Building Candera

Select Features

> CMake feature switches have been renamed for V3.x releases, see Change Log

Select Candera Features

Build settings for the Candera Engine are specified by the following CMake variables:

Name	Description	Possible values
CANDERA_2D_ENABLED	Enables Candera engine's 2D elements. If both 2D and 3D are enabled, the solution will contain both CanderaEngine2D and CanderaEngine3D subprojects.	ON/OFF
CANDERA_LAYOUT_ENABLED	Enables Candera 2D dynamic layout. If disabled, the whole code for dynamic layout is removed.	ON/OFF
CANDERA_LAYOUT_CLIPPING_ENABLED	Enables clipping of Candera 2D dynamic layout. If enabled, the render area of a render node is clipped to the layout area.	ON/OFF
CANDERA_3D_ENABLED	Enables Cander engine's 3D elements.	ON/OFF
CANDERA_VRAM_ALLOCATOR_CHECK_ENABLED		ON/OFF
CANDERA_VRAM_ALLOCATOR_STATISTICS_ENABLED		ON/OFF
CANDERA_RENDER_STATE_CACHING_ENABLED	Whether or not Candera shall cache render states. > For SceneComposer builds (on host), render state caching must be disabled. > For target builds, take care to enable this feature!	ON/OFF

CANDERA_DETAIL_STATISTICS_ENABLED	Whether or not Candera shall keep statistics about render state cache hits and misses. > This works only if CANDERA_RENDER_STATE_CACHING_ENABLED is enabled. > For target builds, disable because of the added overhead!	ON/OFF
CANDERA_GLOBALIZATION_ENABLED	Whether or not Candera Globalization library shall be built. > Candera Globalization is an optional module only available upon special request.	ON/OFF
CANDERA_ASSET_DECOMPRESSION_ENABLED	This option is required to decompress asset data if it was compressed with CGI Studio Asset Library Shaper previously. > Enabling this feature will automatically add ZLib to the project.	ON/OFF

CANDERA_TEXT_SHAPER	Allows the selection of a text shaper. A text shaper resolves certain text features like text direction, context specific substitutions and ligatures. Options: • **HarfBuzz (Default)(Optional)**: Unidirectional text. Unicode defined substitutions. Unicode defined ligature, GSUB and GPOS lookup. > HarfBuzz shaping is an optional module only available upon special request. Upon failure to detect the HarfBuzz source location, CMake will display "ComplexScripLib" as default textshaping option. • **ComplexScriptLib**: Bidirectional text. Arabic context substitutions. Forced Unicode defined ligatures. • **NoShaping**: Disables shaping. Text will be displayed left to right. No substitutions or ligatures are resolved.	NoShaping ComplexScriptLib Harfbuzz - *Default*
CANDERA_BIDIRECTIONAL_TEXT_ENABLED	Allows enabling of bidirectional text parsing. This is needed when combining text written left to right with text written right to left. If the text shaper supports bidirectional text this option is required when using multi-font text styles.	ON/OFF

CANDERA_TEXTENGINE_NATIVE_LINEHEIGHT_ENABLED	Whether or not the native line height metrics of the font engine is used. If disabled then 120% of EM size is applied as line height which is compatible with Adobe.	ON/OFF
CANDERA_ASSETLOADER_MEMORYPOOL_ENABLED	Enables the use of the Memory Pool feature in Candera AssetLoader. Requires the MemoryPool feature. > Only available if FEATSTD_MEMORYPOOL_ENABLED is ON.	ON/OFF
CANDERA_TEXTENGINE_MEMORYPOOL_ENABLED	Enables the use of the Memory Pool feature in Candera TextEngine. Requires the MemoryPool feature. > Only available if FEATSTD_MEMORYPOOL_ENABLED is ON.	ON/OFF
CANDERA_ASSET_LIBRARYHEADER_CACHE_ENABLED	Optional feature which will cache all Asset-Library-Header that have been loaded and cached upon Asset initialization. If dynamic Asset-Partitions are being used, caching will occur when a partition is added, and will continue until it is removed. > The cache will use the standard Asset Cache memory.	ON/OFF

Select FeatStd Features

For information about FeastStd features, please refer to FeatStd > FeatStd Features.

3rd Party Software: FreeType2 Configuration

3rd party software FreeType2 is used by the Candera Engine and can be configured in CMake like this:

Name	Description	Possible values
FREETYPE2_CUSTOMIZE	Enable or disable FreeType2 customization.	ON/OFF
FREETYPE2_FORCE_DEFAULT_CONFIG	Set all Freetype2 configuration flags such that the outcome is equivalent to using the "freetype/include/config/" files.	ON/OFF
FREETYPE2_FORCE_CANDERA_CONFIG	Set all Freetype2 configuration flags such that the outcome is equivalent to using "Candera/TextEngine/FreeType/FtCda *.h" files.	ON/OFF
FREETYPE2_FORCE_EMPTY_CONFIG	Clears all Freetype2 configuration flags.	ON/OFF
FREETYPE2_FORCE_LIGHT_CONFIG	Set all Freetype2 configuration flags to a minimal setting that provides support for using TrueType fonts and anti-aliased rendering with a minimal code size.	ON/OFF

> The "light" configuration (FREETYPE2_FORCE_LIGHT_CONFIG) does not support rendering of Opentype fonts. However, a rather minimal configuration with Opentype support is possible by enabling the following flags on top of the "light" configuration:
> FREETYPE2_MODLOAD_DRIVER_CFF
> FREETYPE2_MODLOAD_MODULE_PSAUX
> FREETYPE2_MODLOAD_MODULE_PSHINTER
> FREETYPE2_MODLOAD_MODULE_PSNAMES
> FREETYPE2_OPT_FT_CONFIG_OPTION_POSTSCRIPT_NAMES
> FREETYPE2_OPT_TT_CONFIG_OPTION_POSTSCRIPT_NAMES The exact features are font specific. The fastest way to evaluate the minimal module set requirements for a font is to get in contact with the font designer.

Candera Device Packages

Supported Device Packages

Candera provides device implementations for following device types:

Name	Description	Possible values
Other proprietary devices	Can be provided upon special request	ON/OFF

Device-Specific CMake Settings

Name	Description	Possible values
CGIDEVICE_NAME	The name of the device package	Each device package following the Candera device CMake rules provided in the folder <CANDERA_SRC>/CanderaPlatform/Device
CGIDEVICE_TARGET_BUILD	Enable or disable building for target	ON/OFF
CGISTUDIO_PATH_3RD_PARTY	Path to the 3rd party repository location	A string value containing the system file location of 3rd party software, auto-detected by finding possible locations of the name "[ic01_]cgi_studio_3psw"
CGIDEVICE_TARGET_ <CGIDEVICE_NAME> _DRIVER_PATH_INCLUDE	If you don't use the default driver path, the path to the driver include directory can be changed with this setting.	Path to driver include directory (by default set to include directory in 3rd party repository)
CGIDEVICE_TARGET_ <CGIDEVICE_NAME> _DRIVER_PATH_LIB	If you don't use the default driver path, the path to the driver library directory can be changed with this setting.	Path to driver library directory (by default set to library directory in 3rd party repository)

To build a certain device package selected it from the drop-down list of the CMake property **"CGIDEVICE_NAME"**:

For each device package, either building OpenGL based host simulation or target binaries can be selected via the Boolean CMake property "***CGIDEVICE_TARGET_BUILD***".

The following table illustrates configurations available for the supported device package:

CGIDEVICE_NAME	CGIDEVICE_TARGET_BUILD	PUB_CGI_CPU	PUB_CGI_OS	Default driver path following <CGISTUDIO_PATH_3RD_PARTY> /[src/lib]/

The CMake properties "PUB_CGI_CPU", "PUB_CGI_OS" usually cannot be configured via the CMake user interface, but are autonomously detected by
the build system. The same applies for the variable "PUB_CGI_COMPILER".

If you don't use the CGI Studio 3rd party software folder structure, the driver paths can be specified in the variables

- CGIDEVICE_TARGET_<CGIDEVICE_NAME>_DRIVER_PATH_INCLUDE and
- CGIDEVICE_TARGET_<CGIDEVICE_NAME >_DRIVER_PATH_LIB

Depending on these properties, either MS Visual Studio solutions or Unix makefiles will be generated into the specified build
directory.

Integrate Candera into Custom Build Environments

Integrate Candera into Custom Build Environments

For building Candera as part of the Player demo application, please refer to Building the Player.

If a custom application should include Candera libraries, this chapter explains the CGI-Studio library build process. To simplify integration of Candera source code into a custom build environment references to Candera compile switches are given below.

Build Cgi-Studio Candera libraries with CMake

Use CMake to create a Visual Studio solution for host or any other build environment (like make files for targets) and build Candera libraries.

1. Start CMake
2. Point "Where is the source code" to *<cgi-studio-root>/cgi_studio_candera/src*
3. Point "Where to build the binaries" to *<cgi-studio-root>/cgi_studio_candera/build*. Note that the name of the build directory can be freely chosen but it is required to create it on the same directory level as "src". Avoid blanks in the path
4. Click "Configure" and select the proper generator for the desired build environment.
   > As an example, for host builds using Visual Studio 2010, select "Visual Studio 10".
   > For target builds, please refer to the given target-specific setup documentation part of CGI Studio releases.
5. Confirm the dialog with OK. All Candera CMake settings which can be configured will appear in the main window, having red lines for all settings to be confirmed.
6. Check and modify the Candera CMake properties and click "Configure" again
7. As long as red lines are visible, those lines must be confirmed by clicking "Configure" again.
8. If no more red lines are visible, the "Generate" button is enabled. Click "Generate" to create the Visual Studio solution (or the make files for make-based build environments).

Build Candera with the CMake-generated build environment (Visual Studio solution for host, e.g.).

> Candera libraries will be available in the separate folders in the given build directory.
> To collect all of the libraries in a specific folder, an output path can be specified during CMake configuration:
> - Tick the "Advanced" checkbox in the CMake GUI.
> - After pressing "Configure", the CMake property "CANDERA_ENABLE_LIBRARY_OUTPUT" will become visible. Tick this and press "Configure" again.
> - Specify the library output directory in the "LIBRARY_OUTPUT_PATH" CMake variable.

Integrate Candera Source Code (Compile Switches)

When Candera source code is built using Candera CMake configuration, all relevant compile switches (defines) are already set by CMake. The template CanderaConfig.h.template (refer to *<cgi_studio_root>/cgi_studio_candera/cmake/templates*) is used by CMake to generate the file CanderaConfig.h in the build directory.

If Candera source code is built using a custom build environment, learn about all configuration defines available for Candera in the Candera API Documentation:

- CANDERA_API_DOCUMENTATION/CANDERA GRAPHIC ENGINE/CANDERA CONFIGURATION