INTRODUCTION TO MODDING
SWAT 4

Original Author: Christopher Kline, Irrational Games LLC

Last Updated: May 17, 2005
Last Updated by: Christopher Kline

SWAT 4 Modding Guide 1
1. Overview 3
1.1 Purpose of this document 3
1.2 Related documents 3
1.3 Terminology 3
1.3.1 Root Directory 3
1.3.2 Mod Name 3
1.3.3 Mod Directory 3
1.3.4 Mod Version 4
2. Creating a basic mod 5
2.1 Create the Mod Directory 5
2.2 Create the mod subdirectories 5
2.2.1 Create the Content directory 5
2.2.2 Create the System directory 5
2.3 Configure your mod 5
2.3.1 Tell the mod where to find your mod-specific content 5
2.3.2 Create a batch file to launch the game 6
2.3.3 Create a batch file to launch the editor 6
2.3.4 Provide information about your mod 7
2.3.5 Testing your mod 7
3. Creating a mod that requires source code modification 8
3.1 Initial setup 8
3.2 Set up the source code for your mod 8
3.2.1 Create the Source directory 8
3.2.2 Modify StartupUCC.ini 8
3.2.3 Create a batch file to compile your source code 9
3.2.4 Make your modifications 9
3.2.5 Compiling and testing your mod 10
4. Distributing and installing your mod 11

1. OVERVIEW

1.1 PURPOSE OF THIS DOCUMENT

This document is intended to help mod authors get started making mods for SWAT 4. It covers the basics of setting up and installing a mod, but does not cover all the details of every modifiable system in the game.

1.2 RELATED DOCUMENTS

None yet.

1.3 TERMINOLOGY

1.3.1 Root Directory

The Root Directory in which SWAT 4 is installed. For example, when the default installer location is used, the directory structure for SWAT 4 looks something like this:

- **C:**
    - **Program Files**
        - **Sierra**
            - **SWAT 4**
                - Content
                    - System
                    - Animations
                    - Art
                    - Classes
                    - HavokData
                    - Maps
                    - Movies
                    - Scenarios
                    - Sounds
                    - StaticMeshes
                    - System
                    - Textures

In this case the path “C:\Program Files\Sierra\SWAT 4” is the **Root Directory**. This document will refer to it from now on as <ROOT>.

1.3.2 Mod Name

This is the name of your mod. For example, if this was your first mod you might call your mod *MyFirstMod*. From now on this document will refer to the Mod Name as <MOD_NAME>.

1.3.3 Mod Directory

The Mod Directory is the base directory of all of your mod content. It is located as a subdirectory of the Root Directory with the same name as the Mod Name, i.e. <ROOT>\<MODNAME>. For example, the Mod Directory for *MyFirstMod* would be C:\Program Files\Sierra\SWAT 4\MyFirstMod\.

From now on the Mod Directory will be refered to as <MOD_DIR>.

1.3.4 Mod Version

You may need to make many changes and improvements to your mod over time. The Mod Version is a number that is used to identify which version of your mod is in use, and to indicate when your mod has changed. This is especially important for online multiplayer games, because the Mod Version is used by the game to help people determine which multiplayer servers are compatible with the version of the mod that they are playing.

The Mod Version is in the form MAJOR_REVISION.MINOR_REVISION. When your mod is first released, it will be version 1.0. As you make changes to your mod, you should update the revision number to reflect the scope of the changes that were made. How you change the major and minor revision numbers is up to you, but the convention is to update the minor revision number when only small changes are made, and to update the major revision number when major changes are made.

From now on this document will refer to the Mod Version as <MOD_VERSION>.

2. Creating a basic mod

This section describes how to create a mod that does not require source code changes. I.e., one that only modifies content (textures, static meshes, maps, etc) and INI files.

To create a mod, a few initial steps have to be taken in order.

2.1 Create the Mod Directory

The first step in making a mod is to create the Mod Directory (<MOD_DIR>) as detailed in section 1.3.3.

2.2 Create the mod subdirectories

Next you must create the mod subdirectories to hold your mod-specific content, source code, initialization files, etc.

2.2.1 Create the Content directory

Create the <MOD_DIR>\Content directory. This directory will hold your mod-specific maps, textures, sounds, and other packages.

You will need to create appropriate subdirectories to hold the different types of mod-specific content. For example, if you want to include your own mod-specific textures, you should make a directory called <MOD_DIR>\Content\Textures and save all your .utx texture packages in this directory. Likewise, if you make a mod-specific map, it should be saved into <MOD_DIR>\Content\Maps.

IMPORTANT: Do NOT save mod-specific content in **<ROOT>**\Content, because this can potentially break the retail game or other mods. All mod-specific content should reside within the **<MOD_DIR>**\Content directory or its subdirectories.

2.2.2 Create the System directory

Create the <MOD_DIR>\System directory. This directory will hold your mod-specific INI files.

After creating the directory, copy the following files from <ROOT>\Content\System to <MOD_DIR>\System:

- All the files ending in ".ini"
- All the files ending in ".int"
- The "shaders.spk" file

Now you must make all the copied INI files be writable so that you can edit them. To do this select all the INI files in <MOD_DIR>\System in Windows Explorer, right click and choose "Properties". Then make sure that the "Read Only" attribute is *not* selected.

2.3 Configure your mod

Now you must take a few steps to configure your mod so that it can properly execute with your content and code.

2.3.1 Tell the mod where to find your mod-specific content

Open the file <MOD_DIR>\System\Startup.ini, and change the line that says

```
FilePath=..\..\Content\System;..\..\Content\Scenarios;..\..\Content\Maps;..\..\Con
tent\Test\Maps
```

to

```
FilePath=..\System;..\Content\Scenarios;..\Content\Maps;..\..\Content\System;..\..
\Content\Scenarios;..\..\Content\Maps;..\..\Content\Test\Maps
```

SWAT 4 searches for directories for content in the order specified in the *FilePath* variable. All file paths are relative to the directory from which SWAT4.exe or UCC.exe is executed (this will be explained in more detail later). Since you will be running your mod from your <MOD_DIR>\System directory, the *FilePath* settings above will tell the game to look first in your mod's directories (the paths starting with "..”), and then search in the original game's content directories (the paths starting with "..\..").

2.3.2 Create a batch file to launch the game

Create a file called <MOD_DIR>\Launch<MOD_NAME>.bat. For example, if your mod is called MyFirstMod, the file should be <MOD_DIR>\LaunchMyFirstMod.bat. Now open that file in your favorite text editor and edit it so that it looks like this:

```
@echo off

REM Tell the user that we are running the mod

echo Launching <MOD_NAME>

REM Run Swat4.exe from inside <MOD_DIR>\System, so that the
REM game uses the mod's initialisation files and settings

cd .\System\

..\..\Content\System\Swat4.exe

REM Tell the user that the game has exited

echo <MOD_NAME> has exited
```

Remember to replace <MOD_NAME> and <MOD_DIR> in the above text with the correct information for your mod!

2.3.3 Create a batch file to launch the editor

If you intend to make a mod, at some point you'll probably need to run the editor. To get the editor to properly recognize your mod content, you'll need to run it from a batch file.

Create a file called <MOD_DIR>\LaunchSwatEd.bat. Now open that file in your favorite text editor and edit it so that it looks like this:

```
@echo off

REM Tell the user that we are running the editor

echo Launching SwatEd

REM Run SwatEd.exe from inside <MOD_DIR>\System, so that the
REM editor uses the mod's initialisation files and settings

cd .\System\

..\..\Content\System\SwatEd.exe

REM Tell the user that the editor has exited

echo SwatEd has exited
```

Remember to replace <MOD_DIR> in the above text with the correct information for your mod!

2.3.4 Provide information about your mod

You need to provide some additional information about your mod. To do this, open <MOD_DIR>\System\Version.ini and change the following lines:

```
        ModName=SWAT 4
```

to

```
        ModName=<MOD_NAME>
```

and

```
        PublicVersion=1.1
```

to

```
        PublicVersion=<MOD_VERSION>
```

Note: since this is likely the first version of your mod, <MOD_VERSION> will probably be 1.0. But sometimes people like to use a version number less than one if they intend to distribute the mod while it is still a work in progress. It’s up to you what version you want to use.

2.3.5 Testing your mod

Now you’ve finished with the basic setup of your mod. The next thing to do is make your modifications and test them.

You can use the LaunchSwatEd.bat file to run the SwatEd program. This will allow you to modify existing maps or create your own, and to change or create textures, sounds, animations, and many other aspects of SWAT missions. You can also use a text editor to modify other aspects of the game by editing the INI files in the <MOD_DIR>\System directory*. A full description of how to modify the different aspects of SWAT 4 is beyond the scope of this document, but this should be enough to get you started.*

Once you are ready to test your mod, you can do so by double-clicking the batch file (<MOD_DIR>\Launch<MOD_NAME>.bat) that you created in Step 2.3.2.

If you are happy with your mod and want to distribute it to other people, please follow the steps below in section 4 (Distributing and installing your mod).

3. Creating a mod that requires source code modification

Note: these steps are only necessary if your mod requires compiling custom UnrealScript code. You can skip this section if you're only modifying textures, INI files, or other non-code aspects of the game.

3.1 Initial setup

To make source-code modifications to your mod, you first you need to undertake the basic mod preparation described in Section 2 (Creating a basic mod):

1. Create the Mod Directory
2. Create the mod subdirectories
3. Configure your mod

3.2 Set up the source code for your mod

Now you need to prepare the source code for modification, compilation, and execution.

3.2.1 Create the Source directory

Create a directory called <MOD_DIR>\Source. This directory will hold the UnrealScript source code for your mod.

Now copy the source code directories named "Unreal" and "Game" from the SDK distribution into this folder. When you're done you should have a directory structure that looks like this:

- <ROOT>
    - Content
        - System
        - Maps
        - <…etc…>
    - <MOD_DIR>
        - System
        - Content
        - Source
            - Unreal
                - Core
                - Engine
                - <…etc…>
            - Game
                - AICommon
                - Gameplay
                - RWOSupport
                - <…etc…>

3.2.2 Modify StartupUCC.ini

Open the file <MOD_DIR>\System\StartupUCC.ini, and change the line that says

```
FilePath=..\..\Source\Game\System;..\..\Source\Unreal\System;..\..\Content;..\..\Content\System;..\..\Content\Classes
```

to

```
FilePath=..\Source\Game\System;..\Source\Unreal\System;..\Content;..\Content\Syste
m;..\..\Content;..\..\Content\Classes
```

This will ensure that the script compiler uses your version of the source code and not the pre-compiled shipping versions.

3.2.3 Create a batch file to compile your source code

Create a file called <MOD_DIR>\Compile<MOD_NAME>.bat. For example, if your mod is called MyFirstMod, the file should be <MOD_DIR>\CompileMyFirstMod.bat. Now open that file in your favorite text editor and edit it so that it looks like this:

```
@echo off

REM Tell the user that we are compiling the mod

echo Compiling source code for <MOD_NAME>

REM Run UCC.exe from inside <MOD_DIR>\System, so that the
REM compiler uses the mod's initialisation files and settings
REM and stores the compiled output in the <MOD_DIR>\System
REM directory

cd .\System\

..\..\Content\System\UCC.exe make -nobind

REM Tell the user that the game has exited, and wait for a keypress

echo Finished compiling <MOD_NAME>
PAUSE
```

Remember to replace <MOD_NAME> and <MOD_DIR> in the above text with the correct information for your mod!

3.2.4 Make your modifications

Now you are ready to start modding the source.

While a full discussion of UnrealScript and the source code is beyond the scope of this document, here are a few tips and resources to help you along.

3.2.4.1 Resources on modding

UnrealScript Language Reference on UDN

UnrealEngine2 Reference on UDN

UnrealWiki

3.2.4.2 Tips on modifying the SWAT 4 source code

- Whenever possible, try make your changes in a subclass of a SWAT 4 class, rather than modifying the class itself.

- If you have to modify a class that has the “native” designator in the class declaration, you must avoid adding or removing class variables or ‘native’ member functions, or else you will may get a “serialization size mismatch” crash when you try to run the game.

- If you create a new UnrealScript package (by making a new folder called <YOUR_PACKAGE> inside the <MOD_DIR>\Source\Unreal or <MOD_DIR>\Source\Game directories and placing UnrealScript .uc classes in it), you must tell the engine to load it when compiling code or running a server. You can do this by editing *both* Swat4.ini and UCC.ini to add a new "EditPackages=<YOUR_PACKAGE>" to the [Editor.EditorEngine] section, and add a new "ServerPackages=<YOUR_PACKAGE>" to the [Engine.GameEngine] section. You should put your new EditPackages line as close to the bottom of the list as possible; otherwise you may experience package-order dependency failures when trying to compile your mod.

- You may want to run the game in a window while developing. To do this, find the [Core.System] section in Swat4.ini and delete the line that says "Suppress=WindowedMode" (alternately, you can also put a semicolon in front of this line to comment it out instead of deleting it). After you do this, you can change back and forth between windowed mode and full-screen mode by pressing ALT-ENTER while the game is running.

- You may want to enable debugging output in the Swat4.log while developing your mod. You can show the log while the game is running, you first need to switch to windowed mode, and then type the console command "showlog". To enable the log debugging output, find the [Core.System] section in Swat4.ini, and delete (or comment out) the following lines:
    - Suppress=ScriptLog
    - Suppress=GuiScriptLog
    - Suppress=ScriptWarning

- For performance and security reasons, many debugging tools (which you can learn about by typing the "help" console command while the log window is open) and debugging information is disabled in the shipping game. You can re-enable it by finding the [Engine.GameEngine] section of Swat4.ini and changing "EnableDevTools=false" to "EnableDevTools=true". *NOTE: DevTools are permanently disabled during online play.*

3.2.5 Compiling and testing your mod

To compile your code, just double-click the batch file (<MOD_DIR>\Compile<MOD_NAME>.bat) that you created in Step 3.2.3. If you see a message that says

```
Success - 0 error(s), 0 warning(s)
```

then your mod has compiled successfully! (Note: sometimes you'll get a few warnings when compiling; these can usually be ignored).

Now you are ready to test out your mod. You can do so by double-clicking the batch file (<MOD_DIR>\Launch<MOD_NAME>.bat) that you created in Step 2.3.2.

If you are happy with your mod and want to distribute it to other people, please follow the steps below in section 4 (Distributing and installing your mod).

4. DISTRIBUTING AND INSTALLING YOUR MOD

It’s easy to prepare your mod for distribution. All you have to do is compress <MOD_DIR> using a program line WinRar or WinZip, and then give it to your friends to try out. As long as they extract the files to the same location (i.e., <ROOT>\<MOD_DIR>) and launch the game by double-clicking the batch file, everything should work like a charm!

*Note: when distributing a mod you do not have to distribute your source code modifications if you do not want to do so, as long as the compiled code (the *.u) files are present in the <MOD_DIR>\System directory.*