CGI Studio 3.3.0

Candera V3.3.0

Release Date: November 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_3_0.

Candera/FeatStd CMake Build System Changes

Candera Engine 3D

Changed Light Specular Color to White

Changed default specular light color to White due to feedback from designers. Material default specular color is black anyway, so no specular highlight will occur by default. But if user (designer, technical artist) changes material specular color, an immediate change of appearance will be visible.

Reduction of glGetUniform calls on Shader Upload

By improving handling of Shader Uniforms, the number of calls to glGetUniform during shader uploading was significantly reduced.

Introduction of LoadingHint for DeviceObject.

Version 3.3.0 introduces a LoadingHint for the DeviceObject. Loading hints can e.g. trigger warnings when uploading while rendering, or force unloading from all contexts.

New methods for NodeListener

New methods have been added to NodeListener class. These are:

- Candera::NodeListener::OnNodeAdded: this is invoked whenever a parent is set for the node;
- Candera::NodeListener::OnChildAdded: this is invoked whenever a child node is added to the node;
- Candera::NodeListener::OnChildRemoved: this is invoked whenever a child of the node is removed.

Candera Engine 2D

Introduction of LoadingHint for DeviceObject2D.

Version 3.3.0 introduces a LoadingHint for the DeviceObject2D. Loading hints can e.g. trigger warnings when uploading while rendering, or force unloading from a context.

New methods for Node2DListener

New methods have been added to Node2DListener class. These are:

- Candera::Node2DListener::OnNodeAdded: this is invoked whenever a parent is set for the node;
- Candera::Node2DListener::OnChildAdded: this is invoked whenever a child node is added to the node;
- Candera::Node2DListener::OnChildRemoved: this is invoked whenever a child of the node is removed.

TextNode2D cloning strategy

TextNode2D adopts the unified cloning mechanism, with the introduction of DeepTextNode2DCloneStrategy.

For more information see Unified Cloning Interface, Tree Cloning Strategies and Tree Cloner

---

Improved Pixel Accuracy

Scissoring is now utilized on platforms that support it. This improves pixel accuracy in 2D rendering when a dirty area or a camera scissor is used.

Candera Base

Renderer2D3D

A new renderer has been added. It renders the cameras interleaved in order of their sequence-number. This means that 3D-cameras can be rendered between two 2D-cameras and vice versa. The interleaved renderer however, has the requirement that buffer swapping and clearing needs to be done by the according Cameras directly. Therefore auto clear and auto swapping becomes deprecated. Please see the Migration Guide.

DXT image formats

New compressed formats support in Bitmap is avaiable on selected platforms: Dxt1CompressedRgbFormat, Dxt1CompressedRgbaFormat, Dxt3CompressedRgbaFormat, Dxt5CompressedRgbaFormat.

Candera V3.3.0

Candera Scripting

Lua Update.

Lua has been updated to version 5.3.3.

New Script Callbacks.

This version adds a new callback:

- FixedUpdate

Time interface.

This version adds a new interface to get time information in Lua.

Added uniform functions.

This version adds new functions to Lua to access uniforms through Candera's shader parameter setter.

Candera System

Update System

This release adds an update system where entities can be registered to receive any of a series of update calls.

Controller System

This release adds a controller system which offers an abstract interface that the user can implement and attach to entities to receive update calls.

Candera Device

Candera Transitions

New Default Transition Types

The transition framework now supports Cut, Fade, Slide and Scale transitions.

Transition Request Handles

Transition requests can now be assigned a handle allowing creation of multiple requests at once. This is a requirement for courier

---

TextEngine

TTC font iterator.

A function to iterate over the style names of the font faces contained by a TTC font was introduced in the Font.h interface. Additional functions to retrieve the font style name and the font family name of the current font set up are available.

Candera Behavior

Added support for Behaviors.

Behaviors have been added to support application development with considering the hierarchy of the scene graphs. Behaviors are small application building blocks that allow the user to create their own application handling. Being similar to Widgets Behaviors have the advantage that they can rely on hierarchical information, as every Behavior has to be attached to a node in the scene graph. By design they can be implemented in such a way that one behavior can even be reused for both 2D and 3D, if no special functionality of the Node below has to be used. Behaviors also support event handling through mechanisms for event routing (to other behaviors) and having a base interface (Behavior::OnEvent) to handle events.

Migration Guide

Following an overview about interface changes between Candera V3.2.2 and Candera V3.3.0.

Description	Candera V3.2.2	Candera V3.3.0

RenderTarget Changes due to Renderer2D3D.	void RenderTarget::SetAutoSwapEnabled(bool enabled);
	bool RenderTarget::IsAutoSwapEnabled() const;
	void RenderTarget::SetAutoClearEnabled(bool enabled);
	bool RenderTarget::IsAutoClearEnabled() const;
	void RenderTarget2D::SetClearMode(const SharedClearMode2D::SharedPointer& clearMode);
	SharedClearMode2D::SharedPointer RenderTarget2D::GetSharedClearMode() const;
	void SetClearMode(const SharedClearMode::SharedPointer& clearMode);
	SharedClearMode::SharedPointer GetSharedClearMode() const;
		Use corresponding Camera and Camera2D settings.
		
		void Camera2D::SetClearColorEnabled(bool isEnabled);
		bool Camera2D::IsClearColorEnabled();
		void Camera2D::SetSwapEnabled(bool isEnabled);
		bool Camera2D::IsSwapEnabled() const;
		void Camera2D::SetClearColor(const Color& clearColor);
		const Color& Camera2D::GetClearColor();
		void Camera::SetClearMode(const ClearMode& clearMode);
		const ClearMode& Camera::GetClearMode();
		void Camera::SetSwapEnabled(bool enable);
		bool Camera::IsSwapEnabled() const;
		
		\ Use dedicated clear and swap cameras if you don't want to manage clearing and swapping on dynamic camera configurations.

RenderDevice Changes due to LoadingHint	static bool RenderDevice::UploadBitmapTextureImage(BitmapTextureImage& textureImage, UInt unit);
	static bool RenderDevice::UploadCubeMapTextureImage(CubeMapTextureImage& textureImage, UInt unit);
	static bool RenderDevice::UnloadBitmapTextureImage(BitmapTextureImage& textureImage);
	static bool RenderDevice::UnloadCubeMapTextureImage(CubeMapTextureImage& textureImage);
	static bool RenderDevice::UploadVertexBuffer(VertexBuffer& vertexBuffer);
	static bool RenderDevice::UnloadVertexBuffer(VertexBuffer& vertexBuffer);
	static bool RenderDevice::UploadShader(Shader& shader);
	static bool RenderDevice::DeleteShader(Handle vertexShaderHandle, Handle fragmentShaderHandle, Handle programHandle);
	static bool RenderDevice::UploadRenderBuffer(RenderBuffer& renderBuffer);
	static bool RenderDevice::UnloadRenderBuffer(RenderBuffer& renderBuffer);
	static bool RenderDevice::UploadDirectTextureImage(DirectTextureImage& textureImage, UInt unit);
	static bool RenderDevice::UnloadDirectTextureImage(DirectTextureImage& textureImage);
		static bool Renderer::UploadBitmapTextureImage(BitmapTextureImage& textureImage, UInt unit, DeviceObject::LoadingHint loadingHint);
		static bool Renderer::UploadCubeMapTextureImage(CubeMapTextureImage& textureImage, UInt unit, DeviceObject::LoadingHint loadingHint);
		static bool Renderer::UnloadBitmapTextureImage(BitmapTextureImage& textureImage, DeviceObject::LoadingHint loadingHint);
		static bool Renderer::UnloadCubeMapTextureImage(CubeMapTextureImage& textureImage, DeviceObject::LoadingHint loadingHint);
		static bool Renderer::UploadVertexBuffer(VertexBuffer& vertexBuffer, DeviceObject::LoadingHint loadingHint);
		static bool Renderer::UnloadVertexBuffer(VertexBuffer& vertexBuffer, DeviceObject::LoadingHint loadingHint);
		static bool Renderer::UploadShader(Shader& shader, DeviceObject::LoadingHint loadingHint);
		static bool Renderer::DeleteShader(Handle vertexShaderHandle, Handle fragmentShaderHandle, Handle programHandle, const Char* shaderName, DeviceObject::LoadingHint loadingHint);
		static bool Renderer::UploadRenderBuffer(RenderBuffer& renderBuffer,

Courier V3.3.0

Release Info

- Release Date: November 2016

Visualization Changes

- Asynchronous Loading Changes
- Candera Transitions for Courier

Asynchronous Loading Changes

Asynchronous loading of views has been improved to allow more views to be loaded asynchronously. Due to this, the interface has been changed in the following way:

1. Courier::IViewHandler::SetCurrentLoadingViewScene and Courier::IViewHandler::GetCurrentLoadingViewScene are no longer used and were made deprecated. Courier::IViewHandler::SchedulePartialLoad can be used instead to set the views for asynchronous loading.
2. Two new values have been added to the protected Courier::ViewScene::LoadState enumeration type. These are SynchronousLoadState and AsynchronousLoadState and are further used for asynchronous loading handling inside Courier.
3. Courier::ViewScene2D::SetUploadNode2DTraverser and Courier::ViewScene3D::SetUploadNodeTraverser were made deprecated as new member methods (not static) were added instead. These are Courier::ViewScene2D::UploadNode2DTraverserSet and Courier::ViewScene3D::UploadNodeTraverserSet.

Candera Transitions for Courier

The Candera Transitions have been integrated into Courier. New Courier messages have been added for handling Candera Transitions. These are:

- Courier::CanderaTransitionRequestMsg / Courier::CanderaTransitionResponseMsg - Messages used for creating transition request fragments (Activate, Deactivate, Finish);

- Courier::CanderaTransitionControlFlowReqMsg / Courier::CanderaTransitionControlFlowResMsg - Messages used for indicating beginning or ending composing a request;
- Courier::CanderaTransitionIndMsg - Notifies start and finish events for Candera Transitions;
- Courier::CanderaTransitionFragmentIndMsg - Notifies start and finish events for Candera Transitions Fragments.

Also, two new methods have been added to Courier::IViewHandler and hence to Courier::ViewHandler: ExecuteCanderaTransitionReqAction and ExecuteCanderaTransitionControlFlowAction. These are further called by Courier::ViewFacade when Courier::CanderaTransitionRequestMsg and Courier::CanderaTransitionControlFlowReqMsg messages are sent.

FeatStd V3.3.0

FeatStd Events

Version 3.3.0 introduces Events. Event serves as a base class for user derived events that are dispatched via an EventSource to registered EventListeners.

FeatStd Util

Simple State Machine

New classes FeatStd::StateMachine and FeatStd::HistoryStateMachine have been added which allow the composition of a simple state machine.

FeatStd Monitor

Logging Memory Pools

FeatStd Monitor supports tracing of memory allocation behavior (alloc/free) now. It supports writing the trace into a file and load this file into the Analyzer afterwards.

Monitor: CMake flags

The new logging feature provides two CMake flags:

Name	Description	Possible values
MONITOR_MEMORYPOOL_BINARY_LOGGING_ENABLED	Activates the Memory Pool activity logging.	positive integer
MONITOR_MEMORYPOOL_BINARY_LOGGING_FILENAME	Location where the log-file will be generated. The extension has to be *.membin.	file path

---

Analyzer V3.3.0

Scene Analyzer

A filter functionality has been added to filter a logfile before it is loaded into the Analyzer.
This filter includes:

- Deselect recorder (Recorder-ID as well as recorder annotation name is supported)
- Timespan
- Hitcount of a recorder

MemoryPool-Analyzer Module

A new module for the Analyzer is available now. This module is used in conjunction with memory pools.
It is possible to trace the behavior of the memory pools (alloc/free) and load it into the Analyzer.
The Analyzer provides different statistics and analysis functionalities.
This includes:

- Memory Over Time Graph
- Snapshot information at selected timestamp
- Memory information listed in tabluar form
- Analysis tools for troubleshooting and typical memory issues

See also:

MemoryPool-Analyzer module documentation

SceneComposer V3.3.0

SceneComposer Usability Improvements

Start Screen

A "Start Up" panel was added (Mantis 7202 & Mantis 7366). This panel is opened first when the application is launched and allows the user to:

- see informations about the current platform
- see the list of Recent solutions and to allow loading them
- allow the creation of a new solution based on a template

In this start screen a welcome page is offered to the user, with multiple friendly functions.

See also:

- Startup Panel in SceneComposer User Manual

Scene Transition Editor

The new added feature of scene transition editor allows the definition of transitions on a per scene basis and test these transitions directly in SceneComposer's display preview. The user can use the editor to set the transitions to or from a scene just by dragging and dropping effects onto the scene list. Additionally, the transition framework was extended to support cut, fade, slide and scale transitions or combinations of these effects by default. Transitions are also integrated in the behavior system, which allows the user to add interactive controls to the application from the start. (Mantis 7355).

See also:

- Transitions in SceneComposer User Manual

Scripting Improvements

Since this version on, a new category is available under "Import" menu: "Import Scripts". This operation is possible due to the implementation of a script importer. After every validation of a given script, a specific information is displayed in the Output. In this way the user is aware about the fulfillment of the process. Any script can be saved in any state (with errors or not), but it can be validated only on the solution opening or when pressing F8 (Mantis 7267 & Mantis 7490).

See also:

- How to Import Resources in SceneComposer User Manual
- Scripting in SceneComposer User Manual

Camera Look-At Node

A new property was added on Camera: Look-at node. When this property is set the camera is automatically oriented towards the look-at-node. Up direction and look-at direction are not shown anymore in property grid. Look-at direction is discarded (Mantis 6819).

See also:

- Camera Manipulation in SceneComposer User Manual

Extended Control Functionality

Auto-expand: when the control is dragged and dropped, the control node will not be created. Instead, the content of the control will be directly inserted into the scene, without creating an intermediate group that replaces the control. Behaviors and widgets will still be expanded. A flag to mark controls as auto-expandable was provided (Mantis 7044).

Export SCL Files

The functionality of FTCCMd.exe was extended. Due to this, it is provided support for exporting SCL files from FTCCmd.exe (Mantis 7794).

Importers Textual Meta-Info

Since now on the importers support author and file meta-info. Only "Title", "Author", "Description" and "Filename" are added in the description field of the imported FBX Scene or imported Photoshop

scene (Mantis 7368).

Add to Animation Group

In the context menu of any animation a new option is available: "Add to Animation Group..." This allows the user to choose an "Animation Group" item and to add the animation to it (Mantis 7044).

Extended Control Functionality

Auto-expand: when the control is dragged and dropped, the control node will not be created. Instead, the content of the control will be directly inserted into the scene, without creating an intermediate group that replaces the control. Behaviors and widgets will still be expanded. A flag to mark controls as auto-expandable was provided (Mantis 7044).

At the same time the control functionality was extended by providing support to add default content to anchors (Mantis 7617).

Export SCL Files

The functionality of FTCCMd.exe was extended. Due to this, it is provided support for exporting SCL files from FTCCmd.exe (Mantis 7794).

Refresh Reference Files

If a SCL is used in a solution and the SCL is exported to the same location where the reference is used from, the modified SCL shall be automatically detected by the application and the user is notified through a dialog box. The user can choose if the application should be reloaded or not. In this way the modifications are imported. If the user chooses not to reload the solution, the SCL reference is marked as outdated (Mantis 7333).

See also:

- Reusable Solution Libraries in SceneComposer User Manual

Importers Textual Meta-Info

Since now on the importers support author and file meta-info. Only "Title", "Author", "Description" and "Filename" are added in the description field of the imported FBX Scene or imported Photoshop scene (Mantis 7368).

Timeslider Improvements

To make the work with the timeslider easier, since now on when double clicking a key frame in both "Animation Timeline" and "Animation Value", the marker will be instantly moved at this key (Mantis 7387). At the same time, the timeslider can be moved by using "Move marker to selected keyframe" option from the context menu of any keyframe available in "Animation Timeline" or "Animation Values" panels (Mantis 7388). Edit Value - edit keyframe popup dialog - is now a context menu item and is not triggered anymore via double click.

Note:

When a shortcut is modified in order for the changes to be visible in SceneComposer, the next file must be regenerated (deleted and automatically recreated):

```
%userprofile%\AppData\Roaming\FTC.Launcher\v3.2.2.2\ShortcutProfiles\Default Profile
```

See also:

- Timeline Editor and
- Animation Values - Keyframes in SceneComposer User Manual

Remove AutoSwap and AutoClear in 2D/3D from RenderTarget

AutoSwap and AutoClear options (on RenderTargets) were simplified and they became only SceneComposer features. Auto-Clearing was simplified and when enabled, only a property for choosing the clear color becomes available. These properties are not serialized into the asset - they will not be available into Player (Mantis 6831).

Interleaved Rendering of 2D and 3D Cameras

Rendering in SceneComposer is no longer done separately (2D and then 3D). Now, cameras are interlevead. This means that sequence numbers are global, and not category-specific (2D or 3D). A consequence of this is that Render Targets Panel / Camera Sequence Editor shows interleaved 2D and 3D cameras - it shows effective rendering order (Mantis 6828).

Saving Asset Location

Together with values as "Shader Compiler Type", "Float Number Format" and "Custom ID", the "Asset Location" value can be saved in a specific user defined "Asset Profile" (Mantis 7227).

See also:

- Save an Asset Profile In SceneComposer Manual

Minor Improvements

- **Control Integration:** The "composite" category was renamed "control". This renaming was done only in UI. The underlying structures (users will notice in SCML) are still named "Composite" (Mantis 7348).
- **Collapsed Categories:** In the "Properties" panel, both categories "General" and "Asset" are collapsed by default because, usually, such categories are not required (Mantis 4119).
- **Navigation Between Items:** Inside the "Generate Asset Library" dialog it is possible to navigate between items by using the "Tab" key. The Combobox border was changed to make it always visible when in focus (Mantis 7255).
- **New Label:** "Item Definition" label from "Add New Render Target" dialog has been changed to "Render Target Template" (Mantis 7244).
- **Default Panels:** When starting SceneComposer the following panels are hidden by default: Candera Logging, Render Order Bins, Appearance Editor, Shader Editor, Scripting Editor, SCML Editor (Mantis 7347).
- **Automatically Created:** The steps to created display and render target are removed from the new solution wizzard. These shall be automatically created (Mantis 7430).
- **Re-enabled Properties:** Initially Depth Clear and Stencil Clear were disabled for 3D cameras, now they are no longer disabled (Mantis 6831).
- **Set Pivot:** Center pivot option is now available for 2DNodes (Mantis 7650).
- **Plugins Info:** The plugins can be specified in the plugins.config.xml file (Mantis 7457).
- **Limited Editing:** Since now on, the appearance templates cannot be edited with drag and drop (Mantis 7349).
- **Type Ahead on Shader:** The "Type ahead" was added in the "Add New Shader Program" dialog. When the user types a letter, the first shader starting with that letter is highlighted (Mantis 7054).
- **Calculator Integration:** A calculator is available for all the properties of type int, float, and sbyte properties where the user has to enter a number (Mantis 5578).
- **Dragged Camera:** The cameras obtained as results after a search operation can be dragged and dropped into camera groups (Mantis 7046).
- **Open in File System:** An "Open in File System" context menu option was added on the top level solution in the "Solution Explorer" panel (Mantis 7237).

- **Create Templates:** An "Add Template" command was added for each category from Templates panel (Mantis 7051).
- **Add to Animation Group:** In the context menu of any existing animation, an "Add to Animation Group..." option is available (Mantis 7044).
- **Description for Nodes:** A new "Description" field is available on nodes (Mantis 7272).
- **New Checkbox:** An "Add Camera and Light" checkbox was added in the "Add New Scene..." dialog (Mantis 7239).
- **Shader Template:** When creating a new shader, a predefined template is offered to the user in the "Shader Editor"(Mantis 7241).
