Your JDK 8
The Devil is in the Dark

Dr. Michael Eichberg
Software Technology Group
Department of Computer Science
Technische Universität Darmstadt

www.opal-project.de
Some Facts

- The Java 8 Development Kit consists of:
  - 1,651 packages, which contain
  - 31,423 classes (including inner and anonymous), with over
  - 265,644 (~190,000 non-abstract) methods and
  - 115,992 fields

- The majority of the code ~65% does not belong to the public API
  (public API = classes in the packages java* or org*)
Some Facts

Number of classes:
- JDK 1.1: 17
- Java 5: 2.056
- Java 7: 1.226
- Java 8: 28.124
Some Facts

- Some metrics:
  - the most instance fields defined by a class: 211
  - the most methods defined by a class: 1,260
  - the longest method contains 20.167 instructions (which is close to the maximum possible length)
  - the most register variables used by a method: 142 (~number of local variables in scope)
Denying Universal Truth

- found in `com.sun.jmx.snmp.SnmpInt`
  
  ```java
  boolean isInitValueValid(int v) {
    if ( (v < Integer.MIN_VALUE) ||
         (v > Integer.MAX_VALUE)) { ... }
    return true;
  }
  ```

- found in `com.sun.java.util.jar.pack.Fixups.storeDesc(…)
  
  ```java
  (bytes(loc+1)==(byte)bigDescs(BIGSIZE))! =999)
  ```
Denying Universal Truth

- `String() values = null;`
- `... values = readTag(tagName, string, pos) ...`
- `if(values.length < 0) {
   throw new InvalidAttributeValueException(...);
}
  the length of an array is never smaller than 0`
Confused Logical Operators

- found in sun.nio.cs.ext.GB18030

- else if (...) && offset < 0x12E248) {
  if (offset >= 0x12E248) ... 
}

- found in com.sun.media.sound.AuFileReader

- if (! (magic == AuFileFormat.AU_SUN_MAGIC) ||
  (magic == AuFileFormat.AU_DEC_MAGIC) ||
  (magic == AuFileFormat.AU_SUN_INV_MAGIC) ||
  (magic == AuFileFormat.AU_DEC_INV_MAGIC))

Confused Logical Operators

- Found in `com.sun.imageio.plugins.png.PNGMetadata`

```java
void mergeStandardTree(org.w3c.dom.Node)

if (maxBits > 4 || maxBits < 8) {
    maxBits = 8;
}
if (maxBits > 8) {
    maxBits = 16;
}
```

- `maxBits` will always be 8; every int is either larger than 4 or smaller than 8.
Too Defensive Programming

Checking Obvious Truths

- found in com...org...internal...SuballocatedByteVector removeElementAt

- if(next!=null)
  
  block(m_blocksize-1)=(next!=null) ? next(0) : 0;

- found in javax.print.attribute.HashAttributeSet

- public boolean containsValue(ATTRIBUTE attribute) {
    return
    attribute != null &&
    attribute instanceof Attribute &&
    attribute.equals(attrMap.get(((ATTRIBUTE)attribute).getCategory()));
}
private static boolean isUnnecessaryTransform(ModelTransform transform) {
    if (transform == null) return false;
    if (!(transform instanceof ModelStandardTransform)) return false;
    ModelStandardTransform stransform = (ModelStandardTransform)transform;
    if (stransform.getDirection() != ModelStandardTransform.DIRECTION_MIN2MAX)
        return false;
    if (stransform.getPolarity() != ModelStandardTransform.POLARITY_UNIPOLAR)
        return false;
    if (stransform.getTransform() != ModelStandardTransform.TRANSFORM_LINEAR)
        return false;
    return false;
}
Null Values...

- found in `java.nio.file.FileTreeWalker next()`
  
  ```java
  if (ioe != null) {
    ioe = e;
  } else {
    ioe.addSuppressed(e);
  }
  ```

- found in `javax.print.ServiceUI printDialog(…)`
  
  ```java
  Window owner = null;
  ... owner is not set by any means
  if (owner instanceof Frame) ...
  ```
The dark corners …

…or how to break static analyses by writing unsafe and “contra-idiomatic” code!
It’s just “null”, isn’t it?
found in java.util.concurrent.FutureTask<V>

- All places where the field **runner** (: java.lang.Thread) is set:
  
  - **Line**   **Code**
    106  private volatile Thread runner;
    279  runner = null;
    317  runner = null;

It’s not a bug!
It’s a feature!
It’s just “null”, isn’t it?
found in java.util.concurrent.FutureTask<V>

- private static final sun.misc.Unsafe UNSAFE;
  private static final long runnerOffset;
  static {
    UNSAFE = sun.misc.Unsafe.getUnsafe();
    Class<?> k = FutureTask.class;
    runnerOffset =
      UNSAFE.objectFieldOffset(k.getDeclaredField("runner"));
    ...
  }
It’s just “null”, isn’t it?
found in `java.util.concurrent.FutureTask<V>`

- public void run() {
  if (state != NEW ||
    UNSAFE.compareAndSwapObject(
      this, runnerOffset,
      null, Thread.currentThread()))
    return;

  ...
}

It’s not a bug!
It’s a feature!
A Bug or a Feature?

- try {
  XmlSchema s = null;
  s.location();
} catch (NullPointerException e) {
  // as expected
}

Used to detect the version of the current xml library.
Use(less|ful) null checks

- found in `java.util.concurrent.ConcurrentSkipListMap`

- ```
private V doPut(K key, V value, boolean onlyIfAbsent) {
    // MANY LINES OF CODE!
    splice: for (int insertionLevel = level;;) {
        int j = h.level;
        for (Index<K,V> q = h, r = q.right, t = idx;;) {
            if (q == null | | t == null) break splice;
        // MANY LINES OF CODE!
    }
```
Use(less|ful) null checks

- found in java.util.concurrent.ConcurrentSkipListMap

- private V doPut(K key, V value, boolean onlyIfAbsent) {
  // MANY LINES OF CODE!
  splice; for (int insertionLevel = level;;) {
    int j = h.level;
    for (Index<K,V> q = h, r = q.right, t = idx;;) {
      if (q == null || t == null) break splice;
      // MANY LINES OF CODE!
    }
    // MANY LINES OF CODE!
  }

Yes. You will find a bunch of these in j.u.c; where null checks never fail, but the JIT does not understand this so would otherwise arrange more expensive exception code rather than skipping. So it is (counter-intuitively) a small performance tweak to include the explicit check. […]

-Doug
Never, ever remove deprecated code!

- found in com.sun.org.apache.xml.internal.serializer.ToStream outputDocTypeDecl

```java
boolean dothis = false;
if (dothis)
{
    // at one point this code seemed right
    // but not anymore - Brian M.
    if (closeDecl)
    {
        ...
    }
}
```
found in com.sun.org.apache.xalan.... TransformerFactoryImpl

/**
As Gregor Samsa awoke one morning from uneasy dreams he found himself transformed in his bed into a gigantic insect. He was lying on his hard, as it were armour plated, back, and if he lifted his head a little he could see his big, brown belly divided into stiff, arched segments, on top of which the bed quilt could hardly keep in position and was about to slide off completely. His numerous legs, which were pitifully thin compared to the rest of his bulk, waved helplessly before his eyes. "What has happened to me?", he thought. It was no dream.... */

protected final static String DEFAULT_TRANSLET_NAME = "GregorSamsa";

„Kafka1906“
Atelier Jacobi: Sigismund Jacobi (1860–1935)
Your JDK
The Devil is in the Dark

Dr. Michael Eichberg

This presentation was created with the help of the OPAL Framework and the BugPicker.