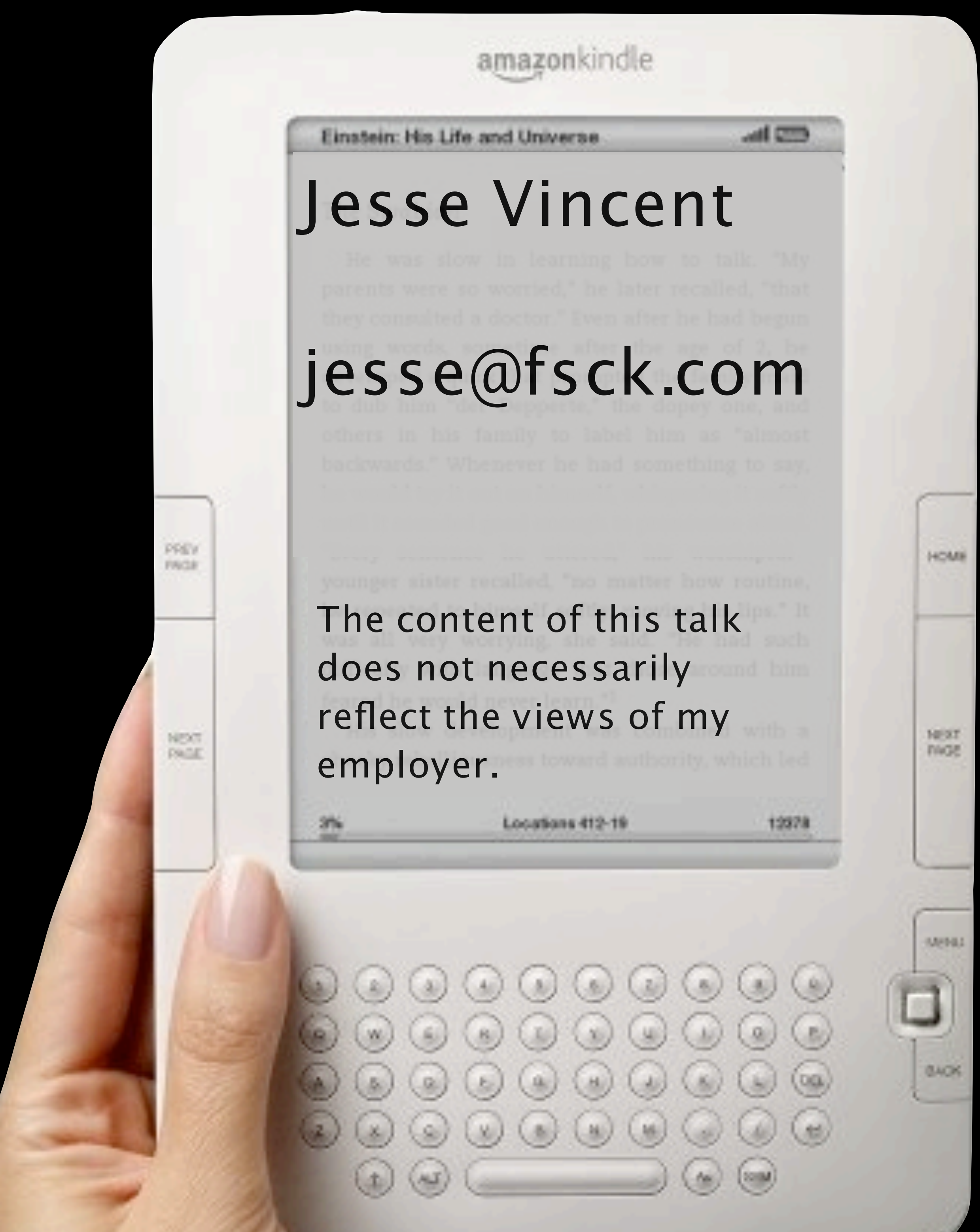
My first ebook reader

Talk Contains NO DRM



Jesse Vincent

jesse@fsck.com

The content of this talk does not necessarily reflect the views of my employer.

I bought the Kindle 2 to read books, but I couldn't leave well-enough alone.

Some quick Kindle facts

ARM 1136JF-S

2Gb flash (4 on the DX)

128 Mb RAM

3G Modem

USB OTG

600x800 16 grey screen

Great battery

Linux 2.6, glibc 2.5, DBus, busybox

Most of /sbin is in sh

2.0 shipped with a USB-Networking debug mode

GUI is Java (Obfuscated)

Browser and JVM provided by ACCESS

Things **not** to do

Don’t steal books

Don’t use your Kindle as a 3G modem

Don’t crack Amazon’s Topaz DRM

Don’t crack Amazon’s .mobi DRM

Think twice before you h4xx0r your Kindle

Amazon knows **where you are**

Amazon knows **who you are**

Amazon has **your credit card number**

Your syslog gets sent to Amazon

You might **brick** your Kindle

Why hack the Kindle?

I like to read. A lot. I *love* the Kindle's 3G modem.

The Kindle has limited format support:

> No ePub. No PDF. No .lit. No .chm.

...so I couldn't read a lot of stuff I wanted to read.

The Kindle *does* support .prc (Mobipocket)
and .cbz (Comic books)

(Open Formats)++

ePub is just zipped HTML and images

PDF is... PDF

.prc (Mobipocket) is HTML 3.2 + extensions + glue

.cbz is just zipped .pngs and .jpgs

Calibre

http://calibre.kovidgoyal.net

Calibre is free and open

It runs on the desktop

That kind of defeats the purpose of the Kindle

Early Hackery

Perl app to convert ePub to Mobipocket

Web-based document conversion system

http://kindle.fsck.com/http://some.com/foo.epub

Custom Kindle book to automate delivery

Then I found the USB Network mode

```
:debug
`usbNetowk
`usbQa
```

`192.168.15.244` ➡ `192.168.15.200`

Now I didn’t need the 3G modem

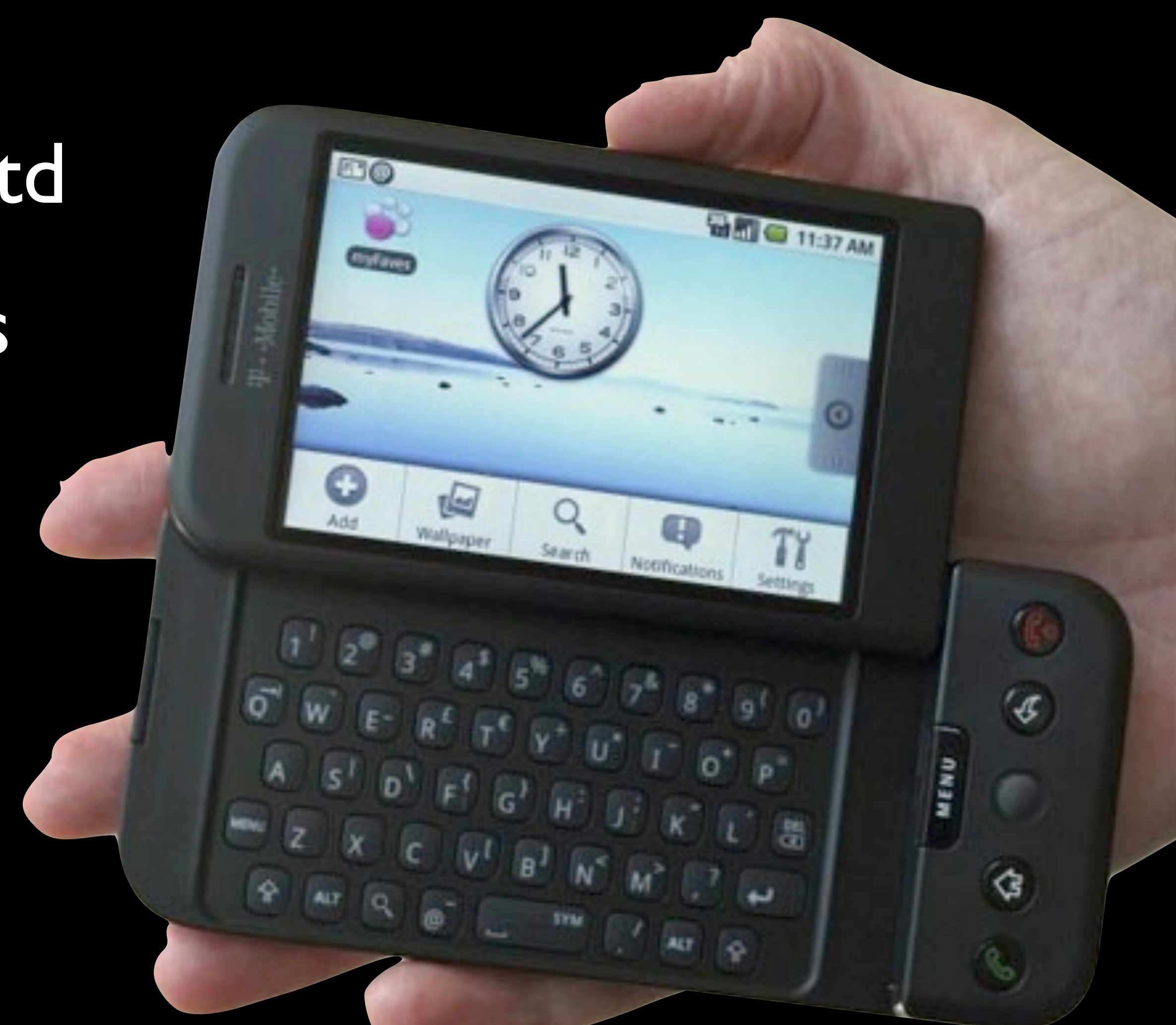
What next?

The Kindle 1 software update format was based on tar, sh and MD5

Reverse engineered (by somebody else) - see http://igorsk.blogspot.com

The Kindle 2 and DX use the exact same updater format

Getting in the first time



Amazon’s busybox is built without telnetd

ARM Linux is pretty standard these days

A statically linked busybox is just fine

Early discoveries:

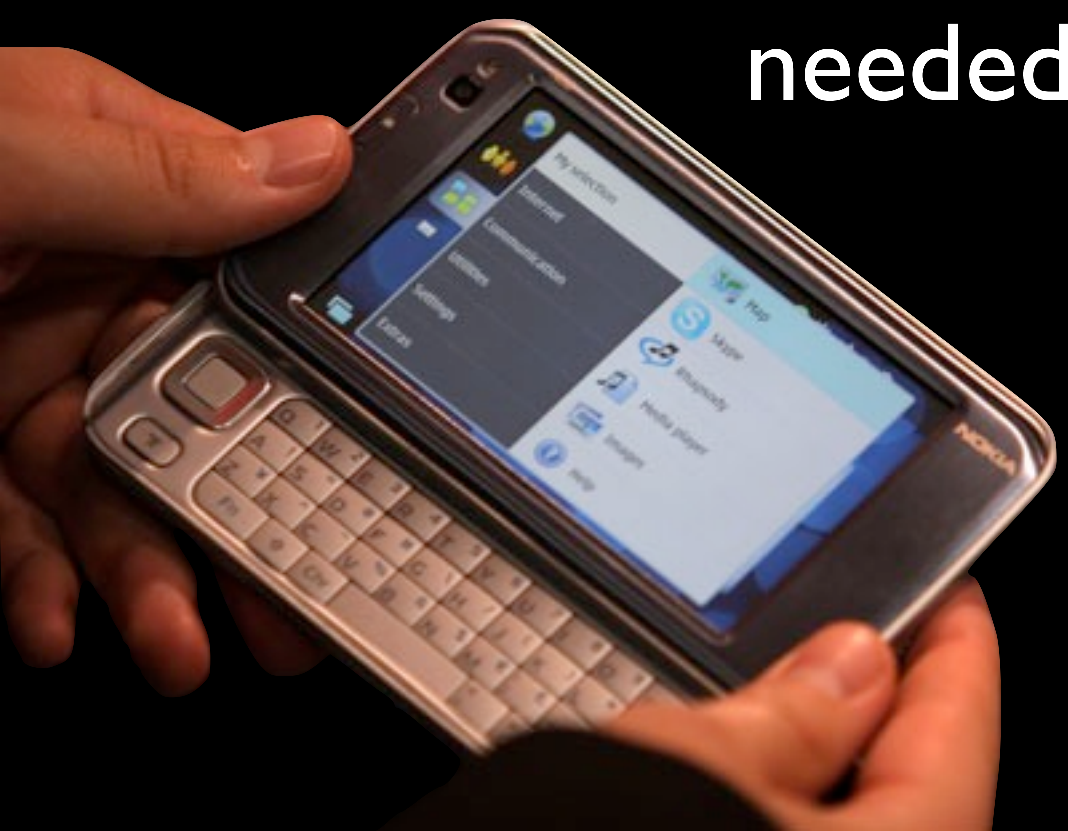
- /proc/config.gz
- Kernel built with NFS
- User-data partition NOT mounted noexec
- Undocumented support for .cbz files

Buildfarm on an N810

Cross-compiling is ... not reliable

Linux 2.6, glibc 2.5 and gcc

Built nfsmount, screen, and everything else needed to get some “work” done



Building Calibre on Kindle

Qt, Python

(unladen swallow)

PyQt

Took 12 hours to

convert a book...after I built

swaptools and gave it 256M

Savory for Kindle

Hacked Calibre down to size

Poppler-based .pdf ➨.cbz engine

inotify and DBus based daemon

Kindle updater that adds an init script

ext2 disk image with Savory runtime

That was good enough...

...until I got DX envy.

The Kindle 2 is codenamed ‘turing’

Test scripts on the device talked about ‘nell’ (and about a turing with a trackball)

When the DX came out, I wanted a real PDF reader...with zoom, search and indexes

This should be easy!

1. Figure out how to paint the screen
2. Figure out how to read the keyboard and fiveway controller
3. Build a custom PDF reader for the Kindle

Screen and Keyboard

- /dev/fb0 - Virtual framebuffer
  - echo “{1,2,3}” > /proc/eink_fb/update_display
- /dev/input/event{0,1} - Keyboard and Keyboard
  - The 5-way is just another keyboard
- Drivers in GPL Kernel release
- Like everything else, bog-standard linux

Ubuntu

Ubuntu Jaunty Jackalope - ported to ARM

Installed on qemu

Tarred up the root image

NFS mounted on the Kindle

```
chroot /tmp/kindle sh
```

X.org

X.org needs a TTY or VT to start up

The Kindle's Kernel is built without CONFIG_VT

I did awful (but small) things to X.org

Xfbdev "just works"

What’s next?

Polish

Documentation

Publication

Fixing X.org’s colormap

Building a useful user experience