JDK 11特性解析与发展趋势

杨晓峰

Principal Member of Technical Staff,
Java Platform Group
Oracle, Corp

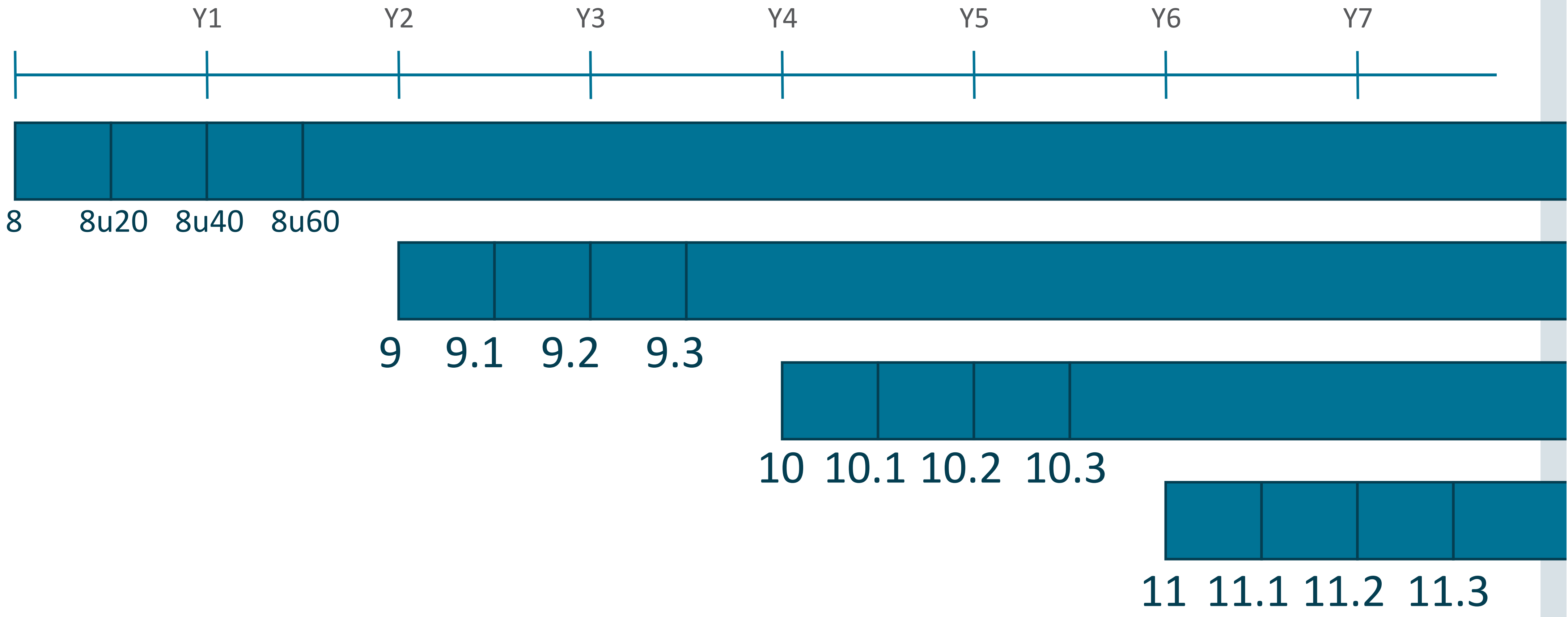
Safe Harbor Statement

日程

- 新的JDK发布模式
- JDK 11特性解析
  - 类库
  - 语言和工具
  - JVM
- Java/JVM发展趋势

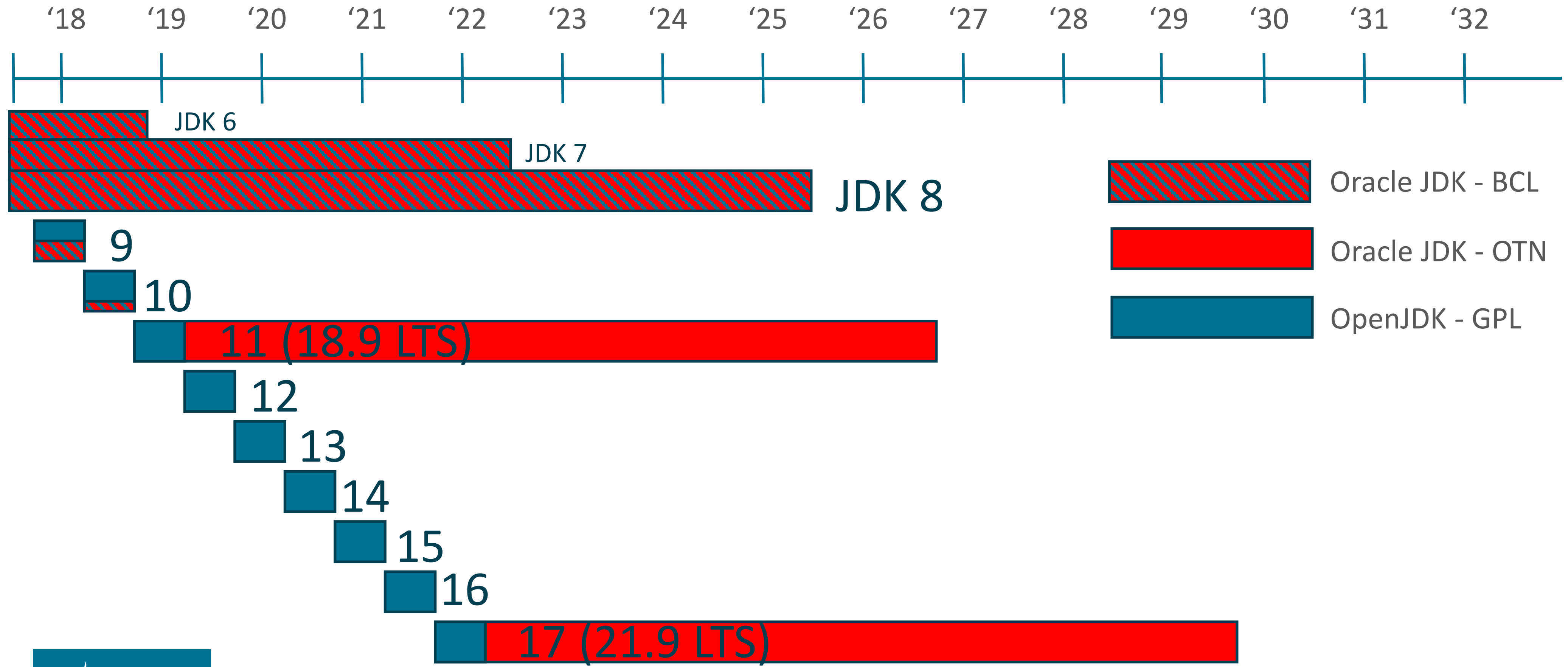
求求你，别再更新，真学不动了！

以前的JDK发布模式

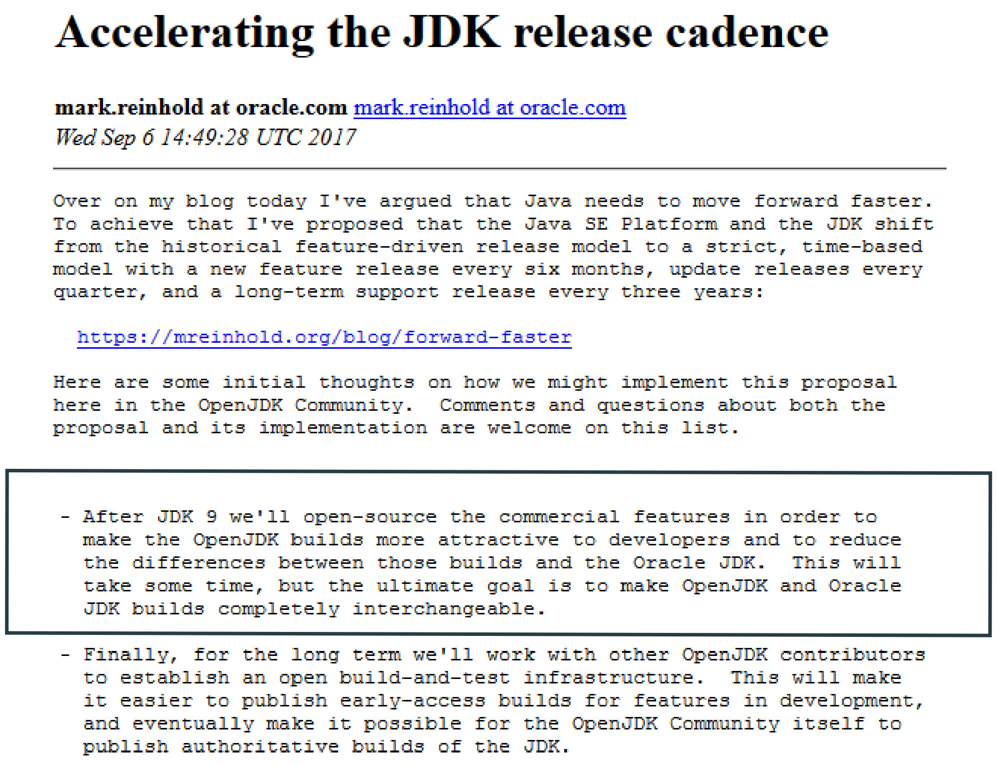


Y1
Y2
Y3
Y4
Y5
Y6
Y7
8
8u20
8u40
8u60
9
9.1
9.2
9.3
10
10.1
10.2
10.3
11
11.1
11.2
11.3

Oracle JDK & OpenJDK



更加开放的Java！



Accelerating the JDK release cadence

mark.reinhold at oracle.com mark.reinhold at oracle.com
Wed Sep 6 14:49:28 UTC 2017

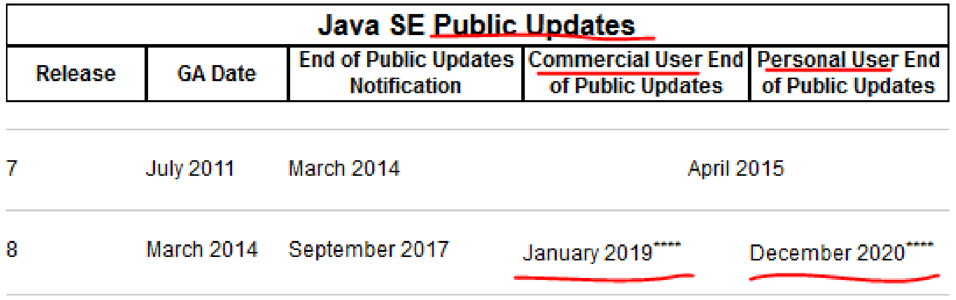
```
Over on my blog today I've argued that Java needs to move forward faster.
To achieve that I've proposed that the Java SE Platform and the JDK shift
from the historical feature-driven release model to a strict, time-based
model with a new feature release every six months, update releases every
quarter, and a long-term support release every three years:

  https://mreinhold.org/blog/forward-faster

Here are some initial thoughts on how we might implement this proposal
here in the OpenJDK Community.  Comments and questions about both the
proposal and its implementation are welcome on this list.

  - After JDK 9 we'll open-source the commercial features in order to
    make the OpenJDK builds more attractive to developers and to reduce
    the differences between those builds and the Oracle JDK.  This will
    take some time, but the ultimate goal is to make OpenJDK and Oracle
    JDK builds completely interchangeable.

  - Finally, for the long term we'll work with other OpenJDK contributors
    to establish an open build-and-test infrastructure.  This will make
    it easier to publish early-access builds for features in development,
    and eventually make it possible for the OpenJDK Community itself to
    publish authoritative builds of the JDK.
```

- 更快速的迭代
- Oracle提供了更多选择
  - OpenJDK builds - GPL V2 with CPE
  - Oracle JDK
- 开源Oracle JDK的商业特性
- 构建更现代的基础设施

URL: http://mail.openjdk.java.net/pipermail/discuss/2017-September/004281.html

是时候考虑升级JDK了

- JDK 8已经快5年了



Java SE Public Updates				
Release	GA Date	End of Public Updates Notification	Commercial User End of Public Updates	Personal User End of Public Updates
7	July 2011	March 2014	April 2015	
8	March 2014	September 2017	January 2019****	December 2020****

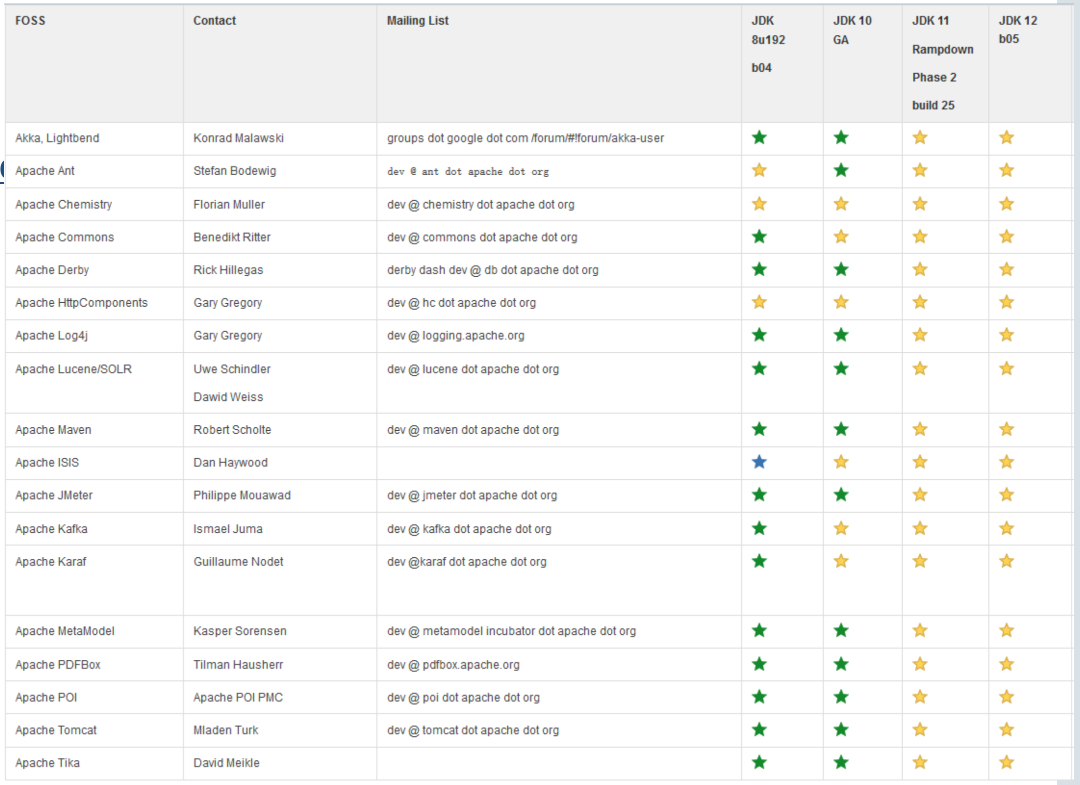
- 无支持的老版本JDK意味着错失：
  - 最新的安全更新
  - 大量的新特性、Bug修复
  - 接近零成本的性能优化
  - …

JDK 11会是好的选择吗

- 主流开源项目密切支持新版JDK:

https://wiki.openjdk.java.net/display/quality/Quality+Outr

- 大量的新特性 + 开源的商业特性:
  - New HTTP Client
  - TLS 1.3
  - ZGC
  - JMC/JFR
  - APPCDS
  - …



FOSS	Contact	Mailing List	JDK 8u192 b04	JDK 10 GA	JDK 11 Rampdown Phase 2 build 25	JDK 12 b05
Akka, Lightbend	Konrad Malawski	groups dot google dot com /forum/#!forum/akka-user	🟩	🟩	🟨	🟨
Apache Ant	Stefan Bodewig	dev @ ant dot apache dot org	🟨	🟩	🟨	🟨
Apache Chemistry	Florian Muller	dev @ chemistry dot apache dot org	🟨	🟨	🟨	🟨
Apache Commons	Benedikt Ritter	dev @ commons dot apache dot org	🟩	🟨	🟨	🟨
Apache Derby	Rick Hillegas	derby dash dev @ db dot apache dot org	🟩	🟩	🟨	🟨
Apache HttpComponents	Gary Gregory	dev @ hc dot apache dot org	🟨	🟨	🟨	🟨
Apache Log4j	Gary Gregory	dev @ logging.apache.org	🟩	🟩	🟨	🟨
Apache Lucene/SOLR	Uwe Schindler Dawid Weiss	dev @ lucene dot apache dot org	🟩	🟩	🟨	🟨
Apache Maven	Robert Scholte	dev @ maven dot apache dot org	🟩	🟩	🟨	🟨
Apache ISIS	Dan Haywood		🟦	🟨	🟨	🟨
Apache JMeter	Philippe Mouawad	dev @ jmeter dot apache dot org	🟩	🟩	🟨	🟨
Apache Kafka	Ismael Juma	dev @ kafka dot apache dot org	🟩	🟨	🟨	🟨
Apache Karaf	Guillaume Nodet	dev @karaf dot apache dot org	🟩	🟨	🟨	🟨
Apache MetaModel	Kasper Sorensen	dev @ metamodel incubator dot apache dot org	🟩	🟩	🟨	🟨
Apache PDFBox	Tilman Hausherr	dev @ pdfbox.apache.org	🟩	🟩	🟨	🟨
Apache POI	Apache POI PMC	dev @ poi dot apache dot org	🟩	🟩	🟨	🟨
Apache Tomcat	Mladen Turk	dev @ tomcat dot apache dot org	🟩	🟩	🟨	🟨
Apache Tika	David Meikle		🟩	🟩	🟨	🟨

Java 11

- 类库
- 语言和工具
- JVM

HTTP Client API （标准化）

- 终于毕业了！
- 新的模块/package：java.net.http/java.net.http
- 支持多种协议:
  - HTTP/2 -- Binary, multiplex, Server Push, Header Compression (Hpack), Priority
  - HTTP/1.1
  - WebSocket
- 高性能和扩展性
  - 非阻塞
  - Reactive Stream机制

HTTP Client API介绍

- 简单易用的API：
  - HttpClient
  - HttpRequest/HttResponse
  - WebSocket
- Fluent风格
- 大量使用CompletableFuture
- 支持全面的连接方式
  - HTTP/2 over clear text TCP (h2c)
  - HTTP/2 over TLS (h2)

HTTP Client API使用

// 构建Client

```
HttpClient client = HttpClient.newBuilder()
       .version(Version.HTTP_2)
        .sslContext(yourSslCtx)
       .proxy(ProxySelector.of(inetAddr))
        .build();
```

// 构建 request

```
HttpRequest request = HttpRequest.newBuilder()
       .uri(URI.create("https://www.github.com/"))
       .GET()
        .build();
```

同步方式：

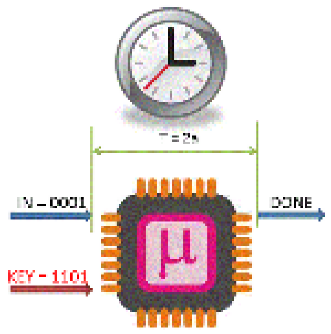
```
  HttpResponse<String> resp = client.send(request,
BodyHandlers.ofString());
   System.out.println(response.statusCode());
   System.out.println(response.body());
```

异步方式：

```
client.sendAsync(request, BodyHandlers.ofString())
      .thenApply(HttpResponse::body)
      .thenAccept(System.out::println);
```

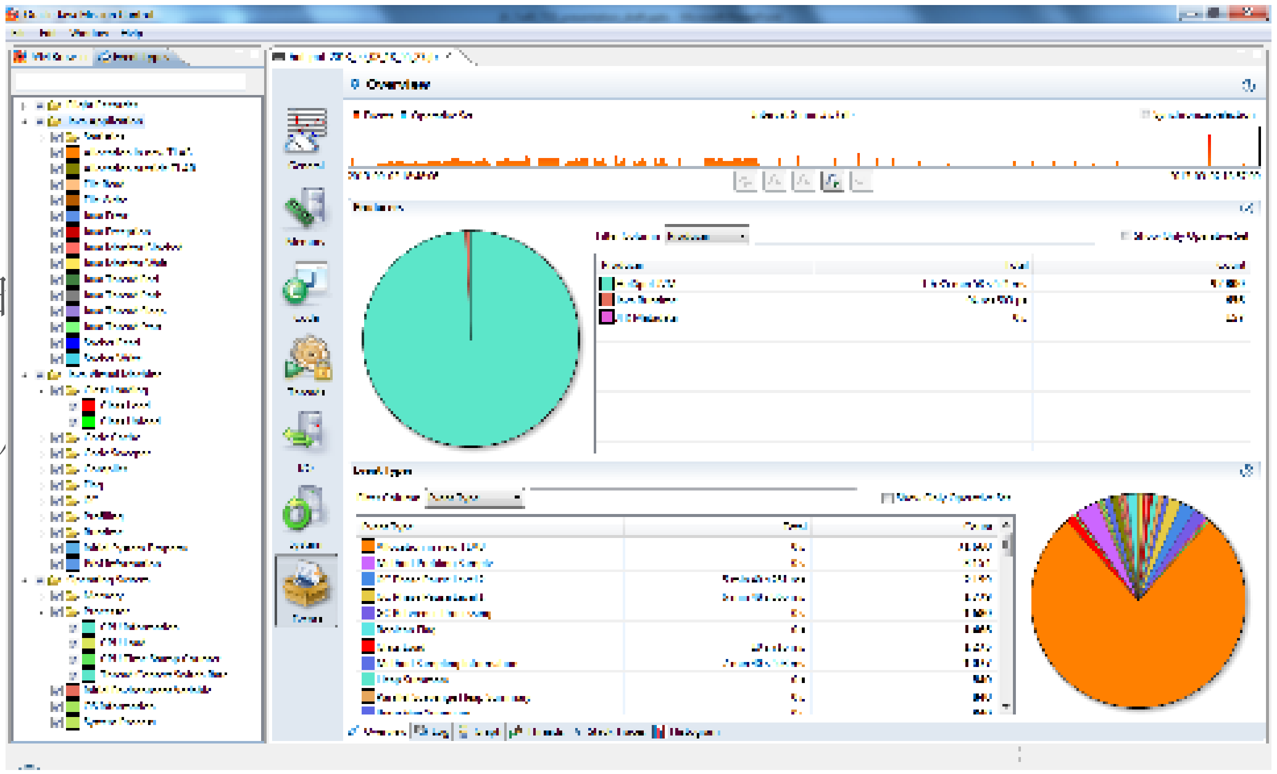
安全类库

- Transport Layer Security (TLS) 1.3
  - 安全标准的重大革新
  - 大幅提高性能和安全
  - 保持Java平台的竞争力
- ChaCha20 and Poly1305 Cryptographic Algorithms
- Key Agreement With Curve25519 and Curve448
  - 纯Java实现
  - Constant time math API



JFR： Flight Recorder

- 基于事件机制的跟踪框架
  - Everything is an Event!
  - 深入集成到JDK/JVM内部
  - 极致性能，具备生产系统开启
  - 方便的API （Java, C/C++）
  - 配合Java Mission Control等工具

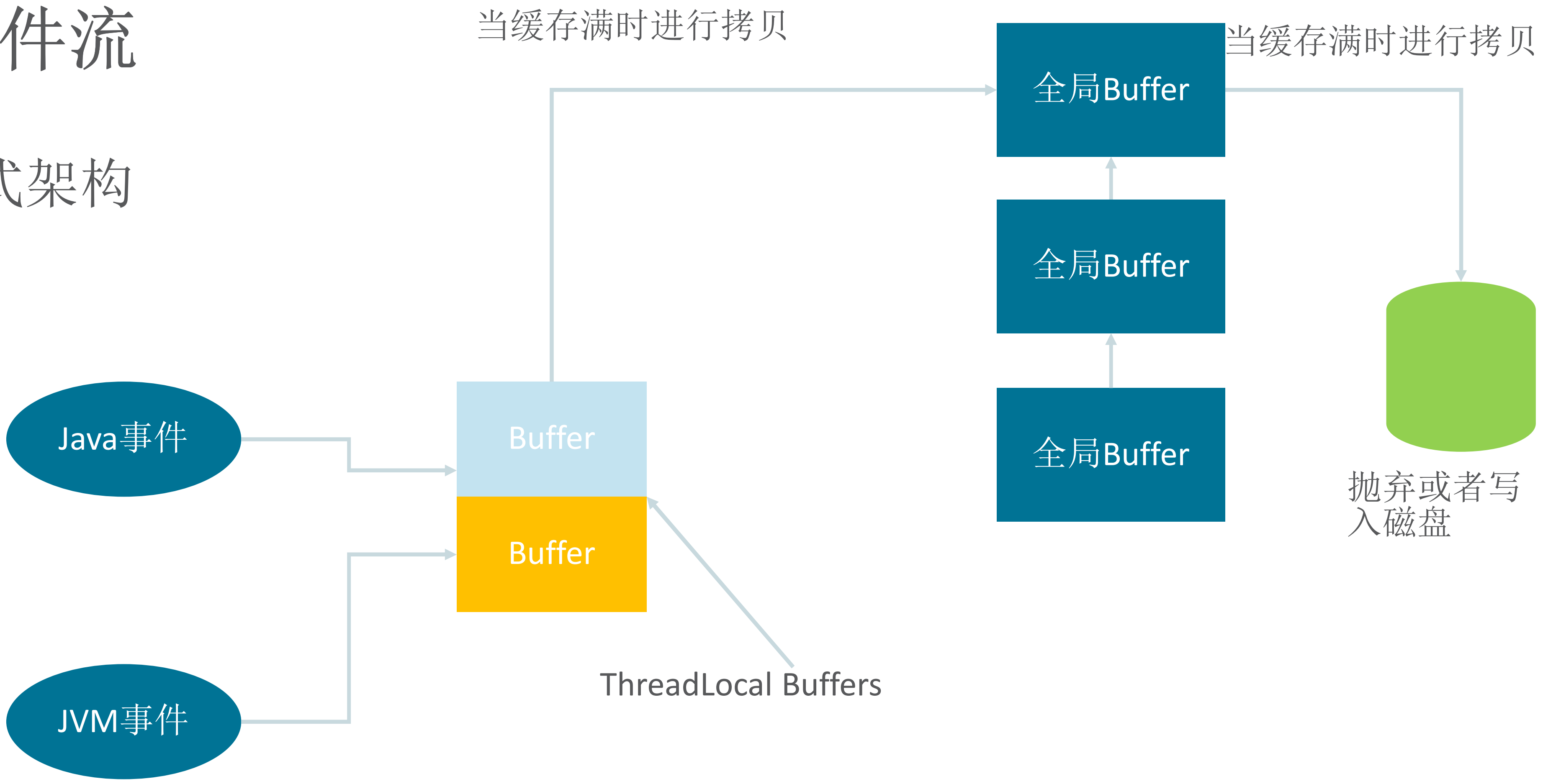


JFR独有的能力

- 不需要字节码操纵就可以Profile对象分配
- 精确的方法采样
- 详细的safepoint信息
- 详细的GC信息：暂停、引用处理…
- 深入JIT内部：热点方法、内联、逆优化…

JFR事件流

- 层级式架构



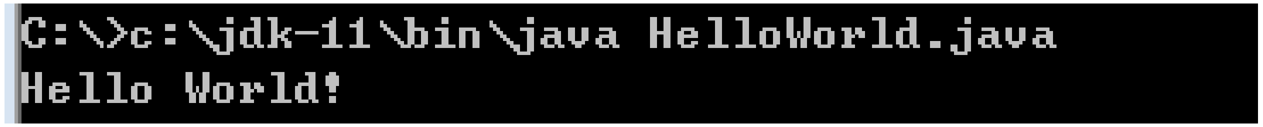
JFR使用

- 简单易用的Public API：
  - jdk.jfr

```
//定义事件
@Label("MyFirstEvent")
@Description("Just an example")
class MyFirstEvent extends Event {
  @Label("Message")
  String message;
}

//运行时提交
MyFirstEvent event = new MyFirstEvent();
event.message = "How ya doin";
event.commit();
```

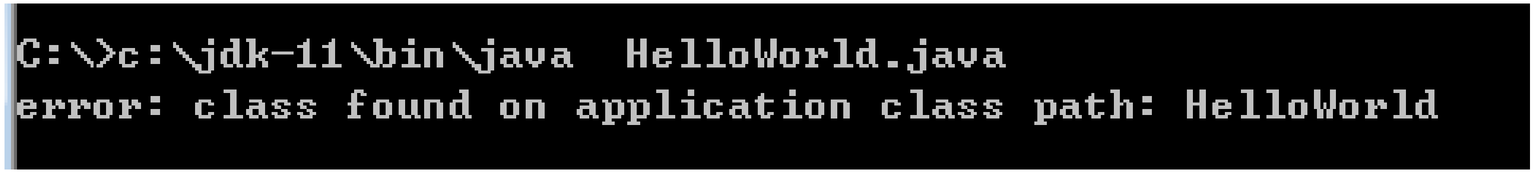
Launch Single-File Source-Code Programs

- Java程序运行的模式
  - java App
  - java -jar App.jar
  - java -m yourModule/com.yourcorp.App
  - 新的模式： java App.**java**



```
C:\>c:\jdk-11\bin\java HelloWorld.java
Hello World!
```

- 限定使用条件，例如已经有相应的.class



```
C:\>c:\jdk-11\bin\java  HelloWorld.java
error: class found on application class path: HelloWorld
```

Local-Variable Syntax for Lambda Parameters

- 将本地变量类型推断应用于Lambda

```
(var x, var y) -> x.process(y)

(@Nonnull var x, @Nullable var y) -> x.process(y)
```

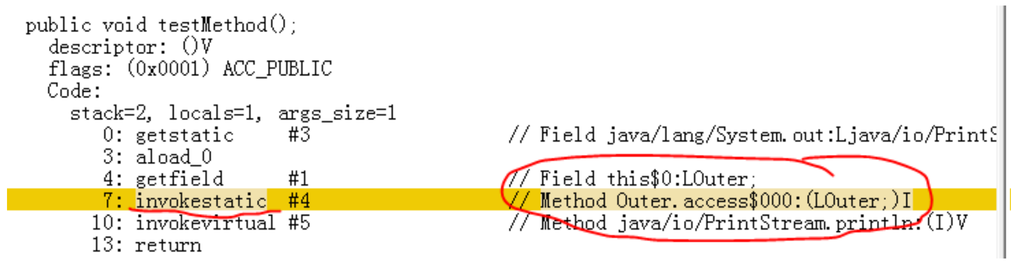
不允许出现:

```
(var x, y) -> x.process(y)
(var x, int y) -> x.process(y)
```

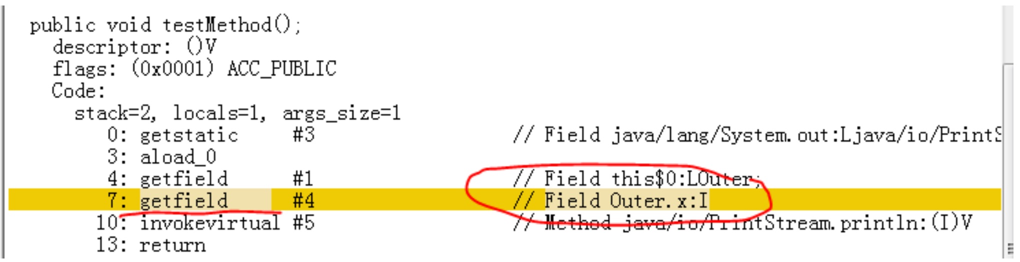
Nestmates： Nest-based Access Control

- 语言层面的修改
- 不再需要编译器魔法

```
// 样例代码：
public class Outer {
    private int x;
    class Inner{
        public void testMethod(){
            System.out.println(x);
        }
    }
}
```



```
public void testMethod();
  descriptor: ()V
  flags: (0x0001) ACC_PUBLIC
  Code:
    stack=2, locals=1, args_size=1
       0: getstatic     #3                  // Field java/lang/System.out:Ljava/io/PrintS
       3: aload_0
       4: getfield      #1                  // Field this$0:LOuter;
       7: invokestatic  #4                  // Method Outer.access$000:(LOuter;)I
      10: invokevirtual #5                  // Method java/io/PrintStream.println:(I)V
      13: return
```



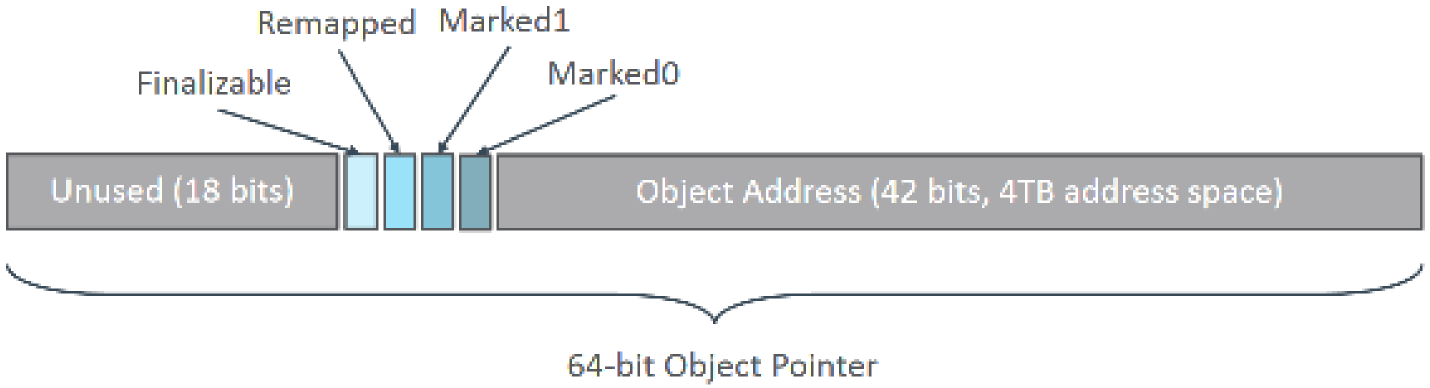
```
public void testMethod();
  descriptor: ()V
  flags: (0x0001) ACC_PUBLIC
  Code:
    stack=2, locals=1, args_size=1
       0: getstatic     #3                  // Field java/lang/System.out:Ljava/io/PrintS
       3: aload_0
       4: getfield      #1                  // Field this$0:LOuter;
       7: getfield      #4                  // Field Outer.x:I
      10: invokevirtual #5                  // Method java/io/PrintStream.println:(I)V
      13: return
```

ZGC: A Scalable Low Latency Garbage Collector

- 主要目标
  - 支持T bytes堆大小
  - <=10 ms暂停时间
  - 暂停时间不会随着Heap增大而变化
  - 相比于G1，吞吐量下降不超过15%

ZGC设计特点

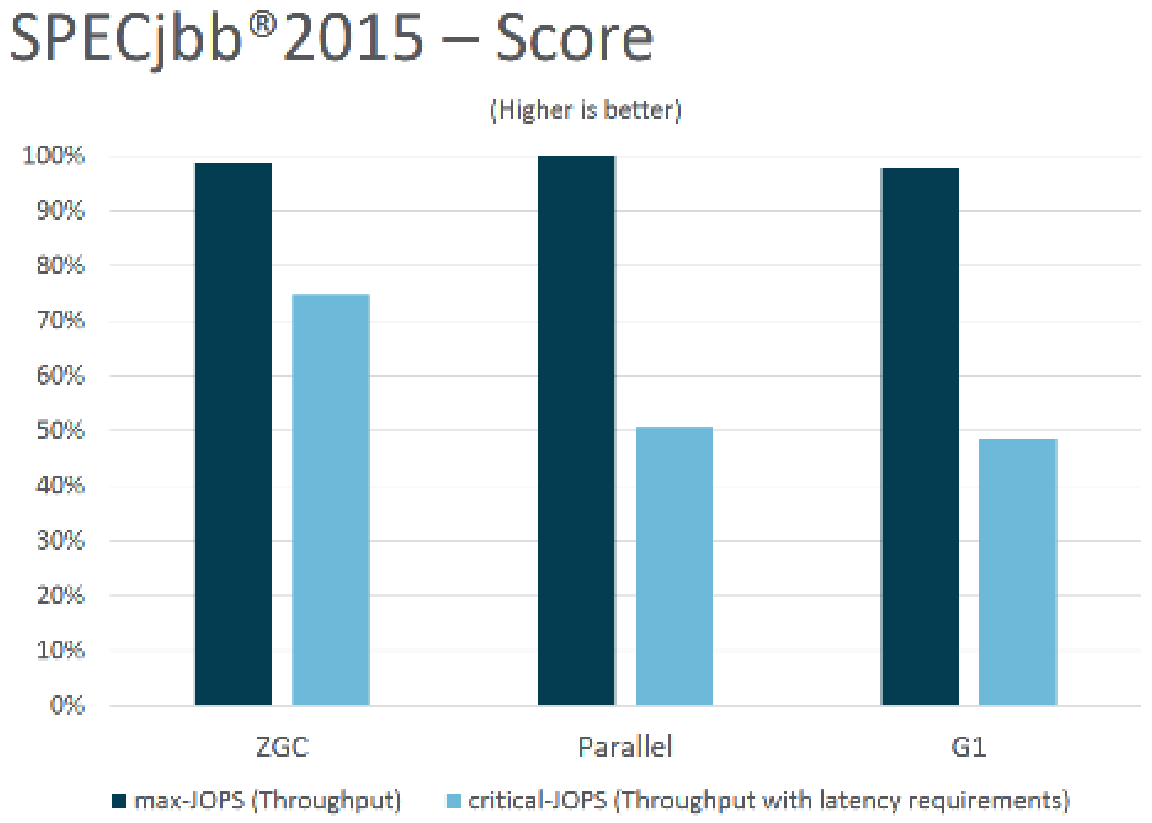
- 无年代
- Region-based
- Colored pointers
  - 信息存储在指针中的空闲位置
  - 不支持32位平台
  - 不支持对象指针压缩
- Load barriers
  - 从堆中加载对象时触发
  - 维护指针颜色
- 部分整理
- 大部分处理都是并发



ZGC使用

- 构建时需要额外指定
  - --with-jvm-features=zgc
- 目前是实验性，仅Linux/x64
  - -XX:+UnlockExperimentalVMOptions -XX:+UseZGC

ZGC -- 实际吞吐量和延迟表现都非常好



Mode: Composite

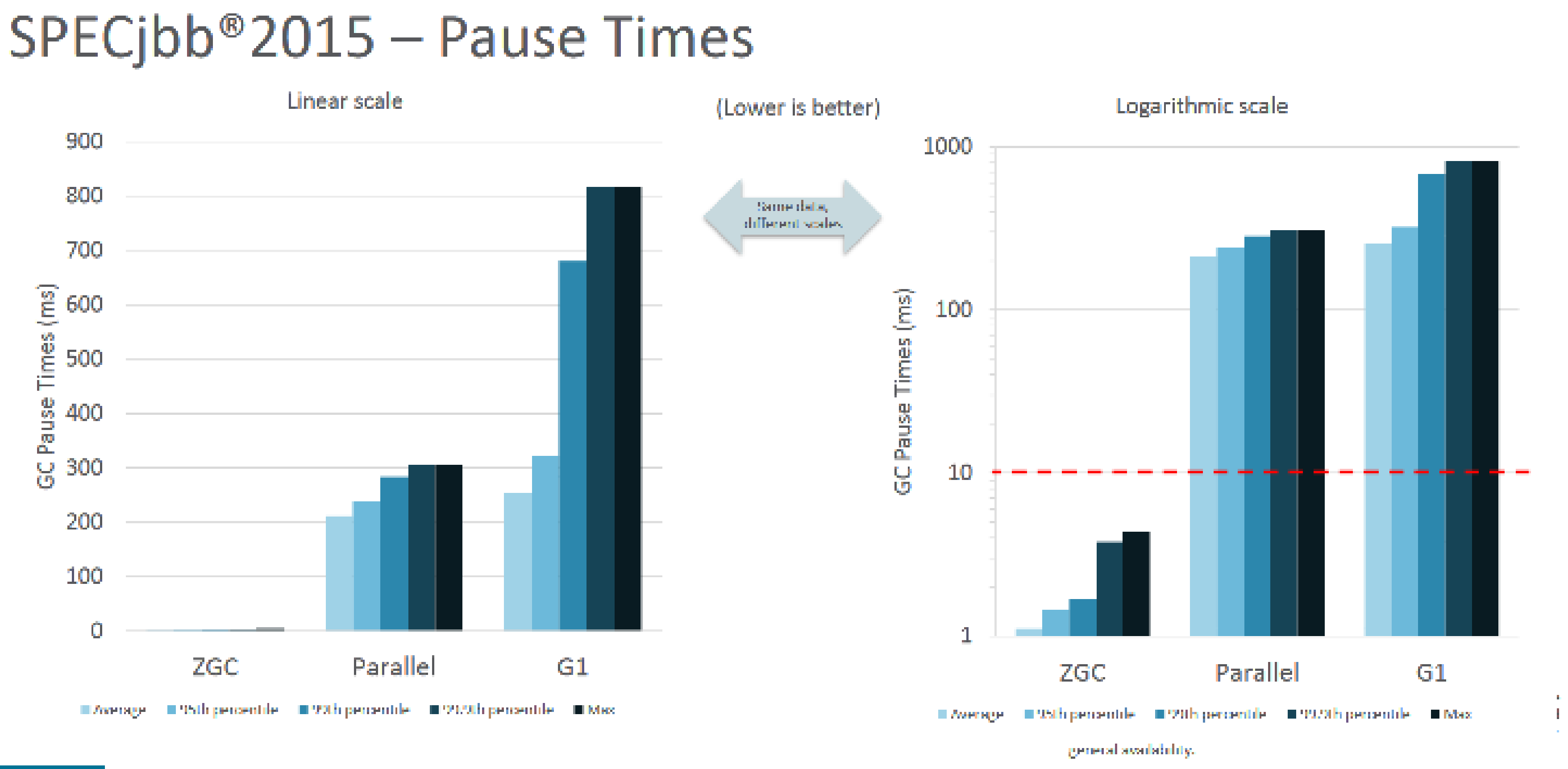
Heap Size: 128G

OS: Oracle Linux 7.4

HW: Intel Xeon E5-2690 2.9GHz
2 sockets, 16 cores (32 hw-threads)

SPECjbb®2015 is a registered trademark of the Standard Performance Evaluation Corporation (spec.org). The actual results are not represented as compliant because the SUT may not meet SPEC's requirements for general availability.

ZGC -- 实际吞吐量和延迟表现都非常好

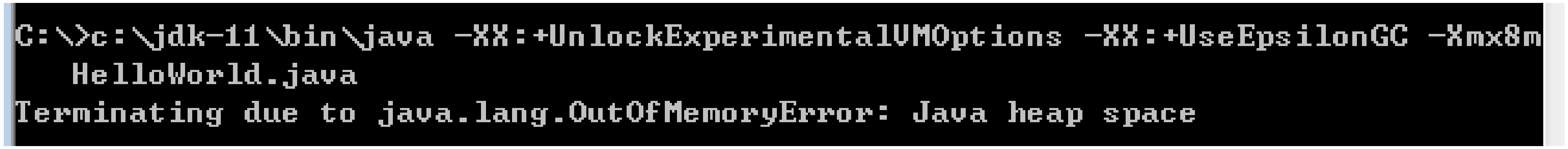


Epsilon: A No-Op Garbage Collector (1)

- 实验性特性
- 强烈的违和感 → 不做垃圾收集的GC!
- 使用：

  -XX:+UnlockExperimentalVMOptions -XX:+UseEpsilonGC

- 内存不足则OOM退出



```
C:\>c:\jdk-11\bin\java -XX:+UnlockExperimentalVMOptions -XX:+UseEpsilonGC -Xmx8m
   HelloWorld.java
Terminating due to java.lang.OutOfMemoryError: Java heap space
```

Epsilon: A No-Op Garbage Collector (2)

- 应用场景:
  - 在性能测试等场合,去掉GC本身的开销
    - GC worker
    - Barrier
    - …
  - 短时间的应用
    - Serverless/Function-as-a-Service ?
  - 如果想自己实现一个GC，Epsilon可以作为一个好的参考:
    - 展示出最小化的GC实现

Java 11以后？

发展趋势（1）

- Java-on-Java
  - 更多JVM编程使用Java语言而不是C++
  - Graal是未来
    - 目前已经是AOT的实现基础
    - 未来也许会逐渐替换 C2 → C1 →其他
    - JDK 11， Deprecated：
      - Nashorn JavaScript script engine
      - jdk.scripting.nashorn API
    - 基于Graal的多语言支持
    - SubstrateVM

发展趋势（2）

- Valhalla：
  - 提高数据密度 → 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

发展趋势（3）

- 改善开发效率：
  - Panama
    - 改进本地代码支持
    - 更好的硬件支持： Vector API
    - 对于大数据、机器学习等应用场景非常重要
    - 网络等类库的本地部分也许会重写
  - Loom
    - 良好粒度的并发
    - Fiber， Continuation
  - Amber适度的语法优化

发展趋势（3）

- 安全性
- 改进启动时间
- 更好的扩展性和可预测性
  - Shenandoah
  - ZGC

参考

- Project JDK 11
- ZGC分享 by Per Liden
- R大在zhihu上的分享
- Six New Trends in the JVM, by John Rose

谢谢

关注我的公众号： xiaofeya

有空儿会更新原创和授权的前沿内容