C/C++ build configuration

- C SDK Instructions
  - C SDK build configuration
- C++ SDK Instructions
  - C++ SDK build configuration

This page describes different configuration options available for C and C++ endpoint SDKs.

CMake is used as a build system in both C and C++ SDK, so basic understanding of how CMake works may be useful but not required.

C SDK Instructions

To build C SDK, proceed as follows:

1. Generate the C endpoint SDK in Admin UI.
2. Download and untar the Kaa C SDK archive.
3. Run the following commands:

C SDK build configuration

To configure the C endpoint SDK build, you can optionally specify the following parameters for CMake:

Parameter	Description	Values	Usage example
CMAKE_INSTALL_PREFIX	Directory for Kaa to be installed in.	Accepted: '/path/to/some/directory' Default: '/usr/local'	cmake -DCMAKE_INSTALL_PREFIX='/home/username/kaa'
CMAKE_BUILD_TYPE	Build type	This is built-in CMake variable. Refer to this page - https://cmake.org/cmake/help/v3.0/variable/CMAKE_BUILD_TYPE.html	cmake -DCMAKE_BUILD_TYPE=Debug
KAA_MAX_LOG_LEVEL	Maximum log level used by the SDK	Accepted: • 0 - NONE (no logs) • 1 - FATAL • 2 - ERROR • 3 - WARN • 4 - INFO • 5 - DEBUG • 6 - TRACE Default: • 6 - If CMAKE_BUILD_TYPE is Debug • 4 otherwise	cmake -DKAA_MAX_LOG_LEVEL=4
KAA_WITHOUT_<MODULE>	Kaa module to be omitted during the build	Accepted: • EVENTS • LOGGING • CONFIGURATION • NOTIFICATION Default: All modules are present in the build	cmake -DKAA_WITHOUT_EVENTS=1

KAA_PLATFORM	SDK target platform **NOTE:** Before running the **cmake** command with the KAA_PLATFORM parameter for a platform other than supported, do the following: 1. Create a folder in $KAA_HOME/listfiles/platform/ and name it using [a-zA-z_-] symbols. You will be able to use the folder name as a value for the KAA_PLATFORM parameter. 2. Put the CMakeLists.txt file into the created folder. This file may contain specific compilation /linking flags, platform-dependent source files, third-party library dependencies, that is all information necessary for building the C endpoint SDK for this platform. 3. Optionally, specify the following parameters in the CMakeLists.txt file. • KAA_INCLUDE_PATHS - full path(s) to folder(s) containing additional header files • KAA_SOURCE_FILES - full path(s) to additional source files • KAA_THIRDPARTY_LIBRARIES - third-party libraries (the name of the library, for example, ssl, crypto)	Accepted: x86-64 ios esp8266 cc32xx Default: x86-64	cmake -DKAA_PLATFORM=x86-64 In the CMakeLists.txt file: set (KAA_INCLUDE_PATHS ${KAA_INCLUDE_PATHS} path_to_folder_1_with_header_files path_to_folder_2_with_header_files) set (KAA_SOURCE_FILES ${KAA_SOURCE_FILES} path_to_source_file_1 path_to_source_file_2) set (KAA_THIRDPARTY_LIBRARIES ${KAA_THIRDPARTY_LIBRARIES} some_library_1 some_library_2)

The following example illustrates the build procedure for the debug build with the INFO log level and disabled EVENTS feature (on Linux system):

```
mkdir build
cd build
cmake -DCMAKE_BUILD_TYPE=Debug -DKAA_MAX_LOG_LEVEL=4 -DKAA_WITHOUT_EVENTS=1 ..
make
make install
```

C++ SDK Instructions

To build the C++ endpoint SDK, proceed as follows:

1. Generate the C++ endpoint SDK in Admin UI.
2. Download and untar the Kaa C++ SDK archive.
3. Run the following commands:

C++ SDK build configuration

To configure the C++ endpoint SDK build, you can optionally specify the following parameters for CMake.

Parameter	Description	Values	Usage example

CMAKE_INSTALL_PREFIX	Directory for Kaa to be installed in	Accepted: '/path/to/some/directory' Default: '/usr/local'	cmake -DCMAKE_INSTALL_PREFIX='/home/username/kaa'
CMAKE_BUILD_TYPE	Build type	This is built-in CMake variable. Refer to this page - https://cmake.org/cmake/help/v3.0/variable/CMAKE_BUILD_TYPE.html	cmake -DCMAKE_BUILD_TYPE=Debug
KAA_MAX_LOG_LEVEL	Maximum log level used by the SDK	Accepted: • 0 - NONE (no logs) • 1 - FATAL • 2 - ERROR • 3 - WARN • 4 - INFO • 5 - DEBUG • 6 - TRACE Default: • 6 - If CMAKE_BUILD_TYPE is Debug • 4 otherwise	cmake -DKAA_MAX_LOG_LEVEL=4
KAA_WITHOUT_<MODULE>	Kaa module to be omitted during the build	Accepted: • EVENTS • LOGGING • CONFIGURATION • NOTIFICATIONS • OPERATION_TCP_CHANNEL • OPERATION_LONG_POLL_CHANNEL • OPERATION_HTTP_CHANNEL • BOOTSTRAP_HTTP_CHANNEL • CONNECTIVITY_CHECKER Default: All modules are present in the build	cmake -DKAA_WITHOUT_EVENTS=1
KAA_WITH_SQLITE_LOG_STORAGE	Point the Kaa LOGGING module to use the SQLite persistent log storage. **NOTE:** Works only if KAA_WITHOUT_LOGGING=0.	Accepted: • 0 - disable SQLite log storage • 1 - enable SQLite log storage Default: 0	cmake -DKAA_WITH_SQLITE_LOG_STORAGE=1

The following example illustrates the build procedure for the debug build, with the INFO log level and disabled EVENTS feature and specified path to the folder Kaa will be installed in (on Linux system):

```
mkdir build
cd build
cmake -DCMAKE_INSTALL_PREFIX='/home/username/kaa' -DCMAKE_BUILD_TYPE=Debug -DKAA_MAX_LOG_LEVEL=4 -
DKAA_WITHOUT_EVENTS=1 ..
make
make install
```