Creating Custom Landscape Importers

Guide to creating new Landscape import formats via plugin.

Creating your own heightmap and weightmap formats for importing landscape data is accomplished by writing a plugin. The plugin will add your new format to the engine, as well as importing the data from your files.

Writing A Custom Importer

- In order to create new importers, your plugin should create instances of objects implementing `ILandscapeHeightmapFileFormat` and `ILandscapeWeightmapFileFormat`, and register those objects with `ILandscapeEditorModulemodule::RegisterHeightmapFileFormat` and `ILandscapeEditorModulemodule::RegisterWeightmapFileFormat`, respectively.
- Implementing the `ILandscapeHeightmapFileFormat` interface in a plugin requires overriding the following functions:
  1. `const FLandscapeFileTypeInfo& GetInfo() const`: Returns type information indicating which file types this class handles, and whether or not exporting is supported.
  2. `FLandscapeHeightmapInfo Validate(const TCHAR* HeightmapFilename) const` - Validates the named file, or rejects it and returns an error code and message.
  3. `FLandscapeHeightmapImportData Import(const TCHAR* HeightmapFilename, FLandscapeFileResolution ExpectedResolution) const` - Actually imports the file.
  4. `void Export(const TCHAR* HeightmapFilename, TArrayView<const uint16> Data, FLandscapeFileResolution DataResolution, FVector Scale) const` - Exports the file, if this format supports exporting (see the return value from `GetInfo`). This is the only function that doesn't need to be overridden in order to compile. However, if it is called without being overridden, it will call `check`.
  5. `(Destructor)` - Classes that implement this interface should use a virtual destructor, as they will be deleted via a pointer to the interface class.
- Implementing the `ILandscapeWeightmapFileFormat` interface is nearly identical, with only minor differences in some return types:

1. `const FLandscapeFileTypeInfo& GetInfo() const` - Returns type information indicating which file types this class handles, and whether or not exporting is supported.
2. `FLandscapeWeightmapInfo Validate(const TCHAR* WeightmapFilename) const` - Validates the named file, or rejects it and returns an error code and message.
3. `FLandscapeWeightmapImportData Import(const TCHAR* WeightmapFilename, FLandscapeFileResolution ExpectedResolution) const` - Actually imports the file.
4. `void Export(const TCHAR* WeightmapFilename, TArrayView<const uint16> Data, FLandscapeFileResolution DataResolution, FVector Scale) const` - Exports the file, if this format supports exporting (see the return value from `GetInfo`). This is the only function that doesn't need to be overridden in order to compile. However, if it is called without being overridden, it will call `check`.
5. `(Destructor)` - Classes that implement this interface should use a virtual destructor, as they will be deleted via a pointer to the interface class.

- For further information and examples, you can see the interfaces in `LandscapeFileFormatInterfaces.h`, the .PNG implementations in `LandscapeFileFormatPng.cpp` and `LandscapeFileFormatPng.h` , and the .RAW implementations in `LandscapeFileFormatRaw.cpp` and `LandscapeFileFormatRaw.h`. All of this code is in the LandscapeEditor module in the engine.