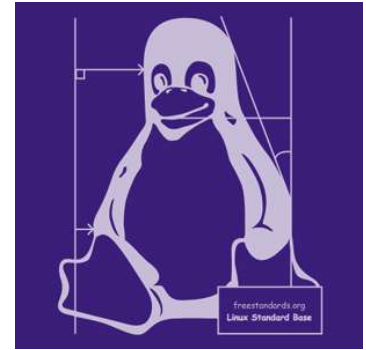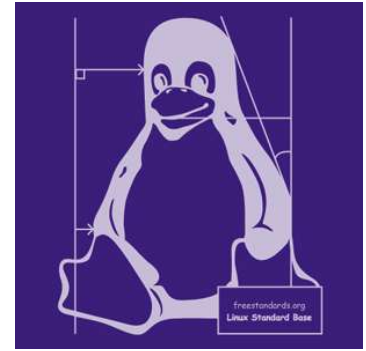# Linux Standard Base and Debian
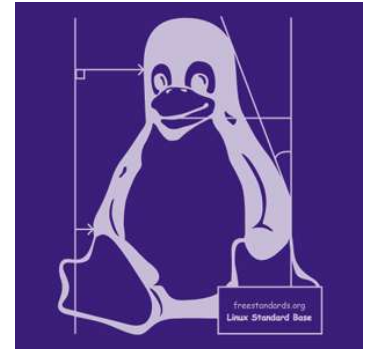
Matt Taggart <taggart@debian.org>
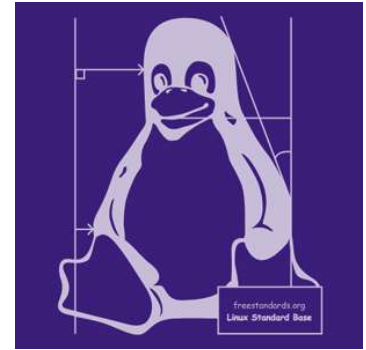debconf2
July 7, 2002

Standards

- 'Standards are great! There are so many to choose from.'

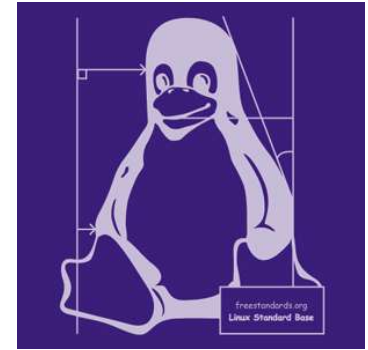- Show of hands: How many have looked at the LSB?

Disclaimer

- Brief introduction targeted at a Debian audience

- Not enough time to cover in depth

- Time for questions, ask as we go
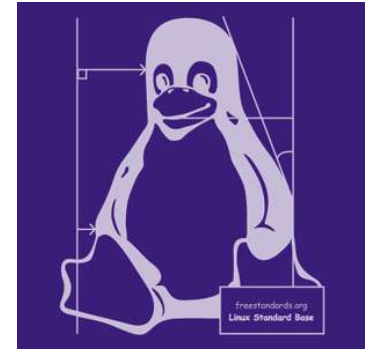
Overview

- Standards, Who needs 'em?

- LSB background

- Debian Involvement

- lsb-futures

- questions

Standards

- Why should Free Software Developers care?
    - Limited interest from Debian so far
    - Most assume it doesn't affect them
    - Early mistakes upset most Debian people
    - Free Standards are actually just as important to Debian than other distributions(if not more)
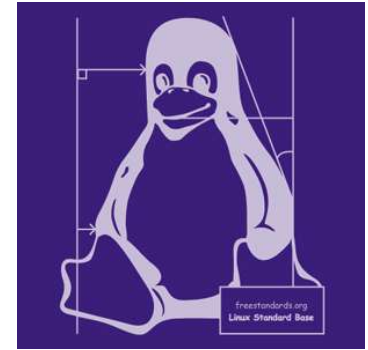
The Rest of the World

- Linux has a minority of OS users

- Debian has a minority of Linux users

- We can't abandon the majority of computer users

- We need to be stay in tune with the world around us and do what we can to provide an alternative

- Similar compromise to non-free, LGPL, and GPL not restricting usage on a propriatary system
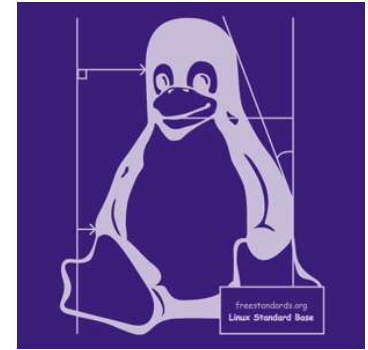
Friend or Foe?
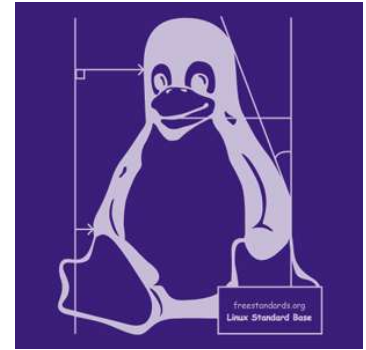
- With commercialization we're starting to see,

  - traditional propriatary software companies embrace free software

  - traditional free software companies embrace propriatary software

  - letter of the law, not spirit

  - grey areas

  - "enterprise" editions of software
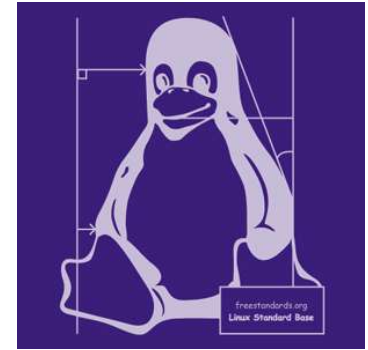
  - United Linux - unclear

Positive Benefits

- Commercialization not all bad

  – Good jobs for hackers

  – Resources to help the community
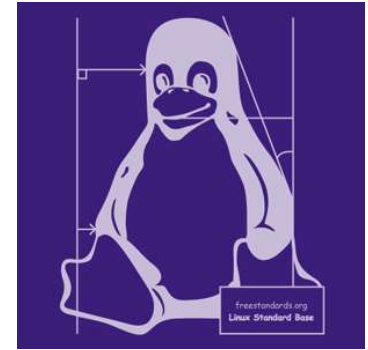
- Keep the good, prevent the bad

UNIX?

- Early UNIX community similar to Linux community

- Fragmentation when propriatary vendors got involved

- Is history repeating itself?

- If so, how do we prevent that?
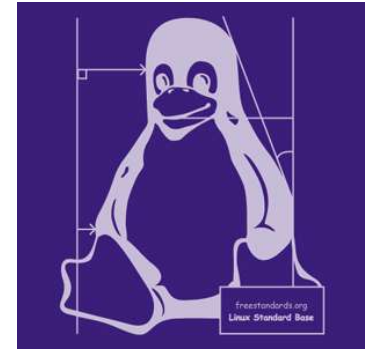
Current Protection

- Debian's protection built on the DFSG
  - protections outlined in the DFSG
  - Licenses that protect against obfuscation, GPL's "preferred form"
- We're safe, but only in our little bubble
- What about things outside of Debian's control that still affect us?
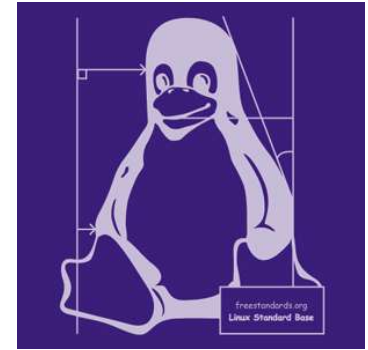
Additional Protection

- Free standards are an additional protection from those that seek to exploit our community

- Driving towards standards helps
  - prevent "lock-in" to propriatary or unique components
  - ensures that others are working with us, not against us
  - solidarity to do the right thing
 - moral high ground, frame the debate

Other Reasons

- Don't care about that? Other practical reasons

- In order to develop the best Debian system possible we need to be efficient

  - need stable API/ABIs

  - Spend time developing, not debugging "quirks"

  - Need to develop in a timely manner, shouldn't have to deal have to deal with unstable unless wanted

  - need applications to continue to work into the future without maintenance
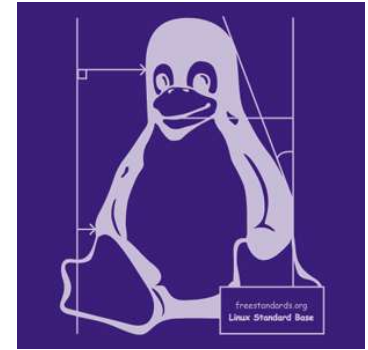
Other Reasons (con't)

- Packaging new software for Debian
  - Developers are also users, may be coming from(and coding in) other environments
  - Example: gwireless-applet

- Compounding Effect
  - These things that helps attract users and developers helps to grow the community and further enforce those things, etc.
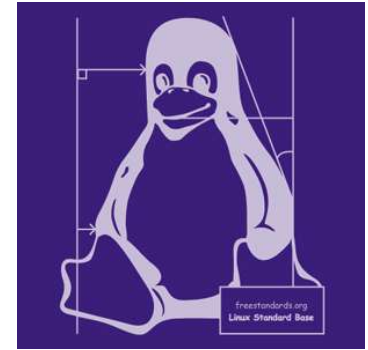
LSB Background
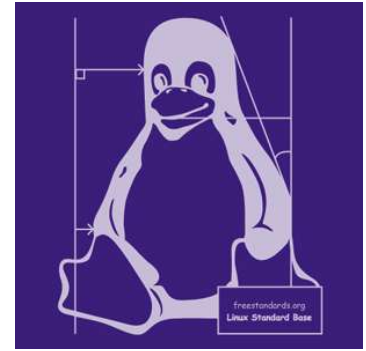
- Need exists, let's do something about it
- Creation of the Linux Standard Base
  - Workgroup of the Free Standards Group
  - Li18nux, LANANA are other workgroups
- LSB gets to leverage lots of existing standards
  - FHS, POSIX, UNIX, etc.
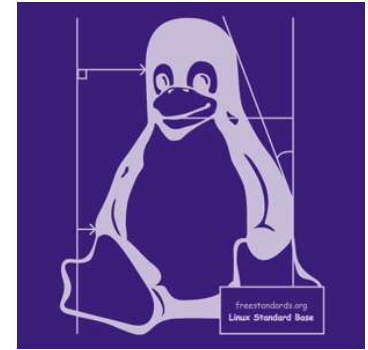
Problem with Standardizing Linux

- "Developing to a standard and walking on water are both easy... as long as they're frozen"

- Developers aiming for a moving target

- Want BINARY-compatibility across Linux systems

- upstream needs to be able to the right thing

- no such thing as 'bug for bug compliant'

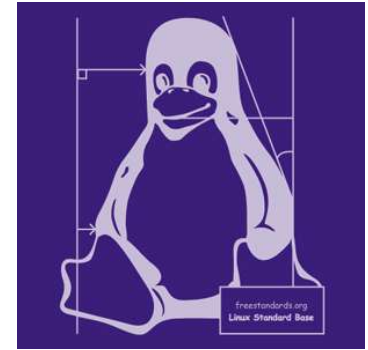- How do we solve both?

Answer

- Most upstreams already do a good job

- Libraries

  - soname versioning

  - Versioned symbols

  - LSB headers and stub libraries

  - safety net, separate linker

    - ls-lsb.so -> ld-linux.so

- Commands

  - minimal, mostly covered by other standards
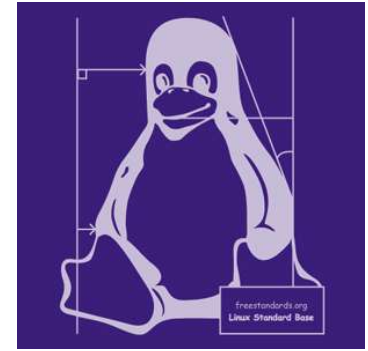
The LSB is...

- a BINARY standard

- a standard for developers

- meant to cover the majority of Linux systems

- a minimal set of universal components
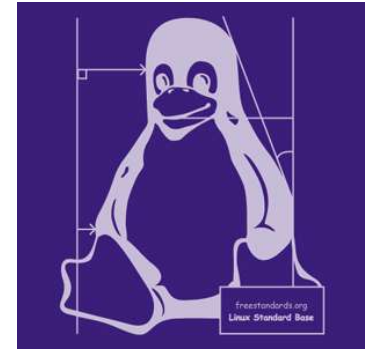
The LSB is not...

- a source only standard

- a standard for user or administration tools

- meant to cover niche areas like RT, embedded, etc.

- meant to include things that only exist in some Linux systems

- everything and the kitchen sink
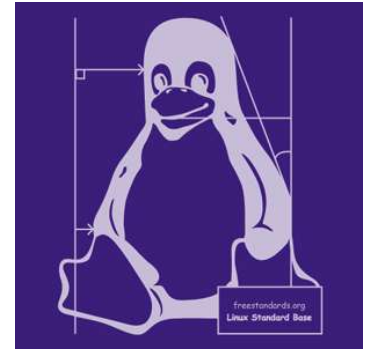
Details

- gLSB vs. archLSB

- runtime vs. application

- test suite

- development tools

- certification

Debian

- Runtime

  - lsb - provides runtime, depends on the things in the LSB. Chris Lawrence <lawrencc@debian.org>

  - alien - support for installing LSB packages. Joey Hess <joeyh@debian.org>

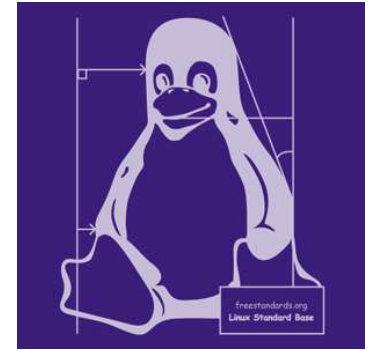 - lsb_release - prints LSB information. Wichert Akkerman <wakkerma@debian.org>
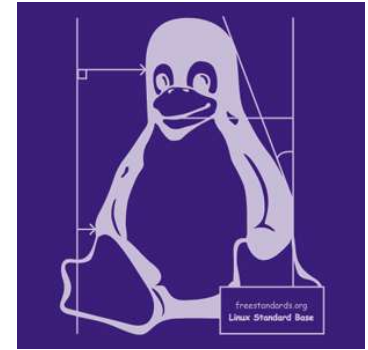
Debian

- Development

  - lsbrpm - Static version of rpm needed for creating LSB packages. Joey Hess <joeyh@debian.org>

  - lsbdev - chroot environment to aid in developing LSB applications. Matt Taggart <taggart@debian.org>

  - new in 1.2, splits lsbdev into
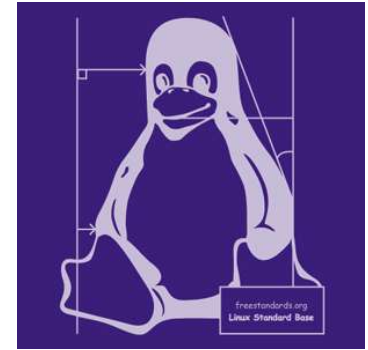
    - lsbdev-base

    - lsb-chroot

    - lsbcc

Debian

- Testing, filing bugs, fixing

  – lots of people

- Compilance

  – Participation in Certification Program. Anthony J. Towns <ajt@debian.org>
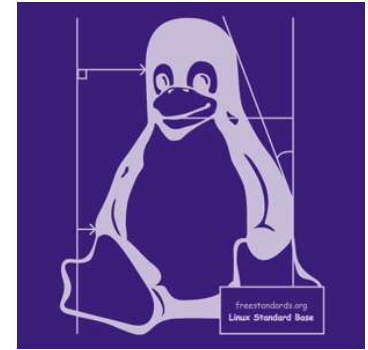
- Others

Status

- Version 1.2 released on June 28

- Fixes previous Debian concerns

- Target for the first wave of certification

- 1.3 planned for December, 2002

  - c++

  - additional architecture support and bug fixes
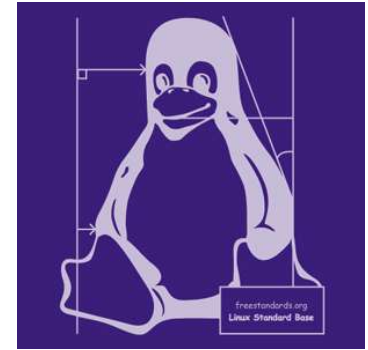
lsb-futures

- June 2001- HP, IBM, Intel, others.

- Standard needed sooner to prevent fragmentation

- Accelerate expansion of the LSB
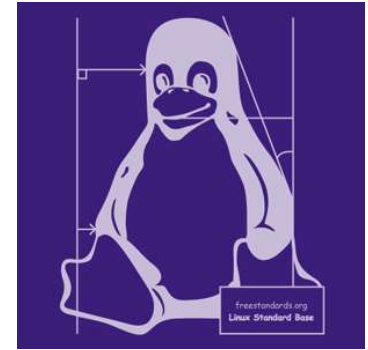
- Improve process for inclusion

Criteria

- document best practice, don't invent standards

  - demand, best pratice, stable

- standardize API/ABI not implementation

  - allows for competing implementations

- licenses and patents

  - maximize participation

  - "no strings attached development environment"

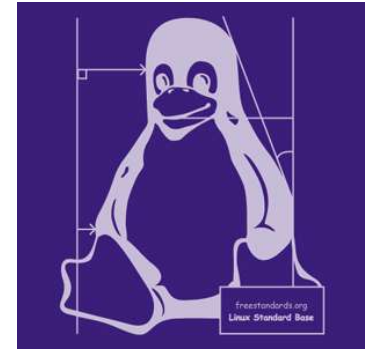 - GPL vs. LGPL

Criteria

- community participation
  - upstream cooperative
  - distro maintainers
  - distro versions
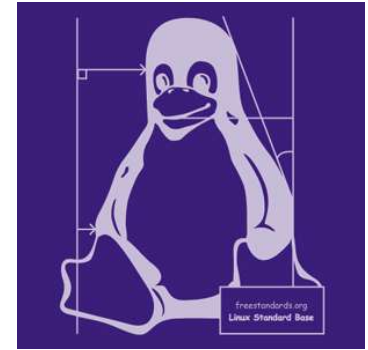 - distro patches

Selection Process

- Gather input from development community

- Analyse existing body of software

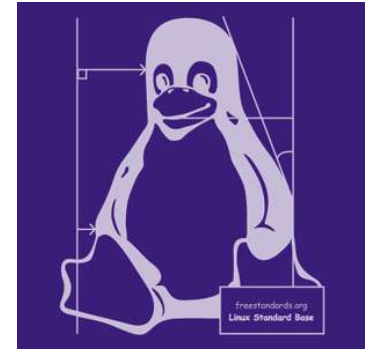- Track dependencies

Contact and Liason

- Work with the component communities to move towards standards

- Examples
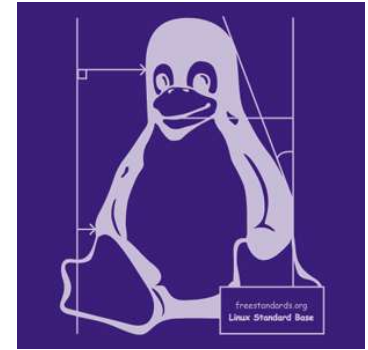
  - c++

  - gtk2

  - not kernel

Tracking

- Track components through the process

- scalable, many candidates in process at any given time

- 'developer rules' - Things that people are motivated to work on go in first.

Offload

- Shield the rest of the LSB Workgroup and allow them to concentrate on the immediate version

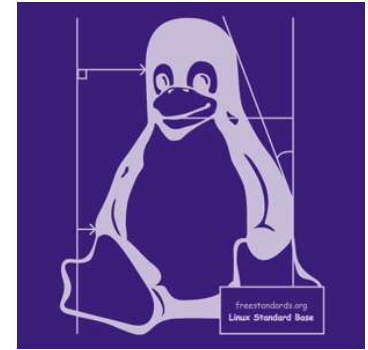- Speeds up releases

- Improves quality

Conclusion

- Debian needs the LSB

- The LSB needs Debian

- Version 1.2 is very reasonable from a Debian perspective

- Future versions will be reasonable

- Continuing(and increased) participation from Debian can ensure protection for the community and a better Debian too

Questions
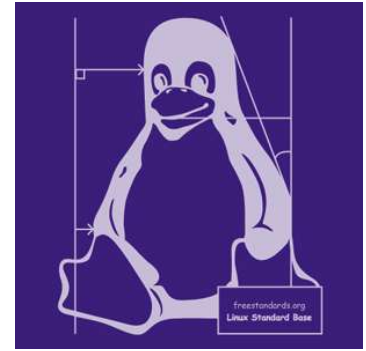
Taxi?

- Anyone want to share a taxi to the airport at 2pm?