Basic Far Cry Mods Guide

2004-09-08

Introduction

This document provides a guideline for creating mods that can be played with the game FarCry. You need to have a registered copy of FarCry in order to create or play mods made by other users. FarCry has been designed from the ground up for real time editing and to be easily modified by final users. This document will provide you with a small overview of the endless modding possibilities offered by FarCry.

Note

This document is NOT a guide or a tutorial explaining how to program or how to model etc: it is assumed thorough the various sections that you are familiar with some of the basic concepts related to the part of game you're interested in modifying. No present or future release of the FarCry SDK will include any commercial tool used for development. The tools, source code and documents included in the SDK are created by Crytek and are therefore the sole property of Crytek unless specified otherwise. Read the FarCry SDK License Agreement for more information.

What's in the SDK

- Root:

*SDK License

- Docs:

*This document

* Contains documents about Lua, physics, shaders, AI, localization, characters and vehicles.

- Sandbox Documents: Sandbox Editor extensive documentation.

- Sources:

- Animations: 3ds Max® and biped animation samples.

    - ccgDump/cfgDump : C++ source code of Crytek's file format for animated and static objects.

    - Objects: 3ds Max® samples of static and animated objects, plus a vehicle.

- ResourceCompilerPC: C++ source code for creating precompiled animated objects. Useful for learning more about file formats and animated models.

- Tools:

    * simple FarCry command line batch files.

    - Bin32: 32 bit tools binaries.

- CryExporter: 3dsMax® and Maya® exporter plugin and documentation.

    - DDSPlugin: Nvidia's® DDS plugin source code and documentation.

    - LypSinch Tool: LypSinch tool binaries and documentation.

    - PolybumpPlugin: Max® , Maya® Polybump plugin and documentation.

Basics

Development Mode

There are two distinct operational modes for running FarCry: DEVMODE (developers mode) and non DEVMODE (or normal mode). To run the game in developers mode, create a shortcut or a batch file to run the main 32 or 64 bit game executable FarCry.exe, with command line –DEVMODE (an example "Far Cry DEVMODE.cmd" is included in the SDK – look in the Tools section). In DEVMODE, all cheats are enabled, and the order of file operations is different than in normal mode, as is explained later to allow an easier creation and testing of new game content (levels, scripts, etc.). However, you cannot play online when in DEVMODE, plus remember that other users will play your MOD in normal mode, so you must make sure that your MOD is working without DEVMODE. Another important difference is that when running the game in DEVMODE (or when running the editor), the file DEVMODE.Lua will be executed at runtime. This file contains a lot of additional commands and features that we used to develop the game, and that will turn out to be useful for you too when developing your MOD.

Pak Files

Data and scripts are stored in .pak files. Pak files are normal zip files that can be opened with any .zip compatible utility program. These files are stored in the folder FCData, on the original FarCry disk. They are:

- Scripts.pak: contains all .lua scripts used in FarCry. These are in simple text format and can be edited with any text-editing program such as Notepad®. For more info about Lua scripts, please consult the Lua Manual included in the Docs section.
- Shaders.pak: contains all shaders used in FarCry. Please refer to the shaders documents in the Docs folder.
- CCGF_Cache.pak: contains precompiled animation data. Please refer to the ccgDump and ResourceCompiler source code and documentation in the Sources folder.
- Effects.pak: contains binary data for particle effects. Please read the Sandbox documentation to learn how to create and use particle effects within the Editor.
- Music.pak: contains music patterns, in the form of .wav and .ogg files, used to play dynamic music in FarCry. Read the Sandbox music documentation section to learn how to add new patterns and edit dynamic music.
- Objects.pak: contains all objects and characters used in FarCry. Refer to the Sources and Docs documentation about static and animated models and data to learn more about this.
- Sounds.pak: contains all FarCry' sounds in .wav format. Read the Sandbox documentation to learn how to place soundspots, sound areas, EAX presets etc.
- Textures.pak: contains all FarCry textures, mainly in DDS format. Check out the DDS plugin in the tools folder to learn more about DDS format.

Engine Data Access

When running the game, all pak files in the FCData folder are opened. All files contained in the pak files are added to the engine virtual file system with their relative folder specified in the pak file, starting from the game's root. When the game tries to load a file, it acts differently depending whether you are running it in DEVMODE or not.

When running in normal mode, the files inside the paks are loaded first; as opposite to DEVMODE, where the files outside the paks are loaded first.

The advantage of DEVMODE is that you can easily test your MOD without bothering too much with pak files, before creating the final distribution package to exchange with other users. For instance let's say that you would like to modify the HUD compass texture, making it blue for a submarine mod. The first step is to create a subfolder in the Mods folder, something like "Mods/Submarine". If you look into the textures pak file, you would find a file called "Textures/Hud/compass.dds"; now just extract this file from the pak to the game root. If you now make changes to the compass texture and run the game normally, you won't notice any difference, because the compass texture has been loaded from the pak file. If you run the game in DEVMODE, you will see the compass replaced with yours, because now the compass texture from disk has been loaded first, before the one in the pak file. If you remove your own custom texture file, the game will now load the default one in the pak file, even if running in DEVMODE, because even though the game will try to load the texture file from disk it won't find one, and it will fallback to the default one in the pak file.

MOD Basics

For the sake of clarity and for easier packing later, instead of placing the files in the game root folder, you should move or create them into the mods subfolder. So for the above-mentioned file, the full path will look like:

"Mods/Submarine/Textures/Hud/compass.dds".

If you play the game using the Submarine MOD, the game will consider the folder "Mods/Submarine" as the root folder, and will always make an attempt to first load files from there, thus replacing the compass texture with the new one on HD, when running in DEVMODE, or with the one in the MOD pak file, when running in normal mode.

To run a MOD, you can either select it from the Mods menu list, or run the game specifying a MOD by command line. If you select it from the menu, the game will restart to ensure that all data gets correctly reloaded or replaced for the new MOD; instead the command line syntax is:

"-MOD:Submarine".

Log files and verbosity

The messages displayed on console and log files have different verbosity levels. The game can be set to display only a restricted amount of log information in the log files and console. During the making of a mod, it is often necessary to check the log file for error messages and warnings, to be sure you aren't making errors while creating your MOD. The main log files to look into are log.txt and editor.log – these files are created when you run the game or the editor. There is a console variable which defines the verbosity level, it is called "log_verbosity" and its value has a range from 0 to 8; level 0 means no messages will be displayed, level 8 means all messages will be printed in the log files and console. By default log_verbosity is set to 0.

Making new levels and content

Probably the first thing you would like to experiment with, and the easiest thing to do, is creating new levels without creating any new content or scripts. To do this all you need is the sandbox editor and the existing scripts and data. For learning how to create, modify and export levels, please refer to the Sandbox documentation. However the thing to highlight here is that as long as you don't replace any of the data included in any of the pak files, your levels don't need to have their own mod folder and they can be placed directly into the main "Levels" folder. Also, levels should stay outside the main pak file when distributing a mod. For more info read "Part 4: Making the MOD ready for distribution". Dropping down the console and typing "\map mapname" where 'mapname' is the name of the map can load any map. When you distribute the MOD, to strip down the file size you don't have to include the .cry file in the package as it contains pure editor data and is not needed in order to run the game; the game needs only the exported level data. However it's up to you to leave the editor source file included in the package to allow others to learn from what you've done. Here at Crytek we decided to leave the game open and include the source files of all levels to allow all users to see how FarCry levels were made.

Making new textures/materials and objects/characters

To make new object to add to your levels, you need a program like Photoshop® to create and edit textures, and a program like 3dsMax® to create new objects. Objects can be exported into Crytek's proprietary CGF format using the Cry Exporter plug in – please refer to the appropriate documentation to learn how to use it. Textures should preferably be exported in the DDS format; check out the DDS plugin in the Tools folder. Please notice that you can also apply new materials to objects and characters in real-time in the editor, without creating new objects. Creating new characters and animations requires a program like 3dsMax®: please look at the examples of animated characters in the "Sources/Animations" and "Sources/objects" folder. The Crytek exporter documentation and the C++ source code include a detailed description of the CGF format in case you would like to write a program to export CGF files from other modelling programs.

Sounds and music

Any music-editing program, which is able to export .wav and .ogg files, can be used to replace or create new sounds for your MOD. Please note that the music .pak file includes files in .wav and .ogg format: however the .ogg format is used only when the user selects high-quality music settings – if the .ogg file is not found, the system will try to open the .wav file; keep this in mind in case you want to replace existing music patterns. Once you make new sounds you can place them into your levels using the Sandbox editor; refer to the sound editor documentation section for more info.

Scripting

Perhaps the most flexible way to completely change Far Cry's game play behaviours and game logic with a game MOD is through scripting. Far Cry's scripting system uses the Lua language (www.lua.org); please refer to the Lua manual for more detailed information regarding the programming language itself, which is not the purpose of this document.

Also please read the Sandbox editor documentation, which includes a nice MOD and scripting section, explaining how to add new entities within the editor and the basics of scripting within FarCry.

Tip: make sure you set log_verbosity to 8 when working on a MOD.

A big chunk of the game is defined through .lua scripts, which offers the flexibility of a real programming language; however it is much simpler to work with than other languages like C++: Lua does not require any long time code compilation, and doesn't require any complex c++ studio environment to work with it: any text editor will suffice.

Additionally, within the Sandbox editor it is possible to modify and reload the scripts in real-time. In this case if you cause script errors, you will see them in the console output and in the log file.

You can also check for Lua compile errors and integrate the Lua compiler in a text editor like UltraEdit®. Script errors can cause undefined behaviours and can slow down the game at runtime and during loading.

These are the steps to integrate Lua compiler in UltraEdit®:

- Go to advanced -> tool configuration
- Command line: luacompiler %F (verify there is a luacompiler.exe in your FarCry folder)
- Working directory: c:\games\FarCry (or whatever you use)
- Menu name: Lua Compiler
- Check "output to list box" & "capture output"
- Now press insert, and then OK
- Additionally, enable view -> display line numbers

You can now use the new menu item under "advanced" or its corresponding shortcut key at any time to check your lua script.

No output means syntactically correct: you can now run the game and have a reduced chance of script errors.

If there is output it will point you to the line containing the error.

The game also includes a custom Lua debugger with a visual interface. By default the debugger is disabled. To enable it, make sure that the file system.cfg contains the line:

sys_script_debugger = "1"

If enabled, the debugger will pop up every time there is a script error. Once enabled you can invoke it any time by pressing the “f8” key, as defined in DEVMODE.lua.

At the time this document is released, there should also be Lua IDEs and debuggers freely available on the web.

Please note that while the compiler will verify the scripts for syntax errors, you can still create logical errors that cannot be found by the compiler (for instance during the game your script could try to access a table that was removed). That’s why it is important to always check the game output.

It is not necessary to manually compile the scripts in order to use them: the game and the editor compile all scripts on the fly when needed.

Although you could release scripts in binary (pre-compiled) format, we strongly encourage you to leave them in text format for other users to learn from them. In FarCry, we decided to leave all scripts in text format, opened for modifications from the community.

Along the lines of the blue submarine hud texture example, we will now modify a script. Let’s make a “one shot one kill” game mode. For this example, you will have to extract and edit the script “Scripts/Default/GameRules.lua” (this is for single player - for multi player the relative game rules script is inside the FFA, ASSULT or TDM folder). Search for the function “function GameRules:OnDamage(hit)”. At the beginning of that function, add this line:

“hit.damage=99999;”

This will inflict a huge damage to any entity causing it to die immediately. Do all these operations in the Submarine folder. Now run the game by specifying the submarine MOD. Now you should kill or get killed by everybody with one shot from any weapon. Probably you will also like to modify the available weapons equipment for players and enemies so that they have only the pistol and a few bullets in the level to start with; refer to the Sandbox editor documentation to learn how to do that.

ModExe.Lua

You have the option to execute Lua code at the moment the Mod is loaded to give the ability to start a cutscene, bind keys, run a level and many other things. To do this, simply add a file called ModExe.Lua in the MOD root folder, and add there any initialisation code you would like to have for your MOD.

Making the MOD ready for distribution

Your cool MOD is now ready to be shared with other FarCry players worldwide. You should make it as simple as possible for other users to install your MOD: the best way will be to make an installer; in case you cannot create one then you should at least provide a zip file, so that the final user can simply extract it into the game MODS folder.

As we mentioned before, a pak file should be created for correctly playing the MOD. There is a batch file provided to automate this process: PackerForDistrib.bat, in the “tools” folder.

Basically everything should go into the pak except the “Levels” folder. You should have this file already in the “Mods” folder in the original FarCry distribution. Run the batch file with the MOD name you wish to create the distributable MOD package, for instance “PackerFordistrib Submarine”. The batch file is looking for PKZIP® 2.5 to create the pak, which is freely available from the pkWare® website; however you can still do this manually using other zip utility programs. The batch file will create a subfolder in the “Mods” folder, called “Distrib”, which contains the distributable package. It is good practice now to run the game without DEVMODE and test your MOD by selecting the newly created Distrib MOD from the MOD game menu. If everything works correctly, the entire content of the “Distrib” folder is what you should zip, or provide an installer for, and share with other users.

For extra clarity you can add additional information about your mod and a preview picture as well. Adding a file called ModDesc.txt does this. The game searches the text for keywords and displays them in the MOD menu. The value of the key should be enclosed by “ ”.

These are the keywords and examples you should use in the text:

- _Title_: This is the title of the MOD. Example: _Title_= “MyMod”
- _Author_: This is your name or the name of the team who made the MOD. Example: _Author_ = “Mario Ranza”
- _Version_: Version of the MOD. Example: _Version_ = “1.0.0.0”
- _Website_: MOD Website. Example: _Website_ = “www.FarCrySubmarineMod.com”
- _Description_: MOD description. You can provide a detailed description of your MOD here.

The box will create a scrollbar to scroll the text up and down if it doesn’t fit in the visible area of the box.

Any other text is ignored so you can also add comments in the mod description that won’t show up in the mod menu (like a troubleshooting section etc.).

The mod picture preview should be in DDS format, size 256*256. Should be put in the MOD root folder, with ModDesc.txt and ModExe.lua, and should be called ModPreview.dds.

After this is done you are ready to create the final distribution package and share it online with other users. Check the official FarCry website and other fansite for additional info and updates.

Make sure you have completed all steps correctly and test the MOD carefully before distributing it; otherwise other players will run into technical issues and will not fully enjoy your MOD.