Error Handling

When encountering an error during Katana scenegraph traversal RfK with either warn or abort according to specific settings on the scene and/or location in question. RfK also allows this behavior to be overridden with a custom AttributeFunction.

Katana Error Handling

Errors within Katana scenegraph traversal are indicated (from Katana or the scene recipe) by an "errorMessage" attribute at that location. A fatal error is indicated when the location "type" has been set to "error" or if the attribute "errorSeverity" has been set to "critical". RfK will skip over error locations by default, which can result in missing geometry in the render. If you want RfK to abort the render when it encounters an error location, you can set the `prman GlobalStatements.errorHandler` to "`abortall`". With this attribute set, all error locations will cause an aborted render. However, if you wish to have more control over the specific errors that should cause an abort, there is a mechanism by which a custom override handler can be added to intercept and reinterpret the error location. An attribute function can be injected into RfK that will make the final determination of whether a location error is fatal or not. This attribute function runs after RfK has done its work for deriving the error's fatality essentially giving the author the last word.

The attribute function is given the location path and the location's attributes as input and is expected to return an attribute containing an integer "abort" attribute indicating whether or not the render should be terminated.

Below you'll find an example of an override attribute function. The function is activated by setting the following string attribute at /root:

```
"prmanGlobalStatements.plugin.katanaErrorHandlerAttrFncs" = "MyOverrideKatanaErrorHandler"
```

```
class MyOverrideKatanaErrorHandlerFnc : public Foundry::Katana::AttributeFunction
{

public:

    static FnAttribute::Attribute run(FnAttribute::Attribute args)
    {
        FnAttribute::GroupAttribute errorHandlerArgs = args;

        // Incoming arguments include location's attributes and the location path
        FnAttribute::GroupAttribute locationAttrs = errorHandlerArgs.getChildByName("attrs");
        FnAttribute::StringAttribute pathAttr = errorHandlerArgs.getChildByName("locationPath");

        std::string path = pathAttr.getValue("", false);

        // Return the integer "abort" attribute true to abort render, false to continue
        FnAttribute::GroupBuilder gb;
                if (dont_want_to_abort_on_error) // for example, if(path == "/root/world")
                {
                gb.set("abort", FnAttribute::IntAttribute(false));
        }
                else
                {
                gb.set("abort", FnAttribute::IntAttribute(true));
        }
                return gb.build();
    }
};

DEFINE_ATTRIBUTEFUNCTION_PLUGIN(MyOverrideKatanaErrorHandlerFnc)

void registerMyKatanaOverride()
{
    REGISTER_PLUGIN(MyOverrideKatanaErrorHandlerFnc, "MyOverrideKatanaErrorHandler", 0, 1);
}
```

RenderMan Error Handling

Errors in RenderMan contain a error code and error message. By default, these errors will show up in the Render Log, but the render will not terminate. You can change the preference for this with the `prmanGlobalStatements.errorHandler` option in PrmanGlobalStatements. Setting the value to "abort" or "abortall" will terminate the render. For finer control, RfK provides a mechanism to override an errorHander set to abort or arbortall based on specific error codes and error messages. An attribute function is injected into prman's error handling system to determine if an error should result in termination of the render.

The arguments to the attribute function are the error code and the error message. The function is expected to return an attribute containing an integer "abort" attribute indicating whether or not the render should be terminated.

```
"prmanGlobalStatements.plugin.rendermanErrorHandlerAttrFncs" = "MyOverridePrmanErrorHandler"
```

```
class MyOverridePrmanErrorHandlerFnc : public Foundry::Katana::AttributeFunction
{

public:

    static FnAttribute::Attribute run(FnAttribute::Attribute args)
    {
        FnAttribute::GroupAttribute errorHandlerArgs = args;

        // Incoming arguments include the error code and error message
        FnAttribute::StringAttribute errorMessageAttr = errorHandlerArgs.getChildByName("errorMessage");
        FnAttribute::StringAttribute rmanCodeAttr = errorHandlerArgs.getChildByName("rmanCode");

        std::string rmanCode = rmanCode.getValue("", false);

        // Return the integer "abort" attribute true to abort render, false to continue
        FnAttribute::GroupBuilder gb;
                if (dont_want_to_abort_on_error) // for example, if(rmanCode == "X00002")
                {
                gb.set("abort", FnAttribute::IntAttribute(false));
        }
                else
                {
                gb.set("abort", FnAttribute::IntAttribute(true));
        }
                return gb.build();
    }
};

DEFINE_ATTRIBUTEFUNCTION_PLUGIN(MyOverridePrmanErrorHandlerFnc)

void registerMyPrmanOverride()
{
    REGISTER_PLUGIN(MyOverridePrmanErrorHandlerFnc, "MyOverridePrmanErrorHandler", 0, 1);
}
```