Develop Yocto* Project Compatible Applications with the Intel® oneAPI IoT Toolkit and Linux* Build Tools

User Guide

Legal Information

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

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.

This document contains information on products, services and/or processes in development. All information provided here is subject to change without notice. Contact your Intel representative to obtain the latest forecast, schedule, specifications and roadmaps.

The products and services described may contain defects or errors known as errata which may cause deviations from published specifications. Current characterized errata are available on request.

Intel technologies may require enabled hardware, software or service activation.

No product or component can be absolutely secure.

Your costs and results may vary.

Performance varies by use, configuration and other factors. Learn more at www.Intel.com/PerformanceIndex.

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

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

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

© Intel Corporation.

Contents

Introduction

With the Yocto* Project Compatible application development plugin in the Eclipse* IDE, you can develop C/C++ applications for Yocto Project targets, using the standard Yocto Project compatible Software Development Kits (SDK) and Extensible Software Development Kits (eSDK).

This guide takes you through the steps to create, build, and debug an application:

Before You Start

Before you can develop a Yocto Project Compatible application using the Intel oneAPI IoT Toolkit, you must install the following on your host system:

- A Linux* development system (Windows* and macOS* are *not* supported)
- Eclipse IDE for C/C++ Developers for Linux x86_64 systems
- Intel oneAPI IoT Toolkit (opt for installation of the "Linux kernel build tools")
- Yocto Project 2.6 or later (optional if you are importing an existing Yocto SDK)

If you are working behind a firewall, you may also need to configure the Eclipse IDE proxy settings.

Step 1. Install the Eclipse IDE for C/C++ Developers

If you have not installed a copy of the Eclipse IDE for C/C++ Developers, do so now.

***NOTE**: the Yocto Project Compatible plugins for Eclipse are only supported on Linux development hosts. Windows and macOS development hosts are not supported.*

The 2021.x versions of the Intel oneAPI IoT Toolkit include a copy of the Eclipse IDE for C/C++ Developers. You can skip this step if you plan to install and use that copy of Eclipse, otherwise:

- Go to www.eclipse.org/downloads/packages
- Click the "**Eclipse IDE for C/C++ Developers**" title in the list of Eclipse IDE packages.
- Locate the "**Linux x86_64**" download link and install Eclipse onto your Linux development host.

Step 2. Install the Intel oneAPI IoT Toolkit

If you have not installed the Intel oneAPI Toolkit, do so now.

***NOTE**: installation of the Intel oneAPI Base Toolkit is optional and is not required to use the Linux kernel build tool plugins for Eclipse. The Linux kernel build tools do not require or use the Intel oneAPI DPC++ compiler, they require gcc for building Yocto Project Compatible system images.*

- Before you start, review the System Requirements and Release Notes to make sure you have the prerequisites you need to install the product.

  System Requirements
  Release Notes

  To install the Intel oneAPI IoT Tookit, click here .

- After you install the product, see the getting started guide, especially the section describing how to install the GNU* development tools.

  Getting Started Guide

Step 3. (Optional) Install Yocto Project

***OPTIONAL**: this step is only required if you are going to build a Yocto Compatible Platform project. If you will be using a Yocto Project Compatible platform SDK, you can skip this step.*

Before creating a Yocto Project platform project, you must prepare your host system and install Yocto Project 2.6 or later.

Prepare your host system

Before you install Yocto Project you must prepare your host system, see Yocto Project Quick Build.

Download and install recommendations

We recommend that you download a minimal Yocto Project installation, and then use Toaster to select and automatically download the desired layers (from the current or past release) plus other extra content during initial development. Once the design is stable, you can use Toaster to explicitly fetch the extra content.

Supported host operating systems

The Yocto Project documentation lists those host platforms that can be used to develop Yocto Project images and applications.

Installation instructions

- Product and download information: https://www.yoctoproject.org/software-overview/downloads/
- Installation information: https://www.yoctoproject.org/docs/latest/brief-yoctoprojectqs/brief-yoctoprojectqs.html

Step 4. (Optional) Configure Your Host System

***OPTIONAL**: this step is only required if you are going to build a Yocto Compatible Platform project. If you will be using a Yocto Project Compatible platform SDK, you can skip this step.*

- The installation must be writeable to create projects and run the **bitbake** build system.
- Do not build content as "root" or "sudo." This is enforced by the build tool **bitbake** and prevents unexpected side effects to the host system from builds and modifications of the kernel and file systems.
- The development user must set up their proxies to allow **git** and **http** access to the network.

Step 5. (Optional) Configure the Eclipse IDE Proxy

If you are on an open network, skip this step.

If you are working behind a proxy server, you may need to configure the Eclipse IDE proxy. Follow these Eclipse proxy instructions.

Consult your network administrator to determine the correct proxy settings for your work environment.

***NOTE**: We recommend that you leave the SOCKS proxy setting empty in the Eclipse IDE proxy settings panel. In some proxy environments, setting the Eclipse SOCKS proxy may interfere with the ability of the Eclipse IDE to access Internet resources. In this case, the error messages provided by the Eclipse IDE may be misleading. This is a known problem with the Eclipse IDE.*

If you encounter problems with cloning git repos, check your **.gitconfig** file to insure it is configured to clone using http and https URLs.

***NOTE:** Use the **insteadOf** rule in your **.gitconfig** file.*

Example:
[url https://git.yoctoproject.org/git]
insteadOf = "git://git.yoctoproject.org"

Prepare a Yocto Project Platform SDK

***NOTE** If you already have a Yocto* Project Compatible platform SDK, you can skip this section and go directly to Import your SDK into the Eclipse IDE for instructions to import your SDK.*

Build a Yocto Project SDK for use with the Eclipse IDE

***NOTE** For these instructions, we will use the default **qemux86** machine type.*

***NOTE** The SSH server is required to deploy and debug applications. The other items are optional depending on the intended usage.*

Add the following to your image layer's **conf/bblayers.conf** file.

```
# Add the target connection and debug agents (GDB, tcf-agent, ...)
EXTRA_IMAGE_FEATURES += " eclipse-debug ssh-server-openssh "
IMAGE_INSTALL_append = " gdbserver "

# Add C++ runtime for C++ applications
IMAGE_INSTALL_append = " libstdc++ "

# Add the 32-bit libraries to the target files system
IMAGE_INSTALL_append = "lib32-glibc lib32-libgcc lib32-libstdc++"
# Add 'file' so that you can test 64 versus 32 bit applications
IMAGE_INSTALL_append = " file "
```

Make sure you are in a clean shell to set up the environment. Enter:

```
$ cd <yp_installation_dir>
$ . oe-initbuild-env
```

By default, the project build directory will be in **<yp_installation_dir>/build**. To specify a different location, enter:

```
$ . oe-init-build-env <my_dir>
```

Build the project (in the same shell).

```
$ bitbake core-image-base
$ bitbake core-image-base -c populate_sdk
```

Extract the SDK somewhere. For example, the following is for the SDK development workflow.

```
$ mkdir /opt/sdk
$ ./tmp-glibc/deploy/sdk/poky-glibc-x86_64-core-image-base-i586-toolchain-
2.6+snapshot.sh -y - d /opt/sdk
```

Results

After building the default SDK, the resulting **tmp/deploy/sdk** directory looks like the following. It is under **tmp** for this BSP, and there is only one SDK variant (32-bit) as opposed to the LTS default for **qemux86-64** (32-bit and 64-bit). Also,

NOTE *This example shows Yocto Project release **2.5**. For Yocto Project 2.6 the release number will be **2.6**, and so on.*

```
[build]$ ls tmp/deploy/sdk/
poky-glibc-x86_64-core-image-base-i586-toolchain-2.5+snapshot.host.manifest poky-glibc-
x86_64-core-image-base-i586-toolchain-2.5+snapshot.sh
poky-glibc-x86_64-core-image-base-i586-toolchain-2.5+snapshot.target.manifest poky-glibc-
x86_64-core-image-base-i586-toolchain-2.5+snapshot.testdata.json
[build]$
```

If you build the image, the directory looks like this:

```
[build]$ ls tmp/deploy/images/qemux86/
bzImage
bzImage--4.14.48+git0+94457657b8_65d1c84953-r0-qemux86-20180812100752.bin
bzImage-qemux86.bin
```

```
core-image-base-qemux86-20180812100752.qemuboot.conf core-image-base-qemux86-
20180812100752.rootfs.ext4
core-image-base-qemux86-20180812100752.rootfs.manifest core-image-base-
qemux86-20180812100752.rootfs.tar.bz2 core-image-base-qemux86-
20180812100752.testdata.json core-image-base-qemux86.ext4
core-image-base-qemux86.manifest
core-image-base-qemux86.qemuboot.conf core-image-base-qemux86.tar.bz2
core-image-base-qemux86.testdata.json
modules--4.14.48+git0+94457657b8_65d1c84953-r0-qemux86-20180812100752.tgz
modules-qemux86.tgz
[build]$
```

Create a Development Environment

Customize for use with the Eclipse IDE

Add the following to `EXTRA_IMAGE_FEATURES` in `conf/local.conf`. This enables the SDK you generate to work with the Eclipse IDE. Here are some example settings, with comments, to enable different parts of the integration:

```
# Add the target connection and debug agents (GDB, tcf-agent, ...) WRTEMPLATE
+= "feature/debug"
EXTRA_IMAGE_FEATURES += " eclipse-debug ssh-server-openssh "
# Add C++ runtime for C++ applications
IMAGE_INSTALL_append = " libstdc++ "
# Add the 32-bit libraries to the target files system
IMAGE_INSTALL_append = "lib32-glibc lib32-libgcc lib32-libstdc++"
# Add 'file' so that you can test 64 versus 32 bit applications
IMAGE_INSTALL_append = " file "
```

Build the project

To build the project, enter these commands:

```
$ bitbake wrlinux-image-glibc-core
$ bitbake wrlinux-image-glibc-core -c populate_sdk
```

Extract the SDK

To extract the SDK, enter these commands:

```
$ mkdir ~/wrl-sdk
$ ./tmp-glibc/deploy/sdk/wrlinux-small-10.17.41.2-glibc-x86_64-qemux86_64-
wrlinux-image-glibc- core-sdk.sh -y -d ~/wrl-sdk
```

The target folder shown at the end of the second command above (`wrl-sdk`) can be replaced. It should point to a folder for which you have Write permissions.

Continue to the next section to import the SDK into the Eclipse IDE.

Import your SDK into the Eclipse IDE

Start the Eclipse IDE

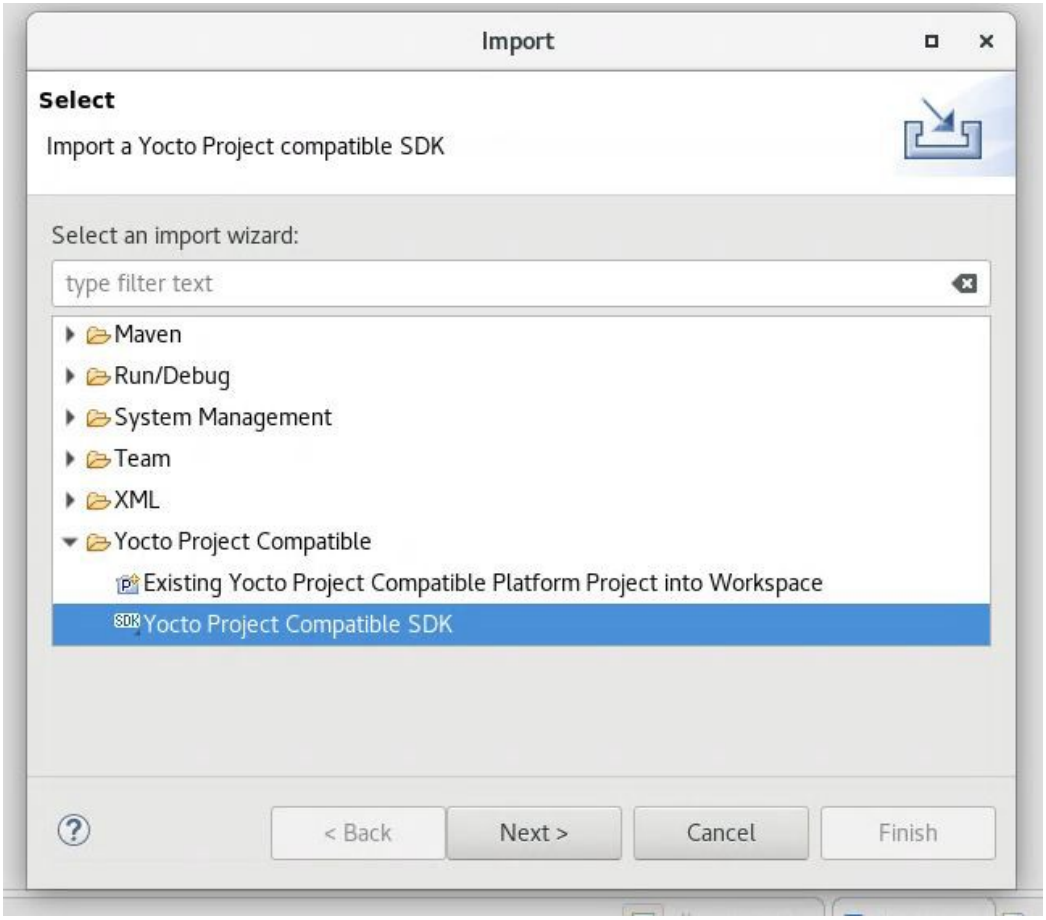
Configure the Intel oneAPI environment variables by sourcing the setvars.sh script or by sourcing the Linux kernel build tools vars.sh script directly:

```
source <install-dir>/intel/oneapi/setvars.sh
```
or
```
source <install-dir>/intel/oneapi/eclipse-iot-plugins/latest/env/vars.sh
```

Launch your copy of Eclipse IDE for C/C++ Developers that includes the Linux build tools plugins *from the command line in the shell from which you sourced the environment variables configuration script*, in the step above.

Import the SDK

- In the Eclipse IDE, choose **File** > **Import** to start the import wizard.
- Expand the **Yocto Project Compatible** folder and select **Yocto Project Compatible SDK**.



- Click **Next**.

- For **Folder**, browse for and select the SDK.
- Click **Finish.**

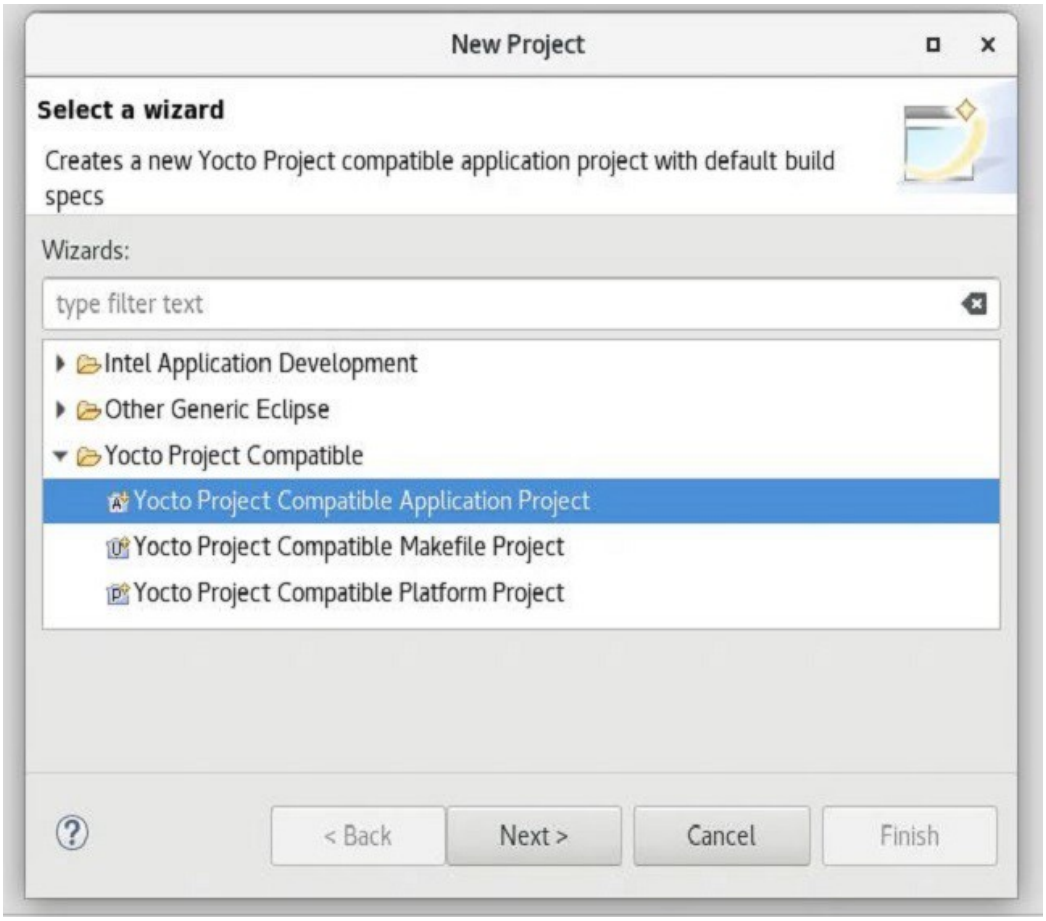
Continue to the next section to create and build a new application in the Eclipse IDE.

Create and Build a New Application

The topics in this section explain how to create and build an application in the Eclipse IDE. Before you continue, make sure you have completed the steps above.

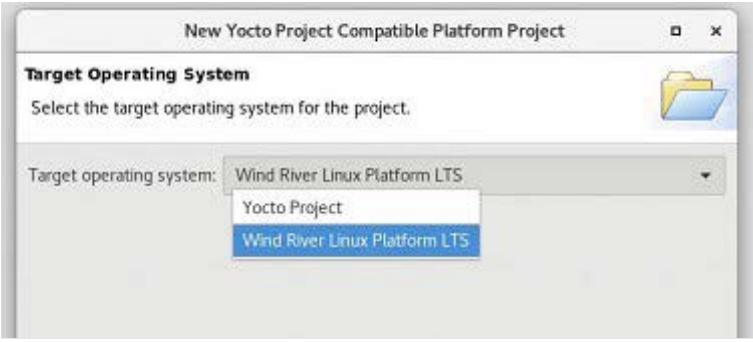
Create a new project

- Choose **File** > **New** > **Project** to start the new project wizard.
- Expand the **Yocto Project Compatible** folder and select **Yocto Project Compatible Application Project**.

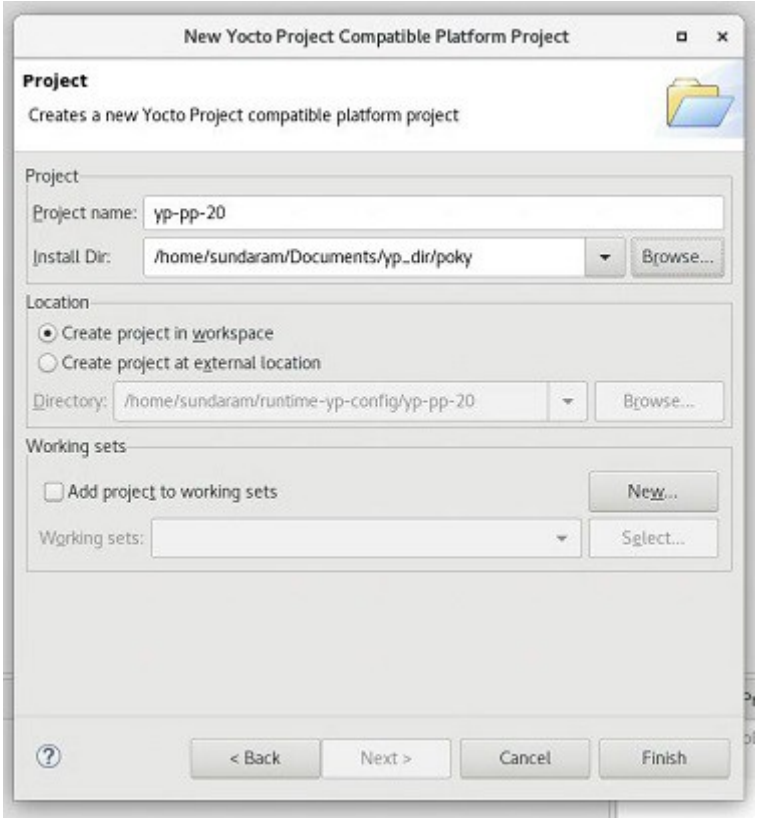


- Click **Next**.

- From the **Target operating system** drop-down list, select **Yocto Project** and click **Next**.

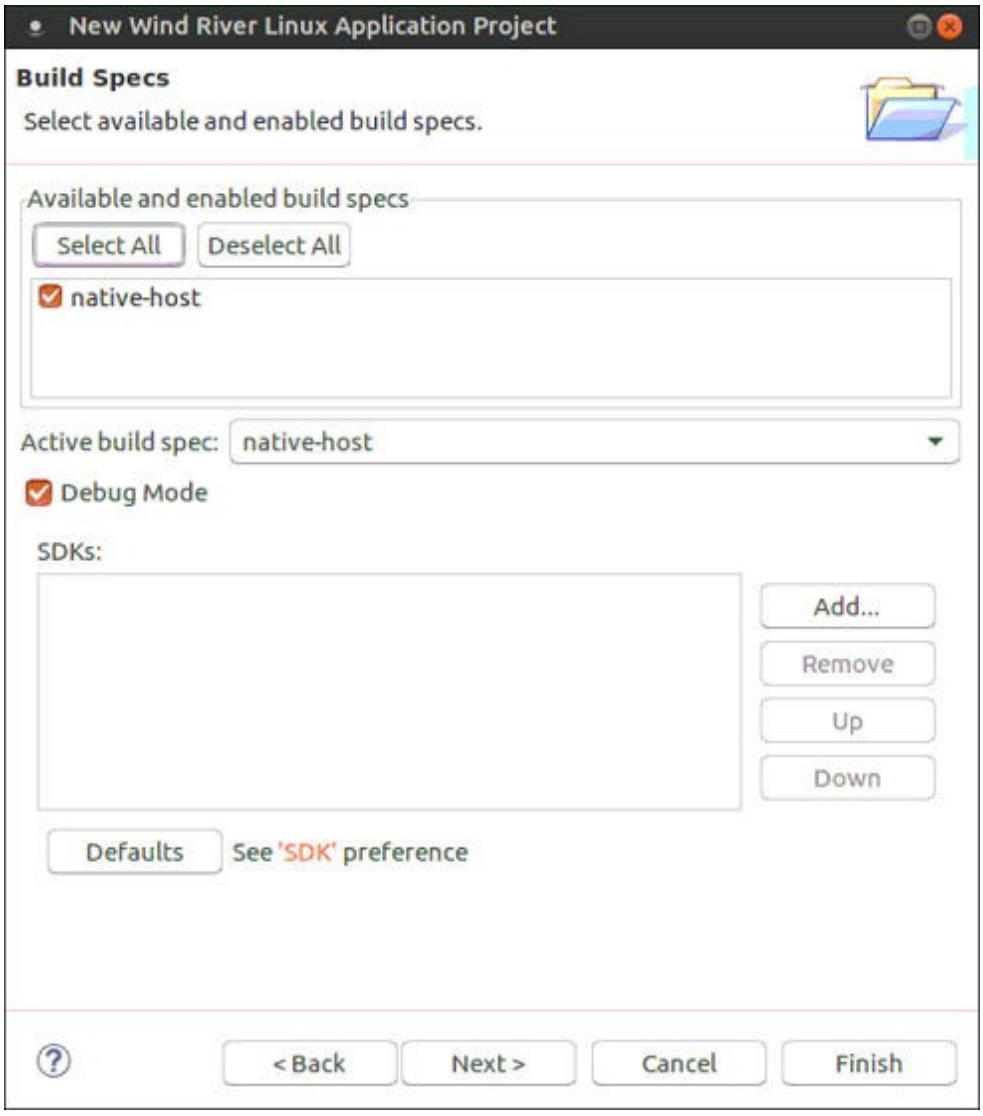


- Enter a **Project name** and browse to the location where **Wind River Linux** or **Yocto Project** is installed.
- Click **Finish.**

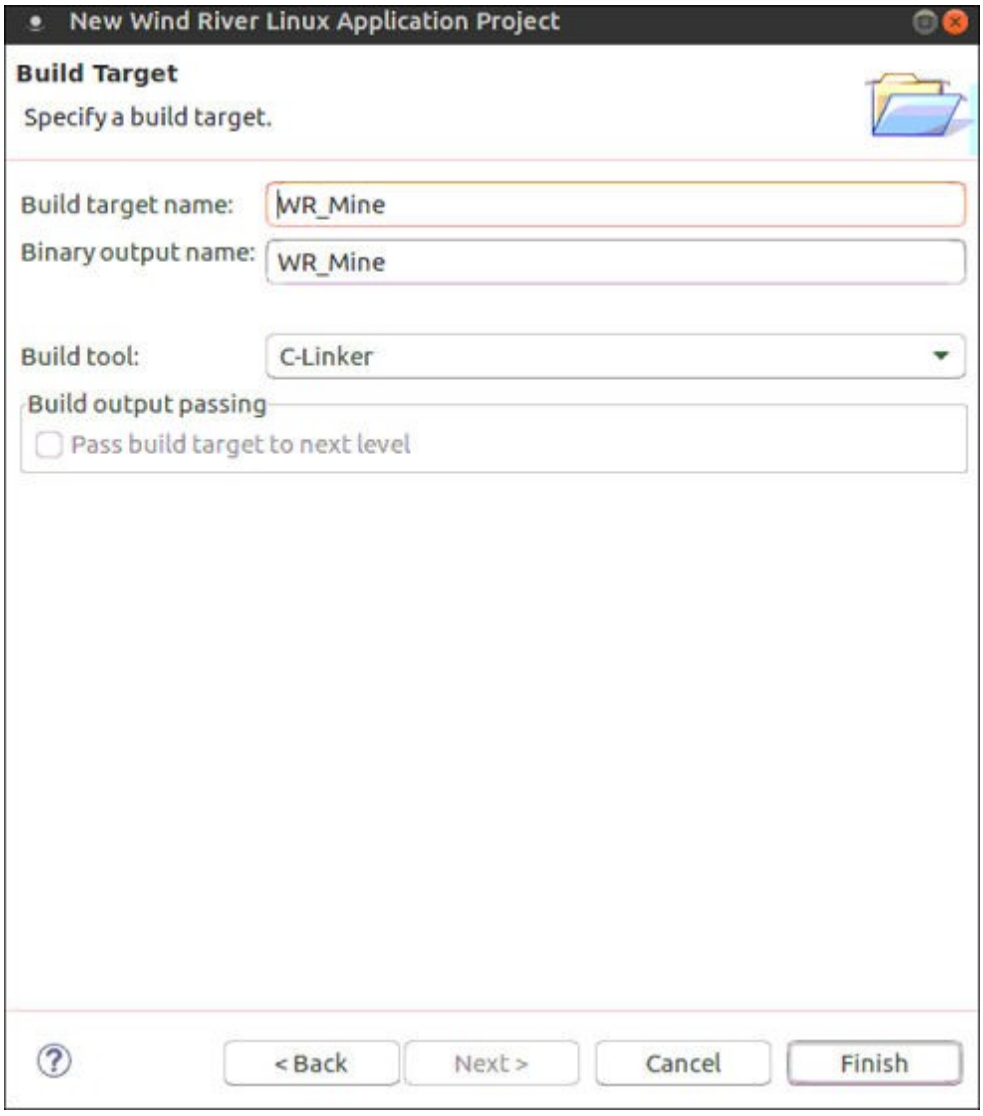


Tip *You can create Working Sets to group related elements in your workspace for display in views or for operations on a set of elements. On the Project screen above, you can add the new project to an existing Working Set or create a new Working Set. For information, click* ***New*** *to open the* ***New Working Set*** *window and then click Help.*

- Select a Build Spec and click **Next**.



- Specify a Build target and Build tool.

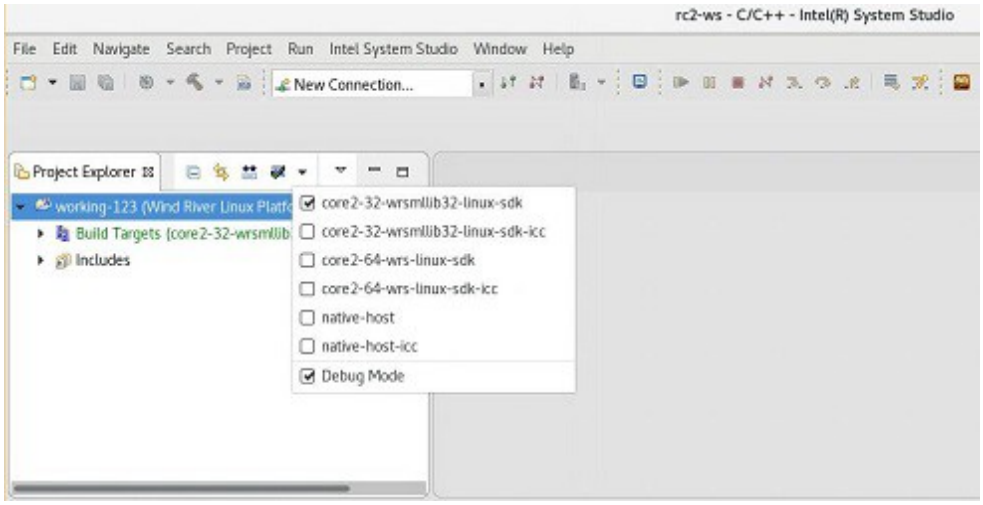


- Click Finish.

Your project is created and displayed in the Project Explorer.

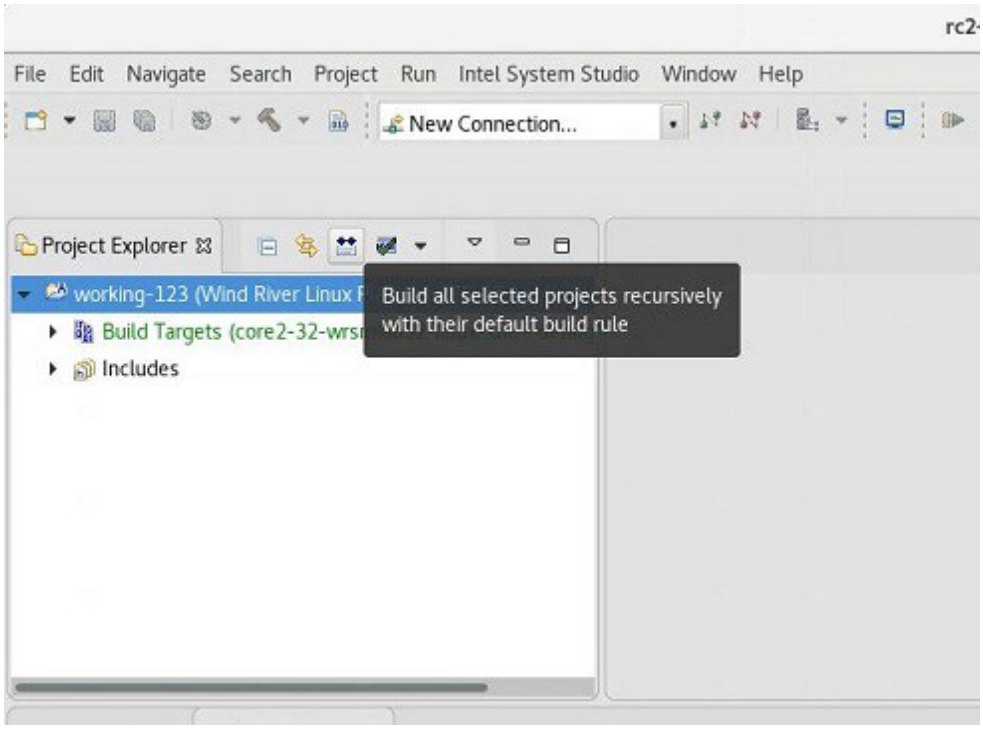
Build the application

- In the **Project Explorer** list, click the name of your project.
- In the project toolbar, click the **Build Specification** drop-down list and select a build specification.

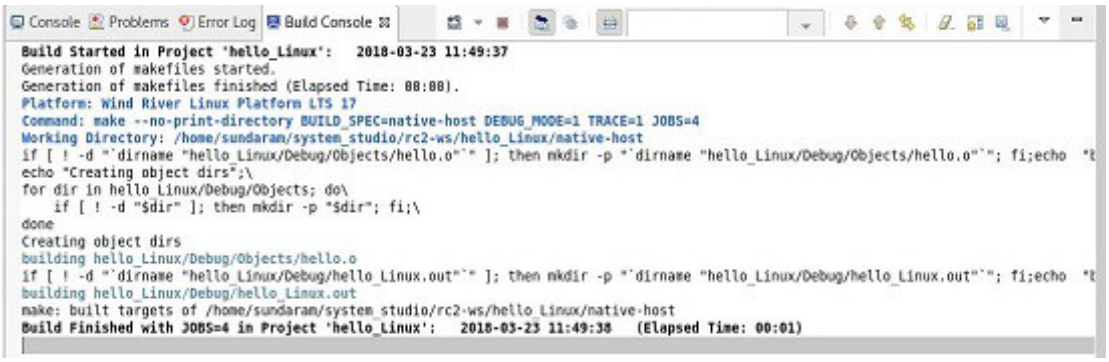


NOTE *The first SDK you imported is checked as the default build specification. If you did not import any SDKs, you can build an application to run on your host development system by selecting* ***native- host*** *or* ***native-host-icc****.*

- In the project toolbar, click the **Build** button.



- The results appear in the Build Console.



Export and Import a Project

To export a project from the Eclipse IDE:

- In the **Project Explorer**, select the project to export.
- Select **File** > **Export**.
- In the **Export** window, select **General** > **Archive File** and click **Next**.
- Select the archive file location and file compress/format options.
- Click **Finish**.

The archive file is exported and stored in the location specified. To import a project into the Eclipse IDE:

- In the Eclipse IDE, select **File** > **Import**.
- In the **Import** window, select one of the following options and click **Next**:

  General > **Projects from Folder or Archive**: use this option based on the absolute/relative paths within the project. An absolute path can be reset for a new user after import is complete from the right-click menu by selecting **Build** > **Clean Project**.

  General > **Existing Projects into Workspace**: use this option if different users have the same path for the shared mount point.

- Leave the **Copy** checkbox unselected and click **Finish**.

When import completes, the imported project appears in the **Project Explorer**.

Create a Target Connection

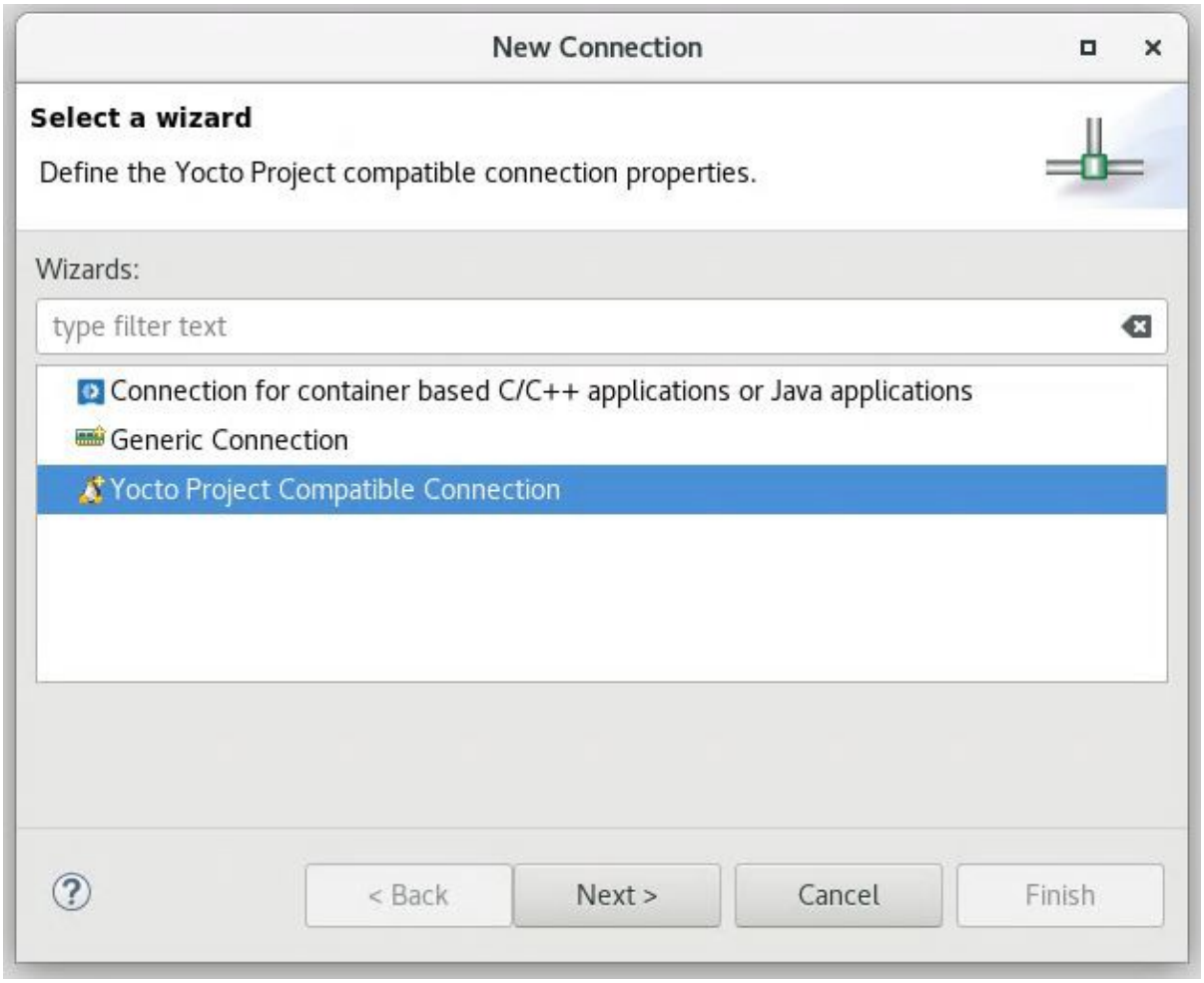
If you want to run or debug a new project, you need to create a new connection.

If you already have a connection, simply click on the arrow next to the connection drop-down list, select your connection, and continue to the next topicRun or Debug a Yocto Project Compatible Application.

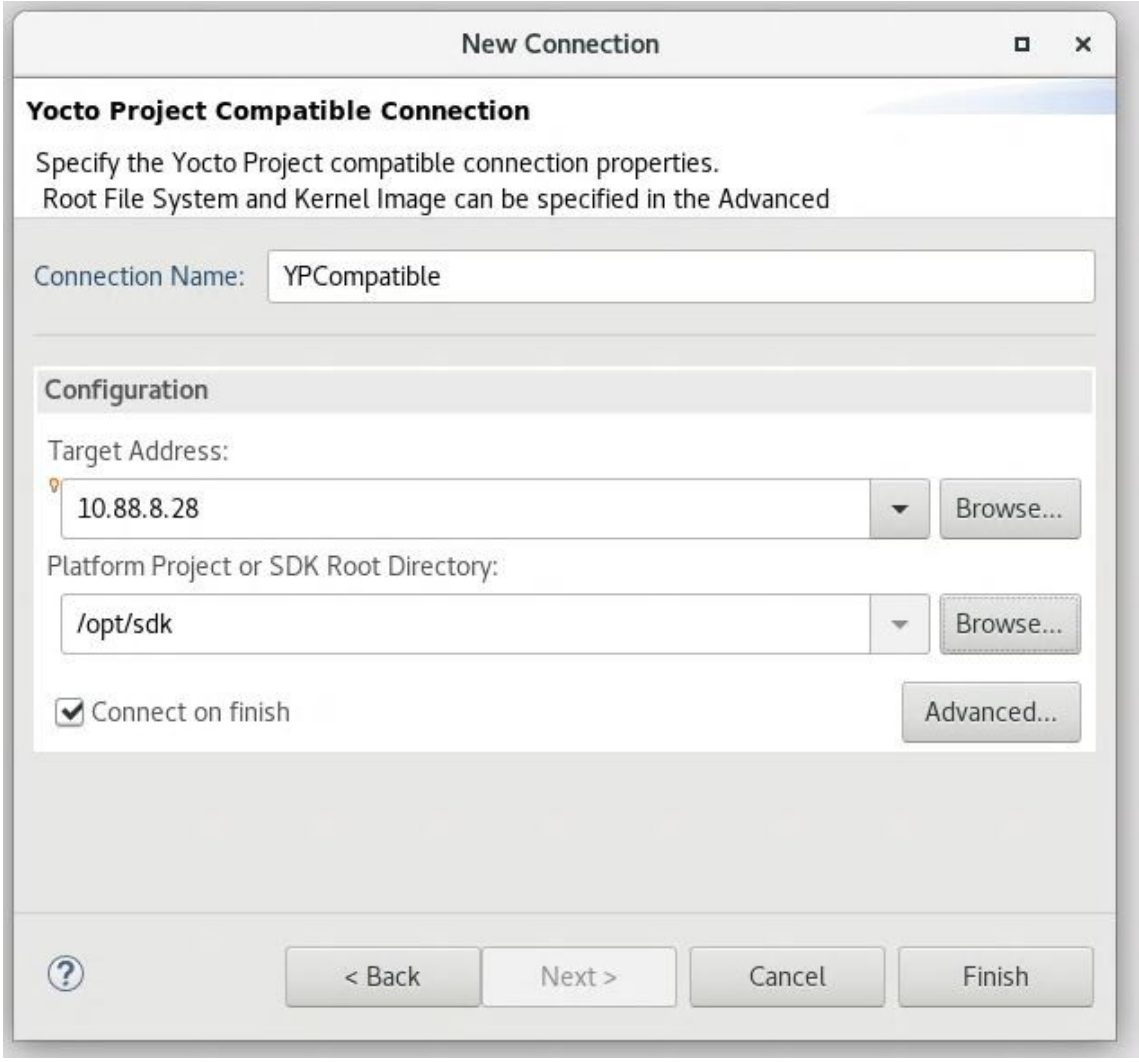
Make sure your target device is physically connected to your host system and functioning correctly.

NOTE *If your target is QEMU, start the QEMU simulator in your project directory by entering the command* `$ runqemu qemux86-64 nographic`. *When prompted, run* `ipconfig` *to get the IP address.*

- Click **New Connection** in the target selection drop-down list in the toolbar.
- Select **Yocto Project Compatible Connection** and click **Next**.



- Browse for and select the folder for the Platform Project or SDK Root Directory.
- Enter an IP address for Target Address and select Connect on finish.



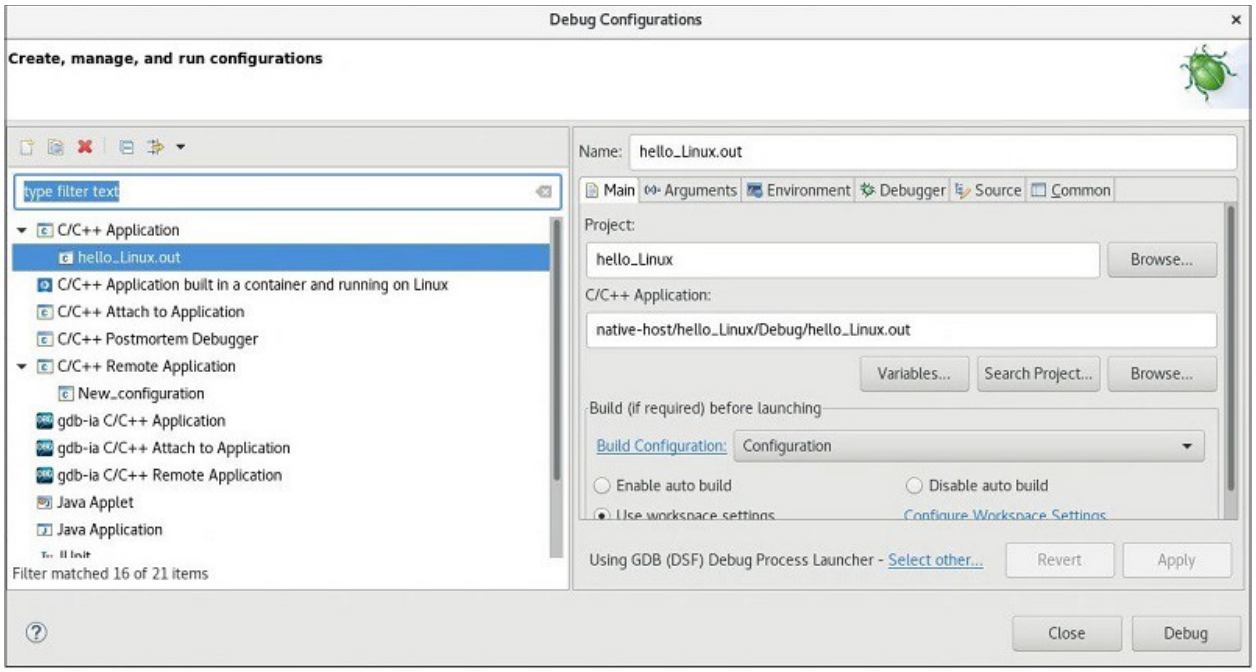
- Click **Finish**. The connection is established.

Run or Debug a Yocto Project Compatible Application

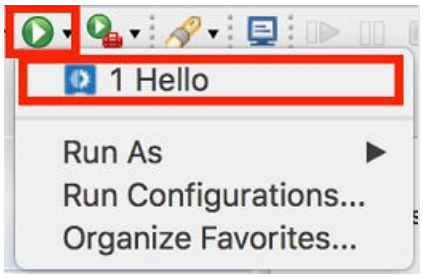
For more information on running and debugging programs in Eclipse*, see the documentation on the Eclipse web site.

- If you haven't already done so, make sure your target device is physically connected to your host system and functioning correctly. Or, if you have a simulated target (QEMU), make sure it is running.
- If the connection is not already open, select the connection name from the target selection drop-down list in the toolbar and click the green **Connect** button.
- After the connection has been established, the red **Disconnect** button becomes active, and the word (Connected) appears in title bar of the Eclipse IDE.
- To create a remote Run or Debug configuration, from the **Run** drop-down list in the toolbar, select either **Run Configurations** or **Debug Configurations**.
- Set up your configuration on the configurations screen. The sample below shows the **Debug Configurations** screen for a **C/C++ Application**.

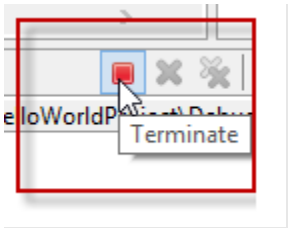
  NOTE *To specify the app you want to run or debug, click the* ***Search Project*** *button and select from a list of built applications. The path of the application you select will be used to fill the* ***Remote Absolute File Path****.*



- To run a project, click the **Run** drop-down list in the toolbar and select the name of your project.



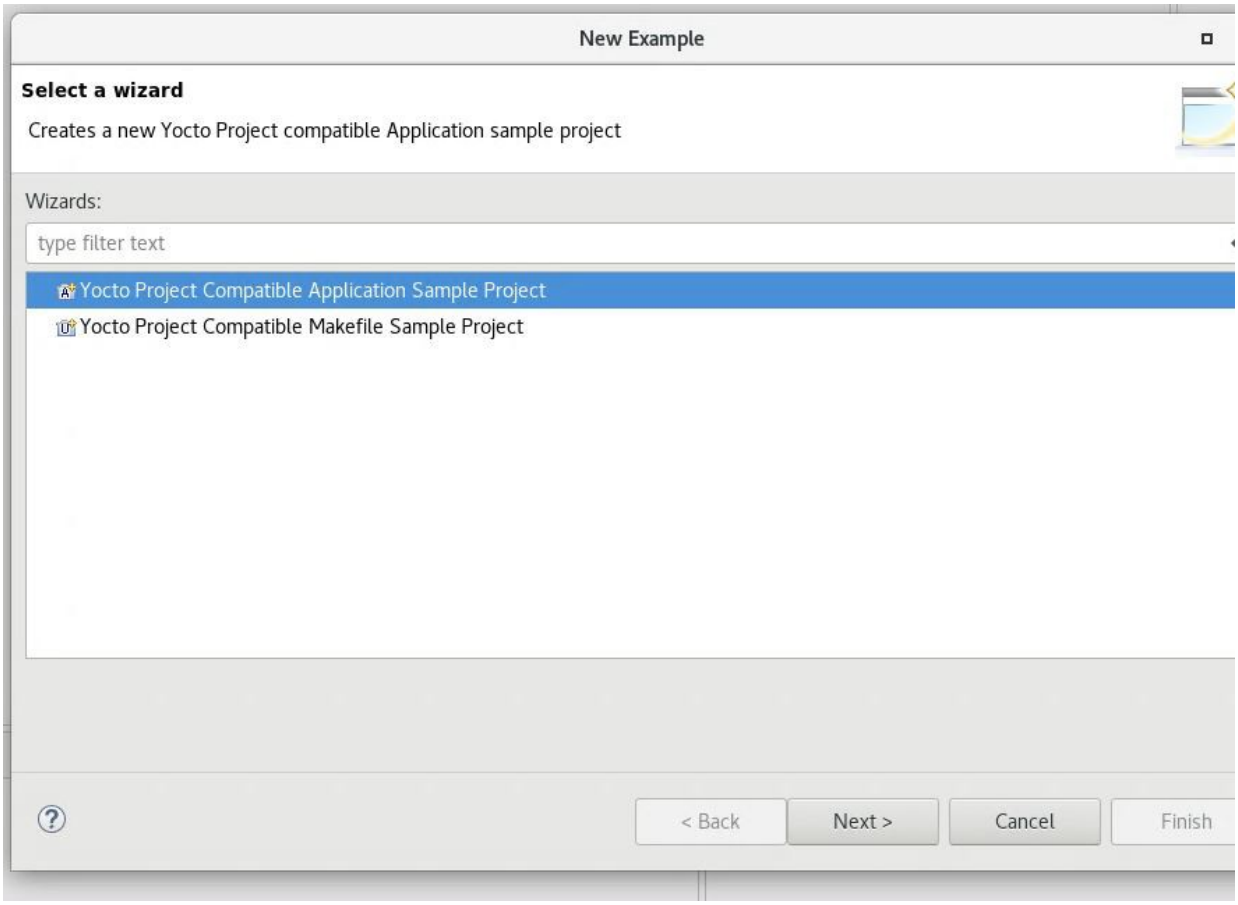
- To debug a project, click the **Debug** drop-down list in the toolbar and select the name of your project.
- If a warning message about host authenticity is displayed, click **Yes** to upload and run your project.
- Your project should be running.
- When finished, click the red **Terminate** button on the **Console** panel to end the currently running process.



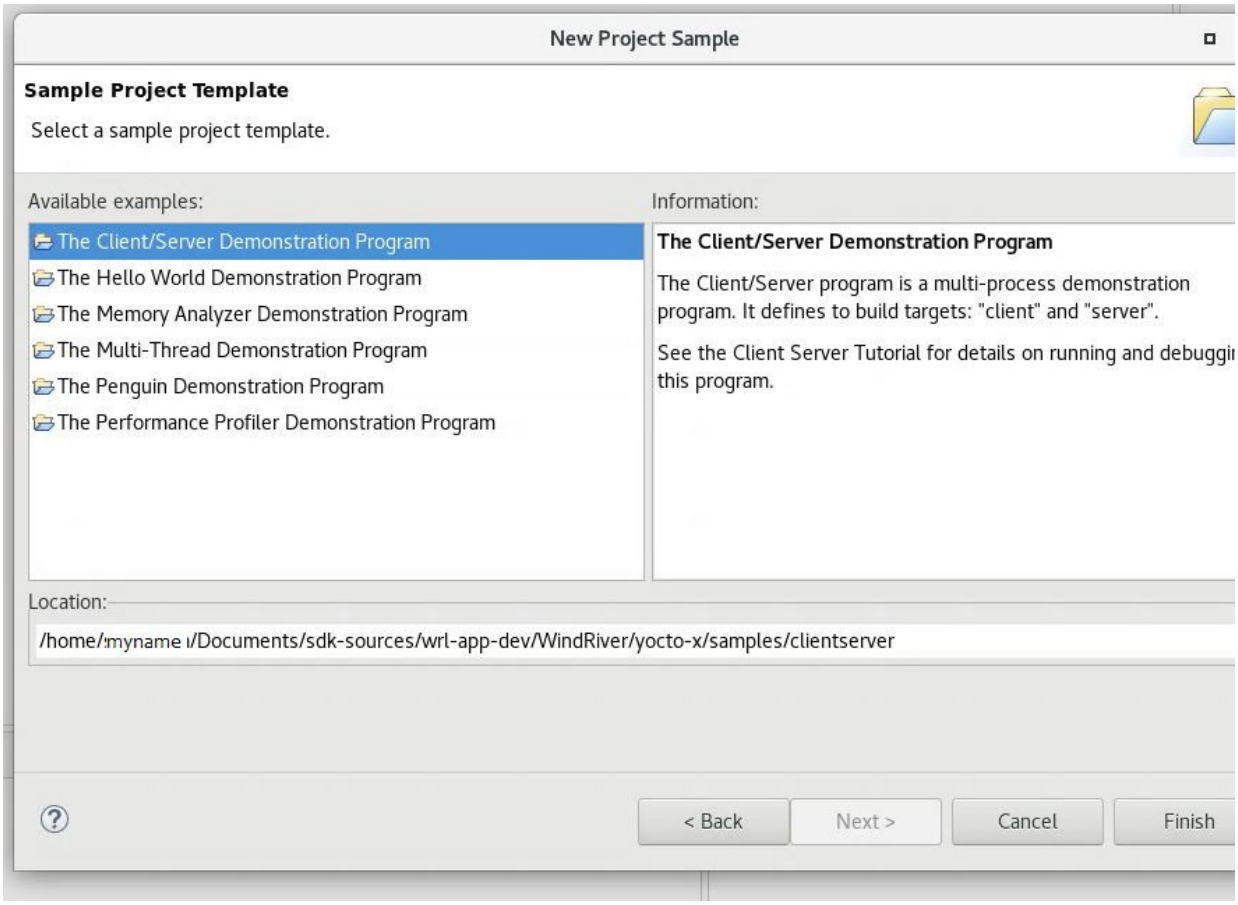
Using Samples - Yocto Project

To create a new project quickly, you can start with one of the Yocto* Project samples available in the Eclipse IDE.

- Choose **File** > **New** > **Example** to start the new example wizard.
- Select **Yocto Project Compatible Application Sample Project** and click **Next**.



- Select a sample, for example, Hello World.
- Click **Finish**.



- The sample project is opened in the Project Explorer.
- In the **Project Explorer**, click the name of your project.
- In the Project Explorer toolbar, click the **Build Specification** drop-down list and select **native-host** and **Debug Mode**.
- In the Project Explorer toolbar, click the **Build** button.
- The results appear in the Build Console.