Unsafe: халява закончилась?

Алексей Федоров,
Одноклассники / JUG.ru

Что такое Unsafe

- **Очень специальный объект**
- Существует с JDK 1.4 — уже 15 лет
- Нужен в Class Library, чтобы оттуда дергать JVM

> *“A VM/library interface, designed strictly for use within the JDK”*
> Mark Reinhold, Java Platform Architect
> JVMLS 2015

Что такое Unsafe

- **Очень специальный объект**
- Существует с JDK 1.4 — уже 15 лет
- Нужен в Class Library, чтобы оттуда дергать JVM

Используется в разных частях JDK

- **1.4** — Reflection, Serialization, NIO
- **1.5** — JSR166 (atomics, locks), CORBA, AWT
- **6.0** — JSR166 (CopyOnWriteArrayList etc.)
- **7** — JSR166 (ForkJoin etc.), BigDecimal, java.lang.invoke
- **8** — JSR166 (много!), Mac OS X objective C bridge

sun.misc.Unsafe

- Лежит в приватном пакете sun.misc
  - раньше в Sun JDK
  - потом в OpenJDK
  - перекочевал в Oracle JDK

А что на других JDK?

А что на других JDK?

```
public final class Unsafe {

    static {
        registerNatives();
        Reflection.registerMethodsToFilter(Unsafe.class, "getUnsafe");
        …
    }

    private Unsafe() {}

    private static final Unsafe theUnsafe = new Unsafe();

    @CallerSensitive
    public static Unsafe getUnsafe() {
        Class cc = Reflection.getCallerClass();
        if (cc.getClassLoader() != null)
            throw new SecurityException("Unsafe");
        return theUnsafe;
    }
```

Как получить Unsafe

```
public static final Unsafe UNSAFE = getUnsafe();

private static Unsafe getUnsafe() {
    try {
        Field f = Unsafe.class.getDeclaredField("theUnsafe");
        f.setAccessible(true);
        return (Unsafe) f.get(null);
    } catch (Exception e) {
        throw new RuntimeException(e);
    }
}
```

И всё???

По умолчанию Secutiry Manager в Java выключен!

И всё???

По умолчанию Secutiry Manager в Java выключен!

```
$ java -Djava.security.manager

java.security.AccessControlException: access denied
("java.lang.RuntimePermission" "accessClassInPackage.sun.misc")
```

«Тебя предупреждали»

```
$ javac Test.java

Test.java:2: warning: Unsafe is internal proprietary API
and may be removed in a future release
import sun.misc.Unsafe;
                ^
```

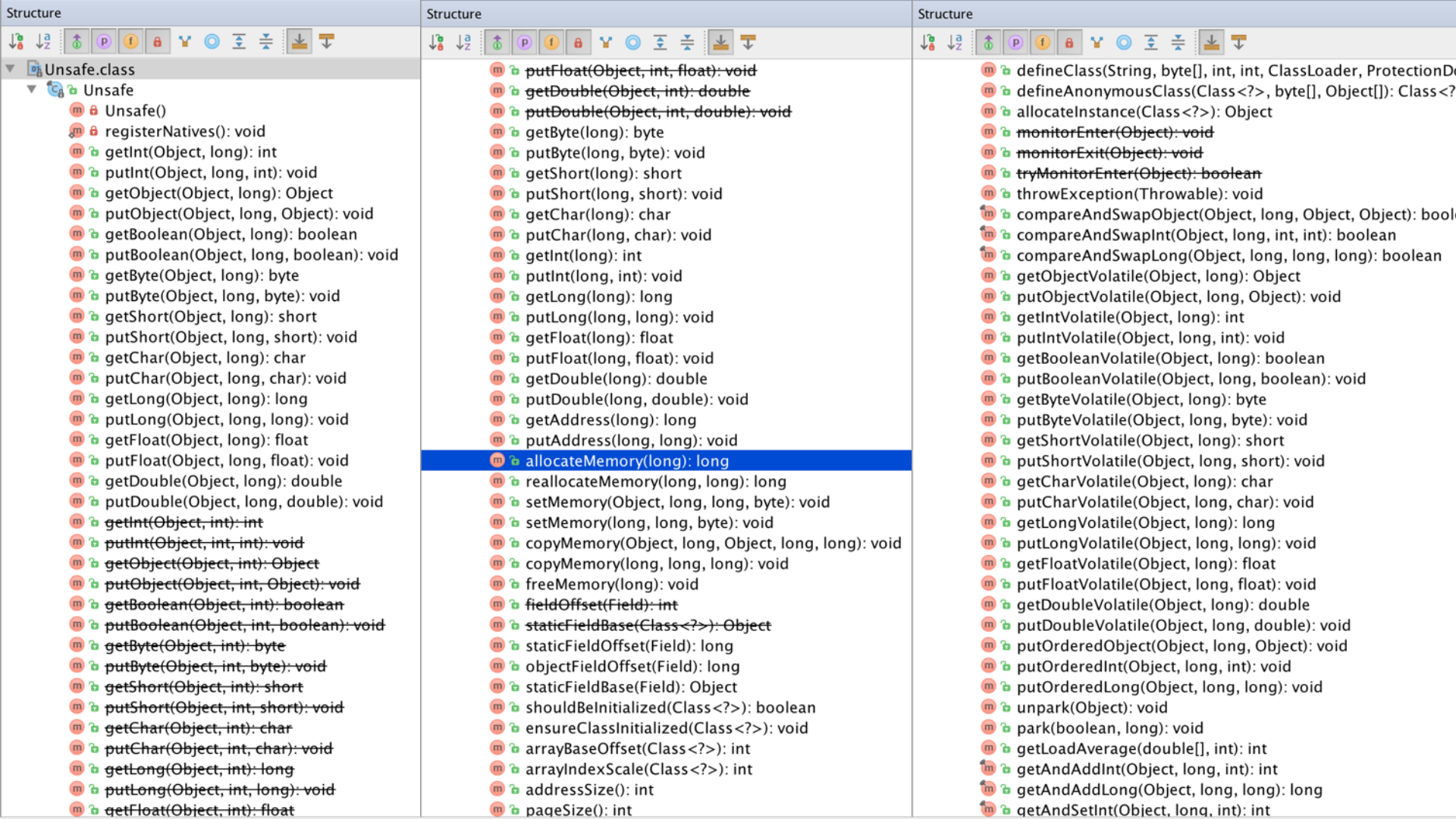
«Да мне пофиг!»

```
$ javac Test.java

Test.java:2: warning: Unsafe is internal proprietary API
and may be removed in a future release
import sun.misc.Unsafe;
                ^
```

```
$ javac -XDignore.symbol.file Test.java
```

Что умеет Unsafe?


Structure
Unsafe.class
Unsafe
Unsafe()
registerNatives(): void
getInt(Object, long): int
putInt(Object, long, int): void
getObject(Object, long): Object
putObject(Object, long, Object): void
getBoolean(Object, long): boolean
putBoolean(Object, long, boolean): void
getByte(Object, long): byte
putByte(Object, long, byte): void
getShort(Object, long): short
putShort(Object, long, short): void
getChar(Object, long): char
putChar(Object, long, char): void
getLong(Object, long): long
putLong(Object, long, long): void
getFloat(Object, long): float
putFloat(Object, long, float): void
getDouble(Object, long): double
putDouble(Object, long, double): void
getInt(Object, int): int
putInt(Object, int, int): void
getObject(Object, int): Object
putObject(Object, int, Object): void
getBoolean(Object, int): boolean
putBoolean(Object, int, boolean): void
getByte(Object, int): byte
putByte(Object, int, byte): void
getShort(Object, int): short
putShort(Object, int, short): void
getChar(Object, int): char
putChar(Object, int, char): void
getLong(Object, int): long
putLong(Object, int, long): void
getFloat(Object, int): float
Structure
putFloat(Object, int, float): void
getDouble(Object, int): double
putDouble(Object, int, double): void
getByte(long): byte
putByte(long, byte): void
getShort(long): short
putShort(long, short): void
getChar(long): char
putChar(long, char): void
getInt(long): int
putInt(long, int): void
getLong(long): long
putLong(long, long): void
getFloat(long): float
putFloat(long, float): void
getDouble(long): double
putDouble(long, double): void
getAddress(long): long
putAddress(long, long): void
allocateMemory(long): long
reallocateMemory(long, long): long
setMemory(Object, long, long, byte): void
setMemory(long, long, byte): void
copyMemory(Object, long, Object, long, long): void
copyMemory(long, long, long): void
freeMemory(long): void
fieldOffset(Field): int
staticFieldBase(Class<?>): Object
staticFieldOffset(Field): long
objectFieldOffset(Field): long
staticFieldBase(Field): Object
shouldBeInitialized(Class<?>): boolean
ensureClassInitialized(Class<?>): void
arrayBaseOffset(Class<?>): int
arrayIndexScale(Class<?>): int
addressSize(): int
pageSize(): int
Structure
defineClass(String, byte[], int, int, ClassLoader, ProtectionD
defineAnonymousClass(Class<?>, byte[], Object[]): Class<?
allocateInstance(Class<?>): Object
monitorEnter(Object): void
monitorExit(Object): void
tryMonitorEnter(Object): boolean
throwException(Throwable): void
compareAndSwapObject(Object, long, Object, Object): bool
compareAndSwapInt(Object, long, int, int): boolean
compareAndSwapLong(Object, long, long, long): boolean
getObjectVolatile(Object, long): Object
putObjectVolatile(Object, long, Object): void
getIntVolatile(Object, long): int
putIntVolatile(Object, long, int): void
getBooleanVolatile(Object, long): boolean
putBooleanVolatile(Object, long, boolean): void
getByteVolatile(Object, long): byte
putByteVolatile(Object, long, byte): void
getShortVolatile(Object, long): short
putShortVolatile(Object, long, short): void
getCharVolatile(Object, long): char
putCharVolatile(Object, long, char): void
getLongVolatile(Object, long): long
putLongVolatile(Object, long, long): void
getFloatVolatile(Object, long): float
putFloatVolatile(Object, long, float): void
getDoubleVolatile(Object, long): double
putDoubleVolatile(Object, long, double): void
putOrderedObject(Object, long, Object): void
putOrderedInt(Object, long, int): void
putOrderedLong(Object, long, long): void
unpark(Object): void
park(boolean, long): void
getLoadAverage(double[], int): int
getAndAddInt(Object, long, int): int
getAndAddLong(Object, long, long): long
getAndSetInt(Object, long, int): int

```java
package cee.secr.misc;

public class MyUnsafe {

    public native int getInt(long address);
```

```c
#include <jni.h>

JNIEXPORT jint JNICALL
Java_snow_misc_MyUnsafe(JNIEnv* env, jobject myUnsafe,
                        jlong address) {
    return *(jint*)address;
}
```

Сколько стоит JNI

```
Create stack frame
    Move arguments according to ABI
        Wrap objects into JNI handles
            Obtain JNIEnv* and jclass
                Trace method_entry
                    Lock if synchronized
                        Lazy lookup and linking
                            in_java → in_native thread transition
                                Call the native function
                            Check for safepoint
                        Switch state to in_java
                    Unlock if synchronized
                Notify method_exit
            Unwrap result, reset JNI handles block
        Handle exceptions
  Remove stack frame
```

Intrinsics

- Большинство методов — intrisics:
  - getInt —> mov
  - compareAndSwapInt —> cmpxchg

CAS

```
// native
public final native boolean compareAndSwapInt(
          Object o, long offset, int expected, int x);

//non-native
public final int getAndAddInt(
              Object o, long offset, int delta) {
    int v;
    do {
        v = getIntVolatile(o, offset);
    } while (!compareAndSwapInt(o, offset, v, v + delta));
    return v;
}
```

AtomicInteger — i.getAndAdd(5)

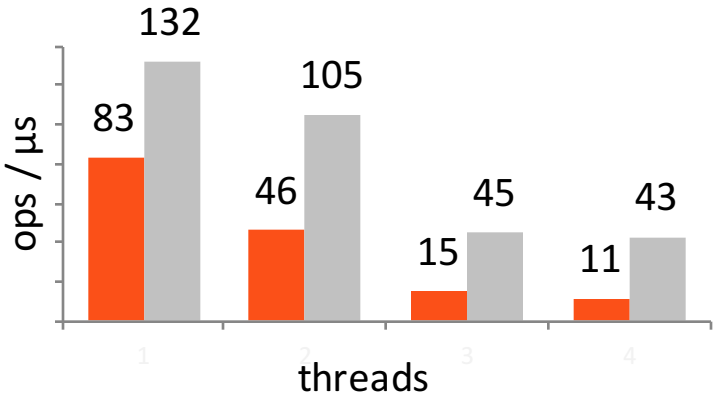
`-XX:+PrintAssembly`

JDK 7u72

```
loop:
   mov     0xc(%r10),%eax
   mov     %eax,%r11d
   add     $0x5,%r11d
   lock cmpxchg %r11d,0xc(%r10)
   sete    %r11b
   movzbl %r11b,%r11d
   test    %r11d,%r11d
   je      loop
```

JDK 8u66

```
lock addl $0x5,0xc(%r10)
```

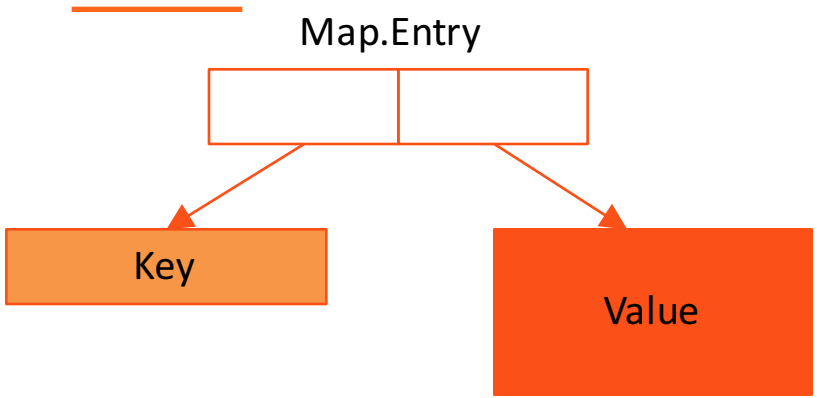


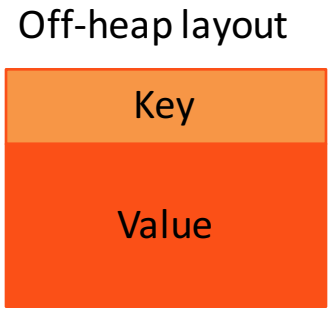
Use cases

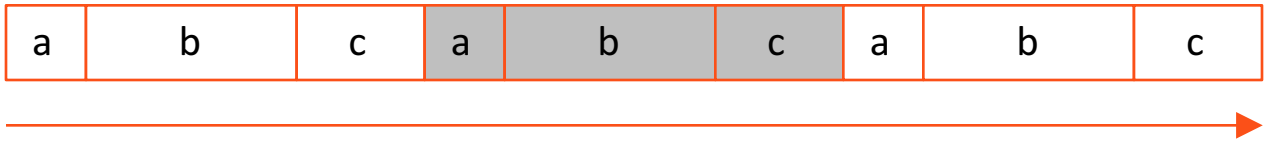
Off-heap Collections

- Lists, sets, maps
  - Large capacity > 2 GB
  - Predictable GC pauses
  - No heap fragmentation
  - Data locality

Data locality







I/O and persistence

- Persistent caches and storages
  - Memory-mapped files (64-bit)
  - Shared memory
- Cooperation with OS
  - Pointer arithmetic, alignment
  - mlock, madvise etc.

IPC, Messaging

- Concurrent off-heap buffers and queues
- High-performance messaging
  - Disruptor
  - Aeron: 6M msg/s
- Shared data structures
  - Chronicle Map

Удаление Unsafe

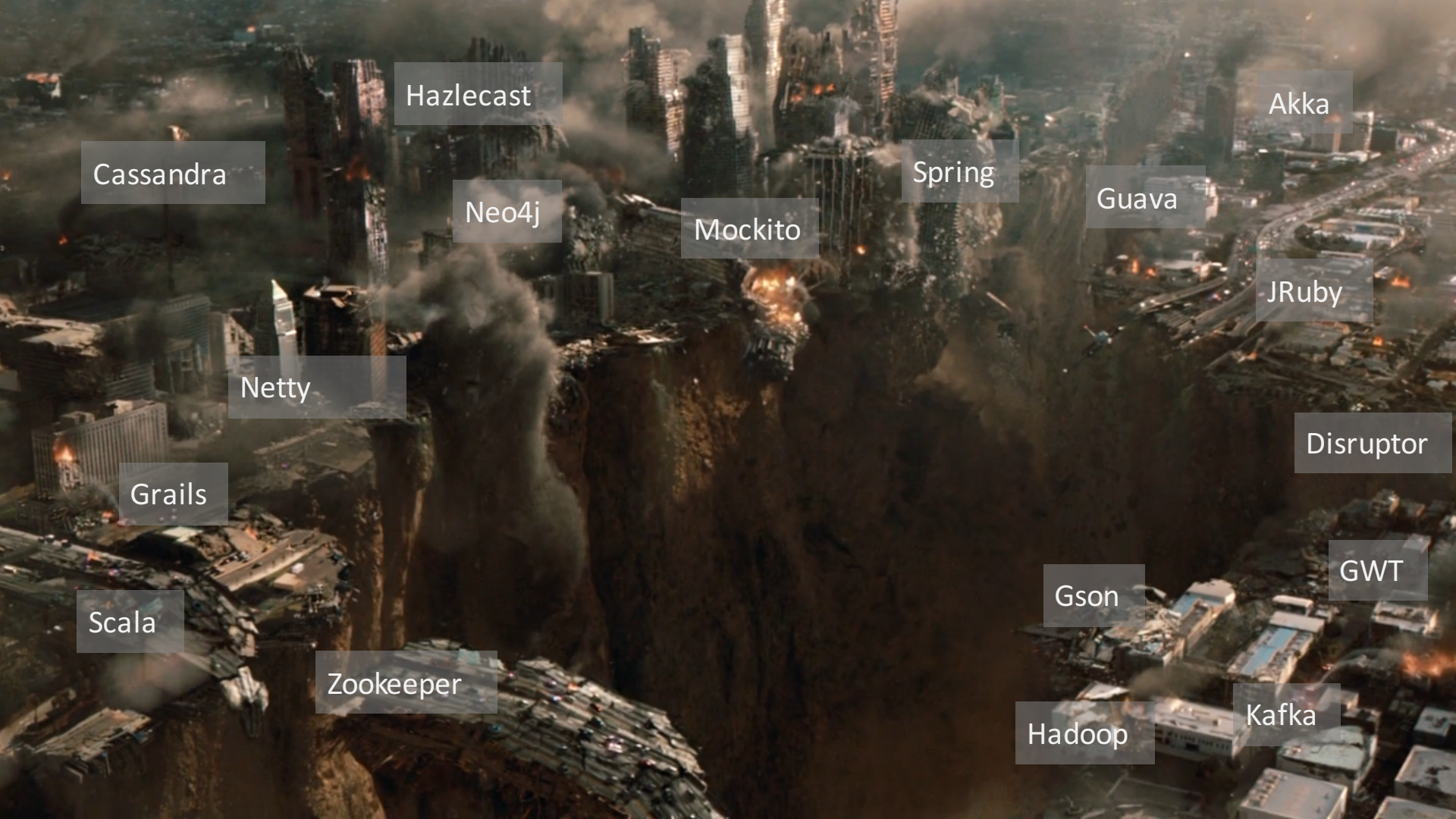
Тактика действий в случие удаления Unsafe

- Переписать куски кода
  - JNI
  - Reflection
  - NIO
  - etc.

Тактика действий в случие удаления Unsafe

- Переписать куски кода
  - JNI
  - Reflection
  - NIO
  - etc.
- Как действовать
  1. Новый layer: wrapper над Unsafe
  2. Заменить на wrapper над public API

Беда 1: Performance Degradation


Hazlecast
Akka
Cassandra
Spring
Guava
Neo4j
Mockito
JRuby
Netty
Disruptor
Grails
GWT
Gson
Scala
Zookeeper
Kafka
Hadoop


Hazlecast
Akka
Cassandra
Spring
Guava
Neo4j
Mockito
JRuby
Netty
Disruptor
Grails
Unsafe
GWT
Gson
Scala
Zookeeper
Kafka
Hadoop

Беда 2: Transitive lock-in

Как так получилось?

```
$ javac Test.java

Test.java:2: warning: Unsafe is internal proprietary API
and may be removed in a future release
import sun.misc.Unsafe;
                 ^
```

Как так получилось?

```
$ javac Test.java

Test.java:2: warning: Unsafe is internal proprietary API
and may be removed in a future release
import sun.misc.Unsafe;
                ^
```

Слишком мало евангелизма, Oracle!

Как развивалась ситуация

Хронология событий

- JEP 200: The Modular JDK

Хронология событий

- JEP 200: The Modular JDK
  - Mark Reinhold выступил на Devoxx 2014

Хронология событий

- JEP 200: The Modular JDK
  - Mark Reinhold выступил на Devoxx 2014
- Пост на [blog.dripstat.com]

Хронология событий

- JEP 200: The Modular JDK
  - Mark Reinhold выступил на Devoxx 2014
- Пост на blog.dripstat.com
- Chris Engelbert из Hazelcast начал дискуссию

Хронология событий

- JEP 200: The Modular JDK
  - Mark Reinhold выступил на Devoxx 2014
- Пост на blog.dripstat.com
- Chris Engelbert из Hazelcast начал дискуссию
- Обсуждение на JCrete

Хронология событий

- JEP 200: The Modular JDK
  - Mark Reinhold выступил на Devoxx 2014
- Пост на blog.dripstat.com
- Chris Engelbert из Hazelcast начал дискуссию
- Обсуждение на JCrete
- Mark Reinhold выступил на JVMLS
  - все немного успокоились

Хронология событий

- JEP 200: The Modular JDK
  - Mark Reinhold выступил на Devoxx 2014
- Пост на blog.dripstat.com
- Chris Engelbert из Hazelcast начал дискуссию
- Обсуждение на JCrete
- Mark Reinhold выступил на JVMLS
  - все немного успокоились
- JEP 260: Encapsulate Most Internal APIs

План замены Unsafe на public API

- Replacement in JDK 8 → hide in JDK 9
  - sun.misc.BASE64Encoder etc.
  - Available via command-line flag
- No replacement in JDK 8 → available outside
  - sun.misc.Unsafe, sun.reflect.ReflectionFactory
  - sun.misc.Cleaner, sun.misc.SignalHandler
- Replacement in JDK 9 → hide in JDK 9, remove in JDK 10

Что приходит на замену

JEP 193: VarHandles

Expected in Java 9

```
class Queue {
    int size;
    ...
}

VarHandle queueSize = VarHandles.lookup()
      .findFieldHandle(Queue.class, "size", int.class);

queueSize.addAndGet(10);
```

```
VarHandle handle = VarHandles.arrayElement(int[].class)

VarHandle viewHandle = VarHandles.arrayElementViewHandle(
        byte[].class, long[].class, true);
```

- get, getVolatile, getAcquire, getOpaque
- set, setVolatile, setRelease, setOpaque
- compareAndSet, compareAndExchangeVolatile
- compareAndExchangeAcquire, compareAndExchangeRelease
- weakCompareAndSet, weakCompareAndSetAcquire, weakCompareAndSetRelease
- getAndSet, getAndAdd, addAndGet

```
ByteBuffer byteBuffer = ByteBuffer.allocateDirect(8);
VarHandle bufferView =
        VarHandles.byteBufferViewHandle(long[].class, true);
```

```
MemoryRegion region = MemoryRegion.allocateNative(
        "myname", MemoryRegion.UNALIGNED, Long.MAX_VALUE);

VarHandle regionView =
     VarHandles.memoryRegionViewHandle(long[].class, true);

regionView.set(region, 0, Long.MAX_VALUE);
return regionView.get(region, 0);
```

Итоги

- Вендор услышал сообщество и изменил планы
  - Вендор готов и дальше слушать
  - Сообщество представляет собой силу, за ним экосистема

Итоги

- Вендор услышал сообщество и изменил планы
  - Вендор готов и дальше слушать
  - Сообщество представляет собой силу, за ним экосистема
- Unsafe все-таки выпилят
  - Но постепенно, в течение нескольких лет

Development @ Одноклассники

- Tech Blog
  - http://habrahabr.ru/company/odnoklassniki
- Open source
  - https://github.com/odnoklassniki
- Career
  - http://v.ok.ru

Благодарю Андрея Паньгина

@AndreyPangin

Вопросы и ответы

Спасибо за внимание!

@23derevo
alexey.fyodorov@corp.mail.ru