



Delta3D Version 2.4.0

dtScript::

Reference Manual

Contents

1	Main Page	1
2	Directory Documentation	3
2.1	src/dtScript/ Directory Reference	3
2.2	inc/dtScript/ Directory Reference	4
2.3	inc/ Directory Reference	5
2.4	src/ Directory Reference	6
3	Namespace Documentation	7
3.1	dtScript Namespace Reference	7
3.1.1	Detailed Description	7
4	Class Documentation	9
4.1	ScriptManager Class Reference	9
4.1.1	Detailed Description	10
4.1.2	Constructor & Destructor Documentation	10
4.1.2.1	ScriptManager	10
4.1.2.2	~ScriptManager	10
4.1.3	Member Function Documentation	10
4.1.3.1	Load	10
4.1.3.2	run	10
4.1.3.3	Run	10
4.1.3.4	Stop	10
4.1.4	Member Data Documentation	10
4.1.4.1	mFilename	10
4.1.4.2	mFileObject	10
5	File Documentation	11
5.1	dtscript.h File Reference	11
5.2	export.h File Reference	12
5.2.1	Define Documentation	12
5.2.1.1	DT_SCRIPT_EXPORT	12
5.3	mainpage.h File Reference	13
5.3.1	Detailed Description	13
5.4	scriptmanager.cpp File Reference	14
5.5	scriptmanager.h File Reference	15

Main Page

Delta3D is an Open Source engine which can be used for games, simulations, or other graphical applications.

The **Delta3D** framework exists as a number of modules, each sitting in its own library, enclosed within its own namespace. At the very core lies the dtCore library. This contains basic, low-level functionality which is mostly required for all 3D applications written in C++.

Around and alongside this sit other supporting libraries, such as dtUtil (containing reusable features which are useful for most applications), dtTerrain (for rendering terrain databases), dtGame, dtNet, etc.

Extensive online documentation is available from the Delta3D **Docs** section to help in using Delta3D.

The project's original reference guides generated by Doxygen from the source code may be viewed at the Delta3D **API Documentation** section.

To download source code, binaries, dependencies and sample datasets visit the Delta3D **Downloads** page.

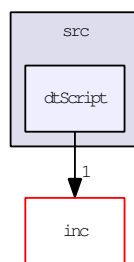
For more about dependencies see the Delta3D **Dependencies** page.

The documentation you are looking at can be downloaded from www.3draum.ch.

Enjoy!

Directory Documentation

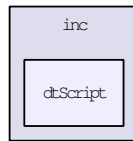
2.1 src/dtScript/ Directory Reference



Files

- file [scriptmanager.cpp](#)

2.2 inc/dtScript/ Directory Reference



Files

- file [dtscript.h](#)
- file [export.h](#)
- file [mainpage.h](#)
- file [scriptmanager.h](#)

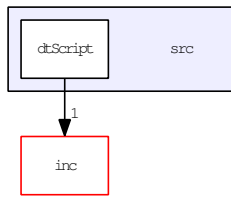
2.3 inc/ Directory Reference



Directories

- directory [dtScript](#)

2.4 src/ Directory Reference



Directories

- directory [dtScript](#)

Namespace Documentation

3.1 dtScript Namespace Reference

Contains the functionality to call Python scripts from a C++ application.

Classes

- class [ScriptManager](#)

This class assists the user in executing Python scripts from within the C++ Delta3D world.

3.1.1 Detailed Description

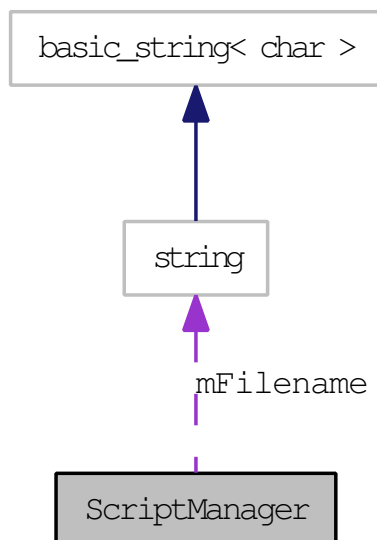
Contains the functionality to call Python scripts from a C++ application.

Class Documentation

4.1 ScriptManager Class Reference

This class assists the user in executing Python scripts from within the C++ Delta3D world.

`#include <inc/dtScript/scriptmanager.h>` Collaboration diagram for ScriptManager:



Public Member Functions

- [ScriptManager](#) (const std::string &name="ScriptManager")
- void [Load](#) (const std::string &filename)
Load a Python script (.py) into memory.
- void [Run](#) (const std::string &filename="")
Load a Python script and execute it immediately. Use the default paramters to use a pre-loaded script.
- void [Stop](#) ()
Stops the [ScriptManager](#) thread. Make sure to call this before you exit your app.

Protected Member Functions

- virtual [~ScriptManager](#) ()
- virtual void [run](#) ()

Protected Attributes

- `std::string mFilename`
- `PyObject * mFileObject`

4.1.1 Detailed Description

This class assists the user in executing Python scripts from within the C++ Delta3D world. Just instantiate a [ScriptManager](#), call `Load(filename)` on your Python script, and then successive `Run()` calls will execute it by kicking off a new thread. Alternatively you can call `Run(filename)` to load and execute in one step.

4.1.2 Constructor & Destructor Documentation

4.1.2.1 `ScriptManager (const std::string & name = "ScriptManager")`

4.1.2.2 `~ScriptManager () [protected, virtual]`

4.1.3 Member Function Documentation

4.1.3.1 `void Load (const std::string & filename)`

Load a Python script (.py) into memory.

4.1.3.2 `void run () [protected, virtual]`

4.1.3.3 `void Run (const std::string & filename = "")`

Load a Python script and execute it immediately. Use the default parameters to use a pre-loaded script.

4.1.3.4 `void Stop () [inline]`

Stops the [ScriptManager](#) thread. Make sure to call this before you exit your app.

4.1.4 Member Data Documentation

4.1.4.1 `std::string mFilename [protected]`

4.1.4.2 `PyObject* mFileObject [protected]`

The documentation for this class was generated from the following files:

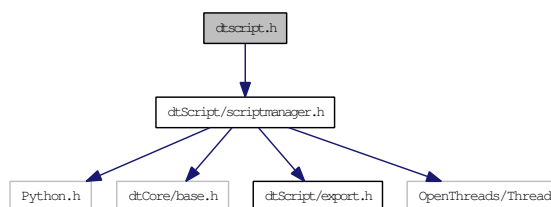
- [scriptmanager.h](#)
- [scriptmanager.cpp](#)

File Documentation

5.1 dtscript.h File Reference

```
#include <dtScript/scriptmanager.h>
```

Include dependency graph for dtscript.h:



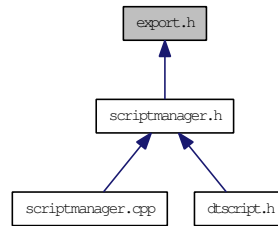
Namespaces

- namespace `dtScript`

Contains the functionality to call Python scripts from a C++ application.

5.2 export.h File Reference

This graph shows which files directly or indirectly include this file:



Defines

- #define [DT_SCRIPT_EXPORT](#)

5.2.1 Define Documentation

5.2.1.1 #define DT_SCRIPT_EXPORT

5.3 mainpage.h File Reference

5.3.1 Detailed Description

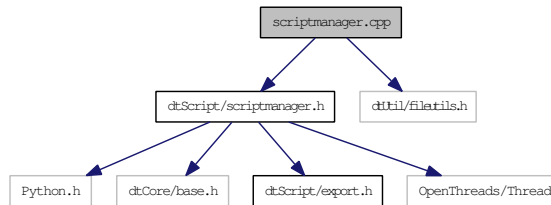
This file contains Doxygen special commands and text for the [Main Page](#) and some other minor aspects of this documentation. It is not part of Delta3D.

5.4 scriptmanager.cpp File Reference

```
#include <dtScript/scriptmanager.h>
```

```
#include <dtUtil/fileutils.h>
```

Include dependency graph for scriptmanager.cpp:



5.5 scriptmanager.h File Reference

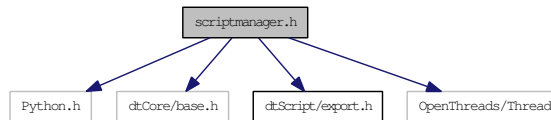
```
#include <Python.h>
```

```
#include <dtCore/base.h>
```

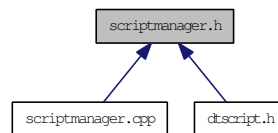
```
#include <dtScript/export.h>
```

```
#include <OpenThreads/Thread>
```

Include dependency graph for scriptmanager.h:



This graph shows which files directly or indirectly include this file:



Classes

- class [ScriptManager](#)

This class assists the user in executing Python scripts from within the C++ Delta3D world.

Namespaces

- namespace [dtScript](#)

Contains the functionality to call Python scripts from a C++ application.

Index

- Symbols -

~ScriptManager
dtScript::ScriptManager, 10

- D -

DT_SCRIPT_EXPORT
export.h, 12
dtScript, 7
dtscript.h, 11
dtScript::ScriptManager, 9
 ~ScriptManager, 10
 Load, 10
 mFilename, 10
 mFileObject, 10
 Run, 10
 run, 10
 ScriptManager, 10
 Stop, 10

- E -

export.h, 12
DT_SCRIPT_EXPORT, 12

- I -

inc/ Directory Reference, 5
inc/dtScript/ Directory Reference, 4

- L -

Load
dtScript::ScriptManager, 10

- M -

mainpage.h, 13
mFilename
dtScript::ScriptManager, 10
mFileObject
dtScript::ScriptManager, 10

- R -

Run
dtScript::ScriptManager, 10
run
dtScript::ScriptManager, 10

- S -

ScriptManager
dtScript::ScriptManager, 10
scriptmanager.cpp, 14
scriptmanager.h, 15
src/ Directory Reference, 6
src/dtScript/ Directory Reference, 3
Stop
dtScript::ScriptManager, 10