer at-
 tribute), 156
 WATER (pyffi.formats.nif.NifFormat.SkyrimLayer at-
 tribute), 167
 water_direction()

	(pyffi.formats.nif.NifFormat.BSWaterShaderProperty), 62	write()	(pyffi.formats.bsa.BsaFormat.ZString method), 10
water_shader_flags()	(pyffi.formats.nif.NifFormat.BSWaterShaderProperty), 62	write()	(pyffi.formats.cgf.CgfFormat.Data method), 13
WEAPON	(pyffi.formats.nif.NifFormat.Fallout3Layer attribute), 77	write()	(pyffi.formats.cgf.CgfFormat.FileSignature method), 14
WEAPON	(pyffi.formats.nif.NifFormat.OblivionLayer attribute), 156	write()	(pyffi.formats.cgf.CgfFormat.Ref method), 17
WEAPON	(pyffi.formats.nif.NifFormat.SkyrimLayer attribute), 167	write()	(pyffi.formats.cgf.CgfFormat.SizedString method), 18
weight()	(pyffi.formats.nif.NifFormat.MorphWeight property), 86	write()	(pyffi.formats.dae.DaeFormat.Data method), 23
weight()	(pyffi.formats.nif.NifFormat.NiVertWeightsExtraData property), 153	write()	(pyffi.formats.dds.DdsFormat.Data method), 25
weight()	(pyffi.formats.nif.NifFormat.SkinWeight property), 164	write()	(pyffi.formats.dds.DdsFormat.HeaderString method), 25
welding_info()	(pyffi.formats.nif.NifFormat.hkTriangle property), 189	write()	(pyffi.formats.egm.EgmFormat.Data method), 28
width()	(pyffi.formats.nif.NifFormat.MipMap property), 85	write()	(pyffi.formats.egm.EgmFormat.FileSignature method), 28
width()	(pyffi.formats.nif.NifFormat.NiPSysBoxEmitter property), 124	write()	(pyffi.formats.egm.EgmFormat.FileVersion method), 29
width()	(pyffi.formats.nif.NifFormat.NiPSysPlanarCollider property), 130	write()	(pyffi.formats.egt.EgtFormat.FileSignature method), 32
width()	(pyffi.formats.nif.NifFormat.NiRawImageData property), 141	write()	(pyffi.formats.egt.EgtFormat.FileVersion method), 32
WING	(pyffi.formats.nif.NifFormat.Fallout3Layer attribute), 77	write()	(pyffi.formats.egt.EgtFormat.Header method), 33
WING	(pyffi.formats.nif.NifFormat.OblivionLayer attribute), 157	write()	(pyffi.formats.esp.EspFormat.Data method), 35
world_center()	(pyffi.formats.nif.NifFormat.NiScreenLODDData property), 142	write()	(pyffi.formats.esp.EspFormat.GRUP method), 36
world_radius()	(pyffi.formats.nif.NifFormat.NiScreenLODDData property), 142	write()	(pyffi.formats.esp.EspFormat.Record method), 36
world_space()	(pyffi.formats.nif.NifFormat.NiParticleSystem property), 133	write()	(pyffi.formats.esp.EspFormat.ZString method), 37
WorldMap1	(pyffi.formats.nif.NifFormat.NiParticleSystem property), 133	write()	(pyffi.formats.kfm.KfmFormat.Data method), 39
WorldMap2	(pyffi.formats.nif.NifFormat.BSLightingShaderProperty), 52	write()	(pyffi.formats.kfm.KfmFormat.HeaderString method), 40
WorldMap3	(pyffi.formats.nif.NifFormat.BSLightingShaderProperty), 52	write()	(pyffi.formats.kfm.KfmFormat.SizedString method), 41
WorldMap4	(pyffi.formats.nif.NifFormat.BSLightingShaderProperty), 52	write()	(pyffi.formats.kfm.KfmFormat.SizedString method), 41
WRAP_S_CLAMP_T	(pyffi.formats.nif.NifFormat.BSLightingShaderProperty), 52	write()	(pyffi.formats.kfm.KfmFormat.SizedString method), 41
WRAP_S_WRAP_T	(pyffi.formats.nif.NifFormat.BSLightingShaderProperty), 52	write()	(pyffi.formats.kfm.KfmFormat.SizedString method), 41
wrap_s_clamp_t	(pyffi.formats.nif.NifFormat.TexClampMode attribute), 173	write()	(pyffi.formats.nif.NifFormat.bool method), 188
wrap_s_wrap_t	(pyffi.formats.nif.NifFormat.TexClampMode attribute), 173	write()	(pyffi.formats.nif.NifFormat.ByteArray method), 194
write()	(pyffi.formats.bsa.BsaFormat.BZString method), 8	write()	(pyffi.formats.nif.NifFormat.ByteMatrix method), 64
write()	(pyffi.formats.bsa.BsaFormat.FileVersion method), 8	write()	(pyffi.formats.nif.NifFormat.Data method), 65
write()	(pyffi.formats.bsa.BsaFormat.Header method), 8	write()	(pyffi.formats.nif.NifFormat.Data method), 71
write()	(pyffi.formats.bsa.BsaFormat.Header method), 8	write()	(pyffi.formats.nif.NifFormat.FileVersion method), 77
write()	(pyffi.formats.bsa.BsaFormat.Header method), 8	write()	(pyffi.formats.nif.NifFormat.Footer method), 77
write()	(pyffi.formats.bsa.BsaFormat.Header method), 8	write()	(pyffi.formats.nif.NifFormat.HeaderString method), 80
write()	(pyffi.formats.bsa.BsaFormat.Header method), 8	write()	(pyffi.formats.nif.NifFormat.LineString method), 80

method), 83
`write()` (`pyffi.formats.nif.NifFormat.Ref` *method*), 161
`write()` (`pyffi.formats.nif.NifFormat.ShortString` *method*), 162
`write()` (`pyffi.formats.nif.NifFormat.SizedString` *method*), 163
`write()` (`pyffi.formats.nif.NifFormat.string` *method*), 189
`write()` (`pyffi.formats.tga.TgaFormat.Data` *method*), 196
`write()` (`pyffi.formats.tga.TgaFormat.FooterString` *method*), 197
`write()` (`pyffi.formats.tri.TriFormat.FileSignature` *method*), 199
`write()` (`pyffi.formats.tri.TriFormat.FileVersion` *method*), 199
`write()` (`pyffi.formats.tri.TriFormat.Header` *method*), 200
`write()` (`pyffi.formats.tri.TriFormat.SizedStringZ` *method*), 201
`write()` (`pyffi.object_models.FileFormat.Data` *method*), 234
`write()` (`pyffi.spells.Toaster` *method*), 233
`writepatch()` (`pyffi.spells.Toaster` *method*), 233

X

`x()` (`pyffi.formats.nif.NifFormat.Quaternion` *property*), 160
`x()` (`pyffi.formats.nif.NifFormat.QuaternionXYZW` *property*), 160
`x_axis()` (`pyffi.formats.nif.NifFormat.NiPSysPlanarCollider` *property*), 130
`xml_alias` (`pyffi.formats.nif.NifFormat` *attribute*), 190
`xml_bit_struct` (`pyffi.formats.nif.NifFormat` *attribute*), 190
`xml_enum` (`pyffi.formats.nif.NifFormat` *attribute*), 190
`xml_file_name` (`pyffi.formats.nif.NifFormat` *attribute*), 190
`xml_file_path` (`pyffi.formats.nif.NifFormat` *attribute*), 190
`xml_struct` (`pyffi.formats.nif.NifFormat` *attribute*), 190
`XYZ_ROTATION_KEY` (`pyffi.formats.nif.NifFormat.KeyType` *attribute*), 81

Y

`y()` (`pyffi.formats.nif.NifFormat.Quaternion` *property*), 160
`y()` (`pyffi.formats.nif.NifFormat.QuaternionXYZW` *property*), 160
`y_axis()` (`pyffi.formats.nif.NifFormat.NiPSysPlanarCollider` *property*), 130

Z

`z()` (`pyffi.formats.nif.NifFormat.Quaternion` *property*), 160
`z()` (`pyffi.formats.nif.NifFormat.QuaternionXYZW` *property*), 160
`z_fail_action()` (`pyffi.formats.nif.NifFormat.NiStencilProperty` *property*), 146
`ZCOMP_ALWAYS` (`pyffi.formats.nif.NifFormat.ZCompareMode` *attribute*), 177
`ZCOMP_EQUAL` (`pyffi.formats.nif.NifFormat.ZCompareMode` *attribute*), 177
`ZCOMP_GREATER` (`pyffi.formats.nif.NifFormat.ZCompareMode` *attribute*), 177
`ZCOMP_GREATER_EQUAL` (`pyffi.formats.nif.NifFormat.ZCompareMode` *attribute*), 177
`ZCOMP_LESS` (`pyffi.formats.nif.NifFormat.ZCompareMode` *attribute*), 177
`ZCOMP_LESS_EQUAL` (`pyffi.formats.nif.NifFormat.ZCompareMode` *attribute*), 177
`ZCOMP_NEVER` (`pyffi.formats.nif.NifFormat.ZCompareMode` *attribute*), 177
`ZCOMP_NOT_EQUAL` (`pyffi.formats.nif.NifFormat.ZCompareMode` *attribute*), 177
`zoom()` (`pyffi.formats.nif.NifFormat.BSInvMarker` *property*), 50