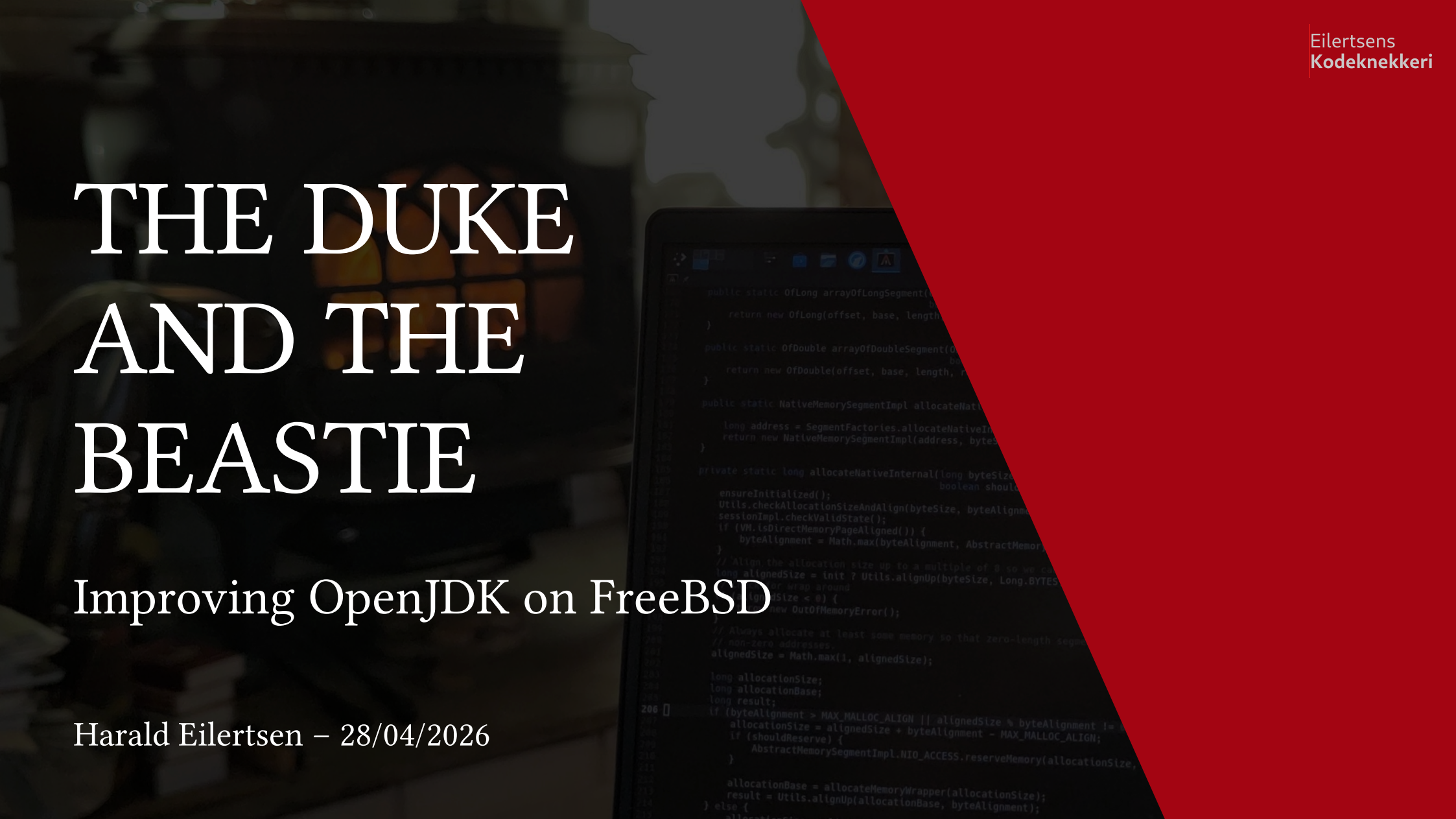


Eilertsens
Kodeknekkeri

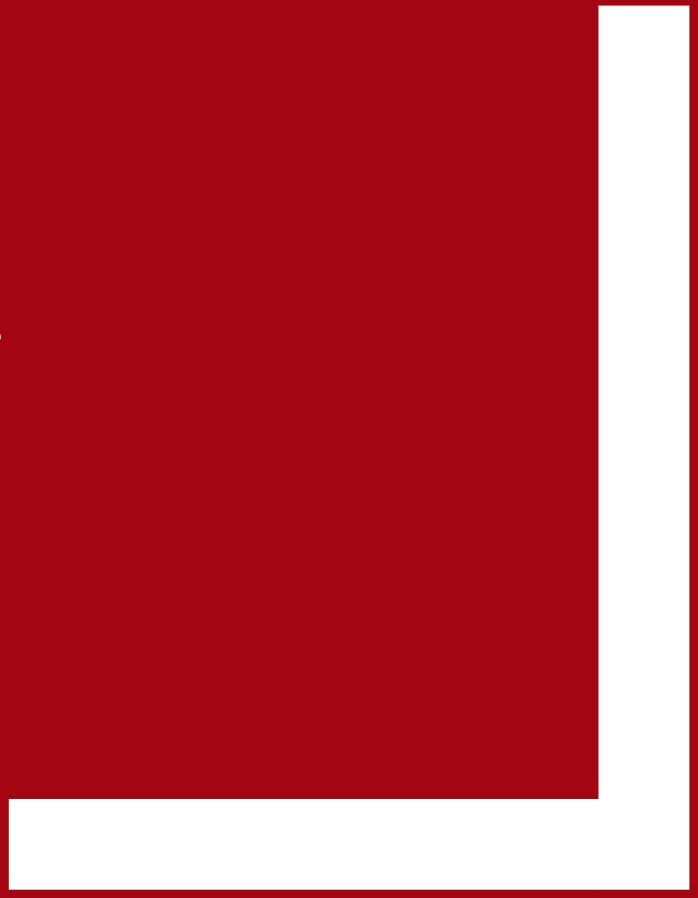
THE DUKE AND THE BEASTIE

Improving OpenJDK on FreeBSD

Harald Eilertsen – 28/04/2026

Contents

INTRODUCTION



No “AI”

This presentation is made by a human!

Harald who?

- Been messing with computers since the early 1980's

Figure 1: My first computer, a Dragon32. Photo by Liftarn, CC BY-SA 2.0

Harald who?

- Anti Virus software, intercepting fs and network on OS/2, Windows and Linux

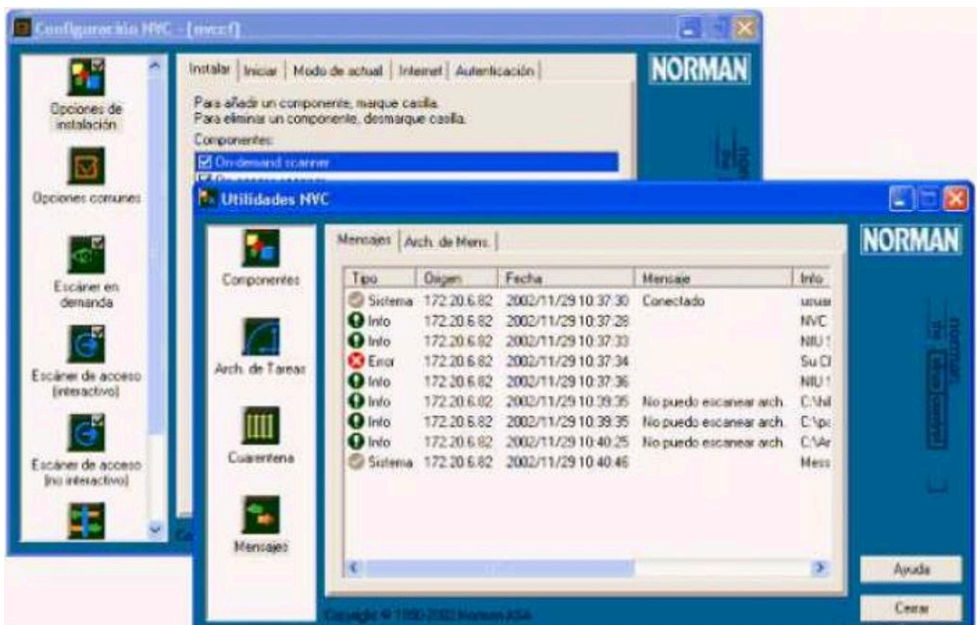


Figure 2: Screenshots of Norman Virus Control, ca 2002

Harald who?

- High end video conferencing, embedded realtime stuff on weird hardware

Figure 3: The Tandberg 8000 mxp video conferencing room system

Harald who?

- Malware/vulnerability research
- WAF development and rules
- Penetration testing/Security auditing

Figure 4: CC0 Lucas Andrade

Harald who?

...and now Improving Java on FreeBSD!

Figure 5: The Duke and the Beastie!

Project sponsor

About FreeBSD

- Free and Open Source OS based on UNIX
- Licensed under a BSD license
- A complete OS, not just a kernel
- Known for: ZFS, Jails, TCP/IP, ...

Figure 7: https://www.mckusick.com/beastie/

About FreeBSD

- Free and Open Source OS based on UNIX
- Licensed under various BSD licenses
- A complete OS, not just a kernel
- Known for: ZFS, Jails, TCP/IP, ...
- **No systemd!**

Figure 8: https://www.mckusick.com/beastie/

About FreeBSD

- Free and Open Source OS based on UNIX
- Licensed under various BSD licenses
- A complete OS, not just a kernel
- Known for: ZFS, Jails, TCP/IP, …
- No systemd!
- **No penguins!**

Figure 9: https://www.mckusick.com/beastie/

About OpenJDK

- Free software implementation of the Java VM and Language Specification.
- Project stewarded by Oracle
- Licensed under the GPLv2

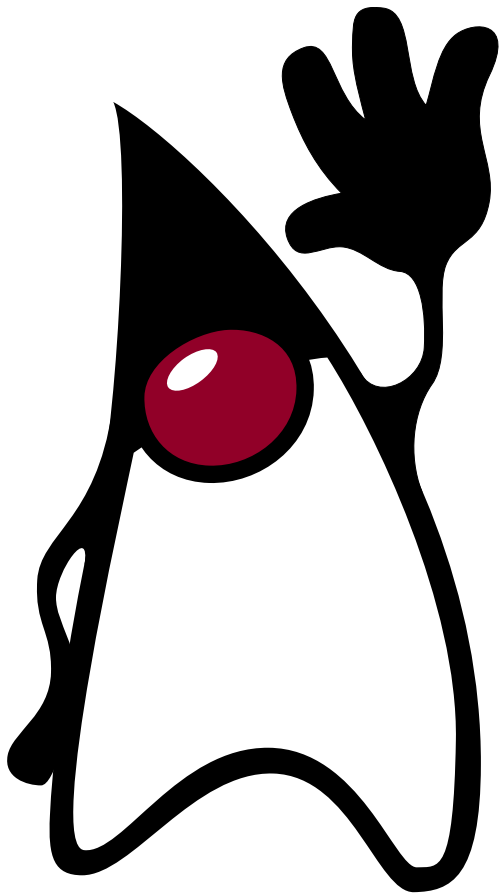


Figure 10: https://wiki.openjdk.org/spaces/duke/pages/51282050/Main

About OpenJDK

- Free software implementation of the Java VM and Language Specification.
- Project stewarded by Oracle
- Licensed under the GPLv2
- ...but need to sign the Oracle Contributor Agreement (OCA)

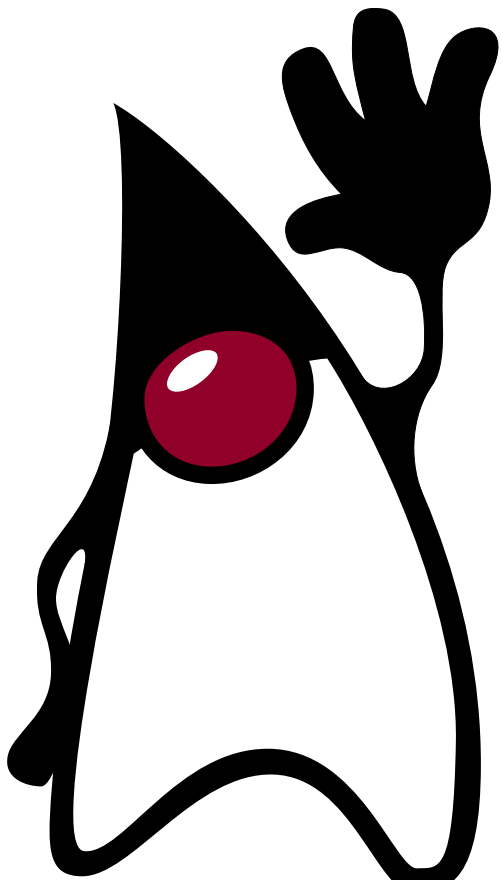


Figure 11: https://wiki.openjdk.org/spaces/duke/pages/51282050/Main

About OpenJDK

- Written in C, C++, Java and assembler
- Supported operating systems: Linux, Windows, MacOS, AIX
- Supported architectures: Aarch64, AMD64, Arm32, PowerPC 64, RISC-V, S390, …

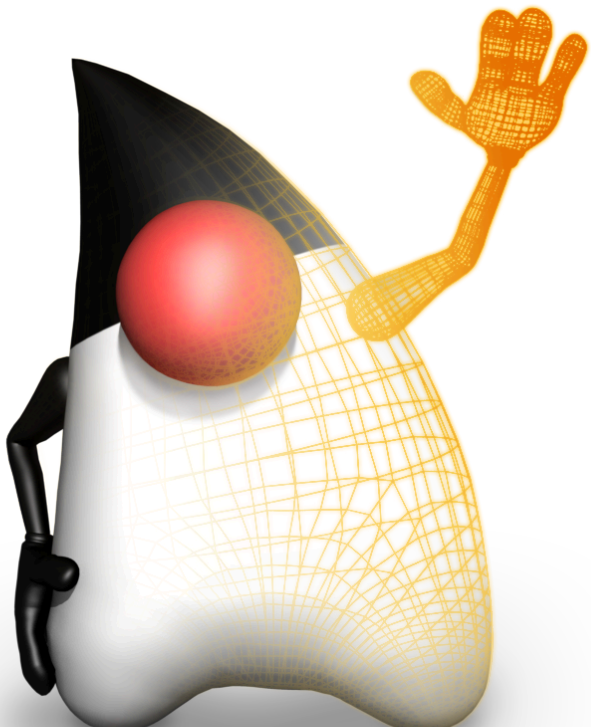


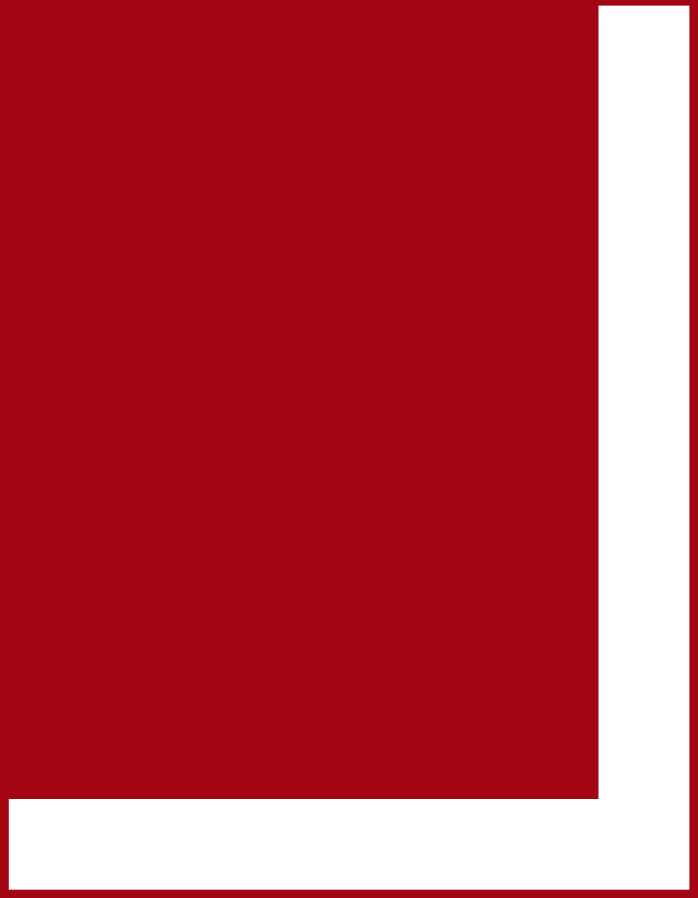
Figure 12: https://wiki.openjdk.org/spaces/duke/pages/51282050/Main

What's Missing?

What's Missing?

No BSD is among the platforms supported by the upstream project.

THE BSD PORT PROJECT



The BSD port project

- **A project to port and keep OpenJDK up to date on FreeBSD, OpenBSD and NetBSD.**

Figure 14: BSD logos

The BSD port project

- A project to port and keep OpenJDK up to date on FreeBSD, OpenBSD and NetBSD.
- **Lives out of tree in the `battleblow/jdk` repo on Github.**

Figure 15: BSD logos

The BSD port project

- A project to port and keep OpenJDK up to date on FreeBSD, OpenBSD and NetBSD.
- Lives out of tree in the `battleblow/jdk` repo on Github.
- **A collaborative project among the BSD's.**

Figure 16: BSD logos

The BSD port project

Goals:

1. Make a solid port of OpenJDK for *BSD.
2. Include the port upstream, supported by the BSD Port Project.
3. Pass the JCK (Java Conformance Kit)

The BSD port project

Status

Goals:

1. Make a solid port of OpenJDK for *BSD. - Mostly done!
2. Include the port upstream, supported by the BSD Port Project.
3. Pass the JCK (Java Conformance Kit)

The BSD port project

Status

Goals:

1. Make a solid port of OpenJDK for *BSD. - Mostly done!
2. Include the port upstream, supported by the BSD Port Project. - In progress
3. Pass the JCK (Java Conformance Kit)

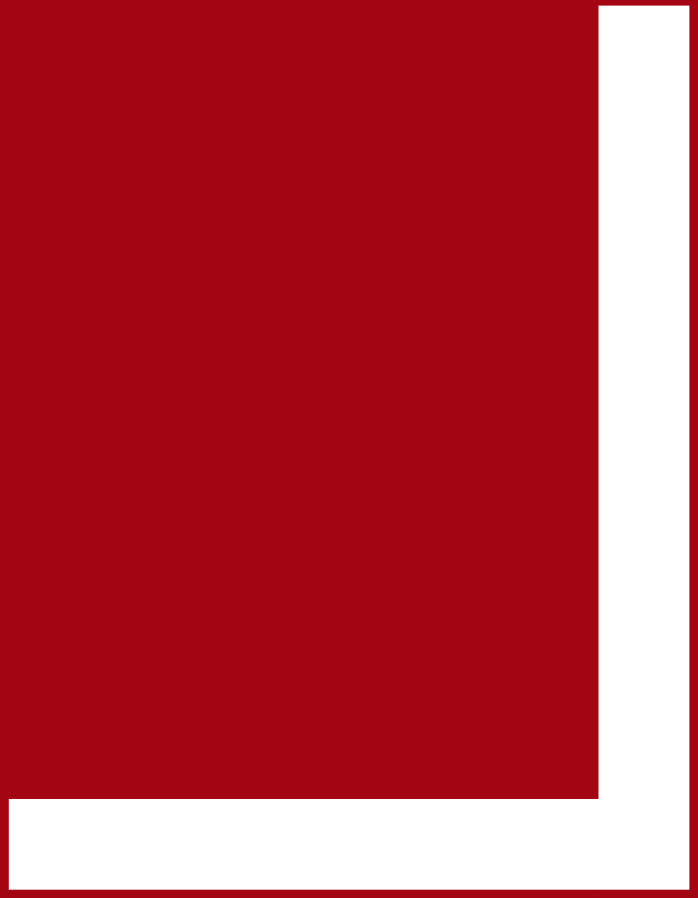
The BSD port project

Status

Goals:

1. Make a solid port of OpenJDK for *BSD. - Mostly done!
2. Include the port upstream, supported by the BSD Port Project. - In progress
3. Pass the JCK (Java Conformance Kit) - Pending

WORKING ON OPENJDK FOR FREEBSD



OpenJDK internals

- Big project with a long history
- Complex build system based on autoconf and GNU make
- Supports lots of different configurations
- On many different architectures
- And many different operating systems

Figure 17: Kurt Schwitters, Merzbau

OpenJDK internals

- Comprehensive test suite (`jtreg`, `googletest`, performance tests, etc)
- Lots of documentation (but never enough!)
- Established governance and development workflows
- Large and helpful community!

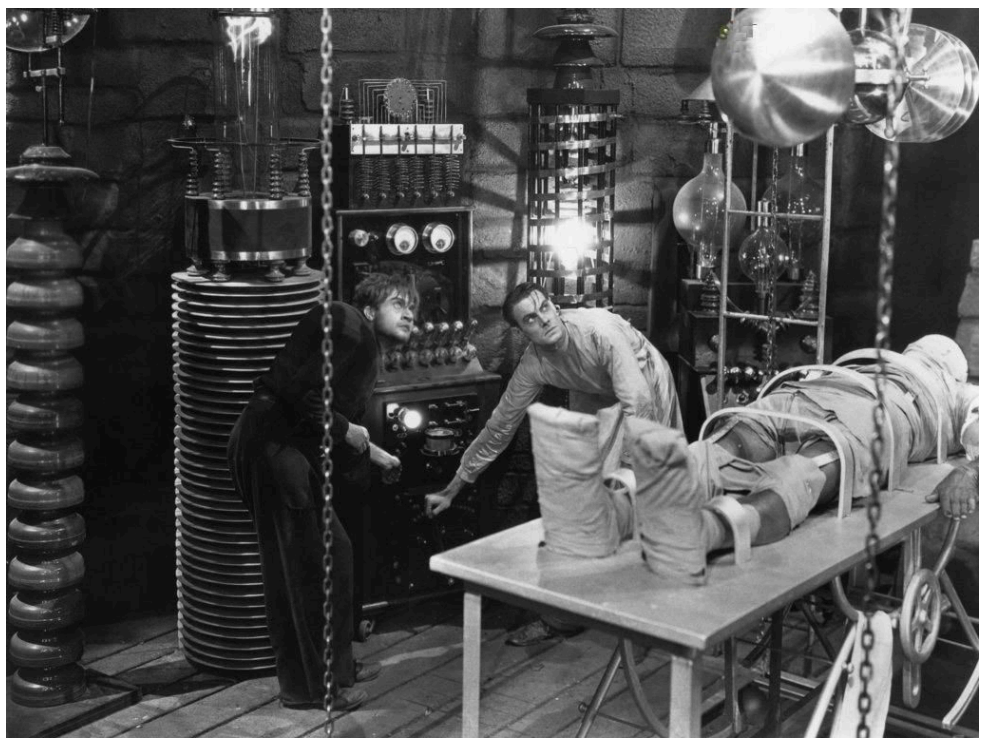


Figure 18: Frankenstein, 1931

Code structure

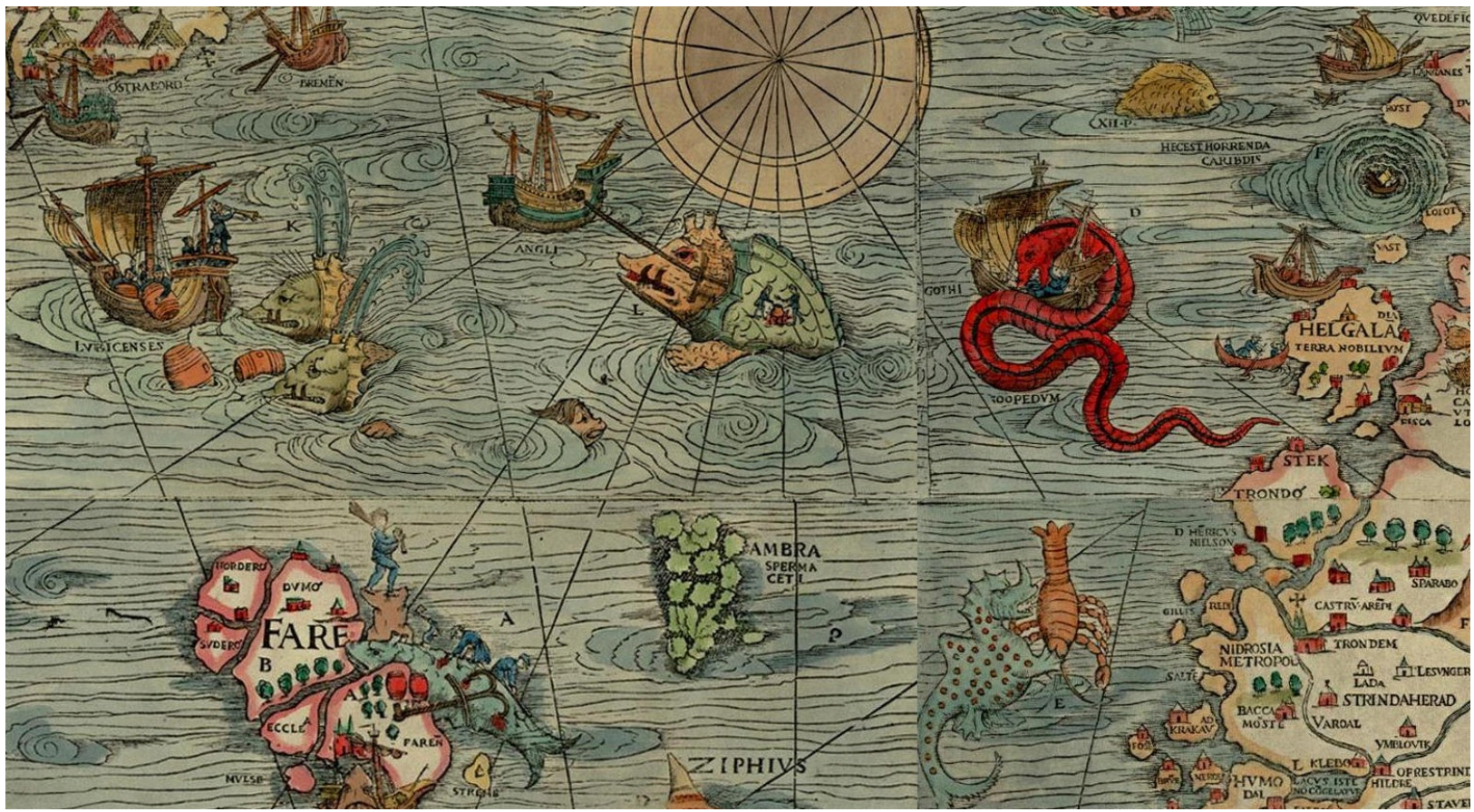


Figure 19: Sea monsters outside Norway, Carta Marina, 1539

Code structure

Hotspot

- The main VM and most low level OS and architecture code.
- Written mainly in C-like C++, and some assembler.

```
src/hotspot
├── cpu
│   ├── aarch64
│   ├── arm
│   ├── ppc
│   ├── riscv
│   ├── s390
│   ├── x86
│   └── zero
├── os
│   ├── aix
│   ├── bsd
│   ├── linux
│   ├── posix
│   └── windows
⋮
```

```
⋮
├── os_cpu
│   ├── aix_ppc
│   ├── bsd_aarch64
│   ├── bsd_x86
│   ├── bsd_zero
│   ├── linux_aarch64
│   ├── linux_arm
│   ├── linux_ppc
│   ├── linux_riscv
│   ├── linux_s390
│   ├── linux_x86
│   ├── linux_zero
│   ├── windows_aarch64
│   └── windows_x86
└── share
```

Code structure

Build system

- A set of autoconf scripts to configure the build.
- A complex hierarchy of GNU Makefiles generates the make rules in memory.
- A set of internal tools to generate matching Java/C++ code for interfacing between the two worlds.

Figure 20: More sea monsters, Carta Marina, 1539

Code structure

Tests

- Custom buildt test system: `jtreg`
- Some googletest for Hotspot
- Microtests for benchmarking
- CI using Github Actions

```
/**
 * @test
 * @bug 8191658
 * @summary Test clhsdb attach, detach, reattach commands
 * @requires vm.hasSA
 * @requires (os.arch != "riscv64" | !(vm.cpu.features ~=
".*qemu.*"))
 * @library /test/lib
 * @run main/othervm ClhsdbAttach
 */

public class ClhsdbAttach {
   ⋮
}
```

Code structure

Modules

- Recognizable from the Java API docs
- This is the largest part of the codebase
- Mostly in Java, but some native and platform specific parts here too

```
src/java.base
├── aix
│   ├── classes
│   ├── conf
│   └── native
├── bsd
│   ├── classes
│   └── native
├── linux
│   ├── classes
│   └── native
├── macosx
│   ├── classes
│   └── native
⋮
```

```
⋮
├── share
│   ├── classes
│   ├── conf
│   ├── data
│   ├── legal
│   ├── man
│   └── native
├── unix
│   ├── classes
│   ├── conf
│   └── native
└── windows
    ├── classes
    ├── conf
    ├── legal
    └── native
```

Repository structure

- openjdk/jdk: Upstream mainline
  - Main development occurs here
  - New major releases (jdk25, jdk26, …) released from stabilizing branches
  - New major release every six months
- openjdk/jdkNNu: Repos for update releases (jdk25.0.x, …)
  - Updates after initial release maintained in separate repositories
  - Forked from stabilizing branch
  - Bug fixes backported from mainline

Bootstrapping

- Large parts of OpenJDK is written in Java
- Iow: We need a working OpenJDK to build OpenJDK

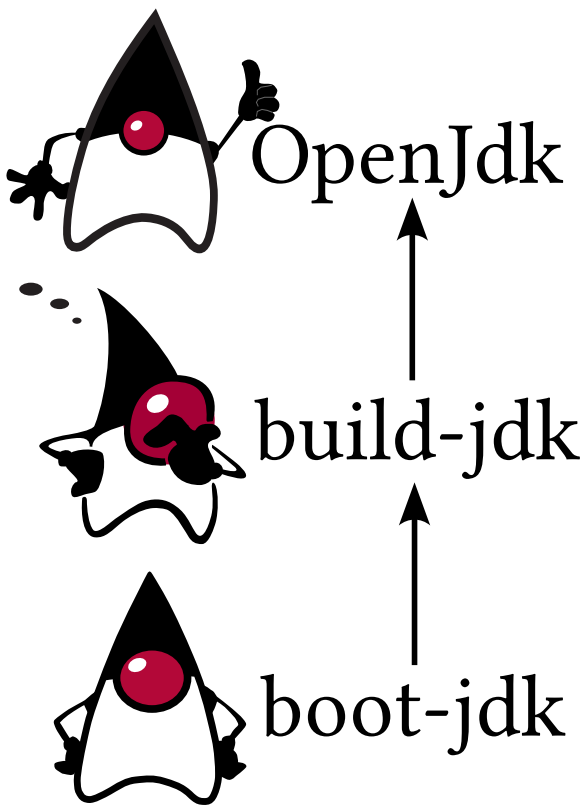


Figure 21: Dukes all the way down

Bootstrapping

- A **boot-jdk** is used to build an intermediate **build-jdk**. Must be *same or previous* major version
- The **build-jdk** is a minimal jdk - just enough to build the full jdk. Must be *exact same* version as the target jdk.

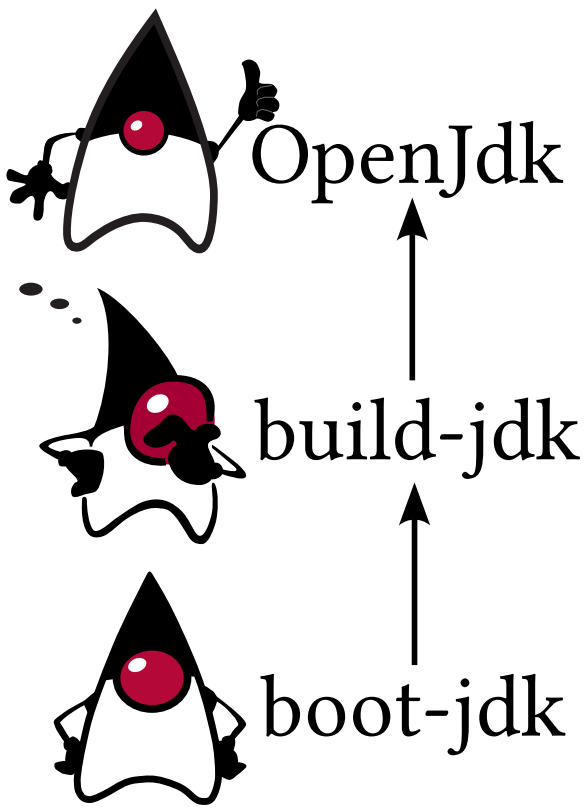


Figure 22: Dukes all the way down

Bootstrapping

When there is no existing port

- Cross-compiling
- Dev kits
- Use a boot-jdk on a separate machine over SSH

Bootstrapping

When there is no existing port

- **Cross-compiling**
  - Configure with e.g. `--openjdk-target=armv7-unknown-freebsd`
  - Only (officially) supported for GCC and a handful of Linux distros
  - Best bet to build *Zero* variant, and bootstrap real build from that
  - Have not had any luck on FreeBSD using clang or gcc
- Dev kits
- Use a boot-jdk on a separate machine over SSH

Bootstrapping

When there is no existing port

- Cross-compiling
- **Dev kits**
  - Premade cross-compiling setups.
  - Configure with `--with-devkit` or `--with-build-devkit`
  - Can be created by `cd make/devkit && gmake tars BASE_OS=... TARGETS=...`
  - Or downloaded from various sources.
  - Only supports Fedora and Oracle Linux
- Use a boot-jdk on a separate machine over SSH

Bootstrapping

When there is no existing port

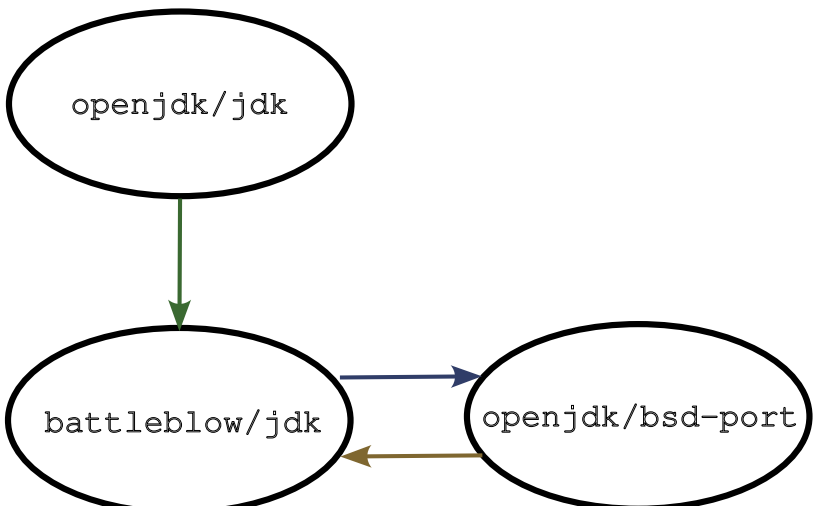
- Cross-compiling
- Dev kits
- **Use a boot-jdk on a separate machine over SSH**
  - Requires a stub boot-jdk with binaries replaced by script forwarding over SSH
  - Source and build tree *must* be shared between the machines
  - ... at the *the same* location. e.g `/home/myuser/src/openjdk`
  - Must be able to log in to other machine without a password

```
$ cat boot-jdk11/bin/javac
#!/usr/local/bin/bash
progname=`basename $0`
exec ssh myuser@othermachine -- /usr/local/openjdk11/bin/$progname "${@@Q}"
```

Listing 1: SSH-boot-jdk-trick, courtesy of Kurt Miller (bsdkurt)

UPSTREAMING THE PORT

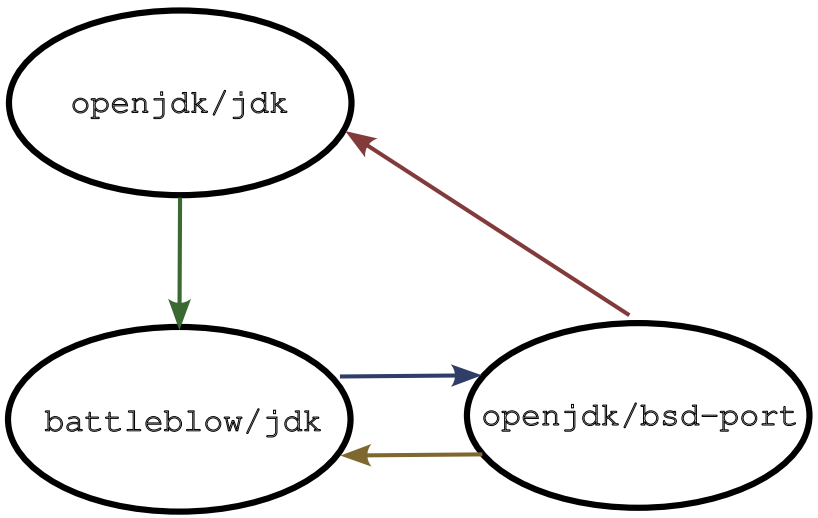
Three main repos



- Upstream mainline: `openjdk/jdk`
  - The base that we build our port on top of
- Existing BSD port: `battleblow/jdk`
  - Our changes on top of the mainline
  - Kept in sync and continuously ported
  - CI includes FreeBSD and OpenBSD
- Upstream BSD port: `openjdk/bsd-port`
  - A separate repo where we merge changes from the existing port
  - Every PR must be reviewed by at least one upstream reviewer
  - Changes fed back into `battleblow/jdk`

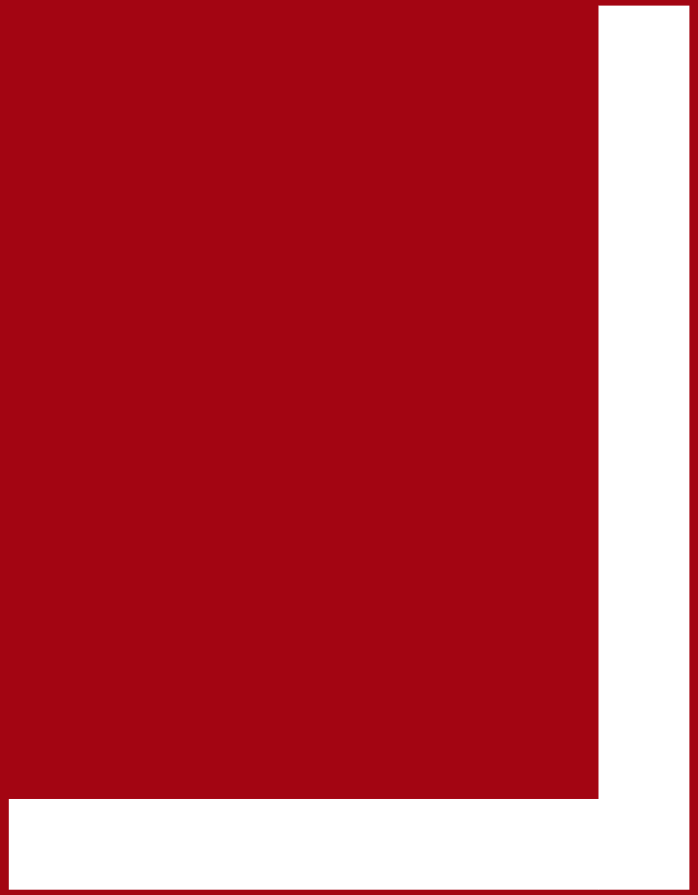
Three main repos

- Upstream mainline: `openjdk/jdk`
  - The base that we build our port on top of
- Existing BSD port: `battleblow/jdk`
  - Our changes on top of the mainline
  - Kept in sync and continuously ported
  - CI includes FreeBSD and OpenBSD
- Upstream BSD port: `openjdk/bsd-port`
  - A separate repo where we merge changes from the existing port
  - Every PR must be reviewed by at least one upstream reviewer
  - Changes fed back into `battleblow/jdk`



When ready, the goal is to merge the bsd-port repo back into mainline

WRAPPING UP



Challenges

- The complexity of the project
  - OpenJDK is in itself a complex code base
  - Interfacing with low level OS primitives (memory, process, threads, debugging, etc)
- Upstream breaking our port (rare, but happens)
- macOS thinks it's a BSD
- Github Actions don't natively support BSD

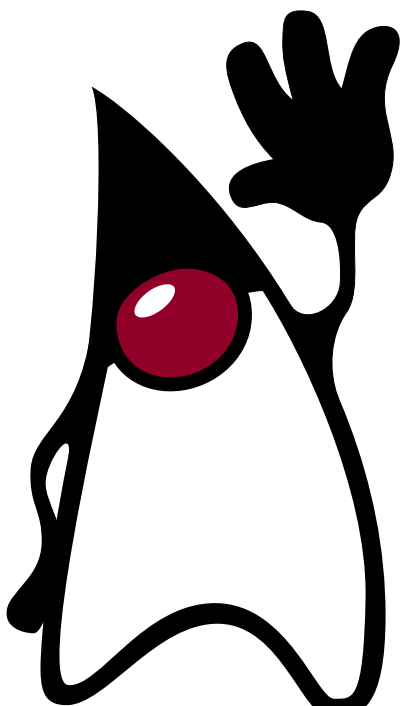
Future

- OpenJDK on FreeBSD is in a pretty good shape, but need to be maintained
- Work is not done after code is upstreamed
  - BSD port project will get a different role
  - Better resilience against breaking changes
  - But also expectation about maintaining the port
- Conformance certification must be repeated for new versions

Thanks

- **The FreeBSD Foundation:** Ed Maste, Joseph Mingrone, …
- **The BSD Port team:** Greg Lewis, Kurt Miller
- **OpenJDK project:** Magnus Ihse Bursie, Erik Joelsson, David Holmes, …
- **FreeBSD ports mentors:** fuz, bofh
- Piotr Kubaj for help with PowerPC bootstraps and more
- Ronald Klop for massive work on upgrading default JDK in FreeBSD
- Everyone who files bugs, gives feedback and has an interest in the project

Thanks



Harald Eilertsen, @harald@hub.volse.no, https://kodeknekkeriet.net/en