OUTLINE

• Why devtool?
• Overview of how devtool works
• Types of Projects Currently Supported
• Most Common devtool Commands
• devtool is Evolving and Improving
WHY DEVTOOL?

Workflow before:

Fire up your trusty editor
Figure out where the recipe is or should be
Copy/Paste errors.
What is the minimum for a valid recipe?
What was that VARIABLE_NAME?

How do I do a md5sum in my editor?
What should I inherit? require? DEPENDS? RDEPENDS?
Oops, I forgot to commit that.
Darn, I should have created a new layer
I just want to build it!
I just want to deploy it!
WHY DEVTOOL?

Workflow after:

- Fire up your trusty editor
- Figure out where the recipe is or should be
- Copy/Paste errors.
- What is the minimum for a valid recipe?
- What was that VARIABLE_NAME?
- How do I do a md5sum in my editor?
- What should I inherit? require? DEPENDS? RDEPENDS?
- Oops, I forgot to commit that.
- Darn, I should have created a new layer
- I just want to build it!
- I just want to deploy it!
OVERVIEW OF HOW DEVTOOL WORKS

devtool add http://<site>/hello-2.10.tar.gz

parsed cached metadata
depends
rdepends
require

recipetooll create

workspace
appends
recipes

hello

hello_2.10.bb

sources

hello

.git

Makefile.am

src

inhibit autotools

built package data

workspace

README

conf

layer.conf

devtool create-workspace workspace

TYPES OF PROJECTS CURRENTLY SUPPORTED

- Autotools (**autoconf** and **automake**)
- Cmake
- **qmake**
- Plain **Makefile**
- Out-of-tree kernel module
- Binary package (i.e. “-b” option)
- Node.js* module
- Python modules that use **setuptools** or **distutils**
**MOST COMMON DEVTOOL COMMANDS**

**devtool add**
- Create a new recipe

**devtool modify**
- Modify the source built by a recipe

**devtool upgrade**
- Upgrade a recipe to a newer version
Devtool has great self-documentation

$ devtool add --help

usage: devtool add [-h] [--same-dir | --no-same-dir] [--fetch URI]
[--version VERSION] [--no-git] [--autorev] [--binary][--also-native]
[--src-subdir SUBDIR] [recipename] [srctree] [fetchuri]

Adds a new recipe to the workspace to build a specified source tree. Can optionally fetch a remote URI and unpack it to create the source tree.

arguments:

recipename  Name for new recipe to add (just name - no version, path or extension). If not specified, will attempt to auto-detect it.

srctree      Path to external source tree. If not specified, a subdirectory of /<workdir>/workspace/sources will be used.

fetchuri    Fetch the specified URI and extract it to create the source tree!
WHY CREATE RECIPES FROM SCRATCH?

Do you have extra time to spare?
DEMO #1

devtool add

A complete workflow from start to finish
WHY MODIFY SOURCE CODE WITH QUILT?

Do you have extra time to spare?
DEMO #2
devtool modify
A simple example
WHY UPGRADE RECIPES BY HAND?

Do you have extra time to spare?
DEMO #3

devtool upgrade

A real layer maintainance workflow example
DEVTOOL IS EVOLVING AND IMPROVING

- **Fido (1.8)**
  - Introduced

- **Jethro (2.0)**
  - Improved

- **Krogoth (2.1)**
  - True Timesaver

- **Morty (2.2)**
  - Refined

- **Pyro (2.3)**
  - Polished

- **2.4+**
  - Possibilities!
  - Your idea

Devtool is evolving and improving.
Cross-Platform Enablement for the Yocto* Project with Containers

- Randy Witt, Intel OTC
- http://sched.co/9Itu

Yocto* Project Extensible SDK: Simplifying the Workflow for Application Developers

- Henry Bruce, Intel OTC
- http://sched.co/9Itz
CALL TO ACTION

• Use the tool whenever practical
• Contribute!
  • devtool is part of OE-Core
  • openembedded-core@lists.openembedded.org
  • http://lists.openembedded.org/mailman/listinfo/openembedded-core
2.4. Using `devtool` in Your SDK Workflow

The cornerstone of the extensible SDK is a command-line tool called `devtool`. This tool provides a number of features that help you build, test and package software within the extensible SDK, and optionally integrate it into an image built by the OpenEmbedded build system.

The `devtool` command line is organized similarly to Git in that it has a number of sub-commands for each function. You can run `devtool --help` to see all the commands.

Three `devtool` subcommands that provide entry-points into development are:

- `devtool add`: Assists in adding new software to be built.
- `devtool modify`: Sets up an environment to enable you to modify the source of an existing component.
- `devtool upgrade`: Updates an existing recipe so that you can build it for an updated set of source files.

GET MORE INFO

https://wiki.yoctoproject.org/wiki/Developer_Workflow_Improvements
THANK YOU
Paul Eggleton, Chris Larson, Leo Sandoval and others
Henry Bruce
Todor Minchev, Randy Witt and Brian Avery
QUESTIONS?
Disclaimer

Intel 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.  
© Intel Corporation