The Yocto Project - A thorough Overview

Robert Berger - Reliable Embedded Systems e.U.  
Consulting Training Engineering  
https://www.ReliableEmbeddedSystems.com  
robert.berger@ReliableEmbeddedSystems.com

Course Objectives

To provide an understanding of the essentials to utilize the Yocto Project on top of your firm knowledge about Embedded Linux and the Linux kernel. We’ll see how a BSP/framework maintainer would use the Yocto Project as well as developers who might not even want/need to know they are using it.

Description

This four day training combines theory with hands-on exercises in order to introduce the Yocto Project. It answers frequently asked questions like:

/> What is the Yocto Project and what is the OE/Yocto workflow?
/> What is BitBake?
/> What are Layers?
/> Is it really necessary to use another version of the toolchain/libraries/packages for each and every Embedded Linux project and an top of that to follow a different work-flow?
/> Can you ensure that the development environment is identical for all developers/suppliers and that you can still produce identical builds like today in 10+ years from now?
/> Can the YP help find out under what software licenses the packages you use are licensed?
/> ... and much more

Hands-on sessions are performed on the host, in a docker container and on some target hardware (e.g. i.mx6 [1]). After the training you will be able to download a docker container based on Ubuntu with all dependencies pre-installed plus the examples in order to work with the course material in your own labs. Please note that this is not an introductory course to Embedded Linux like the one we offer here [2]. You should already know how Embedded Linux works and how to configure/build the Linux kernel and kernel drivers.

Prerequisites

/> Basic familiarity with using a Linux system (e.g. Ubuntu) as an end user in user space
/> Basic familiarity with a command line shell
/> Basic knowledge of user/kernel space programming with Linux
/> Intermediate C programming knowledge
You should have attended “Embedded Linux - From Systems Architecture to Real-Time (5 days)” [2] or “Introduction to Embedded Linux in Theory and Practice - a Crash Course (3 days)” [3]. (strongly recommended) ...except if you are already familiar with the material presented in those courses. You actually need to have experience with Embedded Linux (kernel, userspace, root file-system, tools) to follow this training.

It might be helpful if you attended “Embedded Linux Kernel Internals and Device Drivers (5 days)” [4], but that’s not really a prerequisite. It’s sufficient to know how to build the Linux kernel, in/out of tree kernel drivers and the fdt to follow the BSP and kernel chapter of this training.

The training sessions mentioned above can be delivered on-site and on-line. Feel free to ask for training dates and pricing here [5].

Who should attend?

You already use Linux for your projects and have probably heard about the Yocto Project, but did not dare to have a closer look into it, or had difficulties using it. You don’t know whether and how your daily workflow can be accommodated in the YP and generally find the YP rather complicated. Why do we need all this since up to now everything was (supposedly) much easier? After the training you should be able to decide whether you need the YP or not. The workshop is aimed at software-, development-, system engineers, testers, administrators, engineers and other parties interested in the YP, with a solid knowledge of Embedded Linux.

Delivery options

All training material is in English, but the delivery of it can be in English or in German, as you wish, worldwide. During the training we’ll provide:

<table>
<thead>
<tr>
<th>public (for 1 trainee) on-site (from 2 trainees) instructor driven</th>
<th>on-line instructor driven</th>
<th>on-site/on-line combination instructor driven</th>
</tr>
</thead>
<tbody>
<tr>
<td>workbook</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>in English/pdf printed on request</td>
<td>in English/pdf printed on request</td>
<td></td>
</tr>
<tr>
<td>host machine</td>
<td>✓ <em>laptop</em></td>
<td>✓ <em>hosted vm or laptop</em></td>
</tr>
<tr>
<td>target board</td>
<td>✓ <em>hosted</em></td>
<td></td>
</tr>
<tr>
<td>remote access to host machine and board</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>(web) screen-sharing</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>(web) audio/video</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>(web) whiteboard</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>(web) chat</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>(web) Q&amp;A</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>(phone) audio-bridge</td>
<td>✓</td>
<td>✓</td>
</tr>
</tbody>
</table>

For possible training delivery options check here [6] and/or ask here [5]. The workbooks typically stay with the students after the training. We normally place two trainees on one workspace so they can help each other.
Course Outline

Day 1

Introduction

/> History of Unix/Linux | Licensing | Standards | Unix Philosophy

Embedded Specifics

/> Linux on the desktop compared to Linux in an embedded system | cross-/native toolchains | pre-built toolchains | build systems | C libraries | ...

Eval Board

/> How does Linux boot on a PC and on the Eval Board? | See it booting

Yocto Introduction

/> What is Yocto? | What is the YP? (features/challenges) | Some tools under the YP umbrella (Poky | BitBake | OE-Core | Metadata) | Why use the YP?

How to become part of our Yocto Project community (optional)

/> First steps | How to get in touch | Participate | Contribute | Social Media | Events

The YP Autobuilder

/> What is the YP Autobuilder? | Docker container (pull | launch container) | No docker - no YP-AB

The YP Workflow

/> Intro

/> Workflow | OE architecture

/> Configuration (User | Metadata(Recipes) | Machine(BSP) | Distribution Policy)

/> Features | Recipe Versioning | Layers

/> Sources

/> Source fetching | Patching | Configure/Compile/Staging | SSTATE | Pseudo | recipetool | Examples of Recipes | PACKAGECONFIG | Packages and their contents | Output analysis/Packaging | Image Generation | SDK Generation | Tasks

/> Customizing Images (Intro | local.conf | IMAGE_FEATURES | custom image recipes (.bb files) | custom packagegroups)

Day 2

BitBake

/> History

/> Syntax (Variable Expansion | Variable Assignment | Pre-/-Append | Removal | Variable Flags | Conditional Syntax - OVERRIDES)
BitBake Debugging (debug level | find recipes/images/packagegroups | BitBake environment/tasks/logging | force build/specific task | cleansstate | invalidate stamp | devshell | dependencies | packages | kill all BitBake instances | BitBake graphical wrapper)

Cleaning (gain disk space | rebuild)

Layers

Intro | bitbake-layers tool | dynamic layers

BSP

Intro | System Development Workflow | BSP Developer’s Guide (bsp-tool - ported to recent Poky versions) | BSP creation | non-mainline kernel patches

Kernel

Intro | System Development Workflow | Kernel Development Manual (defconfig | defconfig + configuration fragment | in tree kmod | out of tree kmod | fdt classic | fdt with devicetree.bbclass | ...

Day 3

Software Development Kit

Intro | Cross-Development Toolchain | Sysroot | BBCLASSEXTEND | Multilib | The QEMU Emulator | SDK: Eclipse Yocto Plug-in (deprecated) | User Space Tools | Installing SDKs & toolchains

Cross-Toolchains/SDKs

Building a Cross-Toolchain installer

Using the Standard SDK (Cross-Toolchain + Makefile/Autotools/Autotools lib + App | recipes)

Building/Using the Extensible SDK

Libraries

.so naming convention | What goes where (rootfs/SDK)? | How to use an autotooled .a with poky?

Eclipse (deprecated/optional)

Intro | Application Development Workflow

Working with Eclipse (rootfs features | kernel + fdt | rootfs | install JDK | install Eclipse | configure Eclipse | install Yocto Plugin | Configure the Yocto Plugin | Create "Autotooled Project" | Configuring the Cross-Toolchains | Build the Project)

Day 4

User Space Debugging

Intro | gdb | gdb remote debugging | (gdb remote) Debugging with Eclipse | (remote) Run from Eclipse
Profiling/Tracing

/> Intro | perf | gprof | gcov | strace | ftrace | systemtap | LTTng + Eclipse (data visualization) | top | powertop | powerdebug

Package Management

/> SW update vs. Package Management | Working with Packages | IPK | creating a package feed | installing a package with opkg on the target

Licensing

/> Intro | Add custom license | Add commercial license | Firmware License | Open Source License Compliance

Devtool

/> Intro | Add recipe/Build/Deploy | Create/Add layer | Finish | Modify/Update-Recipe | Build/Run | Build Image

Related Courses

/> “Embedded Linux - From Systems Architecture to Real-Time (5 days)”
http://rlbl.me/elisa

/> “Refresher to Embedded Linux & Intro to the Yocto Project (5 days)”
http://rlbl.me/intely

/> “Embedded Linux Kernel Internals and Device Drivers (5 days)”
http://rlbl.me/ldd

/> “The Yocto Project - A thorough Overview (4 days)”
http://rlbl.me/yocto

/> “Introduction to Embedded Linux & Real-Time, bird’s eye view of the Yocto Project (4 days)”
http://rlbl.me/entirety

/> “Embedded Linux Hardware Interfacing (4 days) - coming soon”
http://rlbl.me/elisha

/> “Compact Linux Driver development (4 days) - coming soon”
http://rlbl.me/coldl

/> “Introduction to Embedded Linux in Theory and Practice - a Crash Course (3 days)”
http://rlbl.me/elin

/> “(Embedded) Linux debugging (3 days)”
http://rlbl.me/linde

/> “FreeRTOS in Theory and Practice (3 days)”
http://rlbl.me/freertos

/> “The Zephyr Project - An Overview (3 days) - coming soon”
http://rlbl.me/zephyr
References

[1] “Target Hardware”
http://rlbl.me/hw

http://rlbl.me/elisa

[3] “Introduction to Embedded Linux in Theory and Practice - a Crash Course (3 days)”
http://rlbl.me/elin

[4] “Embedded Linux Kernel Internals and Device Drivers (5 days)”
http://rlbl.me/ldd

http://rlbl.me/contact

http://rlbl.me/delivery

Trainer

Since 1993, Robert Berger gathered practical and managerial experience in software design and development for embedded systems with and without hard real-time requirements. Since the beginning of the 21st century, he has used Linux on desktop and server class machines, but mainly for embedded practices (automotive, industrial control, robotics, telecoms, consumer electronics, etc.). Robert regularly attends international events such as “Embedded World”, “Embedded Software Engineering Kongress”, “Embedded Systems Conference”, “Embedded Linux Conference” and “Yocto Project Summit” as an expert and lecturer. His specialty is mainly training, but also consulting (in German or English) worldwide. Robert’s expertise ranges from small real-time systems (FreeRTOS) to systems with multiple processors/cores and embedded Linux (user-, kernel-space, device drivers, hardware interfacing, debugging, multi-core, Yocto Project) with a focus on free and open source software. Robert is a globe-trotter. He is CEO & Embedded Software Evangelist at Reliable Embedded Systems e.U. which is based in St. Barbara, Austria, and when not on business trips, lives with his family in Athens, Greece.

Thank you for your interest!

For inquiries please send an email to:
training@ReliableEmbeddedSystems.com