



# Yocto Project and OpenEmbedded development training

On-line seminar, 4 sessions of 4 hours

Latest update: May 06, 2024

<b>Title</b>	<b>Yocto Project and OpenEmbedded development training</b>
<b>Training objectives</b>	<ul style="list-style-type: none"><li>• Be able to understand the role and principle of an embedded Linux build system, and compare Yocto Project/OpenEmbedded to other tools offering similar functionality.</li><li>• Be able to configure and build basic embedded Linux system with Yocto, and install the result on an embedded platform.</li><li>• Be able to write and extend recipes, for your own packages or customizations.</li><li>• Be able to use existing layers of recipes, and create your own new layers.</li><li>• Be able to integrate support for your own embedded board into a BSP layer.</li><li>• Be able to create custom images.</li><li>• Be able to use the Yocto Project SDK to develop applications.</li><li>• Be able to use devtool to generate and modify recipes.</li></ul>
<b>Duration</b>	<b>Four</b> half days - 16 hours (4 hours per half day)
<b>Pedagogics</b>	<ul style="list-style-type: none"><li>• Lectures delivered by the trainer, over video-conference. Participants can ask questions at any time.</li><li>• Practical demonstrations done by the trainer, based on practical labs, over video-conference. Participants can ask questions at any time. Optionally, participants who have access to the hardware accessories can reproduce the practical labs by themselves.</li><li>• Instant messaging for questions between sessions (replies under 24h, outside of week-ends and bank holidays).</li><li>• Electronic copies of presentations, lab instructions and data files. They are freely available at <a href="https://bootlin.com/doc/training/yocto">https://bootlin.com/doc/training/yocto</a>.</li></ul>
<b>Trainer</b>	One of the engineers listed on: <a href="https://bootlin.com/training/trainers/">https://bootlin.com/training/trainers/</a>
<b>Language</b>	Oral lectures: English, French, Italian. Materials: English.



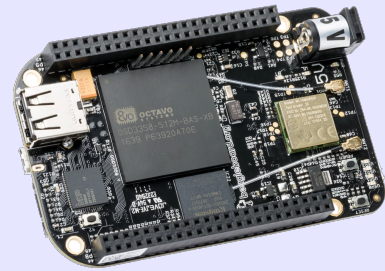
<b>Audience</b>	Companies and engineers interested in using the Yocto Project to build their embedded Linux system.
<b>Prerequisites</b>	<ul style="list-style-type: none"><li>• <b>Knowledge and practice of UNIX or GNU/Linux commands:</b> participants must be familiar with the Linux command line. Participants lacking experience on this topic should get trained by themselves, for example with our freely available on-line slides at <a href="http://bootlin.com/blog/command-line/">bootlin.com/blog/command-line/</a>.</li><li>• <b>Minimal experience in embedded Linux development:</b> participants should have a minimal understanding of the architecture of embedded Linux systems: role of the Linux kernel vs. user-space, development of Linux user-space applications in C. Following Bootlin's <i>Embedded Linux</i> course at <a href="http://bootlin.com/training/embedded-linux/">bootlin.com/training/embedded-linux/</a> allows to fulfill this pre-requisite.</li><li>• <b>Minimal English language level: B1</b>, according to the <i>Common European Framework of References for Languages</i>, for our sessions in English. See <a href="http://bootlin.com/pub/training/cefr-grid.pdf">bootlin.com/pub/training/cefr-grid.pdf</a> for self-evaluation.</li></ul>
<b>Required equipment</b>	<ul style="list-style-type: none"><li>• Computer with the operating system of your choice, with the Google Chrome or Chromium browser for videoconferencing.</li><li>• Webcam and microphone (preferably from an audio headset)</li><li>• High speed access to the Internet</li></ul>
<b>Certificate</b>	Only the participants who have attended all training sessions, and who have scored over 50% of correct answers at the final evaluation will receive a training certificate from Bootlin.
<b>Disabilities</b>	Participants with disabilities who have special needs are invited to contact us at <a href="mailto:training@bootlin.com">training@bootlin.com</a> to discuss adaptations to the training course.



## Hardware, first option

### BeagleBone Black board

- An ARM AM335x processor from Texas Instruments (Cortex-A8 based), 3D acceleration, etc.
- 512 MB of RAM
- 2 GB of on-board eMMC storage (4 GB in Rev C)
- USB host and device
- HDMI output
- 2 x 46 pins headers, to access UARTs, SPI buses, I2C buses and more.



## Hardware, second option

One of these Discovery Kits from STMicroelectronics: **STM32MP157A-DK1**, **STM32MP157D-DK1**, **STM32MP157C-DK2** or **STM32MP157F-DK2**

- STM32MP157 (dual Cortex-A7) CPU from STMicroelectronics
- USB powered
- 512 MB DDR3L RAM
- Gigabit Ethernet port
- 4 USB 2.0 host ports
- 1 USB-C OTG port
- 1 Micro SD slot
- On-board ST-LINK/V2-1 debugger
- Arduino Uno v3-compatible headers
- Audio codec
- Misc: buttons, LEDs
- LCD touchscreen (DK2 kits only)





## Half day 1

---

### Lecture - Introduction to embedded Linux build systems

- Overview of an embedded Linux system architecture
- Methods to build a root filesystem image
- Usefulness of build systems

### Lecture - Yocto Project and Poky reference system overview

- Introduction to the Yocto / OpenEmbedded build system and its lexicon
- Overview of the Poky reference system

### Lecture - Using Yocto Project - basics

- Setting up the build directory and environment
- Configuring the build system
- Building a root filesystem image

### Demo 1 - First Yocto Project build

- Downloading the Poky reference build system
- Configuring the build system
- Building a system image

### Lecture - Using Yocto Project - basics

- Organization of the build output

### Demo 1 - Flashing and booting

- Flashing and booting the image on the board



## Half day 2

---

### Lecture - Using Yocto Project - advanced usage

- Variable assignment, operators and overrides
- Package variants and package selection
- bitbake command line options

### Demo 2 - Using NFS and configuring the build

- Configuring the board to boot over NFS
- Add a package to the root filesystem
- Learn how to use the PREFERRED\_PROVIDER mechanism
- Get familiar with the bitbake command line options

### Lecture - Writing recipes - basics

- Recipes: overview
- Recipe file organization
- Applying patches
- Recipe examples

### Demo 3 - Adding an application to the build

- Writing a recipe for *nInvaders*
- Troubleshooting the recipe
- Troubleshooting cross-compilation issues
- Adding *ninvaders* to the final image

### Lecture - Writing recipes - advanced features

- Extending and overriding recipes
- Virtual packages
- Learn about classes
- BitBake file inclusions
- Debugging recipes
- Configuring BitBake network usage



## Half day 3

---

### Lecture - Layers

- What layers are
- Where to find layers
- Creating a layer

### Demo 4 - Writing a layer

- Learn how to write a layer
- Add the layer to the build
- Move *ninvaders* to the new layer

### Demo 5 - Extend a recipe

- Extend the kernel recipe to add patches
- Configure the kernel to compile the nunchuk driver
- Edit the *ninvaders* recipe to add patches
- Play *nInvaders*

### Lecture - Writing a BSP

- Introduction to BSP layers
- Adding a new machine
- Bootloader configuration
- Linux: the kernel *bbclass* and the *linux-yocto* recipe

### Demo 6 - Create a custom machine configuration

- Create a new machine configuration
- Build an image for the new machine

### Lecture - Distro layers

- Distro configuration
- Distro layers



### Lecture - Images

- Writing an image recipe
- Image types
- Writing and using package groups recipes

### Demo 7 - Create a custom image

- Add a basic image recipe
- Select the image capabilities and packages
- Add a custom package group
- Add an image variant for debugging

## Half day 4

---

### Lecture - Writing recipes - going further

- The per-recipe sysroot
- Using Python code in metadata
- Variable flags
- Packages features and PACKAGECONFIG
- Conditional features
- Package splitting
- Dependencies in detail

### Lecture - Licensing

- Managing open source licenses

### Lecture - The Yocto Project SDK

- Goals of the SDK
- Building and customizing an SDK
- Using the Yocto Project SDK

### Demo 8 - Develop your application in the Poky SDK

- Building an SDK
- Using the Yocto Project SDK

### Lecture - Devtool

- About devtool
- Devtool use cases

### Demo 9 - Using devtool

- Generate a new recipe
- Modify a recipe to add a new patch
- Upgrade a recipe to a newer version



### Lecture - Automating layer management

- Automating layer management

### Lecture - Runtime Package Management

- Introduction to runtime package management
- Build configuration
- Package server configuration
- Target configuration

### Questions and Answers

- Questions and answers with the audience about the course topics
- Extra presentations if time is left, according what most participants are interested in.

## Possible extra time

---

*Extra time (up to 4 hours) may be proposed if the agenda didn't fit in 4 half days, according to the time spent answering questions from participants.*