LinuxFest Northwest

2016

Doc Like an Egyptian

Dru Lavigne
Documentation Lead, iXsystems
LFNW April 23, 2016

All the old paintings on the tombs,
They do the sand dance, don't you know?

If they move too quick (oh whey oh),
They're falling down like a domino.

Walk Like an Egyptian, The Bangles

Outline

Introduction to Sphinx as part of an open source documentation solution

Disclaimer: based on my experiences. While from an open source perspective, discussion also applies to proprietary documentation.
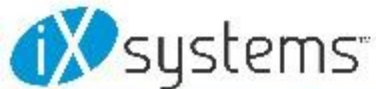
iXsystems

In Theory: Docs...

are published with each software release,

in multiple formats which suit the needs and devices of that software's userbase,

in multiple translations which match that software's global audience,

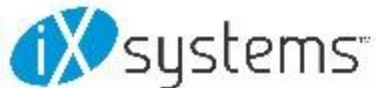and are grammatically and technically correct.

In Reality...

docs are incomplete or outdated (assuming they even exist,

docs are given lower priority than code,

noone wants to write docs / noone reads docs anyways,

no perfect doc management solution exists.

Incomplete/Outdated Docs

Software is a MOVING target

Outdated, unversioned docs are worse than no docs

Need process for versioning and archiving docs, and for finding the correct docs for each software version

I Get No Respect Around Here

Cultural issue: docs are as important as code and Q/A and users actually want useful docs

Gently point users to the pertinent section of the docs and create bug reports or doc updates

Finding the "Perfect" Tool

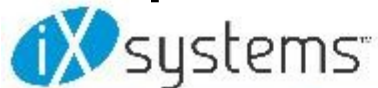Is its formatting language easy to learn or a significant barrier to entry?

Are syntax editors available?

Does it support the required formatting?

Finding the "Perfect" Tool

1st Law of Doc Tools: the number/quality of outputs is inversely proportional to the ease of the markup language.

2nd Law of Doc Tools: the fewer the doc maintainers, the higher the number of desired outputs.

Example: doc/odt
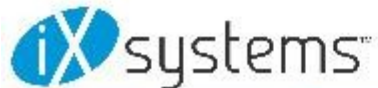
Good: WYSIWYG editor available to any author

Bad: templates are painful; difficult to collaborate; outputs are limited

Example: wiki

Good: entry barrier fairly low and syntax quick to learn

Bad: no concept of ToC, content flow, or versioning; limited outputs

If you build it, they don't necessarily come (except for the spambots)

Example: Latex

Good: rich formatting language; multiple outputs integrate well into build and translation systems

Bad: very high barrier to entry as it takes a dedicated time (and interest) commitment to learn (and teach) the formatting language
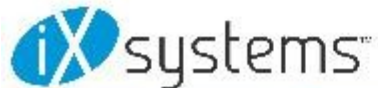
Sphinx Features

Relatively easy-to-learn formatting language that upports many outputs: HTML, PDF, ePUB, LaTeX, Texinfo, man pages, txt, API docs, etc.

Provides a number of extensions such as auto-numbered figures

Sphinx Features

Automatic generation of ToC, index, glossary

Source files are text and easily integrate into existing revision control, translation, build, and CI infrastructures

Sphinx Features

Fully customizable conf.py for controlling doc layout

Several useful built-in builders (e.g. link checker)

iXsystems™

Sphinx Features

Several built-in themes and support for custom themes; layout fully customizable using CSS

Anything can be linked (figures, keywords, headings, etc.)

Sphinx Features

Writers can use any text editor on any system with Python installed (or issue git pull requests)

Sphinx Limitations

Some odd formatting limitations require CSS workarounds (e.g. bold italic)

Documentation is limited, assumes you are familiar with Python, and needs more usage examples

```
.. index:: configuration
.. _Control Panel:

Control Panel
*************


PC-BSD® provides a Control Panel which contains tools for managing your
system. The Control Panel is available from any desktop, meaning it is
available regardless of which desktop you log into.

.. note:: if a desktop does not contain an icon or menu item for Control
Panel, type :command:`pc-controlpanel` from a shell prompt to launch the
Control Panel.

A screenshot of Control Panel started from the KDE desktop can be seen in
:numref:`Figure %s: PC-BSD® Control Panel <control1>`.

.. _control1:

.. figure:: images/control1.png
```
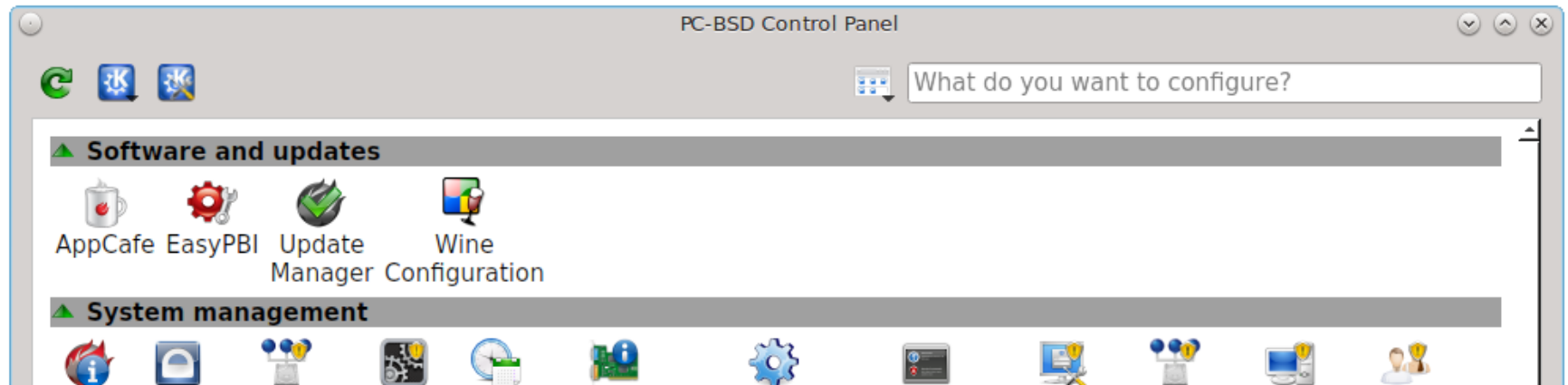
8. Control Panel

PC-BSD® provides a Control Panel which contains tools for managing your system. The Control Panel is available from any desktop, meaning it is available regardless of which desktop you log into.

Note: if a desktop does not contain an icon or menu item for Control Panel, type **pc-controlpanel** from a shell prompt to launch the Control Panel.

A screenshot of Control Panel started from the KDE desktop can be seen in Figure 8.1: PC-BSD® Control Panel.

German × ▾ / PC-BSD Handbook × ▾

Overview   News   **Translate**

More | Hide | Less

To begin the PC-BSD® installation, insert the prepared boot media and boot the system. If the computer boots into an existing operating system instead of the installer, reboot and check your computer's BIOS program to ensure that the drive containing the installation media is listed first in the boot order. Save your BIOS changes and reboot.

Um die Installation von PC-BSD® zu beginnen, fügen Sie das vorbereitete Medium für das Booten ein und booten Sie das System. Wenn der der Computer in ein bestehendes Betriebssystem anstatt in das Installationsprogramm bootet, booten Sie erneut und prüfen Sie das Programm für das BIOS Ihres Computers, um sicherzustellen, dass das Laufwerk, welches das Medium zur Installation beinhaltet, als erstes bei der Rangfolge zum Booten gehört wird. Speichern Sie die Änderungen für das BIOS und booten Sie erneut.

German / PC-BSD Handbook / install.po   String 47341

Locations: ../../install.rst:15

English   W ⧉

The initial boot screen, shown in Figure 3a, offers a choice of using either the graphical or the text based installer. Unless you select otherwise, the graphical installer will load. To instead use the text based installer, either from the console or over a serial connection, use the arrow keys to select an option. If the graphical installer hangs when loading the graphics driver, try selecting the VESA mode option of the graphical installer.

[   ]   Suggest

Comment: 2b02a1475f434fd1a6fa3172c46e387c 57b9bc4fbeee4b44a714c9b7b0973210

Search 🔍   Filter by: Incomplete   ▾   Export View   Previous [1] / 112   Next

Converting Docs

Open source conversion utilities exist for most formats

Experiment by converting a small doc containing required formatting

Expectations

Will you be versioning docs or only creating the latest and greatest?

What outputs are required? Versioned PDFs? HTML on project page? Built into software itself as a help system?

iXsystems™

Source Repo

Determine the doc versioning system and layout of .rst files

Update the README with instructions for authors to create their own doc build environment or how to create pull requests

Create a Cheat Sheet

Include a list of formatting tags and conventions used by your docs as well as any gotchas to help new authors quickly get up-to-speed

Publish the cheat sheet (or include in README) in doc repository

Review What's Available

Spend some time playing with conf.py and experiment with existing themes BEFORE writing new docs as themes affect layout

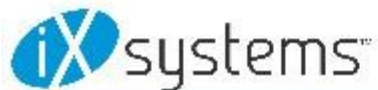Review available extensions and determine if CSS customization is required

Resources & Examples

http://sphinx-doc.org

http://pootle.translatehouse.org

github.com/pcbsd/pcbsd/tree/master/src-qt5/docs

github.com/pcbsd/sysadm/tree/master/api

Questions?

Contact

dru@freebsd.org

URL to slides

http://slideshare.net/dlavigne/lfnw2016

iXsystems™

Evaluate this session

http://lfnw.org/node/3925

Session videos will be posted to the LinuxFest Northwest YouTube channel.

Thank you!