BELLSOFT

JVM on ARM
from sensor to cloud

Dmitry Chuyko
JVM Team

Who we are

Dmitry Chuyko

@dchuyko

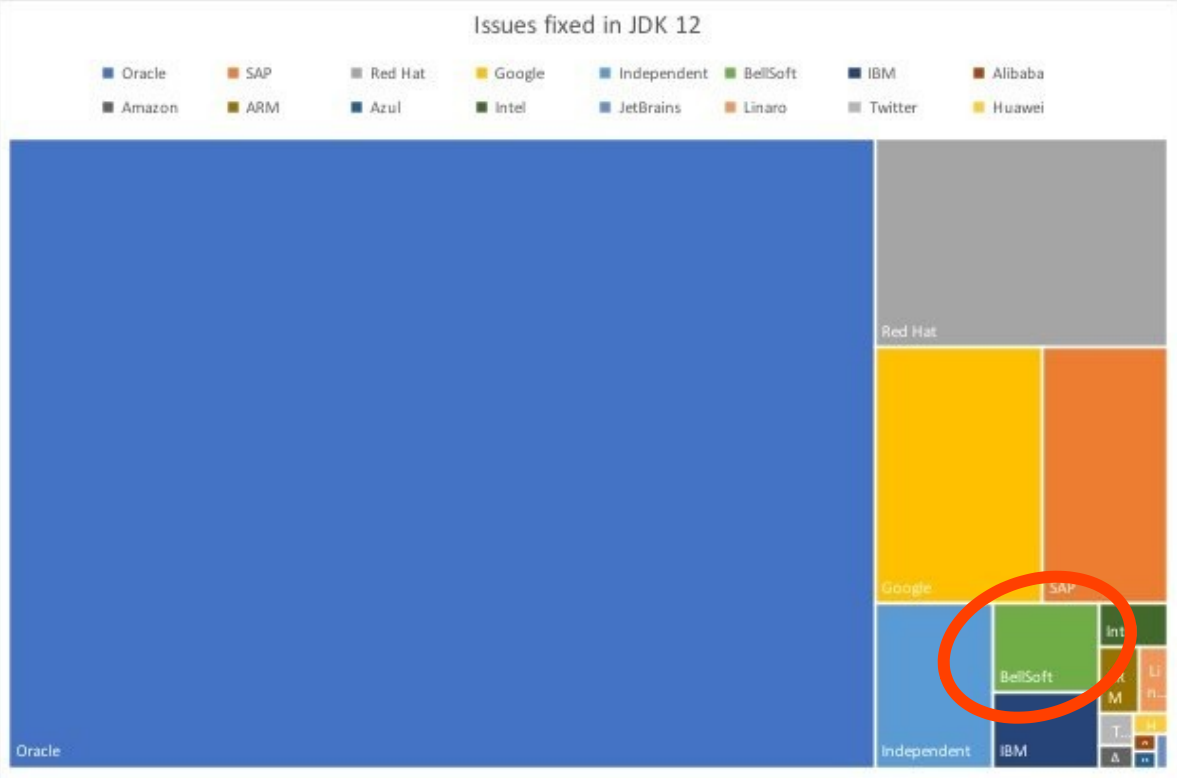
BELLSOFT

Liberica JDK– verified OpenJDK binary
http://bell-sw.com

Ex-employers

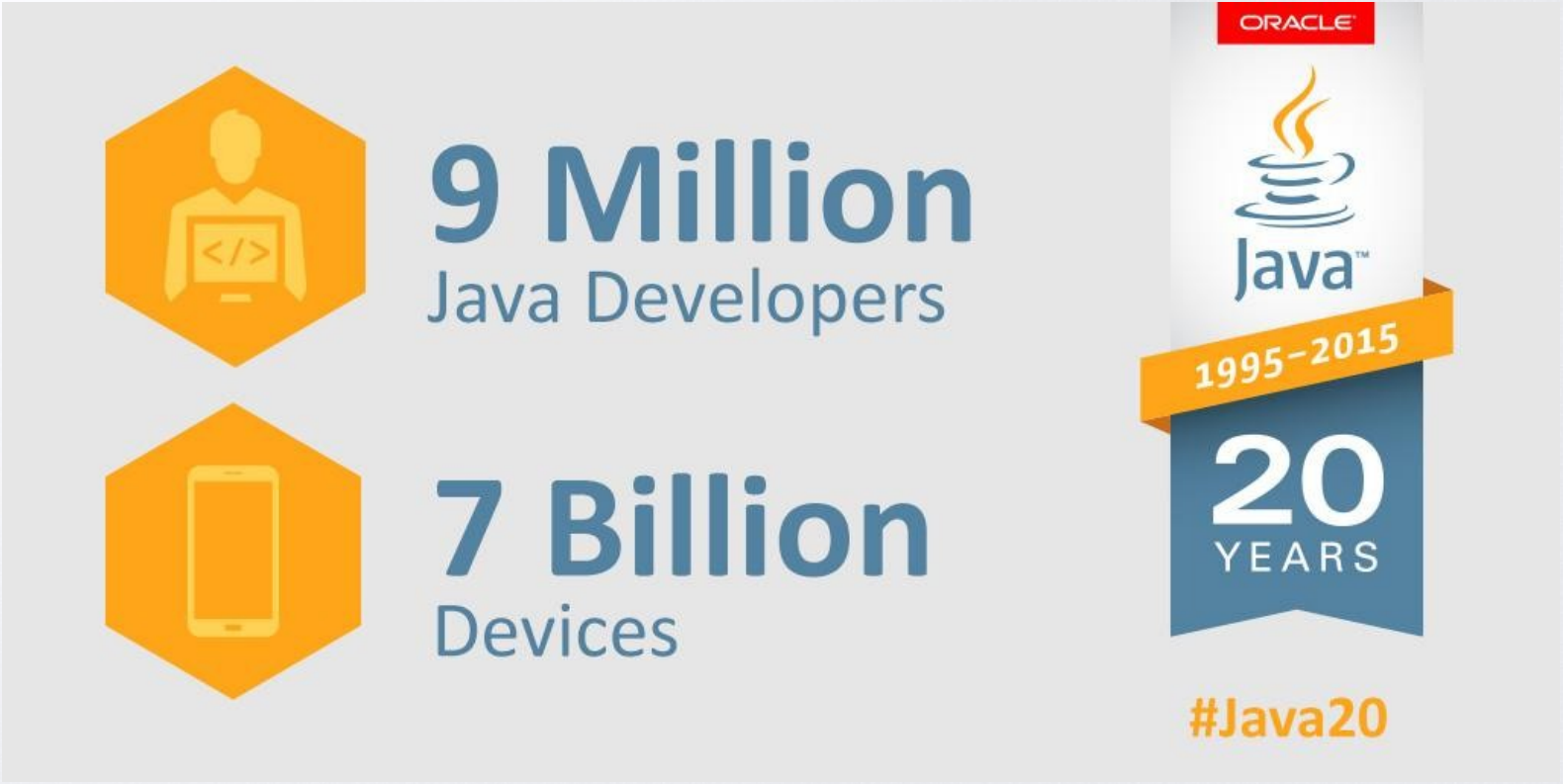
ORACLE

Recent statistics



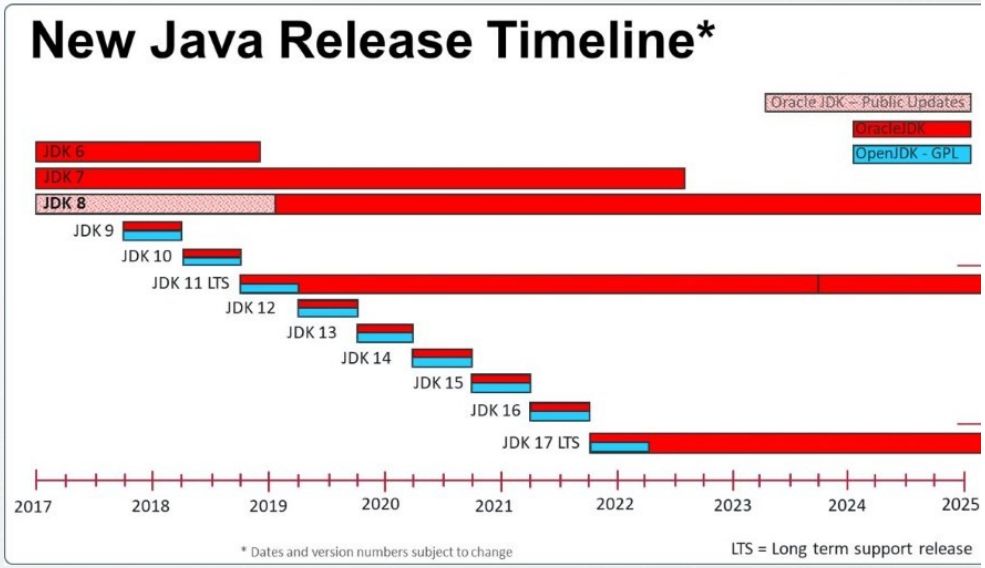
blogs.oracle.com/java-platform-group/the-arrival-of-java-12

Java is 24. Still WORA



Mature evolving open source technology

medium.com/@javachampions/java-is-still-free-2-0-0-6b9aa8d6d244



Java Is Still Free

2.0.3

With the changes to Oracle JDK distribution and support, there has been considerable uncertainty over the rights to use Oracle JDK vs Oracle OpenJDK builds vs OpenJDK builds from other providers. There are various ways to get free updates (**including security**), and (new and existing) paid support models available from various vendors to consider. This document has a Shorter Version and a much Longer Version section with all of the details.

Want to Comment?
Try our Java Is Still Free Mirror post on Medium.

Shorter Version

You can still get the Oracle OpenJDK builds and OpenJDK by other providers for free under an open source license, and the Oracle JDK remains free in some circumstances. See the callout and the rest of the section for the nuances on this.

Java SE / OpenJDK / Oracle OpenJDK Builds / Oracle JDK

The OpenJDK community creates and maintains the (GPLv2+CE) open-source Reference Implementation (RI) of the **Java SE** Specification as governed by the Java Community Process (JCP) and defined through an umbrella Java Specification Request (JSR) for each feature release.

There are implementations (mostly OpenJDK based) of Java SE from various providers such as AdoptOpenJDK, Amazon, Azul, BellSoft, Eclipse, IBM, jClarity, Red Hat, Oracle, SAP, and others.

Oracle JDK 8 is going through the *"End of Public Updates"* process, which means the April 2019 update will restrict commercial use. However, since Java SE 9, Oracle is also providing **Oracle OpenJDK builds** which are free for commercial use (but only updated for 6 months). There are also free **OpenJDK** builds which will be updated (**including security patches**) from other providers like AdoptOpenJDK, Amazon, Azul, BellSoft, IBM, jClarity, Red Hat, Linux distros et al.

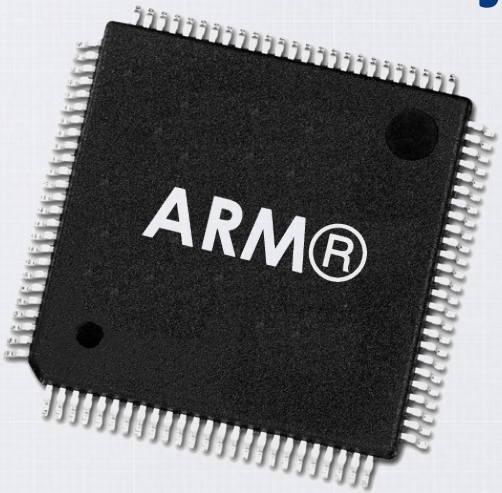
Going forward there are several options to get a JDK. We focus on Java SE 8 and Java SE 11 which is the first Long Term Support (LTS) release under the new release cadence.

- Specification
- Variety of languages
- Ecosystem
- Security
  - Managed Platform
    - Data typing
    - Verification
    - Memory management
  - APIs
    - Cryptography
    - Authentication
    - Public Key Infrastructure

- Variety of implementations
- Variety of platforms

- Binary distributions

- Security
  - Critical patch updates



OpenJDK

Binaries

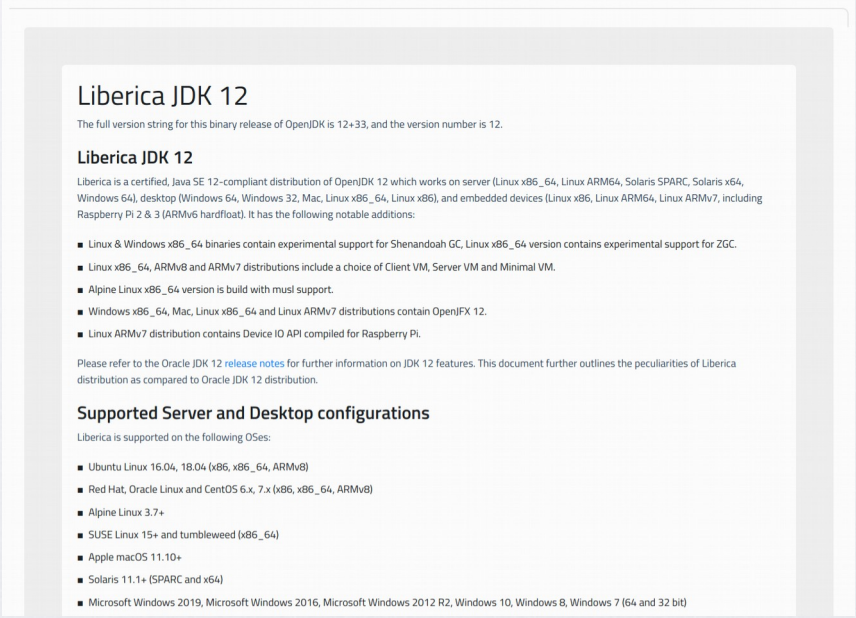
- Tarballs, installers
- Apt, yum
- Docker Hub
- Other channels

bell-sw.com/java.html

Binaries

- For the edge
  - Device IO (I2C, DAC/ADC, GPIO, UART, SPI)
  - OpenJFX (EGL, FB)
- For the cloud
  - Fast Server VM
  - GCs (G1, Shenandoah…)
  - Compilers (C2, Graal)
  - JFR, AOT, AppCDS…

Release Notes



Liberica JDK 12

The full version string for this binary release of OpenJDK is 12+33, and the version number is 12.

Liberica JDK 12

Liberica is a certified, Java SE 12-compliant distribution of OpenJDK 12 which works on server (Linux x86_64, Linux ARM64, Solaris SPARC, Solaris x64, Windows 64), desktop (Windows 64, Windows 32, Mac, Linux x86_64, Linux x86), and embedded devices (Linux x86, Linux ARM64, Linux ARMv7, including Raspberry Pi 2 & 3 (ARMv6 hardfloat). It has the following notable additions:

- Linux & Windows x86_64 binaries contain experimental support for Shenandoah GC, Linux x86_64 version contains experimental support for ZGC.
- Linux x86_64, ARMv8 and ARMv7 distributions include a choice of Client VM, Server VM and Minimal VM.
- Alpine Linux x86_64 version is build with musl support.
- Windows x86_64, Mac, Linux x86_64 and Linux ARMv7 distributions contain OpenJFX 12.
- Linux ARMv7 distribution contains Device IO API compiled for Raspberry Pi.

Please refer to the Oracle JDK 12 release notes for further information on JDK 12 features. This document further outlines the peculiarities of Liberica distribution as compared to Oracle JDK 12 distribution.

Supported Server and Desktop configurations

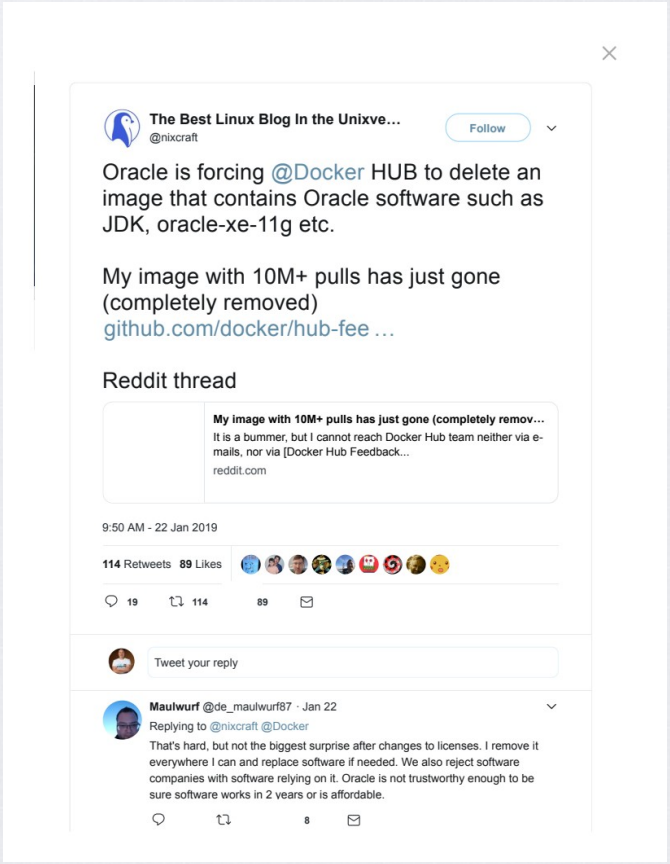
Liberica is supported on the following OSes:

- Ubuntu Linux 16.04, 18.04 (x86, x86_64, ARMv8)
- Red Hat, Oracle Linux and CentOS 6.x, 7.x (x86, x86_64, ARMv8)
- Alpine Linux 3.7+
- SUSE Linux 15+ and tumbleweed (x86_64)
- Apple macOS 11.10+
- Solaris 11.1+ (SPARC and x64)
- Microsoft Windows 2019, Microsoft Windows 2016, Microsoft Windows 2012 R2, Windows 10, Windows 8, Windows 7 (64 and 32 bit)

Derive app runtimes: JDK 11 ARMv7 with Minimal VM and java.base is **17 MB**, including Security manager

Linux Containers

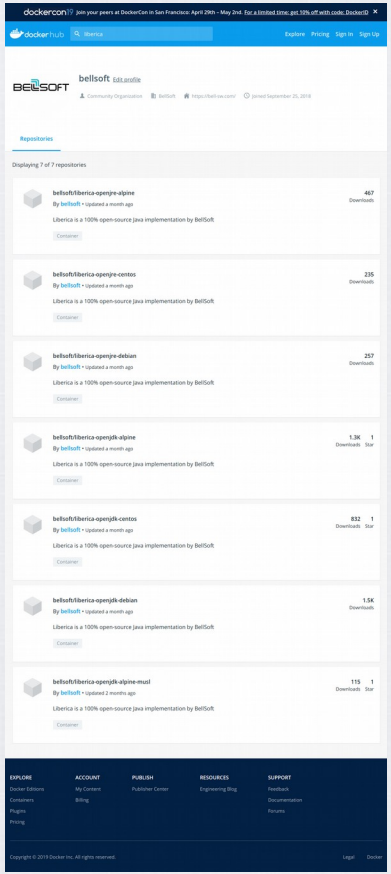
- Standard deployment nowadays
- Java 8, 11, 12, 13+ gets special support
  - cgroups
  - Namespaces
- Everyone needs base images



Base images

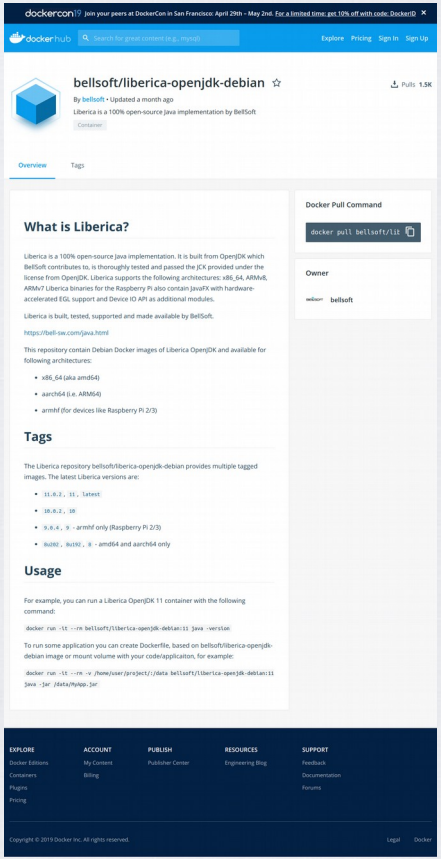
https://hub.docker.com/u/bellsoft

	JRE 8	JDK 11
Debian	227 MB	622 MB
Centos	307 MB	702 MB
Alpine	114 MB	509 MB
Alpine musl base		**37 MB**

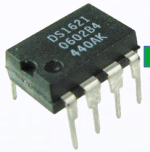


Base images

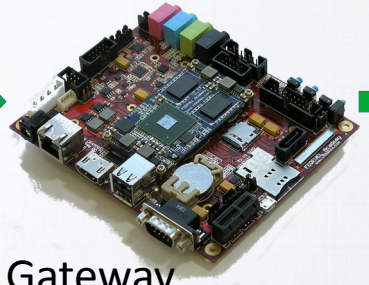
- Java versions
  - 12
  - 11 (LTS)
  - 8 (LTS)
- Distro
  - Debian
  - CentOS
  - Alpine
  - Alpine musl
- Arch
  - x86_64
  - **ARM64**
  - **ARM32**



Example



Sensor



Gateway

Streaming

Kiosk

TCK verified

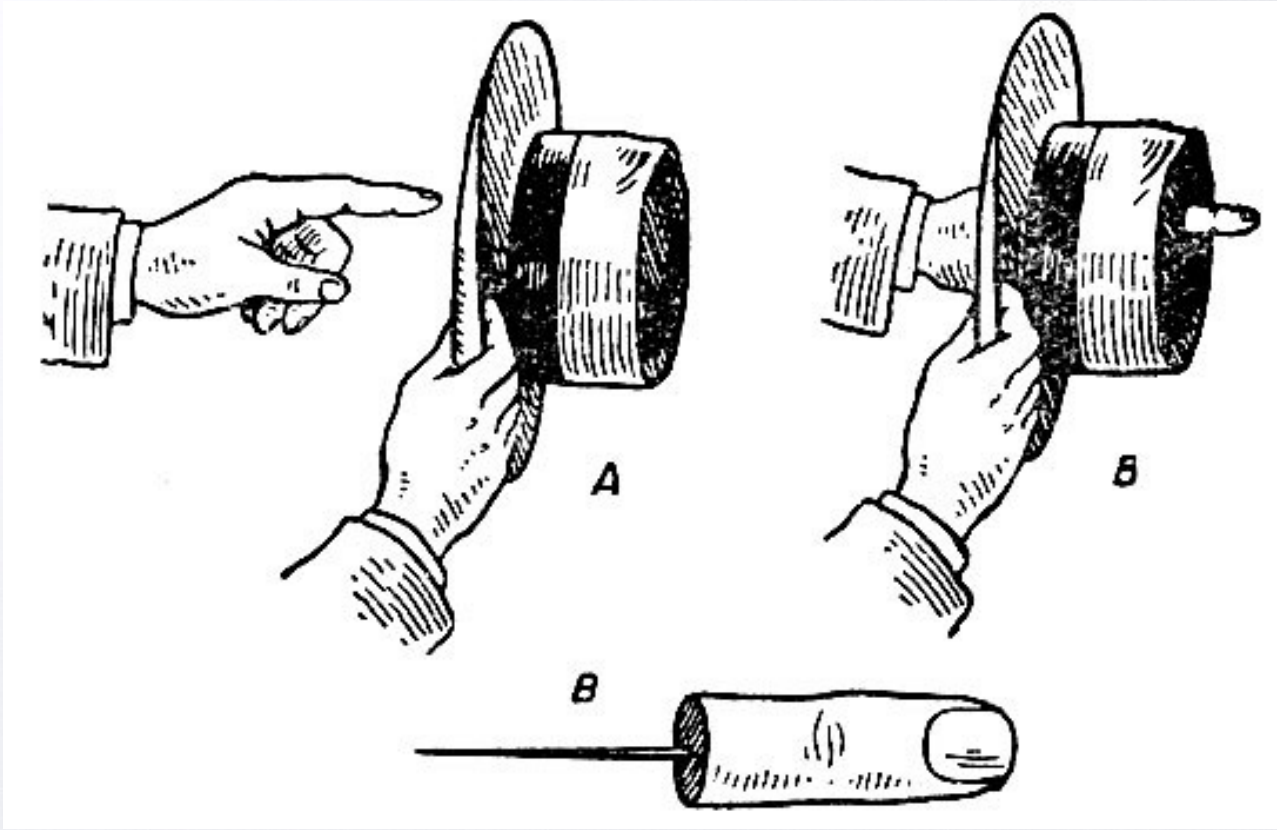
LIBERICA JDK

COMMITTED TO FREEDOM

100% OpenJDK based

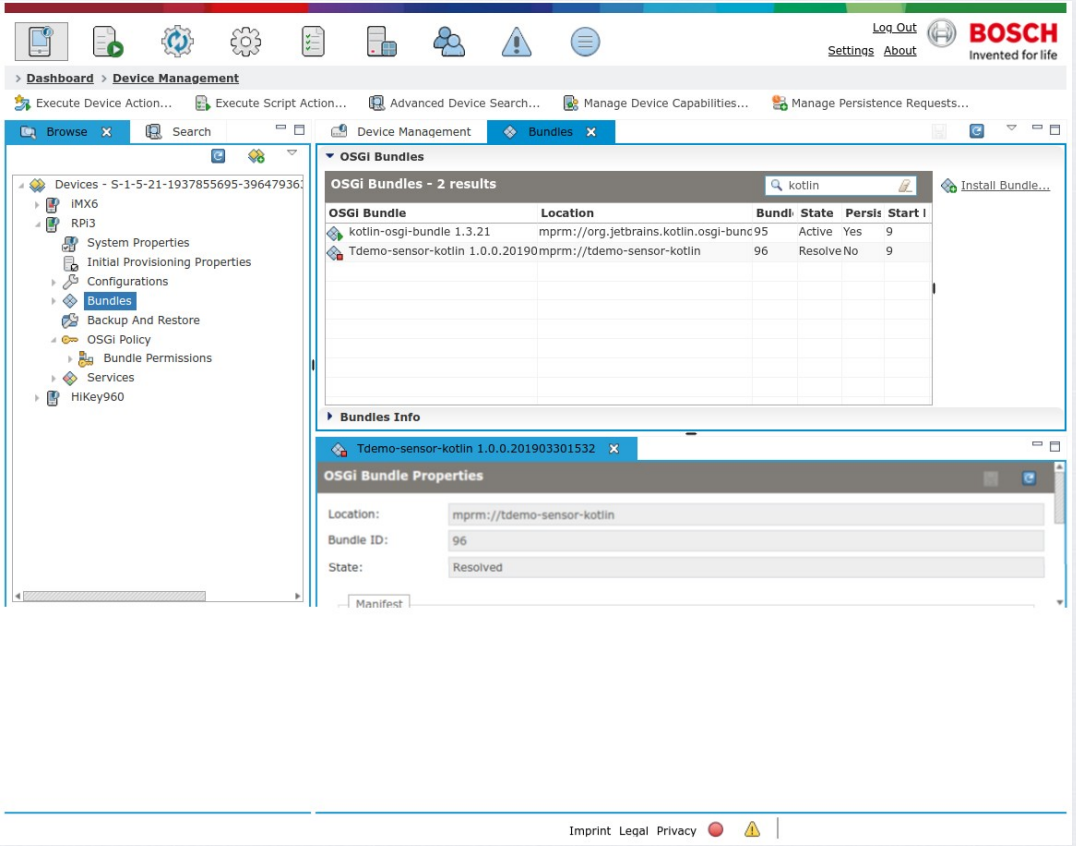
Bundle Management

Demo

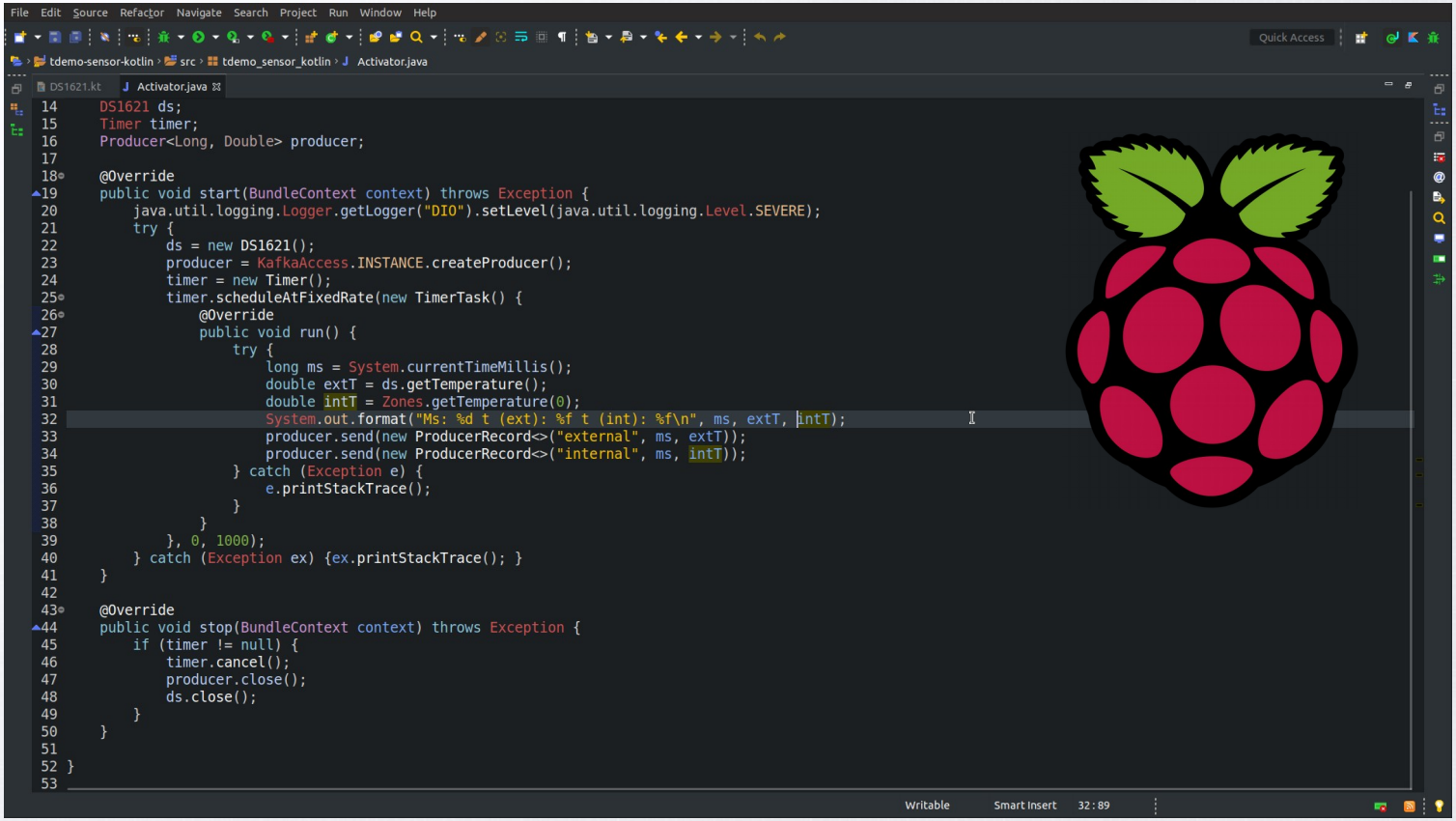


OSGi Device Management

- Manage devices
- Deploy bundles
- Control bundles
- Set permissions



Gateway. Bundle



Gateway. DIO Sensor & Kotlin

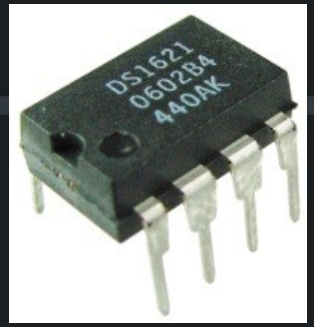
File Edit Source Refactor Navigate Search Project Run Window Help

Quick Access

tdemo-sensor-kotlin › src › tdemo_sensor_kotlin › DS1621.kt

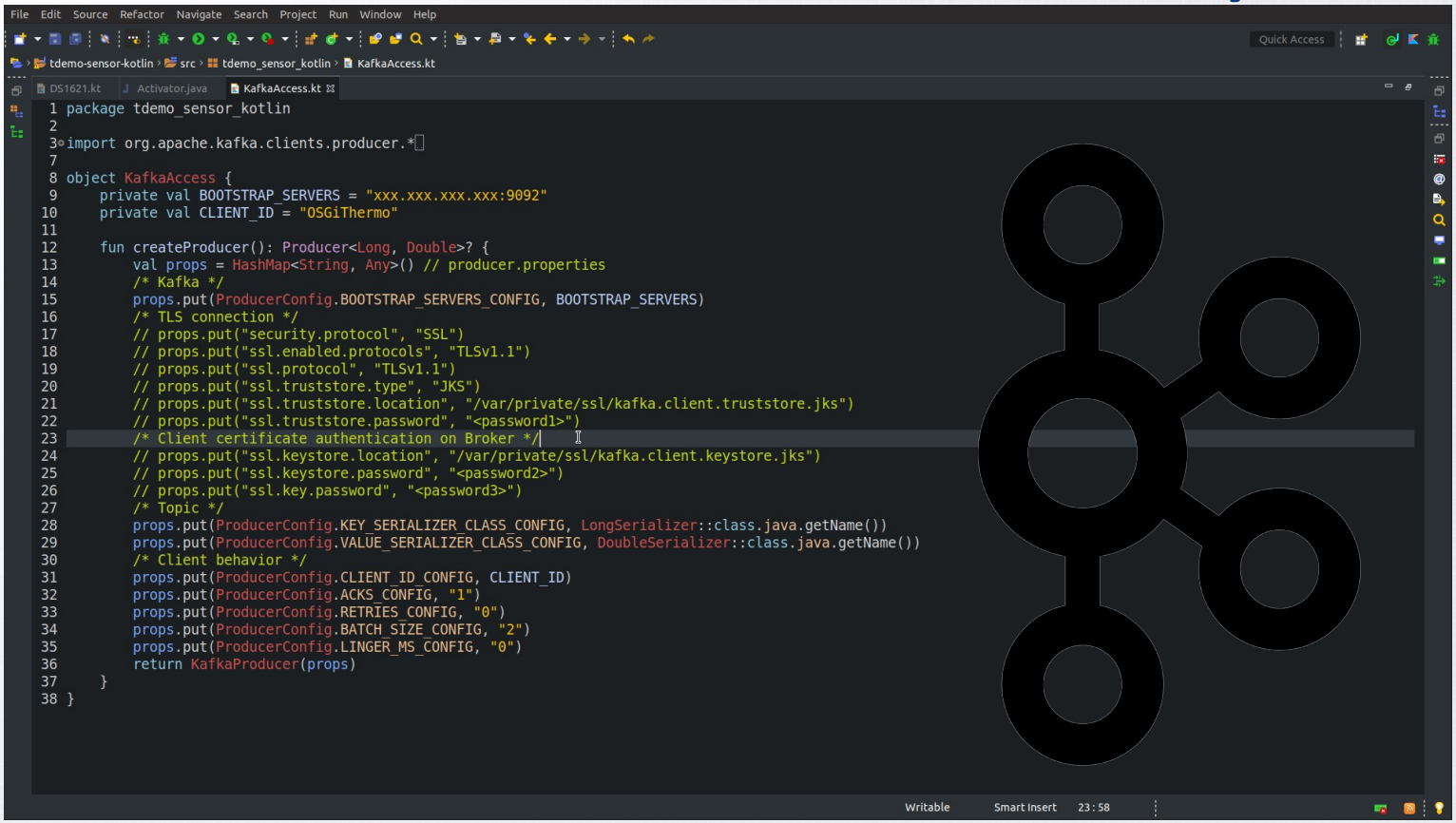
DS1621.kt

```
26      }
27
28      fun close() {
29          i2cDevice!!.close()
30      }
31  }
32
33  internal class DS1621 @Throws(IOException::class)
34  constructor() : I2CSensor() {
35      val temperature: Double
36          @Throws(IOException::class)
37          get() {
38              val cels = read(DS1621_READ_TEMP, 1)!![0].toDouble()
39              val cntr = read(DS1621_READ_CNTR, 1)!![0].toDouble()
40              val slope = read(DS1621_READ_SLOPE, 1)!![0].toDouble()
41              return cels - 0.25 + (slope - cntr) / slope
42          }
43
44      init {
45          val config = I2CDeviceConfig.Builder()
46              .setAddress(I2C_DEVICE_ADDRESS, I2CDeviceConfig.ADDR_SIZE_7)
47              .build()
48          i2cDevice = DeviceManager.open(config) as I2CDevice
49          write(DS1621_SET_ACCESS_CONFIG, 0.toByte())
50          write(DS1621_START_CONVERT)
51      }
52
53      companion object {
54          private val I2C_DEVICE_ADDRESS = 0x4f // Sensor address
55          // values from DS1621 Datasheet
56          private val DS1621_SET_ACCESS_CONFIG = 0xAC.toByte()
57          private val DS1621_START_CONVERT = 0xEE.toByte()
58          private val DS1621_READ_TEMP = 0xAA
59          private val DS1621_READ_CNTR = 0xA8 // read counter register
60          private val DS1621_READ_SLOPE = 0xA9 // read slope register
61
62          /*
63          fun main(args: Array<String>?) {
64              java.util.logging.Logger.getLogger("DIO").setLevel(java.util.logging.Level.SEVERE)
65              try {
```



Writable Smart Insert 32 : 1

Gateway. Kafka



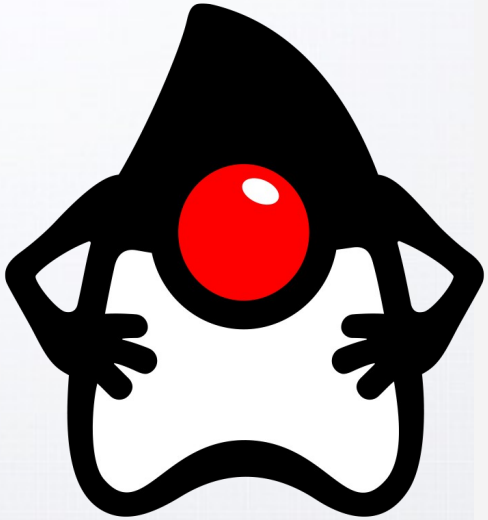
Cloud. Kafka & Docker

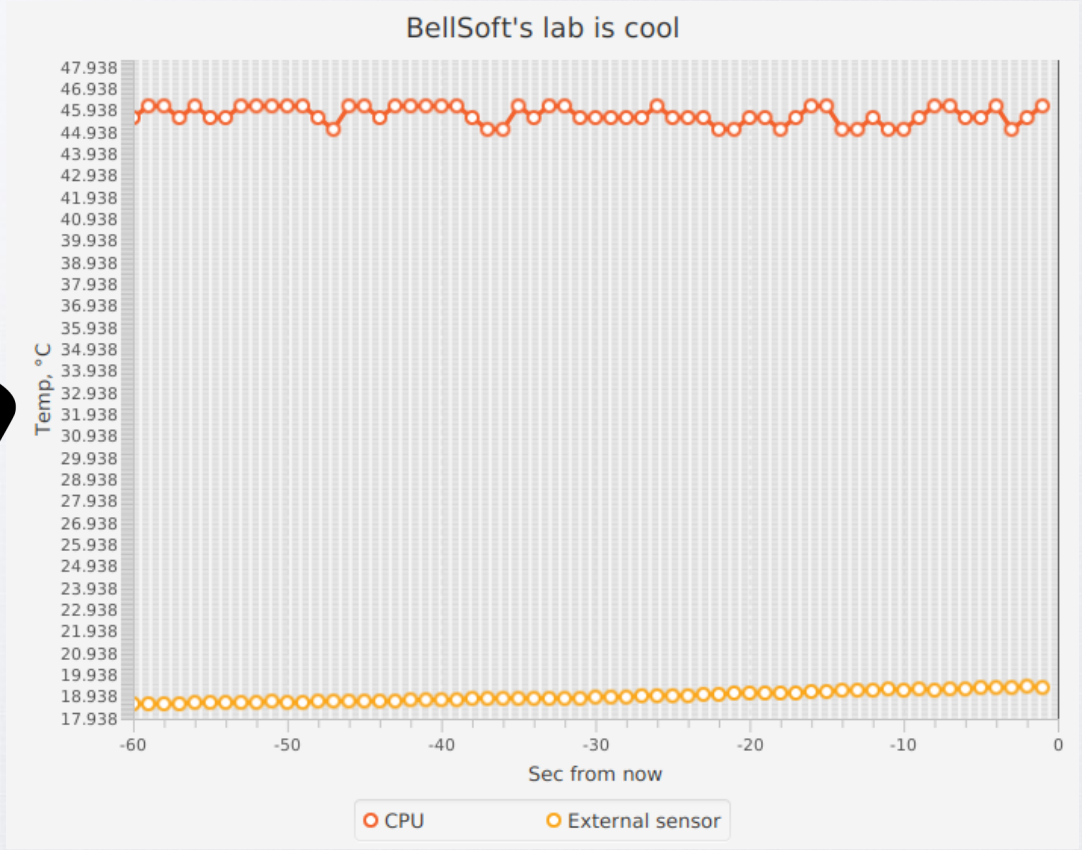

```
File  Edit  View  Search  Terminal  Help

processor       : 93
BogoMIPS        : 200.00
Features        : fp asimd evtstrm aes pmull sha1 sha2 crc32 cpuid
CPU implementer : 0x43
CPU architecture: 8
CPU variant     : 0x1
CPU part        : 0x0a1
CPU revision    : 1

processor       : 94
BogoMIPS        : 200.00
Features        : fp asimd evtstrm aes pmull sha1 sha2 crc32 cpuid
CPU implementer : 0x43
CPU architecture: 8
CPU variant     : 0x1
CPU part        : 0x0a1
CPU revision    : 1

processor       : 95
BogoMIPS        : 200.00
Features        : fp asimd evtstrm aes pmull sha1 sha2 crc32 cpuid
CPU implementer : 0x43
CPU architecture: 8
CPU variant     : 0x1
CPU part        : 0x0a1
CPU revision    : 1

dchuyko@dpochepk:~$ uname -a
Linux dpochepk 4.15.0-20-generic #21-Ubuntu SMP Tue Apr 24 06:16:20 UTC 2018 aarch64 aarch64 aarch64 GNU/Linux
dchuyko@dpochepk:~$ free
              total        used        free      shared  buff/cache   available
Mem:      131963460     4494232    64420424     1047892    63048804   125442672
Swap:       3321056      217680     3103376
dchuyko@dpochepk:~$ docker ps
CONTAINER ID        IMAGE                                COMMAND                  CREATED             STATUS              PORTS                    NAMES
26517e62eccc        bellsoft/liberica-openjdk-debian     "/kafka/bin/kafka-se…"   3 hours ago         Up 3 hours          0.0.0.0:9092->9092/tcp   kafka
08d82b4036fd        bellsoft/liberica-openjdk-debian     "/kafka/bin/zookeepe…"   3 hours ago         Up 3 hours          0.0.0.0:2181->2181/tcp   zookeeper
dchuyko@dpochepk:~$
```

packet

Kiosk. OpenJFX & Kafka

Kiosk




BellSoft's lab is cool
47.938
46.938
45.938
44.938
43.938
42.938
41.938
40.938
39.938
38.938
37.938
36.938
35.938
34.938
33.938
32.938
31.938
30.938
29.938
28.938
27.938
26.938
25.938
24.938
23.938
22.938
21.938
20.938
19.938
18.938
17.938
Temp, °C
-60
-50
-40
-30
-20
-10
0
Sec from now
CPU
External sensor

Summary

- Liberica JDK on Arm brings value to the ecosystem from edge or devices to cloud
  - Min runtime: 17 Mb footprint and energy savings
  - Max runtime: Optimized for throughput, low latency on high-end ARM servers
- JVM runtime unleashes polyglot programming approach
  - Java, Scala, Kotlin, you name it
- Running on JVM runtime allows to re-use applications and libraries on ARM hardware with zero changes
- Liberica JDK is available for edge and cloud
  - Free for use and distribution
  - With commercial support option (24x7 worldwide coverage extending from servers and clouds to edge and any screen)