Candera 3.7

To keep backwards compatibility for any interface change in the public API the old interface has been kept and marked as deprecated. The deprecated interfaces will be removed in future versions. Compiler warnings will be generated when using a deprecated interface through the usage of Candera Macro CANDERA_DEPRECATED_3_7_0.

- Candera Platform
- Candera Base
- Candera Engine 2D
- Candera Engine 3D
- Candera Transitions
- Text engine
- Candera Scripting
- Candera System

Candera Platform

Owner Requirement

OpenGL Offscreen Render Targets (Frame Buffer Objects) are now required to have an owner RenderTarget to ensure that the used EGL context is safe. For this reason RDenderTarget now has the "IsOwnerRequired" function which returns if a parent render target is needed. The concrete RenderTarget implementation needs to implement the function accordingly. If IsOwnerRequired returns true, an Upload of the RenderTarget will only succeed if the owner is set. RenderTarget3D has additionally a protected SetOwner function that classes derived from RenderTarget may call to set their parent render target. This mechanism can of course also be used for non-OpenGL implementations. Widgets and Behaviors need to take care that owners are set correctly.

GlInclude Header File

From now on, the GlInclude header file only includes Open GL header files if a CGIDEVICE_OPENGLES_xx has been defined. E.g. the NullRenderDevice does no longer include them.

Interface to load / store binary shader programs

Binary Shader Programs or Object can be retrieved from the driver and stored to an arbitrary persistence mechanism (e.g. a file.). This is supported by the RenderDevice and can be used by Device Packages. Therefoe RenderDevice::UploadStoredShaderBinaryProgram and RenderDevice::RetrieveShaderBinaryProgram have been added if the CANDERA_SHADER_PROGRAM_PERSIST_INTERFACE_ENABLED flag is defined.

Candera Base

Post Processing System

The new Post Processing System provides an interface to the user to implement their own post processing effects. Such effects are attached to cameras and get callbacks from the system every time the corresponding camera has finished rendering its scene. The effect then processes the camera's render target, and the system routes the render target automatically through the stack of effects.

For ease of use, post processing effects can also be scripted using Lua.

Math numeric stabilization

Standard rotation values like 0, 90, 180, 270 degree should result in stable matrix values. But, due to the calculation of the rotation matrix values the resulting rotated position values have been quite unstable. Therefore, these values are handled separately. Including an optimized handling of rotation value 0 (which no longer leads to sine and cosine calculations). The Math interface of Candera has been extended by NormalizeDegree and DegreeSineAndCosine to implement this optimization.

Candera::WidgetBase::SetStringId with disposer deprecated

The SetStringId interface with a disposer function of WidgetBase has been deprecated due to optimization of the the StringId memory consumption.

Touch event extended

Due to extension in the controls the touch event has been extended by a camera. For details please take a look into controls change log. Beside the additional camera parameter of the touch event the GraphicDeviceUnit interface has been extended by a GetSourceId method to enable an improved sorting of touchable behaviours in the controls.

Glyph Atlas Outline

The glyph atlas has been extended to support glyph outlines.

ThreadedAssetProviderDispatched Outline

Class ThreadedAssetProviderDispatched was updated to offer access the worker thread. This allows to set the worker thread priority.

Candera Engine 2D

Renderer 2D Statistics

Candera's 2D renderer statistics have been extended to keep statistics about uploaded bitmap images, and render camera calls.

Asset References

A scene can now reference assets that are not referenced by scene graph objects, and upload/unload them whenever the scene is uploaded/unloaded.

Candera Engine 3D

Renderer 3D Statistics

Candera's 3D renderer statistics now keep statistics about vertex buffers, shaders, cubemaps, direct textures, render camera calls, and render state cache hit/miss rates (if enabled).

The renderer statistics overlay has also been extended to support the new values.

Canvas Text Outline

The canvas text has been extended to support outlines.

Texel Size Auto Uniform

New auto uniforms have been added, called "u_TexelSize", "u_TexelSize1", .. "u_TexelSize8", which pass a vec2(1.0/(textureWidth*4), 1.0/(textureHeight*4)) to the uniform corresponding with the texture at the given unit. The only purpose for this is to support the outline reference shader.

Asset References

A scene can now reference assets that are not referenced by scene graph objects, and upload/unload them whenever the scene is uploaded/unloaded.

Renderer Functions

New functions were introduced in the Renderer to support implementation of post processing effects. Render targets can now be retrieved from the Renderer for temporary use, while a Blit function provides a convenient way to apply a shader to a full screen quad.

Candera Transitions

Transition reverse improvements

Optimizations on the transition reverse algorithm were done. Reverting an already reversing transition is possible now.

Transition requests

A Candera::Transitions::Request::End() might now return a null pointer depending on whether the request was filled with content or not. The framework changed so that it uses lazy object creation now to minimize memory operations.

Text engine

http://dev.doc.cgistudio.at/APILINK/class_

Candera Scripting

Lua Post Processing Scripts

The post processing system has also been integrated into the Lua script system. Candera Lua Post Processing offers an easy way to create post processing effects directly in Scene Composer using Lua.

Lua Bindings

Candera's Lua bindings have been extended with various functions to facilitate development of post processing effects.

Candera System

Component Handle

The ComponentHandle class from the entity component system has been refactored to use FeatStd's GenericHandle.