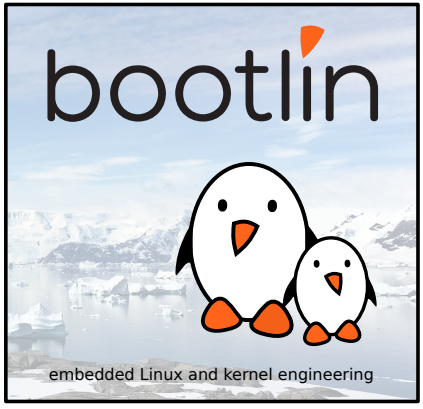
Keep your layer simple – and here's how

Luca Ceresoli
luca.ceresoli@bootlin.com



Luca Ceresoli

- Embedded Linux engineer at **Bootlin**
  - Embedded Linux experts
  - Engineering services: Linux BSP development, kernel porting and drivers, Yocto/Buildroot integration, real-time, boot-time, security, multimedia
  - Training services: Embedded Linux, Linux kernel drivers, Yocto, Buildroot, graphics stack, boot-time, real-time, debugging, audio
- Linux kernel and bootloader development, Buildroot and Yocto integration
- Open-source contributor
- Living in **Bergamo**, Italy
- luca.ceresoli@bootlin.com

https://bootlin.com/company/staff/luca-ceresoli/

Keep your layer simple – and here's how

Great power = great responsibility

Great power = great responsibility

- Bitbake is very powerful and flexible
  - Layers, `.bbappends`, overrides...
  - Lots of ways to modify the build system behaviour
- A very useful feature... but open to misuse
- A desired effect can be obtained in lots of different ways
  - Most of which are wrong is some way
- The “correct”, clean way is not always easy to find
  - Copy&paste from the Internet without understanding opens to bad surprises
  - https://docs.yoctoproject.org/ is better
  - Investing some time in learning is necessary

A real world example

Real Yocto/OpenEmbedded setup of a customer

- 20 layers
  - Including: 2 customer layers, 2 SoC vendor layers, 2 SoM vendor layers, meta-mingw
  - And an additional repository with the git repo manifest

Real Yocto/OpenEmbedded setup of a customer

- 20 layers
  - Including: 2 customer layers, 2 SoC vendor layers, 2 SoM vendor layers, meta-mingw
  - And an additional repository with the git repo manifest
- 4000+ lines in `.bbappend` files in non-core layers
  - Largely unneeded, sometimes hurting
  - Time spent in discovering the origin of issues

Real Yocto/OpenEmbedded setup of a customer

- 20 layers
  - Including: 2 customer layers, 2 SoC vendor layers, 2 SoM vendor layers, meta-mingw
  - And an additional repository with the git repo manifest
- 4000+ lines in `.bbappend` files in non-core layers
  - Largely unneeded, sometimes hurting
  - Time spent in discovering the origin of issues
- As of October 2022 based on Yocto sumo, released April 2018
  - Does not build on recent distros, needs a container
  - Custom scripts to set it up
  - Using a vendor kernel based on 4.9 (December 2016), almost EOL
  - `linux-backports` for drivers
  - Several recipes copied and adapted from more recent releases

Real Yocto/OpenEmbedded setup of a customer

- 20 layers
  - Including: 2 customer layers, 2 SoC vendor layers, 2 SoM vendor layers, meta-mingw
  - And an additional repository with the git repo manifest
- 4000+ lines in `.bbappend` files in non-core layers
  - Largely unneeded, sometimes hurting
  - Time spent in discovering the origin of issues
- As of October 2022 based on Yocto sumo, released April 2018
  - Does not build on recent distros, needs a container
  - Custom scripts to set it up
  - Using a vendor kernel based on 4.9 (December 2016), almost EOL
  - `linux-backports` for drivers
  - Several recipes copied and adapted from more recent releases
- Attempts to upgrade broke lots of things

Real Yocto/OpenEmbedded setup of a customer

- 20 layers
  - Including: 2 customer layers, 2 SoC vendor layers, 2 SoM vendor layers, meta-mingw
  - And an additional repository with the git repo manifest
- 4000+ lines in `.bbappend` files in non-core layers
  - Largely unneeded, sometimes hurting
  - Time spent in discovering the origin of issues
- As of October 2022 based on Yocto sumo, released April 2018
  - Does not build on recent distros, needs a container
  - Custom scripts to set it up
  - Using a vendor kernel based on 4.9 (December 2016), almost EOL
  - `linux-backports` for drivers
  - Several recipes copied and adapted from more recent releases
- Attempts to upgrade broke lots of things
- 1 product

The proposed solution

Can't fix it? Throw it!

- Write a new setup from scratch
- Based on the latest Yocto LTS (Kirkstone)
- Using only the *really* useful layers
- Using recent mainline Linux and U-Boot
  - Product based on i.MX6, well supported
  - Send to mainline any additional patches needed

The result

- 5 layers: oe-core, meta-oe, meta-qt5, meta-rauc (new!), meta-*company-name*
  - meta-*company-name* total: 60 files, 3150 lines
  - Recipe to build mainline Linux: 29 lines using `kernel.bbclass`
  - Recipe to build mainline U-Boot: 25 lines using `u-boot.inc` from oe-core
  - Recipe to build product Device Tree: 5 lines using `devicetree.bbclass`
- Image size: less than half
- 1 git repository, including `kas` config file
- Quick and easy to set up *without custom scripts*
- Builds without containers on modern distributions

The result

- 5 layers: oe-core, meta-oe, meta-qt5, meta-rauc (new!), meta-*company-name*
  - meta-*company-name* total: 60 files, 3150 lines
  - Recipe to build mainline Linux: 29 lines using `kernel.bbclass`
  - Recipe to build mainline U-Boot: 25 lines using `u-boot.inc` from oe-core
  - Recipe to build product Device Tree: 5 lines using `devicetree.bbclass`
- Image size: less than half
- 1 git repository, including `kas` config file
- Quick and easy to set up *without custom scripts*
- Builds without containers on modern distributions
- Customer happy

Keep your layer simple – and here's how

Cool!
How can I do it?

Keep your layer simple – and here's how

Cool!
How can I do it?
Here's how!

Keep your layer simple...

- We wrote a fictitious but realistic YP/OE setup for a product company
  - To act as an example for companies making products
  - Not for BSP/Distro/Software layers
- Clean
  - 1 layer
  - 2 (fictitious) products
  - 1 distro
  - A few recipes: mainline Linux and U-Boot and a userspace application
  - Two `.bbappend` files for fixes

Keep your layer simple...

- Goal: provide a practical example of how to cleanly achieve common needs
  - Fetching layers
  - Writing clean, simple, maintainable recipes
  - Use overrides
  - Use mysterious things such as `MACHINE_ESSENTIAL_EXTRA_RRECOMMENDS`
  - Have distro, machine and image settings in the proper place
  - Avoid too many 3rd-party layers (cost/benefit ratio)
  - Keep it simple!

...and here's how

https://github.com/bootlin/simplest-yocto-setup

Questions? Suggestions? Comments?

Luca Ceresoli
luca.ceresoli@bootlin.com