Candera V3.2.0

Release Date: February 2016

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_2_0.

Candera/FeatStd CMake Build System Changes

i.MX6 Custom Driver Includes

With the newest board support packages the drivers are available together with the other includes and libraries required when cross-compiling for the i.MX6. Therefore, it is now no longer required to specify custom driver include and library paths. The CGIDEVICE_CUSTOM_PATHS CMake parameter has been removed, together with the CGIDEVICE_TARGET_IMX6_DRIVER_PATH_INCLUDE and CGIDEVICE_TARGET_IMX6_DRIVER_PATH_LIB parameters. Ensure that the driver includes and libraries are available in the sysroot specified for cross-compilation.

Candera Engine 3D

2D Composition in 3D using Canvas Objects.

New node types have been added to the 3D scene graph, namely CanvasSprite and CanvasGroup. Both offer means to realize 2D items in 3D space. These canvas objects are transformable in their local XY plane. CanvasSprite reacts on the size of the set texture and offers means to provide a pivot offset, such that rotations or scale is done around this pivot point. With these objects 2D planes containing several 2D elements can be defined, but even be transformed in 3D space. Using an orthographic even pixel exact positioning is possible.

RendererListener Camera Hook

A new hook is defined for RendererListener - RendererListener::OnCameraPostRender that is called after a camera has been rendered.

Node intersection in Screen Coordinates

An interface has been added to compute whether a node intersects with a pick in screen coordinates - Math3D::IsPickIntersectingGeometry.

Added MemoryPool property to TextureImage

TextureImage class has now a memory pool property which indicates where to store the image data for rendering. This property may be ignored on RenderDevices that only support uploading to VideoRam.

Candera Engine 2D

2D Node Render Benchmark Property

New methods have been added for getting/setting of render measure value of Node2D: Node2D::SetRenderBenchmark and Node2D::GetRenderBenchmark. These values are used by a camera render strategy and the rendering of the nodes is done until the sum of these render benchmark values exceed a defined threshold.

Renderer2DListener Camera Hook

A new hook is defined for Renderer2DListener - Renderer2DListener::OnCameraPostRender that is called after a camera has been rendered.

GridLayouter Cell Spanning

A new feature has been added to GridLayouter that allows merging of adjacent columns and/or rows into larger single cells. Two new properties are defined in this case for each element inside a GridLayouter, **RowSpan** and **ColumnSpan**, which can be accessed by corresponding getter/setter methods: GridLayouter::SetRowSpan, GridLayouter::GetRowSpan and respectively GridLayouter::SetColumnSpan, GridLayouter::GetColumnSpan.

2D Scene Graph support

CGIAnalyzer supports 2D Scene Graphs now.

BitmapImage2D::MemoryPool changed

Enumeration values available for BitmapImage2D::MemoryPool have been modified to BitmapImage2D::MemoryPool::VideoMemory and BitmapImage2D::MemoryPool::ClientMemory. The previous values SystemMemory and FlashMemory are now deprecated.

Node2D::LayoutingRectangle deprecated

The dynamic property Node2D::LayoutingRectangle has been deprecated. The only use for this parameter was to allow layouting of text in a uniform way with images. This was made obsolete by the introduction of TextNode2DLayouter. The funcitonality introduced by this property has been moved to the layouter, as follows:

For controlling the size of the layout rectangle, one shall use the dynamic property Node2D::Size (accessible via Layouter::*Size*) For controlling the position of the layout rectangle, one shall use the dynamic property Node2D::Margin (accessible via Layouter::*Margin*) For computing the layout rectangle, a generic interface is available as Layouter::GetComputedLayoutRectangle. This is also available in a non-generic flavour as: Node2D::GetComputedBoundingRectangle for nodes that don't display text and TextNode2D::GetLayoutTextRectangle for text.

Candera Base

Event Listener

This version introduces the foundation for a new event/listener system based on two classes, Implementing the event handlers of IEventListener and EventListener. The motivation for this is to reduce the amount of virtual functions, reduce code needed for predefined event listeners as well as to improve extensibility of existing events. Events are passed to a single OnEvent function which should use Candera RTTI to differentiate between event types and call concrete non-virtual handling functions. Candera will not provide implementations of listeners, only interfaces, further reducing code size.

Specializations for StringBufferAppender added

Specializations for the FeatStd::StringBufferAppender::Append method have been added for all Candera types used with DataType:

- Margin
- Layouter *
- HorizontalAlignment
- VerticalAlignment
- Color
- Rectangle
- Vector2
- Vector3
- TextRendering::Font
- SharedPointer<Image2D>
- Node2D *
- Camera2D *
- RenderNode *
- Group2D *
- Node *
- SharedPointer<Shader>
- Group *

- Mesh *
- Light *
- Camera *
- RenderDevice2D::BlendFactor
- RenderDevice2D::BlendOperation
- RenderDevice2D::Filter
- RenderDevice2D::MipMapFilter

Candera Scripting

This version introduces a scripting language agnostic base component system, as well as an implementation that binds the Lua scripting language to Candera. For an introduction to Lua scripting in Candera, refer to this section.

Candera System

Entity Component System

This version offers an entity component system to facilitate development using the entity component programming paradigm. An entity component system is a model to attach data (components) to objects (entities). It provides fast iteration over data at the cost of slightly higher setup and teardown times. Entities do not contain components as members, so Entity derived classes do not increase the size of your objects. This approach lends itself to data oriented programming, and is an alternative to, not a substitute for, object oriented programming.

SharedPtrProperty::Set(T* value) deprecated

The method SharedPtrProperty::Set(T* value) has been deprecated. Widgets/Applications should use SharedPtrProperty::Set(const TPtr& value) instead.

2D and Custom Experiments added

Global Experiment "Ignore Draw Calls" is available for 2D applications. Custom Experiments allow self-constructed Experiments which can be (de-)activated by CGIAnalyzer. These experiments are based on function pointers allowing more functionality and a wider range of usages.

Candera Device

No changes.

Candera Transitions

This version introduces a basic transition framework to Candera to allow activation and deactivation of nodes or scenes using configurable transition fragments based on a set of rules and a request. Currently only fading and switching is supported. For details on how to use the transition framework please refer to PID_app_dev_tutorial_11 on how to use the framework as well as the documentation of the Transitions::TransitionStrategy, which lies at the core of the framework.

TextEngine

Support for TTC (True Type Font Container)

Support for TTC (True Type Font Container) has been added to TextRendering::Font and TextRendering::FontStore classes. The single fonts inside a collection can be accessed by specifying a Face Index.

Internal Namespace for Font Engine Integrations

The namespaces of the Font Engine integrations - "FreeType" and "BitmapFont" - have been changed to "Internal".

Migration Guide

Following an overview about interface changes between Candera V3.1.1 and Candera V3.2.0.

Description	Candera V3.1.1	Candera V3.2.0
BitmapImage2D::MemoryPool enumeration changed	`bitmapImage2D->S` `bitmapImage2D->S`	`bitmapImage2D->SetMemoryPool(` `BitmapImage2D::ClientMemory);`
Deprecated SharedPtrProperty::Set (T* value), replaced with SharedPtrProperty::Set (const TPtr& value).	`Image2D* image;` `m_image.Set(imag`	`SharedPtrProperty<Image2D> image` `m_image.Set(image);`
Face Index introduced for TTC (True Type Font Container) support.	`UInt8 faceId = t`	`UInt8 faceId = fontStore->GetDes`