SockNastre 01/07/2021

Real File System

To start off on the Real File System of Mirror's Edge Catalyst we should probably talk about what a *Real* File System is. We will discuss this simply and briefly.

Intro: Real File System (RFS)?

A Real File System, abbreviated RFS, is the standard File System used on your operating system. It houses files and data in the way you expect it to be seen on the operating system you use. For this document the focus will be on the Windows version of the game, with occasional text about the other platforms if applicable.

Example (Windows):

Above is an example of the RFS of Windows, but of course there is much more from System32 folders to the Programs directory, and in fact essentially the whole drive that is being used for Windows is part of its RFS (probably)!

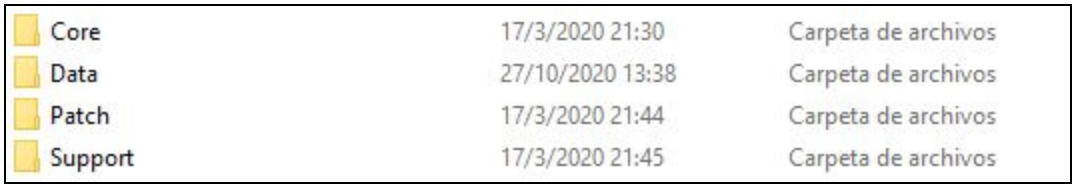
In the case of a game console like the PlayStation 4 if you were to FTP into it you would see a lot of directories and drives the PlayStation uses to run itself and manage games and such. All that is part of the RFS of the PlayStation 4!

Part 1: Root Folder

Let's look at the root folder of Mirror's Edge Catalyst, this is the place where we find the two folders "Data" and "Patch"; both folders are responsible for housing and organizing data in the game like textures, meshes, sounds, etc.

The "Patch" folder may not be there if no patch (AKA update) was installed onto your game. On other platforms (PS4 or Xbox One) the update system may work differently in how the console organizes game data.

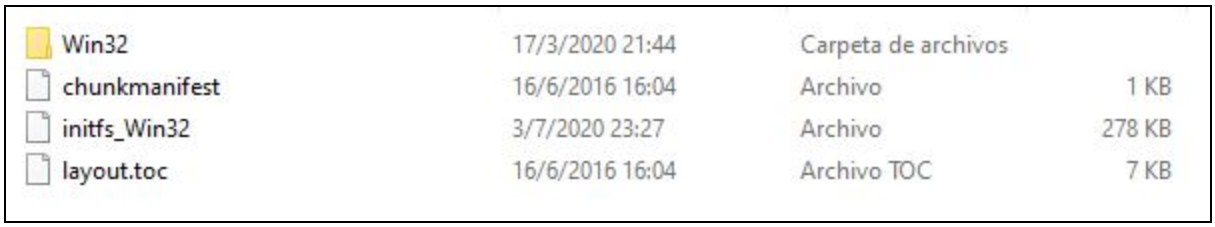
Folders (Windows):



For right now these two folders ("Data" and "Patch") are the most important to us, in regards to modding at least. So the rest of this document will largely focus on those two…

Part 2: Data Folder

Looking inside of the "Data" folder we see four things:



Depending on your platform, you may see a folder named "Win32", "Gen4a", "Gen4b", etc. This also applies for "initfs_Win32", on PS4 this is "initfs_Gen4b" and so on.

Let's break down each thing:

Win32 (Folder)	Holds (asset) data and metadata (bundle, superbundle, etc.) related to assets and EBX class files. We will delve deeper into this folder later.
chunkmanifest (File)	A file that initializes IDs for "chunks" of data.

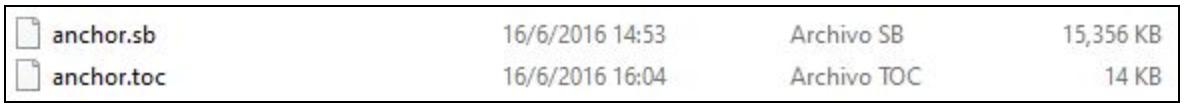
initfs_Win32 (Archive)	Holds scripts, properties, and other things that aren't strictly limited to inside the game like stuff in the "Win32" folder.
layout.toc (Table of Contents File)	Has information about layout, properties, chunk ID setup, and more. Lays out stuff in the "Win32" folder.

Once we get to the *Virtual File System* document chunkmanifest, initfs_Win32, and layout.toc will be discussed in more detail.

Part 3: Win32 Folder

Let's now look into the Win32 folder we saw before. Inside of it there are quite a bit of files and folders, but don't worry! It's a lot simpler than you think to understand.

Aside from the "gameconfigurations" folder (which we will look at later) you will find two different file formats with the name scattered all throughout. They look something like:



The "sb" extension stands for "superbundle" and "toc" for "table of contents". The "sb" files contain a list of bundles and the "toc" files lay out those bundles alongside other properties. *The "toc" files may also lay out chunks*.

The name of each file signifies what its data pertains to. *All chunk metadata is stored inside of "chunks0.sb" and "chunks0.toc". Other files still contain chunk data relevant to them though.*

Part 4: gameconfigurations Folder

Looking at the "gameconfigurations" folder we saw before, we notice that we don't see any .sb or .toc here. Just folders and inside those folders .cat and .cas files.

Each of the folders you see are known as "install chunks". They represent different data in the game that is "installed" (loaded) to be used at certain points.

Inside of the folders you find one "cas.cat" and a variable amount of "cas_XX.cas" archives where the "XX" is any number from 01 to 99.

Let’s break down the responsibilities of each file:

cas.cat	Is a catalog of data inside each “cas_XX.cas” archive. File names are not kept here, rather IDs are used to find specific data.
cas_XX.cas	Contains raw (compressed) data in no particular order with no header.