
LOOT API Python Module Documentation

Release latest

WrinklyNinja

Mar 25, 2019

Contents

1	Usage	3
1.1	Installing the wrapper	3
1.2	Using the wrapper	3
2	API Reference	5
2.1	Enumerations	5
2.2	Public-Field Data Structures	6
2.3	Functions	7
2.4	Classes	7

Contents:

1.1 Installing the wrapper

Build archives contain two binaries:

- `loot_api.pyd` is the Python wrapper
- `loot_api.dll` is the C++ API DLL that the Python wrapper was built against.

The C++ DLL requires the [Visual C++ 2017 Redistributable \(x86\)](#) to be installed.

To use the wrapper, copy both files to wherever you want to import them from (they must be in the same folder), and you're done!

1.2 Using the wrapper

1.2.1 Checking Compatibility

To check if the module loaded is compatible with the version of the API that you developed against:

```
>>> import loot_api
>>> loot_api.is_compatible(0,14,0)
True
>>> loot_api.is_compatible(0,9,0)
False
```

1.2.2 Getting a Plugin's Bash Tag Suggestions

To get a plugin's Bash Tag suggestions from a `masterlist.yaml` metadata file:

```
>>> import loot_api
>>> loot_api.initialise_locale()
>>> db = loot_api.create_game_handle(loot_api.GameType.tes4,
↳ 'C:\\path\\to\\oblivion\\directory')
>>> db.load_lists('masterlist.yaml')
>>> tags = db.get_plugin_tags(u'Unofficial Oblivion Patch.esp')
>>> tags.added
set([u'Scripts', u'Relations', u'C.Owner', u'Actors.AIPackages', u'Actors.Stats', u
↳ 'Actors.ACBS', u'C.Music', u'Factions', u'Invent', u'Relev', u'Names', u'C.Light', u
↳ 'Delev', u'C.Name', u'C.Climate', u'NPC.Class', u'Stats', u'Actors.DeathItem', u
↳ 'Creatures.Blood', u'Actors.CombatStyle', u'Actors.AIData'])
>>> tags.removed
set([u'C.Water'])
>>> tags.userlist_modified
False
```


As this API is just a wrapper for LOOT's C++ API, its documentation is linked to for all non-Python-specific information.

2.1 Enumerations

The wrapped enumeration types below are classes in Python, but the distinction makes no difference in practice, so they're grouped here for semantics. All values are unsigned integer constants.

class `loot_api.GameType`

Wraps `loot::GameType` to expose the LOOT API's game codes.

`fo3`

`fo4`

`fonv`

`tes4`

`tes5`

`tes5se`

class `loot_api.LogLevel`

Wraps `loot::LogLevel` to expose the LOOT API's log level codes.

`trace`

`debug`

`info`

`warning`

`error`

`fatal`

class loot_api.**MessageType**

Wraps `loot::MessageType` to expose the LOOT API's message type codes.

error

say

warn

class loot_api.**PluginCleanliness**

Codes used to indicate the cleanliness of a plugin according to the information contained within the loaded masterlist/userlist.

clean

Indicates that the plugin is clean.

dirty

Indicates that the plugin is dirty.

do_not_clean

Indicates that the plugin contains dirty edits, but that they are part of the plugin's intended functionality and should not be removed.

unknown

Indicates that no data is available on whether the plugin is dirty or not.

2.2 Public-Field Data Structures

Classes with public fields and no member functions.

class loot_api.**MasterlistInfo**

Wraps `loot::MasterlistInfo`.

revision_id

A Unicode string containing a Git commit's SHA-1 checksum.

revision_date

A Unicode string containing the date of the commit given by *revision_id*, in ISO 8601 format (YYYY-MM-DD).

is_modified

A boolean that is true if the masterlist has been modified from its state at the commit given by *revision_id*.

class loot_api.**SimpleMessage**

Wraps `loot::SimpleMessage`.

type

A *loot_api.MessageType* giving the message type.

language

A Unicode string giving the message text language.

text

A Unicode string containing the message text.

condition

A Unicode string containing the message condition.

class loot_api.**PluginTags**

Wraps `loot::PluginTags`.

added

A set of Unicode strings giving Bash Tags suggested for addition.

removed

A set of Unicode strings giving Bash Tags suggested for removal.

userlist_modified

A boolean that is true if the suggestions contain metadata obtained from a loaded userlist.

2.3 Functions

`loot_api.set_logging_callback(callback) → NoneType`

Set the callback function that is called when logging. Wraps `loot::SetLoggingCallback()`.

`loot_api.is_compatible(int, int, int) → bool`

Checks for API compatibility. Wraps `loot::IsCompatible()`.

`loot_api.InitialiseLocale(unicode = u'') → NoneType`

Initialise the current global locale using the given ID. A blank value means that the system default locale will be initialised. Wraps `loot::InitialiseLocale()`.

`loot_api.create_game_handle(game : loot_api.GameType, game_path : unicode[, game_local_path : unicode = u'']) → loot_api.GameInterface`

Initialise a new game handle. Wraps `loot::CreateGameHandle()`.

2.4 Classes

class `loot_api.GameInterface`

Wraps `loot::GameInterface`.

`loot_api.get_database() → loot_api.DatabaseInterface`

Get a database handle. Wraps `loot::GetDatabase()`.

`loot_api.load_current_load_order_state() → NoneType`

Load the current load order state, discarding any previously held state. Wraps `loot::LoadCurrentLoadOrderState()`.

class `loot_api.DatabaseInterface`

Wraps `loot::DatabaseInterface`.

`get_masterlist_revision(loot_api.DatabaseInterface, unicode, bool) → loot_api.MasterlistInfo`

Gets the give masterlist's source control revision. Wraps `GetMasterlistRevision()`.

`get_plugin_metadata(loot_api.DatabaseInterface, plugin : unicode[, includeUserMetadata : bool = True[, evaluateConditions : bool = False]]) → loot_api.PluginMetadata`

Get all a plugin's loaded metadata. Wraps `GetPluginMetadata()`.

`get_plugin_cleanliness(loot_api.DatabaseInterface, plugin : unicode[, evaluateConditions : bool = False]) → loot_api.PluginCleanliness`

Determines the database's knowledge of a plugin's cleanliness. Outputs whether the plugin should be cleaned or not, or if no data is available.

`get_plugin_tags(loot_api.DatabaseInterface, plugin : unicode[, evaluateConditions : bool = False]) → loot_api.PluginTags`

Outputs the Bash Tags suggested for addition and removal by the database for the given plugin.

load_lists (*loot_api.DatabaseInterface*, *masterlist_path* : *unicode*[, *userlist_path* : *unicode* = *u''*])
 → *NoneType*

Loads the masterlist and userlist from the paths specified. Wraps `LoadLists()`.

update_masterlist (*loot_api.DatabaseInterface*, *unicode*, *unicode*, *unicode*) → *bool*

Updates the given masterlist using the given Git repository details. Wraps `UpdateMasterlist()`.

write_minimal_list (*loot_api.DatabaseInterface*, *unicode*, *bool*) → *NoneType*

Writes a minimal metadata file containing only Bash Tag suggestions and/or cleanliness info from the loaded metadata. Wraps `WriteMinimalList()`.

class `loot_api.Version`

Wraps `loot::LootVersion`.

major

An unsigned integer giving the major version number.

minor

An unsigned integer giving the minor version number.

patch

An unsigned integer giving the patch version number.

revision

A Unicode string containing the SHA-1 of the Git revision that the wrapped C++ API was built from.

static string() → *unicode*

Returns the API version as a string of the form `major.minor.patch`

class `loot_api WrapperVersion`

Provides information about the version of the LOOT API Python wrapper that is being run.

major

An unsigned integer giving the major version number.

minor

An unsigned integer giving the minor version number.

patch

An unsigned integer giving the patch version number.

revision

A Unicode string containing the SHA-1 of the Git revision that the wrapped C++ API was built from.

static string() → *unicode*

Returns the API version as a string of the form `major.minor.patch`

class `loot_api.PluginMetadata`

Wraps `loot::PluginMetadata`.

get_simple_messages (*loot_api.PluginMetadata*, *unicode*) → *list*<*loot_api.SimpleMessage*>

Get the plugin's messages as `SimpleMessage` objects for the given language. Wraps `GetPluginMessages()`.

A

added (*loot_api.PluginTags attribute*), 6

C

clean (*loot_api.PluginCleanliness attribute*), 6
condition (*loot_api.SimpleMessage attribute*), 6

D

debug (*loot_api.LogLevel attribute*), 5
dirty (*loot_api.PluginCleanliness attribute*), 6
do_not_clean (*loot_api.PluginCleanliness attribute*), 6

E

error (*loot_api.LogLevel attribute*), 5
error (*loot_api.MessageType attribute*), 6

F

fatal (*loot_api.LogLevel attribute*), 5
fo3 (*loot_api.GameType attribute*), 5
fo4 (*loot_api.GameType attribute*), 5
fonv (*loot_api.GameType attribute*), 5

G

get_masterlist_revision()
(*loot_api.DatabaseInterface method*), 7
get_plugin_cleanliness()
(*loot_api.DatabaseInterface method*), 7
get_plugin_metadata()
(*loot_api.DatabaseInterface method*), 7
get_plugin_tags() (*loot_api.DatabaseInterface method*), 7
get_simple_messages()
(*loot_api.PluginMetadata method*), 8

I

info (*loot_api.LogLevel attribute*), 5
is_modified (*loot_api.MasterlistInfo attribute*), 6

L

language (*loot_api.SimpleMessage attribute*), 6
load_lists() (*loot_api.DatabaseInterface method*), 7
loot_api.create_game_handle() (*built-in function*), 7
loot_api.DatabaseInterface (*built-in class*), 7
loot_api.GameInterface (*built-in class*), 7
loot_api.GameInterface.loot_api.get_database()
(*built-in function*), 7
loot_api.GameInterface.loot_api.load_current_load_
(*built-in function*), 7
loot_api.GameType (*built-in class*), 5
loot_api.InitialiseLocale() (*built-in function*), 7
loot_api.is_compatible() (*built-in function*), 7
loot_api.LogLevel (*built-in class*), 5
loot_api.MasterlistInfo (*built-in class*), 6
loot_api.MessageType (*built-in class*), 5
loot_api.PluginCleanliness (*built-in class*), 6
loot_api.PluginMetadata (*built-in class*), 8
loot_api.PluginTags (*built-in class*), 6
loot_api.set_logging_callback() (*built-in function*), 7
loot_api.SimpleMessage (*built-in class*), 6
loot_api.Version (*built-in class*), 8
loot_api WrapperVersion (*built-in class*), 8

M

major (*loot_api.Version attribute*), 8
major (*loot_api WrapperVersion attribute*), 8
minor (*loot_api.Version attribute*), 8
minor (*loot_api WrapperVersion attribute*), 8

P

patch (*loot_api.Version attribute*), 8
patch (*loot_api WrapperVersion attribute*), 8

R

removed (*loot_api.PluginTags attribute*), 7

`revision` (*loot_api.Version* attribute), 8
`revision` (*loot_api.WrapperVersion* attribute), 8
`revision_date` (*loot_api.MasterlistInfo* attribute), 6
`revision_id` (*loot_api.MasterlistInfo* attribute), 6

S

`say` (*loot_api.MessageType* attribute), 6
`string()` (*loot_api.Version* static method), 8
`string()` (*loot_api.WrapperVersion* static method), 8

T

`tes4` (*loot_api.GameType* attribute), 5
`tes5` (*loot_api.GameType* attribute), 5
`tes5se` (*loot_api.GameType* attribute), 5
`text` (*loot_api.SimpleMessage* attribute), 6
`trace` (*loot_api.LogLevel* attribute), 5
`type` (*loot_api.SimpleMessage* attribute), 6

U

`unknown` (*loot_api.PluginCleanliness* attribute), 6
`update_masterlist()` (*loot_api.DatabaseInterface*
 method), 8
`userlist_modified` (*loot_api.PluginTags* at-
 tribute), 7

W

`warn` (*loot_api.MessageType* attribute), 6
`warning` (*loot_api.LogLevel* attribute), 5
`write_minimal_list()`
 (*loot_api.DatabaseInterface* *method*), 8