



oY1 - Yocto Project Development

Building a Linux Embedded image using Yocto

Objectives

- Using and customizing Yocto
- Creating Yocto-based Embedded Linux platforms
- Using Yocto to develop components

Labs are conducted QEMU ARM-based board

We use a recent version of Yocto

Prerequisite

- Good C programming skills
- Knowledge of Linux Embedded systems (see our [oD1 - Embedded Linux](#) course)
- Preferably knowledge of Linux user programming (see our [oD0 - Linux User Mode Programming](#) course)
- You may be interested also by the [Yocto Expert](#) course or the combined ([oY12 - Comprehensive Yocto Project Usage](#) course)

Course environment

- Theoretical course
 - PDF course material (in English)
 - Course dispensed using the Teams video-conferencing system
 - The trainer to answer trainees' questions during the training and provide technical and pedagogical assistance through the Teams video-conferencing system
- Practical activities
 - Practical activities represent from 40% to 50% of course duration
 - One Online Linux PC per trainee for the practical activities
 - The trainer has access to trainees' Online PCs for technical and pedagogical assistance
- Downloadable preconfigured virtual machine for post-course practical activities

Duration

- Total: 18 hours
- 3 sessions, 6 hours +/-30 min each (excluding break time)
- From 40% to 50% of training time is devoted to practical activities
- Some Labs may be completed between sessions and are checked by the trainer on the next session

Target Audience

- Any embedded systems engineer or technician with the above prerequisites.

Course Outline

Session 1

Introduction to Yocto

- Overview of Yocto
 - History
 - Yocto, Open Embedded and Poky
 - Purpose of the Yocto project
 - The main projects
- Yocto architecture
 - Overview
 - Recipes and classes
 - Tasks

The Yocto build system

- Build system objectives
 - Building deployable images
 - Layers and layer priorities
 - Directory layout
 - Configuration files (local, machine and distribution)
 - The bitbake tool
- Using Yocto
 - Building a package
 - Building an image (root file system + u-boot + kernel)
- Miscellaneous tools around Yocto
 - Yocto SDK
 - Extensible SDK

Exercise: Building a root file system using Yocto

Exercise: Use bitbake commands to build package & images

Exercise: Building a root file system using Yocto

Exercise: Build an extensible SDK for the generated image

Exercise: Deploy the generated image

Yocto package recipes structure

- Recipe architecture
 - Tasks
 - Task dependencies
 - Recipe dependencies
- The bitbake language
 - Standard variables and functions
 - Classes and recipes
 - The base Yocto classes
 - Main bitbake commands
- Adding a new layer
 - Layer structure
 - Various kinds of layers

Exercise: Adding a new layer

Session 2

Writing package recipes for Yocto

- Various kind of recipes and classes
 - Bare program
 - Makefile-based package
 - autotools-based package
 - u-boot
 - kernel
 - Out-of-tree module
- Recipe creation strategies
 - From scratch
 - Using devtool
 - Using recipetool
 - From an existing, similar, recipe
- Debugging recipes
 - Debugging recipe selection
 - Debugging dependencies
 - Debugging tasks
- Defining packaging
 - Package splitting
- Automatically starting a program

Exercise: Writing a recipe for a local user-maintained package

Exercise: Writing and debugging a package recipe for an autotools-based package

Exercise: Starting a program at boot (systemd)

Modifying recipes

- Customizing an existing package recipe (.bbappend)
- Recipe dependencies
- Creating and adding patches
 - Creating a patch for a community-provided component
 - Creating a patch for an user-maintained component
- Defining new tasks
 - Task declaration
 - Coding tasks

Exercise: Adding patches and dependencies to a community package

Exercise: Adding a rootfsinstall task to directly copy the output of a user package in the rootfs image

Session 3

Creating new kinds of recipes

- Creating classes
 - Creating new independent classes
 - Inheriting from an existing class

Exercise: Create a class to generalize the “rootfsinstall” task

Creating a root file system

- Building a root file system with Yocto
 - Creating a custom root file system
- Writing an image recipe
 - Selecting the packages to build
 - Selecting the file system types

- The various kinds of images
- Inheriting and customizing images
 - Customizing system configuration files (network, mount points, ...)
- Users and groups management
- Package management
 - rpm
 - opkg

Exercise: Writing and building an image recipe

Exercise: Add new users to the image

Exercise: Create an image with package support for OTA deployment

Exercise: Test OTA update on the generated image