与「十」俱进，码出未来

2018源创会年终盛典

主办方 开源中国 oschina.net

重新认识现代 Java™

自我介绍

- 杨晓峰
- OpenJDK Committer，京东大数据中心架构师，JVM/Performance
- 曾领导Oracle北京Java核心类库等团队，负责或参与了Java 8～11核心类库等部分新特性相关任务
- 出品了畅销专栏《Java核心技术36讲》等
- 欢迎加入： yangxiaofeng@jd.com

日程

- Java，一个经常“被”不行了的语言
- 重新认识今天的Java
- 未来Java/JVM之路

日程

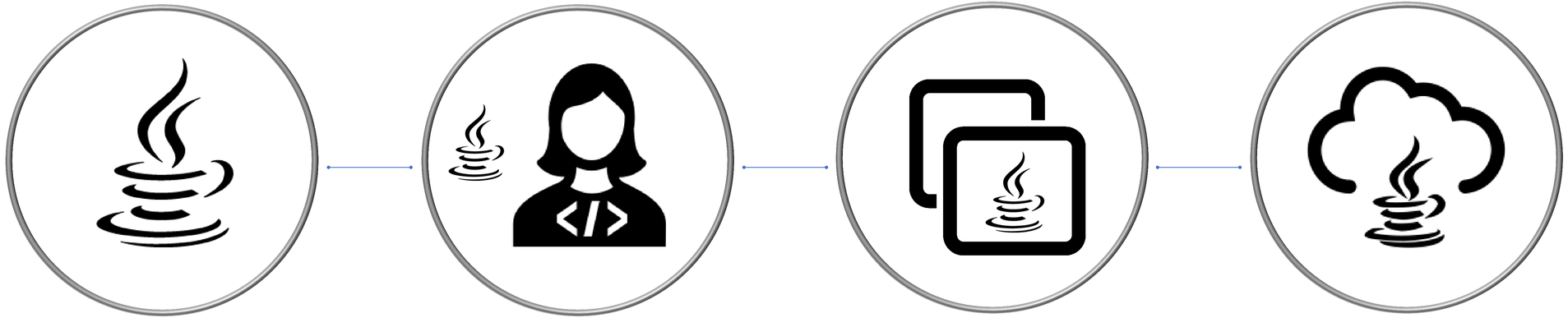
- **Java，一个经常“被”不行了的语言**
- 重新认识今天的Java
- 未来Java/JVM之路

自媒体的眼中 – Java年年都要不行了

- 作为高利贷、保险和爸妈之外最关心我们的人：
  - “**时代抛弃你不会和你打招呼**，Java没人用了，要死了。。。”
  - “面向对象早过时了，函数式才高效！”
  - “臃肿，慢！”
  - “语法太繁琐，手疼！”
  - …

快跑吧，Java要收费了！

现实中的Java – 无与伦比的生态



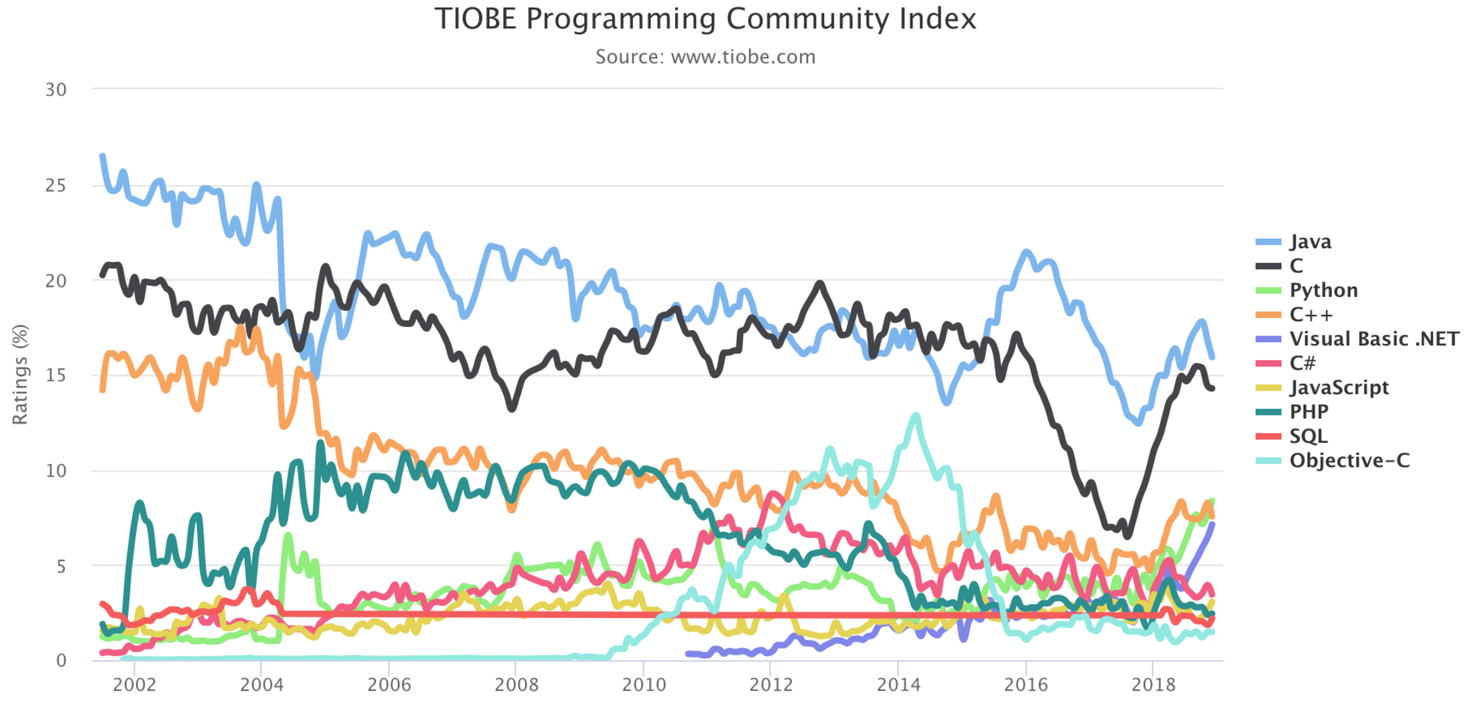
NO.1
编程语言

1200W+
开发者

380亿+
活跃JVM

210亿+
连接到云的JVM

长期处于编程语言榜首


TIOBE Programming Community Index
Source: www.tiobe.com
Ratings (%)
30
25
20
15
10
5
0
2002
2004
2006
2008
2010
2012
2014
2016
2018
Java
C
Python
C++
Visual Basic .NET
C#
JavaScript
PHP
SQL
Objective-C

Java仍然免费!

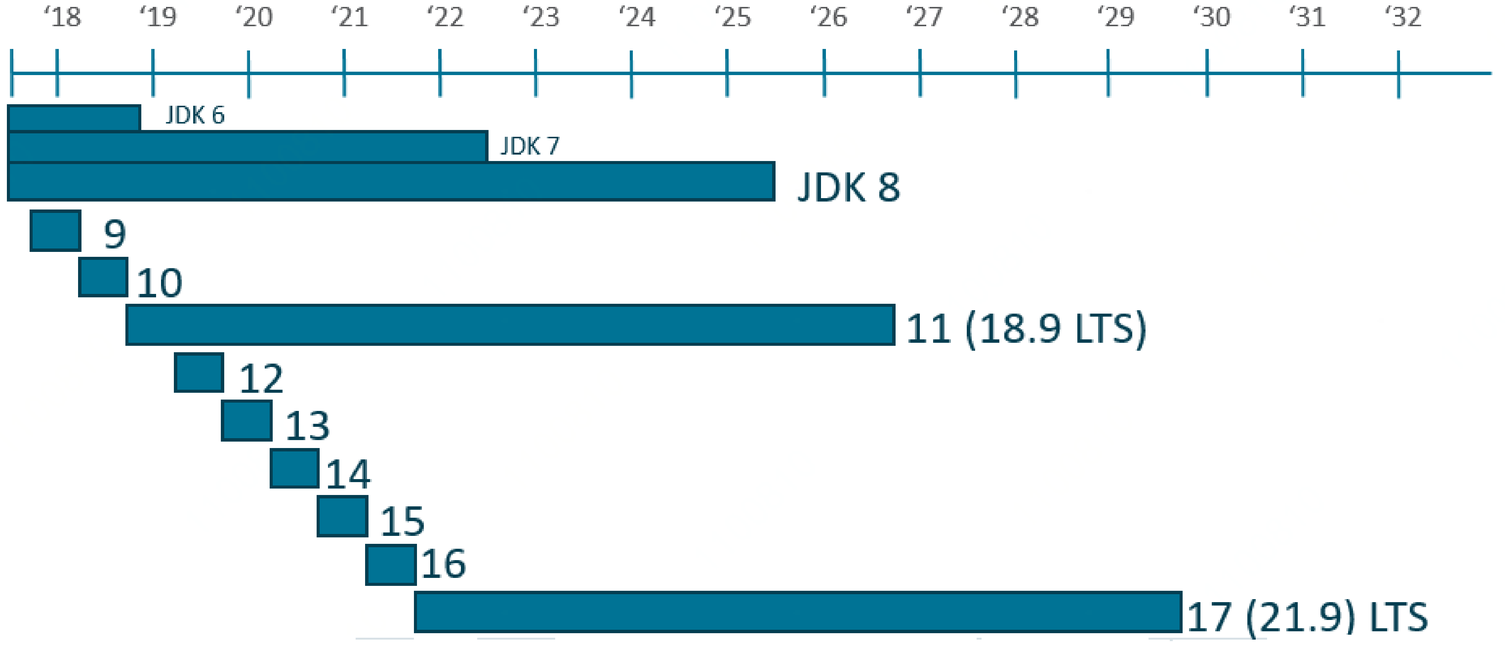
- Oracle开源了AppCDS、JFR、JMC、ZGC等强大的特性
- 针对不同用户，提供了：
  - 商业授权的**Oracle JDK** 和 GPL v2+CPE授权的**OpenJDK**
  - **除了License，没有任何不同！**
- 可以从各种渠道获得JDK发行版
  - Oracle, AdoptOpenJDK, Redhat, Azul, Amazon (Corretto)…
  - **长期（免费/商业）升级**支持

更多厂商投入+更加活跃的OpenJD

- 以自身为例：
- 京东大数据中心已经签署了OCA，并积极参与
- 有且不仅：
  - 深入JVM或者性能等领域的场景
  - 数万台高性能服务器
  - 海量数据+ 各种前沿负载，大数据、机器学习…

JDK新的发布模式

- 特性驱动→ 时间驱动
- 每半年一个主版本
- 每3年一个长期支持版本(LTS)



日程

- Java，一个经常“被”不行了的语言
- **重新认识今天的Java**
- 未来Java/JVM之路

我们真的了解Java吗？

- Java在人的心目中：
  - 纯粹面向对象
  - 跨平台 → 一次编译，任意执行
  - 安全、可靠
  - 强大的并发、IO等
  - 但，笨重的工具，教条、古板的语法...

With some help from Lynda.com, we've compiled a list of 10 of the most sought-after programming languages to get you up to speed.

1. Java

IMAGE: MASHABLE COMPOSITE. IMAGE: WIKIMEDIA COMMONS

What it is: Java is a class-based, object-oriented programming language developed by Sun Microsystems in the 1990s. It's one of the most in-demand programming languages, a standard for enterprise software, web-based content, games and mobile apps, as well as the Android operating system. Java is designed to work across multiple software platforms, meaning a program written on Mac OS X, for example, could also run on Windows.

Where to learn it: Udemy, Lynda.com, Oracle.com, LearnJavaOnline.org.

2. C Language

Java真的“很慢”？

- 抛开场景说快慢是无意义的
  - 长时间运行的服务器端场景的无争议霸主
  - 在微服务等新型架构下，Java/JVM仍然是首选
  - 云计算的NO.1语言，电商、大数据、企业软件、IoT、移动设备…
- 京东、阿里、亚马逊、Google、IBM、Oracle、Netflix、Twitter…

事实证明了其性能和扩展能力！

Java也可以玩儿转函数式

- JDK 8引入Lambda/Stream：

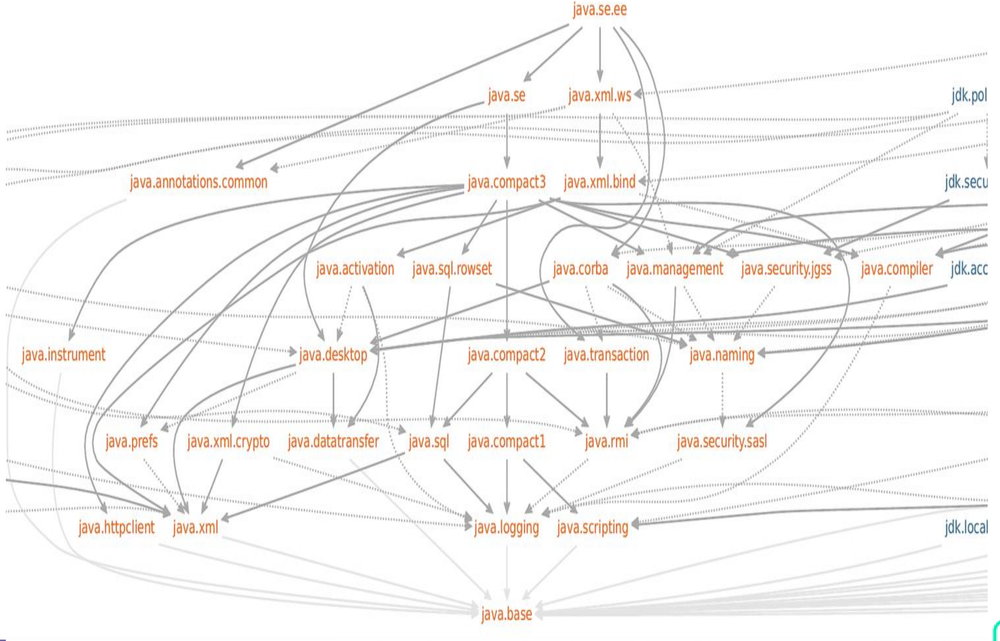
```
List<Block> blocks = List.of(…);
int maxWeight = blocks.stream()
                      .filter(block -> block.getColor() == BLUE)
                      .mapToInt(blue -> blue.getWeight())
                      .max();
```

轻松又强大的并行计算支持

```
double average = roster.parallelStream()
                        .filter(p -> p.getGender() == Person.Sex.MALE)
                        .mapToInt(Person::getAge)
                        .average()
                        .getAsDouble();
```

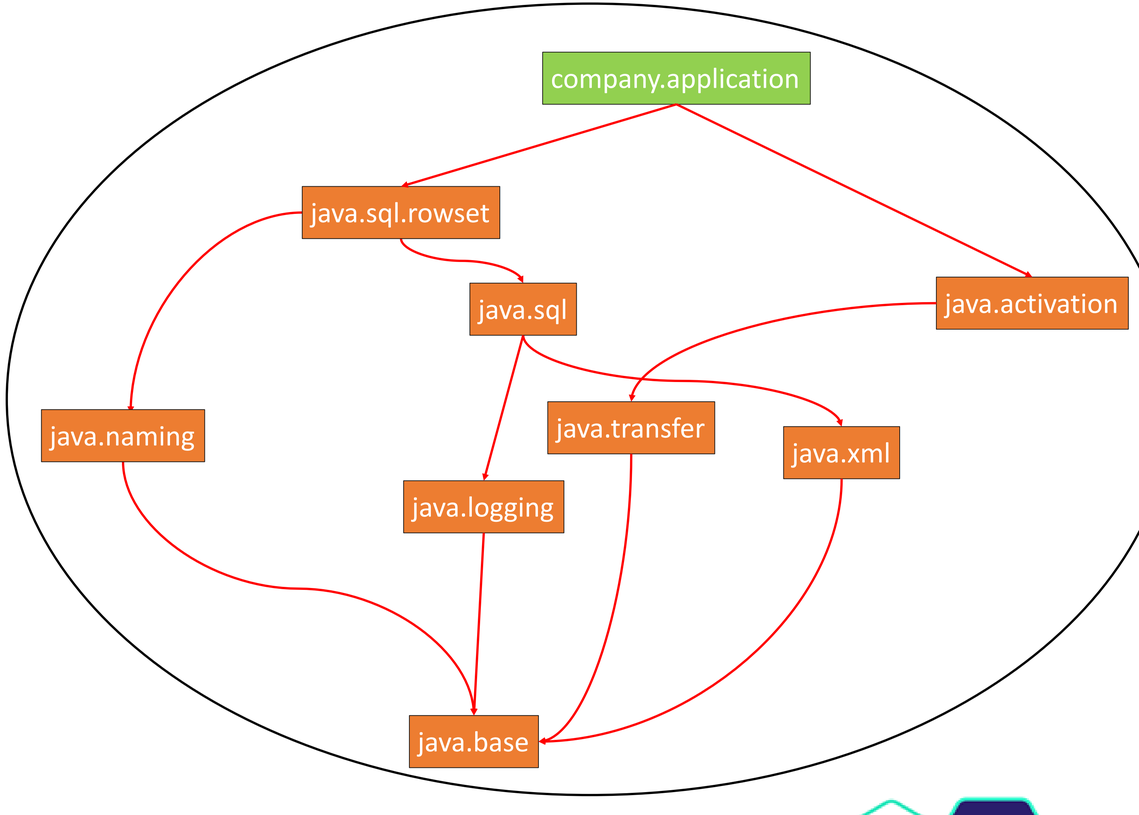
JDK[redacted] 可以很“苗条”（1）

- JDK 9 Jigsaw项目（JPMS，Java Platform Module System）
  - Java平台模块化系统
  - 未来Java快速演进的基础
  - 快速地得到了主流工具的支持



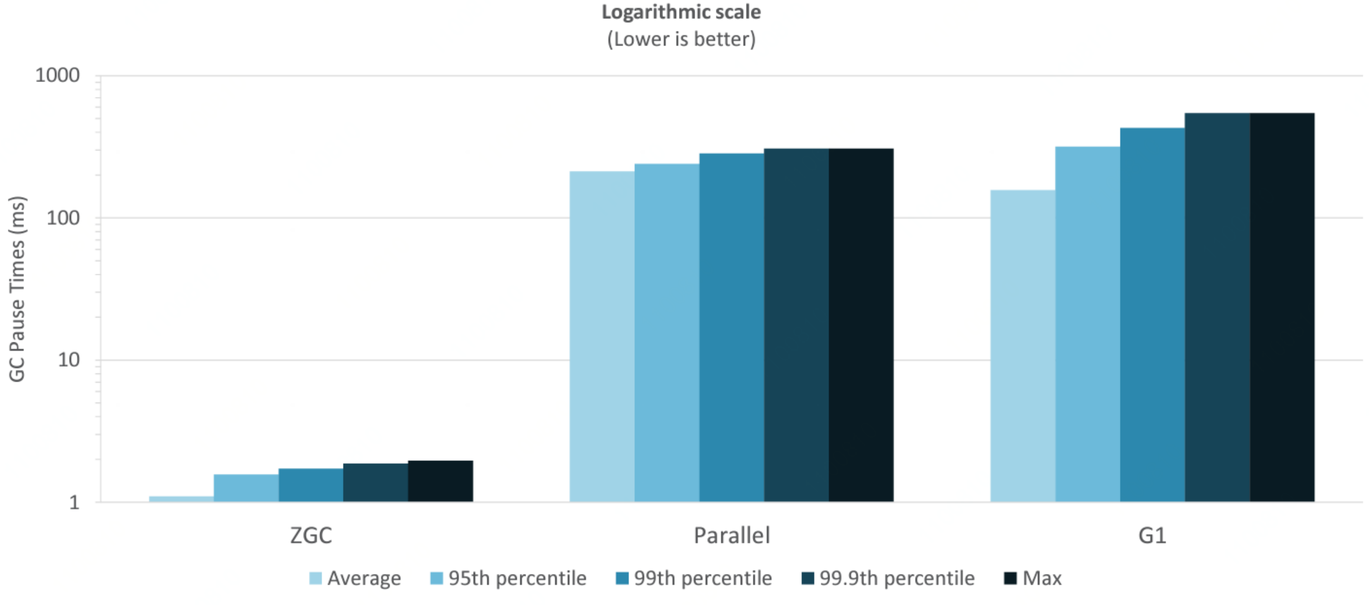
定制轻量级Java Runtime

```
$ jlink --module-path jmods/ \
        --add-modules java.sql.rowset,java.activation \
        --output myimage
#本例定制后只有~40M bytes
$ myimage/bin/java –m company.application
```



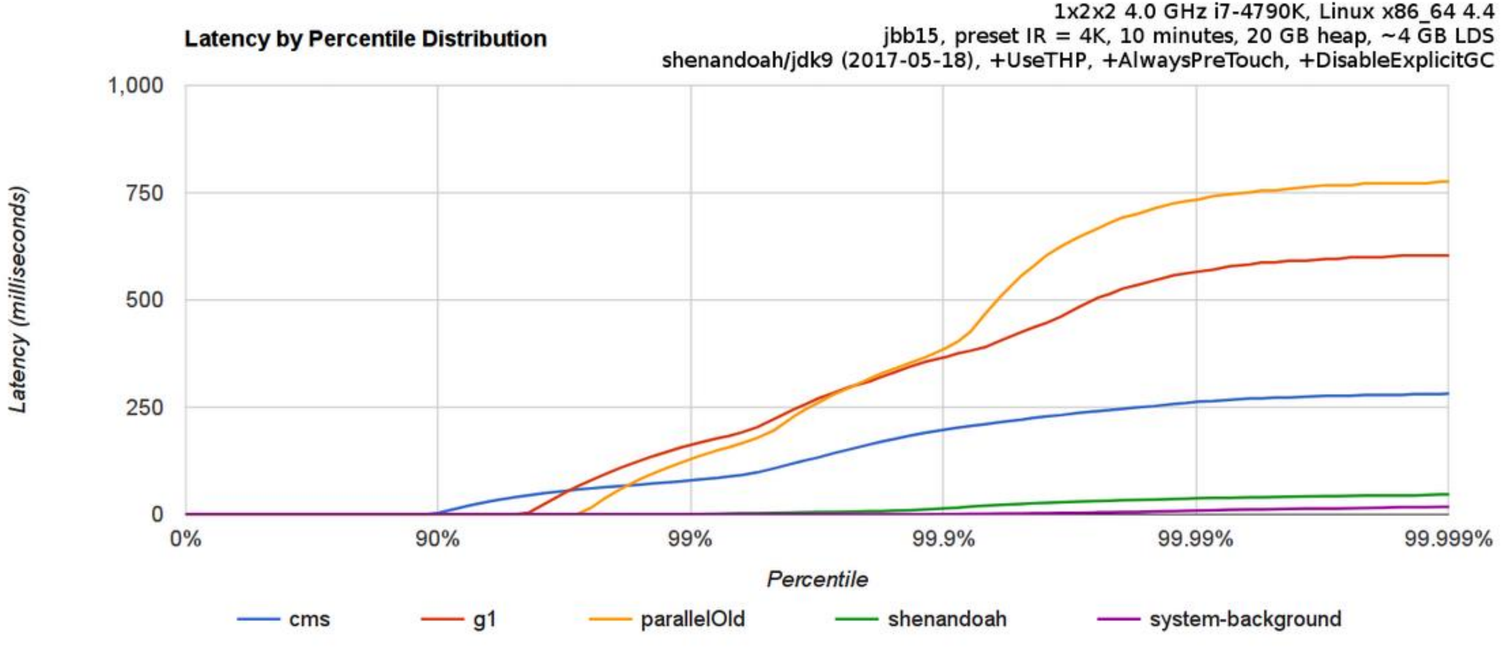
Java GC 越来越智能（1）

- ZGC
  - Oracle开源
  - 轻松扩展到T bytes Heap
  - 低于10ms的暂停
  - 暂停不随堆变大而增长
  - 良好的吞吐量



Java GC 越来越智能（2）

- Shenadoah GC
  - Redhat开源
  - 已经集成到JDK 12
  - 另一个Pauseless GC！

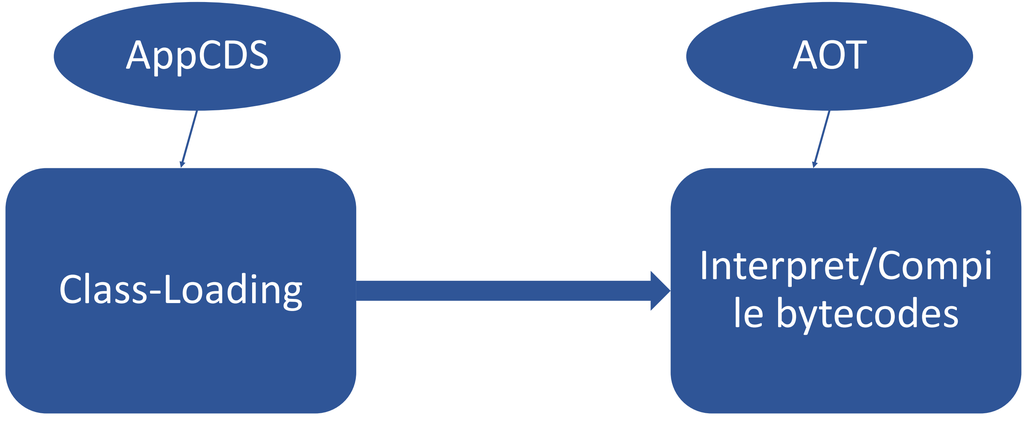


Java GC 越来越很智能（3）

- G1 GC
  - 从JDK 9开始， server模式下的默认GC
  - 简化的GC调优
- 不断完善：
  - Parallel Full GC
  - Faster card scan
  - Abortable Mixed Collector
  - Promptly Return Unused Committed Memory
  - …

Java应用启动也可以很快

- JDK启动加速：
  - 模块化
  - AOT
  - CDS/AppCDS
- 获取30+%的加速并不困难
- 使用Graal甚至获得数量级程度的提高



更加现代化的类库

- Compact Strings
- HTTP/2 Client
- Process API Updates
- Convenience Factory Methods for Collections
- Variable Handles
- …

不断简化的语法 -- 本地变量类型推断

```
Url url                    = new URL(“http://www.oracle.com/”);
URLConnection conn = url.openConnection();
Reader reader              = new InputStreamReader(conn.getInputStream());

var url          = new URL(“http://www.oracle.com/”);
var conn         = url.openConnection();
var reader       = new InputStreamReader(conn.getInputStream());
```

不再古板的体验（1）

JDK 9中引入JShell

- ${JAVA_HOME}/bin/jshell

```
jshell>/help
jshell> ProcessHandle ph = ProcessHandle.current();
jshell> ph.getPid();
jshell> ph.info().command();
```

不再古板的体验（2）

- 试验Hello World也要至少要两步：编译 →运行？

~~${JAVA_HOME}/bin/javac HelloWorld.java~~

// JDK11之后，可以直接执行简单Java文件

${JAVA_HOME}/bin/java HelloWorld.java

日程

- Java，一个经常“被”不行了的语言
- 重新认识今天的Java
- **未来Java/JVM之路**

Java/JVM的未来

- **By Rich Sharples, Redhat：**

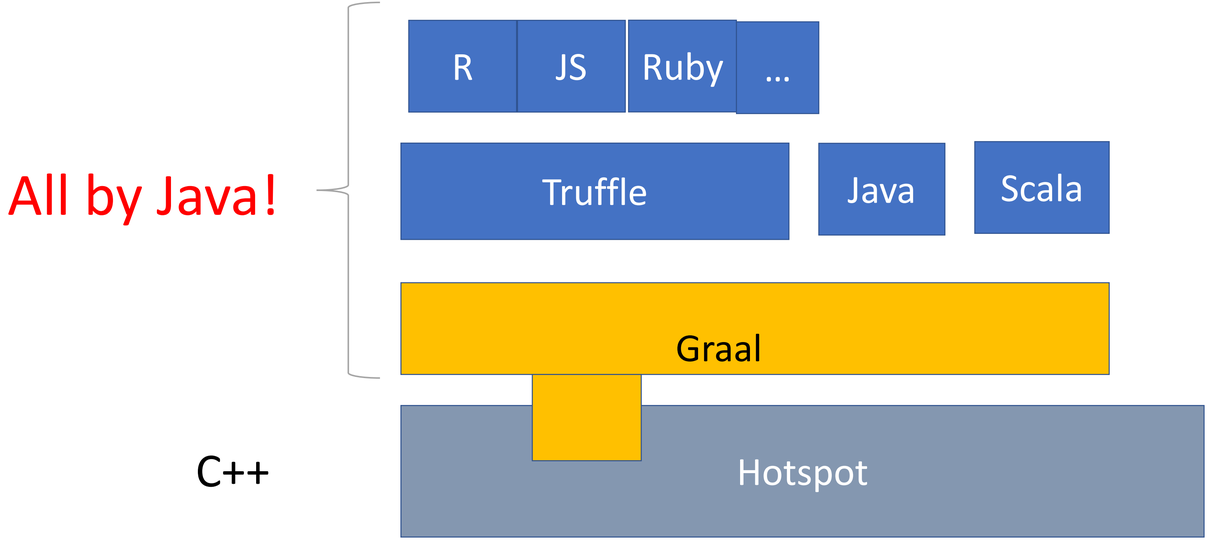
  “

  Java 必须超越**当前的位置，也就是作为大规模关键业务应用程序的可扩展语言运行时**，并**与更轻量、更灵活的语言和运行时展开竞争——**特别是在内存占用和延迟非常敏感的云原生环境中。

  ”

更加轻量的JVM

- Graal (Oracle Lab出品）很可能是未来
  - Java on Java !
  - 目前已经是AOT的实现基础
  - 未来也许会逐渐替换 C2 → C1 →其他
  - 强大的多语言支持
  - 轻量的SubstrateVM



另一个突破性的飞跃

- Valhalla：
  - Value types
  - 原生的Immutable
  - 原始数据类型泛型支持
  - 更灵活的数据类型: tuple, vector…

更高的开发效率和基础能力

- Java也会有协程 – Project Loom
  - Fiber
  - Continuation
- Panama
  - 改进本地代码开发效率 → 终于不用头痛JNI了
  - 更好的底层支持，例如，Vector API，充分利用CPU计算能力
  - 进一步优化大数据、机器学习等应用场景

更人性和高效的语法

```
// 目前的语法
String html = "<html>\n" +
          "    <body>\n" +
          "                <p>Hello World.</p>\n"
+
          "    </body>\n" +
          "</html>\n";
 System.out.print(html);
```

VS

```
// 新的语法
String html = `
              <html>
                <body>
                  <p>Hello World.</p>
                </body>
              </html>
            `.align();
  System.out.print(html);
```

Pattern Matching

- 不仅是语法“简化”，更是**思维方式**的转变
- 目前的繁琐代码：

```
String formatted = "unknown";
if (obj instanceof Integer) {
    int i = (Integer) obj;
    formatted = String.format("int %d", i);
}
else if (obj instanceof Byte) {
    byte b = (Byte) obj;
    formatted = String.format("byte %d", b);
}
else if (obj instanceof Long) {
    long l = (Long) obj;
    formatted = String.format("long %d", l);
}
else if (obj instanceof Double) {
    double d = (Double) obj;
    formatted = String.format(“double %f", d);
}
else if (obj instanceof String) {
    String s = (String) obj;
    formatted = String.format("String %s", s);
}
```

改进的语法

```
String formatted;
switch (obj) {
    case Integer i: formatted = String.format("int %d", i); break;
    case Byte b:    formatted = String.format("byte %d", b); break;
    case Long l:    formatted = String.format("long %d", l); break;
    case Double d:  formatted = String.format(“double %f", d); break;
    case String s:  formatted = String.format("String %s", s); break
    default:        formatted = obj.toString();
}
```

Statements → Expressions!

```
String formatted =
    switch (obj) {
        case Integer i -> String.format("int %d", i);
        case Byte b    -> String.format("byte %d", b);
        case Long l    -> String.format("long %d", l);
        case Double d  -> String.format("double %f", d);
        case String s  -> String.format("String %s, s);
        default        -> String.format("Object %s", obj);
    };
```

谢谢！

欢迎加入京东： yangxiaofeng@jd.com