Intel® RealSense™ SDK 2.0 Github

User Guide

May 2018

Revision 002

Document Number: 337595-002

You may not use or facilitate the use of this document in connection with any infringement or other legal analysis concerning Intel products described herein. You agree to grant Intel a non-exclusive, royalty-free license to any patent claim thereafter drafted which includes subject matter disclosed herein.

No license (express or implied, by estoppel or otherwise) to any intellectual property rights is granted by this document.

Intel technologies' features and benefits depend on system configuration and may require enabled hardware, software or service activation. Performance varies depending on system configuration. No computer system can be absolutely secure. Check with the system manufacturer or retailer or learn more at intel.com.

Intel technologies may require enabled hardware, specific software, or services activation. Check with the system manufacturer or retailer.

The products described may contain design defects or errors known as errata which may cause the product to deviate from published specifications. Current characterized errata are available on request.

Intel disclaims all express and implied warranties, including without limitation, the implied warranties of merchantability, fitness for a particular purpose, and non-infringement, as well as any warranty arising from course of performance, course of dealing, or usage in trade.

All information provided here is subject to change without notice. Contact the Intel representative to obtain the latest Intel product specifications and roadmaps.

Copies of documents which have an order number and are referenced in this document may be obtained by calling 1-800-548-4725 or visit www.intel.com/design/literature.htm.

Intel, RealSense and the Intel logo are trademarks of Intel Corporation in the U.S. and/or other countries.

*Other names and brands may be claimed as the property of others.

§ §

Contents

§ §

Revision History

Document Number	Revision Number	Description	Revision Date
337595	001	• Initial release	May 2018
	002	• Added Mac OS/Android installation guide	May 2018

§ §

1 Introduction

1.1 Purpose and Scope of this Document

This document guides user how to navigate through the Github web page and install the librealsense pre-built package or compile from source code.

§ §

2 LibRealSense SDK 2.0 Overview

2.1 Intended Users

LibRealSense SDK 2.0 is designed for developers.

2.2 Features

- Cross platform API for Intel® RealSense™ D400 series and SR300 cameras
- Tool supports Windows* 10, MAC OS and Linux*

2.3 Description

LibRealSense SDK 2.0 is a cross platform library designed for end users/developers to implement custom applications using the RealSense™ SR300 and D400 series cameras. The SDK allows depth and color streaming, and provides intrinsic and extrinsic calibration information. The library also offers synthetic streams (pointcloud, depth aligned to color and vise-versa), and a built-in support for record and playback of streaming sessions.

§ §

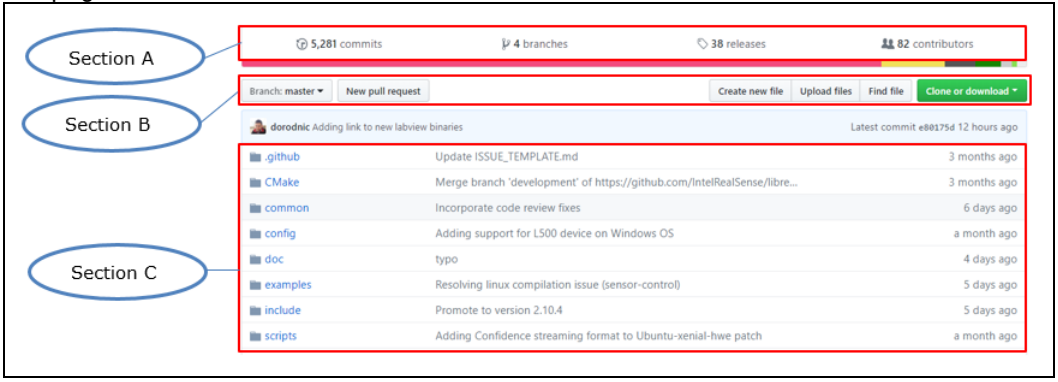
3 GitHub

3.1 Navigating GitHub Content

The section guides developers how to navigate through the repository:

3.1.1 Code

The tab contains various code related information/feature, such as clone or download the code, change to different branch, get the public release etc. Below will break down the page into three sections for more detailed information:



3.1.1.1 Section A

- Commits
  - List detailed information about each commit to the branch
- Branches
  - Shows the number of available branches in the repository
- Releases
  - List the available public releases
- Contributors
  - Provide statistics of contribution from developers

3.1.1.2 Section B

- Branches
  - Drop box, select to bring up the up to date files within specific branch
- New pull request
  - Button, creates a new commit request to repository owner to review then merge into repository. The feature requires developers to fork the repository to their own repository, complete the changes within local repository then submit the pull request. For more detailed information refer to https://github.com/IntelRealSense/librealsense/blob/master/CONTRIBUTING.md
- Create/Upload files
  - The feature is not available, for developers that want to create or upload new file, follow the pull request guidance
- Clone or download
  - Drop box, copy the URL to clone from local git tool, or download the packed source code as zip file directly

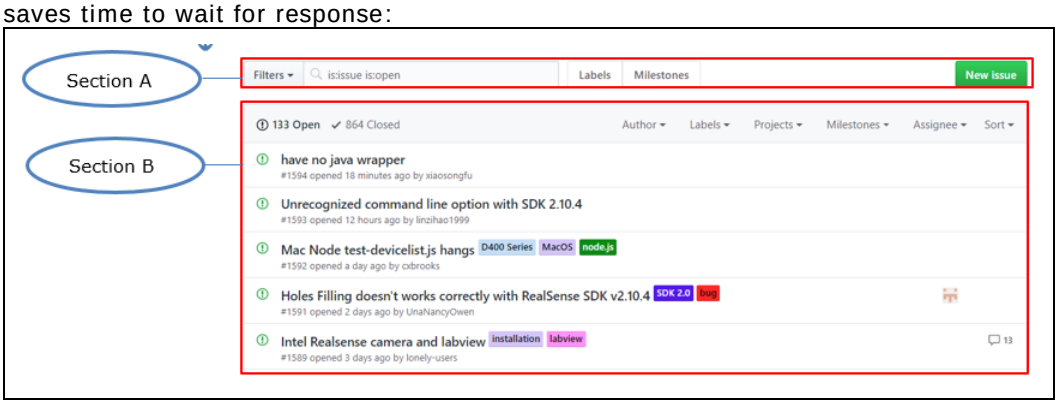
3.1.1.3 Section C

Viewing the architecture of each folder or code:

- Folder
  - Click to view the file architecture of the folder
- Source code
  - Click to invoke github native viewer to view the code

3.1.2 Issues

The tab contains issues from developers, the page provides easy to search and filter tools. Developer can search if any existing similar issue before create new one, which saves time to wait for response:



3.1.2.1 Section A

- Filters
  - Change the search scope, such as searching with the own issues, any issue assigned to/mentioning you etc.
- Search field
  - Type any keyword for searching
- Labels
  - List down available labels, click the label will redirect to page with all issues with the label
- Milestones
  - List down available milestones, click the milestone will redirect to page with all issues related to the milestone
- New issue
  - Creates a new issue, following the predefined format to describe the issue as detail as possible, such as test environment, FW/SDK version, camera model, how to reproduce the issue etc.

3.1.2.2 Section B

- Open/Closed
  - List open or closed issues
- Advanced filters
  - List issues with specific filter item selected, such as issues from specific author, assigned to specific developer or sorted with date or popularity etc

3.1.3 Pull Request

The tab contains pull requests from developers, the page provides easy to search and filter tools. Developer can search if any existing similar pull request before create new one, which saves time to wait for response:

3.1.3.1 Section A

- Filters
  - Change the search scope, such as searching with the own pull request, any pull request assigned to/mentioning you etc.
- Search field
  - Type any keyword for searching
- Labels
  - List of available labels, clicking on the label will redirect the user to a page with all the pull requests with the selected label
- Milestones
  - List of available milestones, clicking the milestone will redirect the user to a page with all pull request related to the milestone
- New pull request
  - Create new pull request, since creating new pull request requires to fork the repository first, the function will not work directly inside the page. For creating pull request, refer to https://github.com/IntelRealSense/librealsense/blob/master/CONTRIBUTING.md

3.1.3.2 Section B

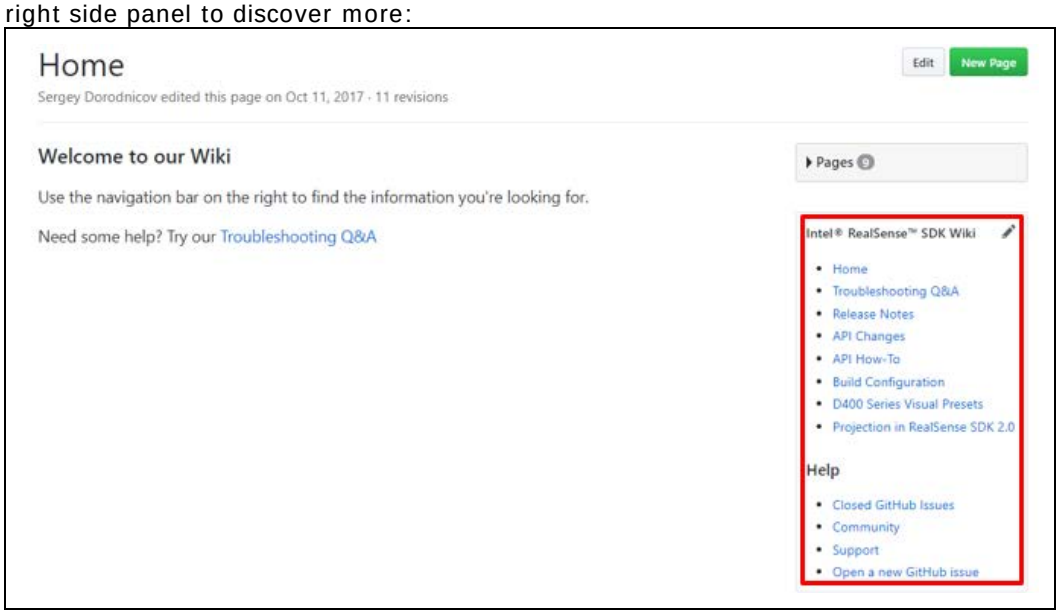
- Open/Closed
  - List open or closed pull request
- Advanced filters
  - List pull request with specific filter item selected, such as issues from specific author, assigned to specific developer or sorted with date or popularity etc.

3.1.4 Project

Listing down all available projects inside the repository, currently there is only one project (Intel® RealSense™ SDK) inside the librealsense repository.

3.1.5 Wiki

The page list down many useful information related to librealsense, click links inside right side panel to discover more:



3.1.6 Insights

The page contains statistics from different point of view, such as contributors, commits and forks etc.

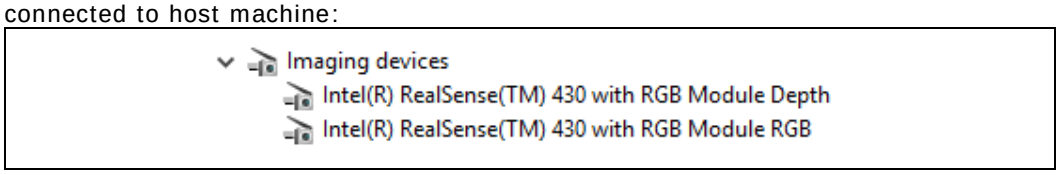
§ §

4 Install/Build LibRS

4.1 Install Pre-Built LibRS

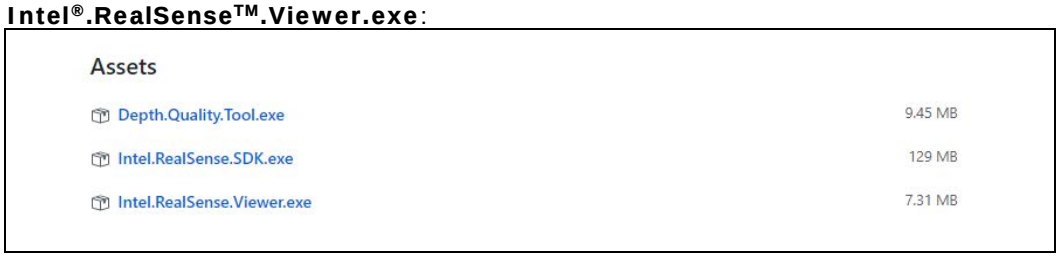
4.1.1 Windows*

Intel® RealSense™ SDK 2.0 provides tools and binaries for the Windows* platform using GitHub Releases. After plugging the camera into a USB3 port, you should be able to refer the newly connected device in the Device Manager, the device name should be various, depend on which device is connected. The picture below indicating D430 is connected to host machine:

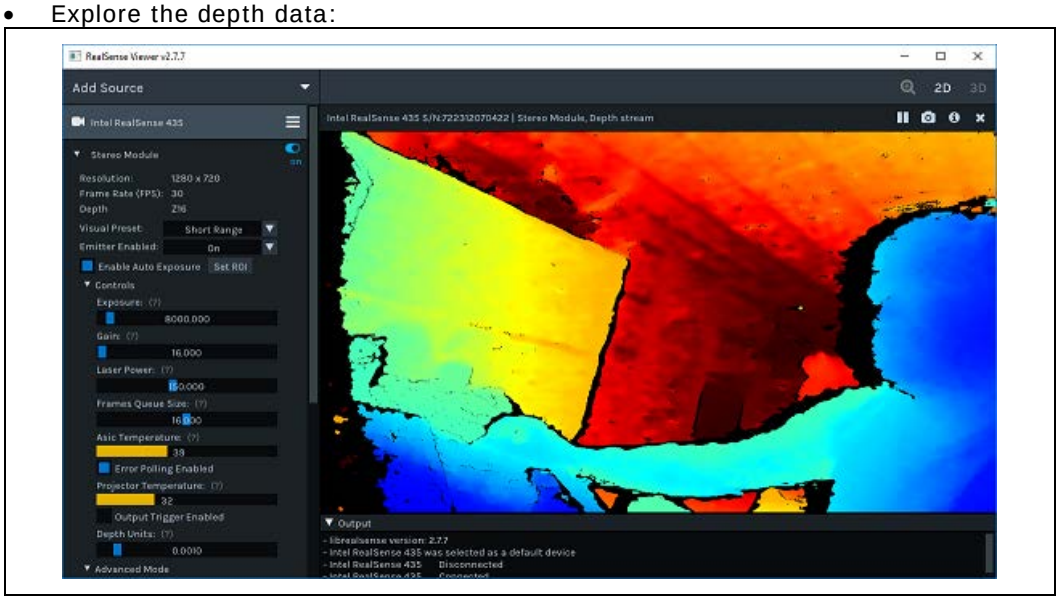


4.1.1.1 Intel® RealSense™ Viewer

Go to the latest stable release, navigate to the **Assets** section, download and run **Intel®.RealSense™.Viewer.exe**:

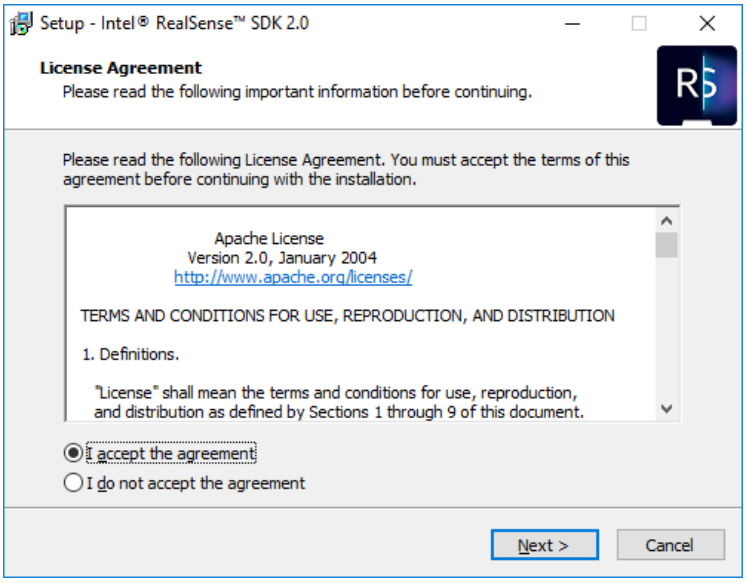


- Explore the depth data:

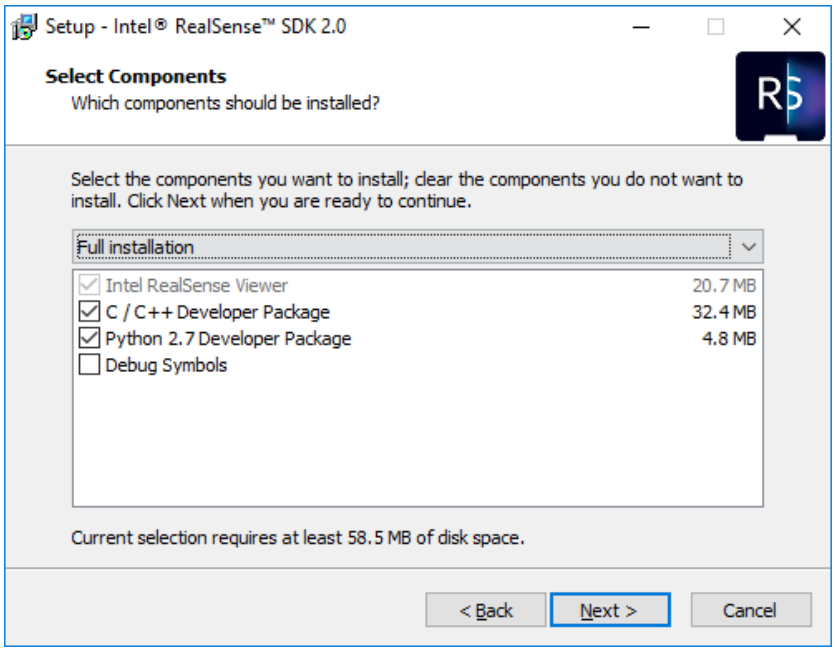


4.1.1.2 Installing the SDK

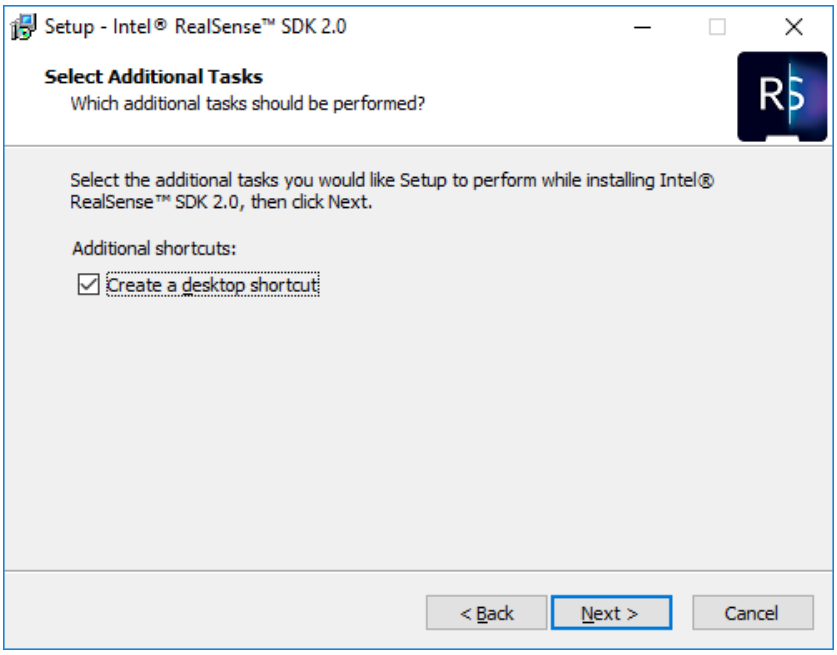
- Go to the latest stable release, navigate to the **Assets** section, download and run **Intel®.RealSense™.SDK.exe**
- Click through several simple steps of the installer:
  1. Intel® RealSense™ SDK 2.0 is distributed under the Apache 2.0 permissive open-source license:



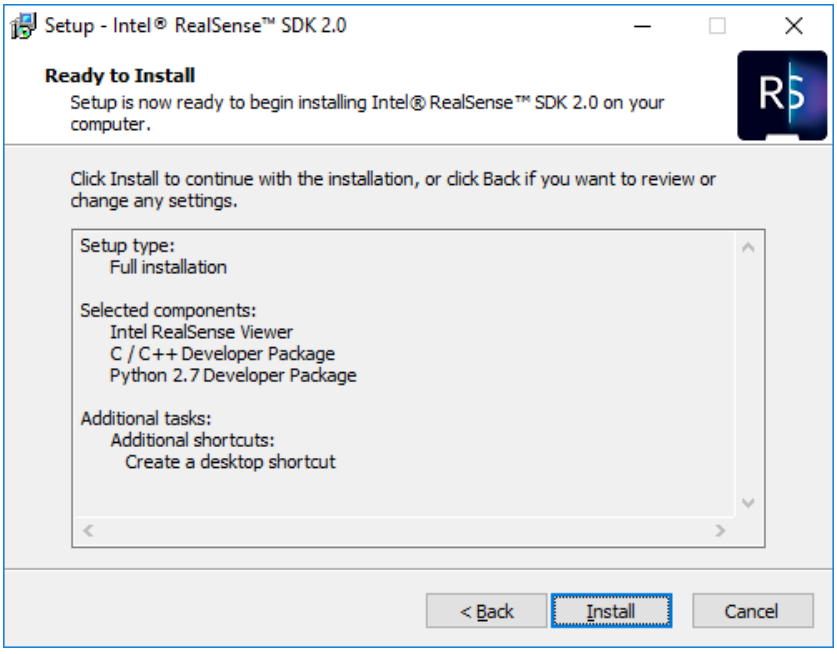
  2. The SDK includes the RealSense™ Viewer, as well as development packages for various programming languages:



3. Approve adding two shortcuts to the desktop:

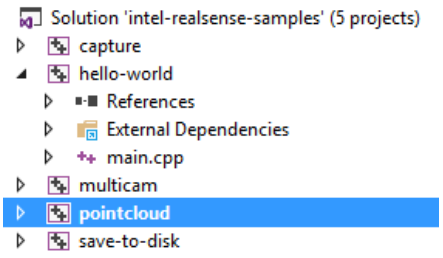


4. Review before installing:



5. Open the **Intel® RealSense™** Samples solution:

6. Press **F5** to compile and run the demos:



4.1.2 Linux*

4.1.2.1 Supported Distribution

Intel® RealSense™ SDK 2.0 provides installation packages in dpkg format for Ubuntu* 16 LTS*.

Note: The Realsense™ DKMS kernel drivers package (librealsense2-dkms) supports Ubuntu* LTS kernels 4.4, 4.10 and 4.13.

4.1.2.2 Install Steps

- Add Intel server to the list of repositories:

  `echo 'deb http://realsense-hw-public.s3.amazonaws.com/Debian/apt-repo xenial main' | sudo tee /etc/apt/sources.list.d/realsense-public.list`

  It is recommended to backup `/etc/apt/sources.list.d/realsense-public.list` file in case of an upgrade

- Register the server's public key:

  `sudo apt-key adv --keyserver keys.gnupg.net --recv-key 6F3EFCDE`

- Refresh the list of repositories and packages available:

  `sudo apt-get update`

- In order to run demos install:

  `sudo apt-get install librealsense2-dkms`

  `sudo apt-get install librealsense2-utils`

The above two lines will deploy librealsense2 udev rules, kernel drivers, runtime library and executable demos and tools. Reconnect the Intel® RealSense™ depth camera and run: **realsense-viewer**

- Developers shall install additional packages:

  `sudo apt-get install librealsense2-dev`

  `sudo apt-get install librealsense2-dbg`

  With **dev** package installed, compile an application with librealsense `using g++ -std=c++11 filename.cpp -lrealsense2` or an IDE of the choice

- Verify that the kernel is updated:

  `modinfo uvcvideo | grep "version:"` should include **realsense™** string

4.1.2.3 Uninstall Steps

Important: Removing Debian package is allowed only when no other installed packages directly refer to it. For example removing **librealsense2-udev-rules** requires **librealsense2** to be removed first.

- Remove a single package with:

  `sudo apt-get purge <package-name>`

- Remove all RealSense™ SDK-related packages with:

  `dpkg -l | grep "realsense" | cut -d " " -f 3 | xargs sudo dpkg --purge`

4.1.2.4 Package Details

The packages and their respective content are listed below:

Name	Content	Depends on
librealsense2-udev-rules	Configures RealSense device permissions on kernel level	-
librealsense2-dkms	DKMS package for Depth cameras-specific kernel extensions	librealsense2-udev-rules
librealsense2	RealSense™ SDK runtime (.so) and configuration files	librealsense2-udev-rules
librealsense2-utils	Demos and tools available as a part of RealSense™ SDK	librealsense2
librealsense2-dev	Header files and symbolic link for developers	librealsense2
librealsense2-dbg	Debug symbols for developers	librealsense2

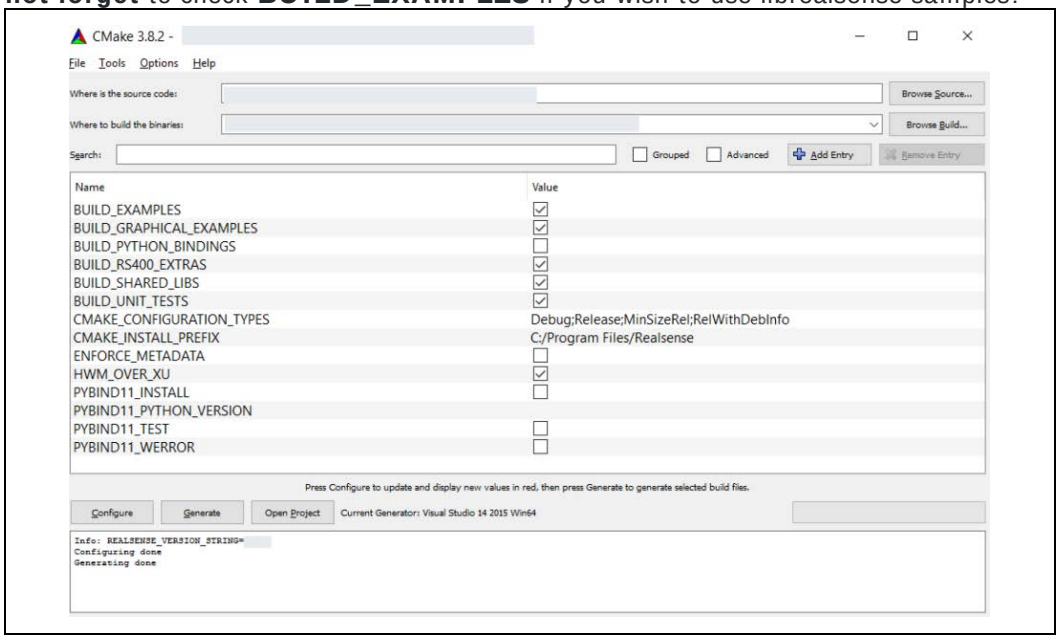
Note: The packages include binaries and configuration files only. Use the github repository to obtain the source code.

4.2 Compile LibRS Source Code

4.2.1 Windows* 8.1 and Windows* 10 Installation

Note: Due to the USB 3.0 translation layer between native hardware and virtual machine, the librealsense team does not recommend or support installation in a VM.

Librealsense shall be built on Windows* using CMake and Visual Studio 2015 / 2017: (MSVC2013 and older are not fully compatible with the C++11 feature-set). Also **do not forget** to check **BUILD_EXAMPLES** if you wish to use librealsense samples:



4.2.1.1 Windows* 8.1

When working on Windows* 8.1, make sure you have KB3075872 and KB2919355 installed. These patches are addressing issues specific to 8.1 video drivers that were later resolved in Windows* 10.

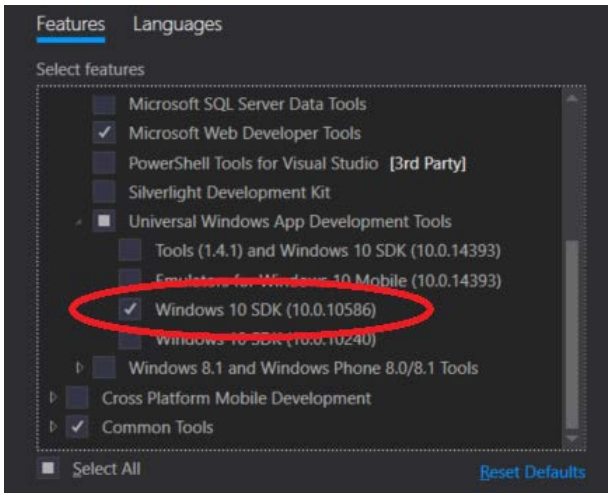
4.2.1.2 Enabling Metadata on Windows*

Metadata attributes is an advanced capability provided by librealsense. Read more on this feature in link.

Follow the instructions to enable metadata generation:

- Prerequisites:
  - Windows* 10 with administrator login
  - WinSDK ver 10 (10.0.15063) or later
- Installation:
  - Verify OS version
  - Run **winver** command from desktop/terminal - "Version 1607" or later is expected:

  - Install WinSDK ver10
  - Navigate to "Control Panel" -> "Programs and Features"
  - Double-click on "Microsoft* Visual Studio" and select "Modify"
  - Check that SDK version 10.0.10586 or later is present, install if needed:



- Update Registry:
  - Windows* OS requires a dedicated registry entry to be present for each unique video device in order to provide metadata:
    - **Use automation script:**
      - Launch Windows* **Powershell** tool as Admin and navigate to the script directory
      - Run the following:

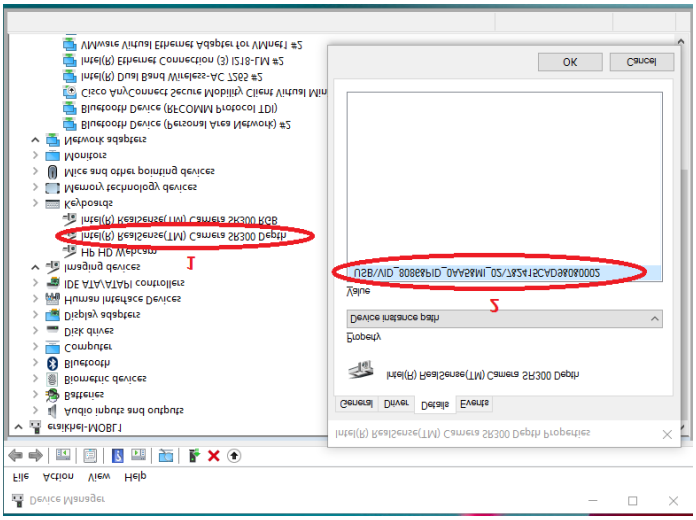
        `PS > .\realsense_metadata_win10.ps1 <optional:operation>`

        The supported parameters are
        - `-op install` adds registry keys for connected Intel® Realsense™ devices
        - `-op install_all` adds keys for all RealSense™ devices logged in the registry
        - `-op remove` removes registry keys for connected Intel® Realsense™ devices
        - `-op remove_all` removes the keys for all RealSense™ devices logged in the registry Running the script without arguments will is similar to -op install operation.
      - If you receive … cannot be loaded because running scripts is disabled on this system message, run:

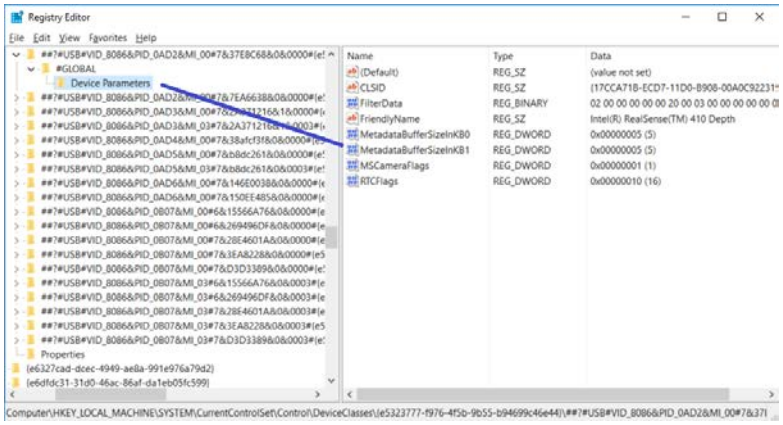
        ```
        `PS > Set-ExecutionPolicy RemoteSigned` , answer `Y` and
        then rerun the script

        **Important** - The registry keys are device-unique.
        Therefore, __*the script must be executed each time a new
        RealSense device is attached*__ to the PC.
        ```
    - **Modifying the registry manually:**

      In case the script cannot be executed due to permissions, or other Host-related issue follow the instructions to update the registry manually:
      - Connect Intel® RealSense™ device to the host
      - Navigate to "Control Panel" -> "Device Manager"
      - Browse for Intel® RealSense™ devices
      - Select the first device from the list, e.g. Intel® RealSense™ Camera SR300 Depth (Step 1)
      - Find device's path (Step 2) and the additional interfaces (Step 3)



- Modifying the Windows* Registry:
  - For each interface found (Steps 2 and 3) perform
    - Using Registry Editing tool such as "regedit" navigate to HKLM\SYSTEM\CurrentControlSet\Control\DeviceClasses\{e5323777-f976-4f5b-9b55-b94699c46e44} branch
    - Browse into the subdirectory with the name identical to the **Device instance path** obtained from the previous step
      - Expand the entry into **#GLOBAL -> Device Parameters**
      - Add **DWORD 32bit** value named **MetadataBufferSizeInKB0** with value 5
      - Add an additional DWORD 32bit value named MetadataBufferSizeInKB1 with value 5 for RS400 device zero interface **##?###USB#VID_8086andPID...**
        ****MI_00**..**



- Repeat the previous step for HKLM\SYSTEM\CurrentControlSet\Control\DeviceClasses\{65E8773D-8F56-11D0-A3B9-00A0C9223196} branch

– Repeat the procedure for all the additional RealSense™ devices (e.g. Intel® RealSense™ Camera SR300 RGB)

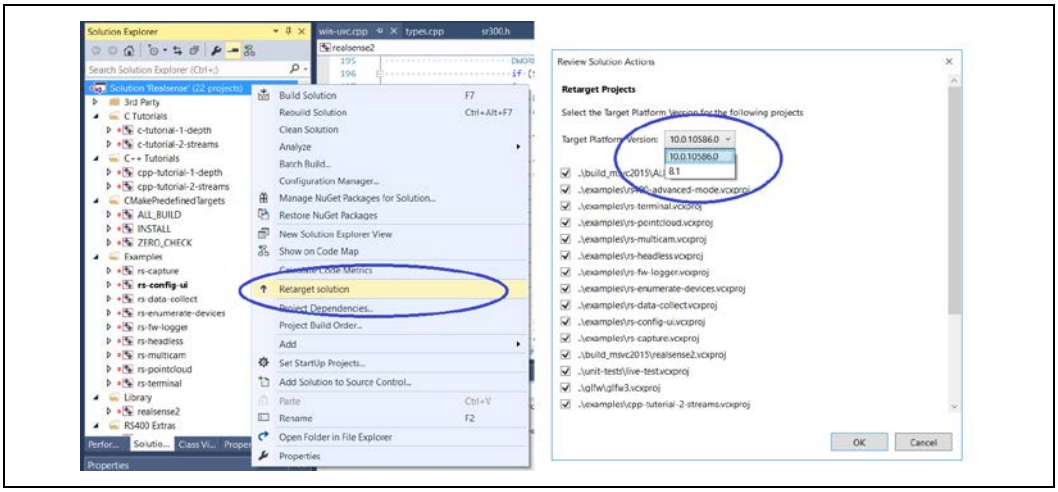
4.2.1.3 Compiling Librealsense with Metadata Support

During compilation the library will automatically detect and enable/disable metadata functionality according to the WinSDK version selected. In case the installed WinSDK does not expose metadata the user will be notified that the build does not include metadata generation:

Librealsense notification: Featuring UVC Metadata requires WinSDK 10.0.10586.0 toolset. The library will be compiled without the metadata support!

In order to ensure compilation with metadata support check the **ENFORCE_METADATA** in the Cmake configuration stage. When set, the compiler will check the target WinSDK version, and abort the build if it is not compatible with metadata requirements.

Note: In case of multiple WinSDK version installed, CMake automatically selects the latest version of SDK. In some cases, however, the automatic selection may fail. Then you need to manually retarget the solution for the proper WinSDK version:



4.2.2 Linux*

Note: Due to the USB 3.0 translation layer between native hardware and virtual machine, the librealsense team does not support installation in a VM. If you choose to try it, we recommend using VMware Workstation Player, and not Oracle VirtualBox for proper emulation of the USB3 controller.

Ensure to work with the supported Kernel versions listed here and verify that the kernel is updated properly according to the instructions.

4.2.2.1 Ubuntu* Build Dependencies

Make sure to have git and cmake installed as these are required for librelasense build:

```
sudo apt-get install git cmake3
```

Several scripts below invoke wget, git, add-apt-repository which may be blocked by router settings or a firewall. Infrequently, apt-get mirrors or repositories may also timeout. For librealsense users behind an enterprise firewall, configuring the system-wide Ubuntu proxy generally resolves most timeout issues.

4.2.2.2 Prerequisites

Note: Running RealSense™ Depth Cameras on Linux* requires patching and inserting modified kernel drivers. Some OEM/Vendors choose to lock the kernel for modifications. Unlocking this capability may requires to modify BIOS settings

- Make Ubuntu* Up-to-date:
  1. Update Ubuntu* distribution, including getting the latest stable kernel
     - sudo apt-get update and sudo apt-get upgrade and sudo apt-get dist-upgrade

Note: On stock Ubuntu* 14 LTS systems with Kernel prior to 4.4.0-04 the basic apt-get upgrade command is not sufficient to bring the distribution to the latest recommended baseline. On those systems use: `sudo apt-get install --install-recommends linux-generic-lts-xenial xserver-xorg-core-lts-xenial xserver-xorg-lts-xenial xserver-xorg-video-all-lts-xenial xserver-xorg-input-all-lts-xenial libwayland-egl1-mesa-lts-xenial`

- Update OS Boot and reboot to enforce the correct kernel selection with `sudo update-grub` and `sudo reboot`
- Interrupt the boot process at Grub2 Boot Menu -> "Advanced Options for Ubuntu" and select the kernel version installed in the previous step. Press and hold SHIFT if the Boot menu is not presented.
- Complete the boot, login and verify that a supported kernel version (4.4.0-.., 4.8.0-.., 4.10.0-.. or 4.13.0-.. as of Feb 2018) is in place with `uname -r`

- Video4Linux backend preparation:

1. Ensure no Intel® RealSense™ cameras are plugged in.
2. Install openssl package required for kernel module build:
   - `sudo apt-get install libssl-dev`
3. Install udev rules located in librealsense source directory:
   - `sudo cp config/99-realsense-libusb.rules /etc/udev/rules.d/`
   - `sudo udevadm control --reload-rules and udevadm trigger`
4. Install the packages required for librealsense build:
   - libusb-1.0, pkg-config and libgtk-3:
   - `sudo apt-get install libusb-1.0-0-dev pkg-config libgtk-3-dev.`

Note: glfw3 and gtk are only required if you plan to build the examples, not for the librealsense core library.

```
*glfw3*:
*On Ubuntu 16.04 install glfw3 via `sudo apt-get install libglfw3-dev`
*On Ubuntu 14.04 or when running of Ubuntu 16.04 live-disk, use
`./scripts/install_glfw3.sh`
```

5. Build and apply patched kernel modules for
   - Ubuntu 14/16 LTS the script will download, patch and build several kernel modules (drivers). Then it will attempt to insert the patched module instead of the active one. If failed the original uvc module will be preserved.
     - `./scripts/patch-realsense-ubuntu-xenial.sh`
   - Intel® Joule™ with Ubuntu Based on the custom kernel provided by Canonical Ltd.
     - `./scripts/patch-realsense-ubuntu-xenial-joule.sh`
   - Arch-based distributions
     - You need to install the base-devel package group.
     - You also need to install the matching linux*-headers as well (i.e.: linux-lts-headers for the linux-lts kernel).

–Navigate to the scripts folder: `cd ./scripts/`
–Then run the following script to patch the uvc module: `./patch-arch.sh`

- Check installation by examining the patch-generated log as well as inspecting the latest entries in kernel log: `sudo dmesg | tail -n 50`
  The log should indicate that a new uvcvideo driver has been registered.
  Refer to the "troubleshooting" chapter in case of errors/warning reports.

6. TM1-specifics:
   - Tracking Module requires hid_sensor_custom kernel module to operate properly. Due to TM1's power-up sequence constrains, this driver is required to be loaded during boot for the HW to be properly initialized.
     In order to accomplish this add the driver's name hid_sensor_custom to /etc/modules file, eg:
     `echo 'hid_sensor_custom' | sudo tee -a /etc/modules`

4.2.2.3 Building Librealsense2 SDK

- On Ubuntu* 14.04, update the build toolchain to gcc-5:
  - `sudo add-apt-repository ppa:ubuntu-toolchain-r/test`
  - `sudo apt-get update`
  - `sudo apt-get install gcc-5 g++-5`
  - `sudo update-alternatives --install /usr/bin/gcc gcc /usr/bin/gcc-5 60 --slave /usr/bin/g++ g++ /usr/bin/g++-5`
  - `sudo update-alternatives --set gcc "/usr/bin/gcc-5"`

Tip: You can check the gcc version by typing: `gcc -v` If everything went fine you should refer gcc 5.0.0.

- Navigate to librealsense root directory and run `mkdir build and cd build`
- Run CMake:
  - `cmake ../` - The default build is set to produce the core shared object and unit-tests binaries in Debug mode. Use `-D CMAKE_BUILD_TYPE=release` to build with optimizations.
  - `cmake ../ -DBUILD_EXAMPLES=true` - Builds librealsense along with the demos and tutorials
  - `cmake ../ -DBUILD_EXAMPLES=true -DBUILD_GRAPHICAL_EXAMPLES=false` - For systems without OpenGL or X11 build only textual examples

Recompile and install librealsense binaries:

- `sudo make uninstall and make clean and make and sudo make install`

  The shared object will be installed in `/usr/local/lib`, header files in `/usr/local/include`

  The demos, tutorials and tests will be located in `/usr/local/bin`.

Tip: Use make -jX for parallel compilation, where X stands for the number of CPU cores available:

`sudo make uninstall and make clean and make -j8 and sudo make install`

This enhancement may significantly improve the build time. The side-effect, however, is that it may cause a low-end platform to hang randomly

Note: Linux* build configuration is presently configured to use the V4L2 backend by default.

Note: If you encounter the following error during compilation gcc: internal compiler error it might indicate that you do not have enough memory or swap space on the machine. Try closing memory consuming applications, and if you are running inside a VM increase available RAM to at least 2 GB.

- Install IDE (Optional): We use QtCreator as an IDE for Linux* development on Ubuntu

4.2.3 Mac OS Installation

Note: macOS support for the full range of functionality offered by the SDK is not yet complete. If you need support for R200 or the SR300, legacy librealsense offers a subset of SDK functionality.

1. Install XCode 6.0+ via the AppStore
2. Install the Homebrew package manager via terminal - link
3. Install the following packages via brew:
   - brew install libusb pkg-config
   - brew install homebrew/core/glfw3
   - brew install cmake
4. Generate XCode project:
   - mkdir build and cd build
   - sudo xcode-select --reset
   - cmake -DBUILD_EXAMPLES=true -DBUILD_WITH_OPENMP=false -DHWM_OVER_XU=false -G Xcode
5. Open and build the XCode project

4.2.4 Android* Installation

The SDK 2.0 delivers cross-platform open source libraries and tools that allow users to develop on multiple Operating Systems and development environments. Intel has validated SDK2.0 on Windows and Linux* platforms. Check latest Release for the build versions. While Intel has not explicitly validated SDK2.0 on Android* platforms, it is expected to work on Android* as well. Refer to the build instructions in the section below. Calibration and firmware update tools that would be used in production and manufacturing processes are not available on Android* at this time. Contact the Intel representative for additional information.

Disclaimer: There is a limitation in access to the camera from a native code due to Android* USB permissions policy. Therefore, building Intel® RealSense™ SDK 2.0 for Android* devices will run only on rooted devices.

4.2.4.1 Build Intel® RealSense™ SDK 2.0 for Android* OS

Before jumping to the instructions section ensure you have all the required accessories.

1. Linux* host machine with Ubuntu 16.04.
2. Rooted Android target device.

3. USB3 OTG cable.

Building SDK Samples for Android* OS

1. Root the Android* device.
2. Download the Native Development Kit (NDK) for Linux* to the host machine.
3. Install CMake 3.6.1 or newer.
4. Download ADB to the host machine by typing sudo apt-get install adb.
5. Clone the latest RealSense™ SDK 2.0 to the host machine.
6. Change the streaming width and height to 480 and 270 respectively in rs-depth and rs-distance examples using Linux* text editor.
7. Open Terminal on the host machine, navigate to librealsense root directory and type the following lines:
   - mkdir build &and cd build
   - cmake .. -DANDROID_ABI=<Application Binary Interface> -DCMAKE_TOOLCHAIN_FILE=<Path to NDK folder>/build/cmake/android.toolchain.cmake
   - make

   Initialize ANDROID*_ABI with one of the supported ABIs (armeabi-v7a for example).
8. When compilation done type the following lines to store the binaries at the same location to easily copy them to the Android* device.
   - mkdir lrs_binaries &and cd lrs_binaries
   - cp ../librealsense2.so ./
   - cp ../examples/C/color/rs-color ./
   - cp ../examples/C/depth/rs-depth ./
   - cp ../examples/C/distance/rs-distance ./
   - cp ../examples/save-to-disk/rs-save-to-disk ./
   - cp ../tools/data-collect/rs-data-collect ./
   - cp ../tools/enumerate-devices/rs-enumerate-devices ./
   - cp ../tools/fw-logger/rs-fw-logger ./
   - cp ../tools/terminal/rs-terminal ./
9. Connect the Android* device to the host machine using USB OTG cable.
10. Create new folder and copy the binaries to the Android* device using ADB by the following lines:
    - adb shell mkdir /storage/emulated/legacy/lrs_binaries
    - adb push . /storage/emulated/legacy/lrs_binaries/
11. Open ADB Shell and move to Super User mode by the following line:
    - adb shell su
12. Copy the binaries to the internal storage and change their permission to be executables by the following lines:
    - cp -R /storage/emulated/legacy/lrs_binaries /data/
    - cd /data/lrs_binaries
    - chown root:root *
    - chmod +x *
13. Use the USB OTG cable to connect the RealSense™ camera to the Android* device.
14. Install Terminal Emulator on the Android* device from Google* Play Store.
15. Open the Terminal Emulator application and type below lines in order to move to Super User mode and run one of the RealSense™ examples/tools.
    - su
    - cd /data/lrs_binaries
    - ./rs-depth

Expected Output

- Streaming Depth data using rs-depth sample on rooted Samsung* Galaxy S4 device.

§ §

5 Additional reference

RealSense™ Website:

- https://realsense.intel.com/

RealSense™ White Paper:

- https://realsense.intel.com/intel-realsense-downloads/#whitepaper

§ §