Nicolas Dechesne, Linaro
Yocto Project Community Manager

Yocto Project® Documentation migration to Sphinx

Yocto Project Virtual Summit Europe, October 29-30, 2020
About me!

- Yocto Project® Community Manager
- Engineering manager at Linaro
- Maintainer of meta-qcom BSP layer, and a few other Linaro OE layers
- Linaro representative at Yocto Project Board
- Involved with Sphinx migration *by accident*
Agenda

- Overview of Yocto Project Docs
- Why change?
- Why Sphinx?
- Timeline
- Tour of the Sphinx based docs
- How can you help?
Yocto Project Documentation

- A *conscious* priority for the Yocto Project, since its inception, to release professional quality documentation
- A pillar of the Yocto Project community throughout the years
- Documentation mostly contributed by Scott Riftenbark
Why change?

- DocBook semantics and tools usually preferred by tech writers, book authors
- However seen as a barrier to entry for new contributors (our developers)
- Discussed at one weekly Technical meeting
- Led to an initial discussion on our mailing list, and quickly some really positive feedback
- LWN article on Kernel migration to Sphinx
  "if the documentation can be written in plain text rather than DocBook XML, it's more likely to be written in the first place"
High level objectives

- Ensure documentation source is 'easier' to read and write
- Encourage more contributors
- Simple to manage and use
- Support many output types (html, pdf, ...)
- Preserve reputation of the quality of the project's documentation.
Why Sphinx?

- Choice of lightweight markup language of some sort was obvious
- Studied differences between Sphinx, Markdown, Asciidoc, ... and studied other open source projects
- Initial plan was to use Asciidoc, but Sphinx/reST was preferred:
  - This is what Python uses, and our project is intimately linked to Python.
  - Sphinx can be extended with plugins in Python
  - Successful migration for the Linux kernel community. Also well documented: https://lwn.net/Articles/671496/, https://lwn.net/Articles/692704/, https://lwn.net/Articles/692705/
Timeline

Idea
Discussed at weekly call
Tossing around with Sphinx and various other markdown tools

April
We have a plan for YP and Bitbake docs!
Initial proof of concept was shown. Semi automatic conversion with Pandoc + initial structure, guidelines and templates

May
We have a plan for YP and Bitbake docs!

'Sphinx' branch available
Dev branches hosted in yocto-docs and bitbake repos. Initial proof of concept ‘rebased’ and refreshed.

June
Dev branches hosted in yocto-docs and bitbake repos. Initial proof of concept ‘rebased’ and refreshed.

July
Hacking, Hacking!
Working on the conversion, extensions, learning its way with Sphinx.

August
Getting there!
CSS, figures, index, intersphinx, version switcher and mega manual, unified docs website for new and old releases

September
Merged!
Intensive hacking for the last mile. Sphinx conversion merged in dev branch. Launched docs.yoctoproject.org. Plan proposed to backport to 3.1 LTS.

October
Yocto Project docs website

- [https://docs.yoctoproject.org/](https://docs.yoctoproject.org/)
- New, unified website site dedicated to all Yocto Project documentation
  - imported all documentations from all releases
  - uniform look and feel for legacy docs vs Sphinx based docs
  - inspired from [https://docs.python.org/](https://docs.python.org/)
- [https://www.yoctoproject.org/docs/](https://www.yoctoproject.org/docs/) will redirect to new website (soon)
- CI/CD for docs:
  - [https://docs.yoctoproject.org/](https://docs.yoctoproject.org/) *(master and transition branch)*
  - [https://docs.yoctoproject.org/next/](https://docs.yoctoproject.org/next/) *(master-next branch)*
  - [https://docs.yoctoproject.org/gatesgarth/](https://docs.yoctoproject.org/gatesgarth/) *(gatesgarth branch)*
  - [https://docs.yoctoproject.org/3.1.3/](https://docs.yoctoproject.org/3.1.3/) *(legacy release)*
  - [https://docs.yoctoproject.org/3.2/](https://docs.yoctoproject.org/3.2/) *(release)*
Overview of Sphinx docs

- **Yocto Docs files:**
  - 208 files changed, 100194 deletions(-)
  - 110 files changed, 57006 insertions(+)

- **Bitbake Docs files:**
  - 25 files changed, 11898 deletions(-)
  - 13 files changed, 6363 insertions(+)

- Preserved (for now) the directory structure for each manual

- [README](#)
- [conf.py](#)

```
$ pip3 install sphinx sphinx_rtd_theme pyyaml
$ cd documentation
$ make html
```
reStructuredText (reST), Docutils, Sphinx

- **reST** is the plaintext markup language
  - paragraph, lists, sections, ...
  - roles
  - directives
- **Docutils** provides the “tools” to process reST into a “tree of nodes”
- **Sphinx** is the documentation builder
  - uses docutils output
  - extends reST to support additional functionality such as extensions, theming, ...
  - top level user command line tools
Extensions

- **Key concept in Sphinx**
  - many extensions provided by Sphinx for added features
  - many extensions exist out there (contrib)
  - ‘simple’ to write custom extensions in a Sphinx project, as Python module.

- **Hooked to an ‘event’ during Sphinx build phases:**
  - Init/config
  - Read all input files
  - Checks
  - Resolving/transforms
  - Writing
Design / implementation details

- Defined **sections/headings layout**, to be applied in all docs
- Built-in support for figures, images, tables
- **Literal blocks** and **code blocks** (highlights with Pygments)
- Built-in **glossary**
  - general index automatically generated
  - :term:`PACKAGES`
- Custom **substitution extension**
  - shortcomings of built-in subst mechanism
  - Based on poky.ent, converted to yaml
  - git checkout &DISTRO_NAME_NO_CAP;
Design / implementation details

- Links and references with `autosectionlabel` extension
  - :ref:`References in the Same Document`
  - Please check :ref:`section <References in the Same Document>`
  - :ref:`overview-manual/development-env:source repositories`

- Intersphinx extension to link Yocto Project and Bibtake cross references
  - See "`:ref:`-D <bitbake:bitbake-user-manual/intro:usage and syntax>`" option
  - :term:`bitbake:BB_NUMBER_PARSE_THREADS`

- Links to **external website**, such as :yocto_wiki:, :yocto_git:...
  - Please check this :yocto_wiki:`wiki page </Weekly_Status>`
Welcome to the Yocto Project Documentation

Introduction and Overview

- Quick Build
- What I wish I’d known about Yocto Project
- Transitioning to a custom environment for systems development
- Yocto Project Software Overview
- Tips and Tricks Wiki

Manuals

- Overview and Concepts Manual
- Reference Manual
- Board Support Package (BSP) Developer’s guide
- Development Tasks Manual
- Linux Kernel Development Manual
- Profile and Tracing Manual
- Application Development and the Extensible SDK (eSDK)
- Toaster Manual
- Test Environment Manual
- Bitbake User Manual

'Mega' Manual

- All-in-one 'Mega' Manual

Manuals/Variable Index
3.1 Understanding and Creating Layers

The OpenEmbedded build system supports organizing Metadata into multiple layers. Layers allow you to isolate different types of customizations from each other. For introductory information on the Yocto Project Layer Model, see the "The Yocto Project Layer Model" section in the Yocto Project Overview and Concepts Manual.

3.1.1 Creating Your Own Layer

It is very easy to create your own layers to use with the OpenEmbedded build system. The Yocto Project ships with tools that speed up creating layers. This section describes the steps you perform by hand to create layers so that you can better understand them. For information about the layer-creation tools, see the "Creating a new BSP Layer Using the bitbake-layers Script" section in the Yocto Project Board Support Package (BSP) Developer's Guide and the "Creating a General Layer Using the bitbake-layers Script" section further down in this manual.

Follow these general steps to create your layer without using tools:

1. **Check Existing Layers**: Before creating a new layer, you should be sure someone has not already created a layer containing the Metadata you need. You can see the OpenEmbedded Metadata Index for a list of layers from the OpenEmbedded community that can be used in the Yocto Project. You could find a layer that is identical or close to what you need.

2. **Create a Directory**: Create the directory for your layer. When you create the layer, be sure to create the directory in an area not associated with the Yocto Project Source Directory (e.g. the cloned poky repository).

While not strictly required, prepend the name of the directory with the string "meta-". For example:

- meta-my-layer
- meta-GUI_xyz
- meta-my-machine

With rare exceptions, a layer’s name follows this form:

- meta-root_name

Following this layer naming convention can save you trouble later when tools, components, or variables “assume” your layer name begins with “meta-”. A notable example is in configuration files as shown in the following step where layer names without the “meta-” string are appended to several variables used in the configuration.

3. **Create a Layer Configuration File**: Inside your new layer folder, you need to create a conf/layer.conf file. It is easiest to take an existing layer configuration file and copy that to your layer’s conf directory and then modify the file as needed.

The meta-yocto-bsp/conf/layer.conf file in the Yocto Project Source Repositories demonstrates the required syntax. For your layer, you need to replace “yocto BSP” with a
Yocto Project 2.7.4 Documentation

Introduction and Overview

- Quick Build
- Overview and Concepts

Manuals

- Reference Manual
- Board Support Package (BSP) Developer's guide
- Development Tasks Manual
- Linux Kernel Development Manual
- Profile and Tracing Manual
- Application Development and the Extensible SDK (eSDK)
- Toaster Manual
- Bitbake User Manual

'Mega' Manual

- All-in-one 'Mega' Manual

© Copyright 2010-2020, The Linux Foundation

Built with Sphinx using a theme provided by Read the Docs.
We need your help!

- Automatic conversion was imperfect, many typos or glitches left over
- Offline documentation (PDF-like)
- Improve main page, nice looking front page example
- Improve ‘search box’, perhaps with DocSearch (see example)
- Backport Sphinx docs for YP 3.1 LTS (dunfell)
- Better integration with Yocto Project website
- Improve existing manuals
- Join the mailing list and review patches!
GUI improvements

Welcome to the Project ACRN (version latest) documentation. ACRN is a flexible, lightweight reference hypervisor, built with real-time and safety-criticality in mind, optimized to streamline embedded development through an open source platform.

Source code for Project ACRN is maintained in the Project ACRN GitHub repository, and is provided under the BSD 3-clause license.
GUI improvements
Thank you for joining us today!