Building a binary distribution of OpenJDK
Strong foundation
20 years of history

1997
MCST – Moscow Center of SPARC Technologies
The company worked as a contractor for Sun Microsystems

2004
Sun Microsystems founded its own Development Center in Saint-Petersburg.

2010
Oracle Corp. acquired Sun Microsystems

2017
Open-source contributions

We contribute to

- Hadoop
- OpenJDK
- GraalVM
Number of external contributions to OpenJDK upstream (8.2017 - 8.2018)

Red Hat | SAP | Google | BellSoft | IBM | ARM | Qualcomm | Intel | Linaro | Longsoon | Azul | JetBrains
Why build your own OpenJDK binary?
Liberica JDK binaries history

Oracle decided not continue to release binaries for ARM after JDK8

2017
- ARM64 Binary
- ARM32 Binary

2018
- Linux x86 64-bit,
- Windows x86 64-bit
- Alpine glibc
- MacOS,
- Solaris SPARC
- Solaris x86

2019
- Installers for Mac, Win, Linux
- Linux x86 32-bit
- Windows x86, 32-bit
- Alpine musl libc

Oracle announced JDK11 licence change and a new release cadence
Define supported platforms list wisely

- Need hardware to compile, build and test
- It is expensive and worthless to buy it
- Having virtual servers can not give a 100% assurance about tests results and performance measurements
- Solution is hybrid infrastructure: local servers and cloud instances
- To support Liberica JDK several hundreds of servers are in use
Compilers and toolchains

There is nothing complex

$ hg clone
$ ./configure
$ make images
Compilers and toolchains

- Compilers in Linux distributions not always yield successful results
- Proper compilers need to be built.
- Using right compilers can give additional benefits
  - Oracle & BellSoft works on moving to gcc 8.
  - Meltdown and Spectre fixes
  - Stability
  - ARM specific gcc improvements
What if we have the proper compiler but build is not successful?
What can we do to fix build process?

Make a hack for your build environment and publish it at GitHub.

Fix OpenJDK build process or toolchain
What can we do to fix build process?

- Fix OpenJDK build process or toolchain
The build is successful!

Test it.

$ git clone https://github.com/spring-projects/spring-petclinic.git
$ cd spring-petclinic
$ ./mvnw spring-boot:run
It works!
You are production ready.

<table>
<thead>
<tr>
<th>Name</th>
<th>Address</th>
<th>City</th>
<th>Telephone</th>
<th>Pets</th>
</tr>
</thead>
<tbody>
<tr>
<td>Jeff Black</td>
<td>1450 Oak Blvd.</td>
<td>Monona</td>
<td>6085555387</td>
<td>Lucky</td>
</tr>
<tr>
<td>Jean Coleman</td>
<td>105 N. Lake St.</td>
<td>Monona</td>
<td>6085552654</td>
<td>Max Samantha</td>
</tr>
<tr>
<td>Betty Davis</td>
<td>638 Cardinal Ave.</td>
<td>Sun Prairie</td>
<td>6085551749</td>
<td>Basil</td>
</tr>
<tr>
<td>Harold Davis</td>
<td>563 Friendly St.</td>
<td>Windsor</td>
<td>6085553198</td>
<td>Iggy</td>
</tr>
<tr>
<td>Maria Escobito</td>
<td>345 Maple St.</td>
<td>Madison</td>
<td>6085557683</td>
<td>Mulligan</td>
</tr>
<tr>
<td>Carlos Estaban</td>
<td>2335 Independence L.</td>
<td>Waunakee</td>
<td>608555487</td>
<td>Lucky Sly</td>
</tr>
<tr>
<td>George Franklin</td>
<td>110 W. Liberty St.</td>
<td>Madison</td>
<td>6085551023</td>
<td>Leo</td>
</tr>
<tr>
<td>Peter McTavish</td>
<td>2387 S. Fair Way</td>
<td>Madison</td>
<td>6085552765</td>
<td>George</td>
</tr>
<tr>
<td>Eduardo Rodriguez</td>
<td>2693 Commerce St.</td>
<td>McFarland</td>
<td>6085558763</td>
<td>Jewel Rosy</td>
</tr>
<tr>
<td>David Schroeder</td>
<td>2749 Blackhawk Trail</td>
<td>Madison</td>
<td>6085559435</td>
<td>Freddy</td>
</tr>
</tbody>
</table>
Why customers hate me?
TCK – Java SE specification compliance guarantee

• TCK is:
  – 500 engineer * years investment
  – 300 pages of documentation
  – 200K+ tests

• Binary TCK verification guarantees compliance with the umbrella Java SE specification and JSRs specific to the particular Java version
Would it be just good to pass TCK?

You MUST! To be Java SE spec compatible.
TCK always passes if we build OpenJDK properly?

Wrong statement!
TCK is important

- Issues identified by TCK
  - java -XX:+UseCompactStrings
  - Win32 VM test (regression tests passed)
- Any platform can be affected at any time
- Solution: Fix OpenJDK
Compatibility is like pregnancy, you can not be almost compatible.

Only TCK 100% is acceptable.
100% TCK passed!

Are you ready for production?

No.
Regression testing

- jtreg regression test harness
  - 20K + tests. Since 1997
  - Must pass 100%, excluding the exception list
  - Shit happens
    - One of the JTreg tests is red in 1% of runs on one platform.
    - What should we do if we have red tests?
    - Fix OpenJDK
Java Concurrency Stress tests

• jcstress - functional testing for the correctness of concurrency support in:
  – JVM
  – Class libraries
  – Hardware.

• Additional benefit behavioral details of the particular platform
Benchmarks

- Industry benchmarks
  - SPECjvm
  - SPECjbb
- Microbenchmarks
Functional testing

- Testing by application and application tests running
  - Hadoop - HighBench and functional tests
  - ElasticSearch, Lucene
  - If everything is good you are good.
  - What if not? We can not nothing.
- Testing by application tests is a good add-on but can not be a basic quality criteria
What should we test to be sure?

- There are $10^{50}$ combinations for platforms, binaries and VM parameters.
- We would be happy to test all of them but life is too short.
- The only way is experimental conclusion.
You are ready to deliver

- The delivery is YourJDK.tar.gz
  - Nobody wants you. Eclipse does not work.
  - Linux packages
  - Win installer
  - Mac installer, SDK man
  - Making users happy is invaluable.
Now we are ready to release super high quality OpenJDK binary!

Consider 20 000 open issues in OpenJDK bug tracker.
Support gives protection from rare cases at critical enterprise environment