The Fresh Asset Processor

Design 2011-Feb-19

The goal of the Fresh Asset Processor (FAP) is to simplify the development and inclusion of audio, texture, and potentially other asset types into Fresh-based applications. The FAP accomplishes this goal by automating a variety of asset processing tasks:

1. Conversion of file types to other file types
2. Compression
3. Ensuring that asset formats are correct (e.g. 16-bit sound, power-of-2 textures)
4. Recording important metadata (such as original texture dimensions prior to conversion)
5. Automated standardized renaming
6. Placement of assets into desired folder hierarchies

These last two features also speak to the FAP’s ability to aid in processing assets for a variety of game flavors, and platforms and formats within a platform (e.g. Crush the Castle Lite vs. Crush the Castle Full; Windows vs. iOS platforms; iPhone 480x320 vs. iPad 1024x767 format ).

# The FAP Pipeline Process

Taking a game like Crush the Castle as an example, let’s look at how the FAP fits into the development of multiple flavors of the game.

Note first that the FAP on its own sits in the Fresh/tools/ folder (or the system $PATH) and therefore does not change from flavor to flavor.

Each particular flavor of the game (e.g. Crush the Castle Lite on iPhone Retina) has an Asset Transform Set (an XML file) that tells FAP how to generate the particular assets for that flavor. Asset Transform Set (ATS) can include other ATSs, so a flavor may if it wishes simply modify another ATS. Therefore the Crush the Castle Lite for iPhone Retina ATS may extend the Crush the Castle Lite ATS, which in turn may extend the base Crush the Castle ATS.

In addition to the set of interrelated ATSs, the FAP also draws from a central source pool of assets. In the case of images, these are typically PSDs. They may be scattered over a variety of subfolders within a common root.

Given this ATS (or perhaps ATS graph) and reservoir of source files, the FAP processes each asset within the ATS according to the directives indicated in the ATS. This results in the production of output files as well as a unified output manifest. The result is a set of pre-processed, simplified assets and a manifest describing these assets to the game.

# The ATS Format

The Asset Transform Set file is an XML file with root element <asset-transform-set>. The root element has attributes defining the source root folder and the destination root folder.

Each child element defines the processing of a single asset. The child element is named after the type of asset it processes: currently either <image> or <audio>.

Each asset element indicates the relative or absolute path to the source asset file. This is the only required datum.

The asset element may also indicate the relative or absolute path of the output file. If no output file is indicated, the file is placed in the destination root and has the same name and extension as the input file.

The asset element may also indicate an asset name. By default this is identical to the file name and extension of the output file.

Additional data are type specific.

## Images

Image elements may specify:

* Whether and how the image should be converted to a power-of-2 size image. Options are: don’t resize, upsize (the default), downsize, and closest size.
* A color key used to make pixels with the given color (+/- an error allowance) totally transparent (disabled by default).
* Whether to premultiply the source alpha into the RGB components (false by default).

Image elements also cause the output manifest file to specify an image entry in the manifest. This entry indicates:

* The asset name
* The image file path
* The image’s file size in bytes
* The image’s actual dimensions
* [Optional] The image’s original dimensions (if different from the actual dimensions)

# Additional Features

The FAP will also verify that all asset file names are unique. This honors the expectation within Xcode that all resources sit together in the Resource root within the bundle and therefore should have distinct names.

Optionally, the FAP may modify a specified .xcodeproj project file to automatically reference the assets indicated in the manifest so that they are correctly copied to the application bundle.