10 Years of The Java Specialists' Newsletter

Dr Heinz M. Kabutz
heinz@javaspecialists.eu
http://www.javaspecialists.eu

Javaspecialists.eu
java training

© 2010 Heinz Kabutz – All Rights Reserved

Brief Biography

- **Dr Heinz Kabutz**
  - **Live on Island of Crete in Mediterranean Sea (Europe)**
  - **PhD Computer Science from University of Cape Town**
  - **The Java Specialists' Newsletter**
  - **Java programmer**
  - **Java Champion since 2005**
  - **Java instructor to corporates**
    - **Java Specialist Master Course**
      - **Threads, Java NIO, Memory, Optimizations, etc.**
    - **Offered in NYC via MVP telepresence**
      - **http://www.exitcertified.com/mvp/**

Javaspecialists.eu

Why Crete?

- Airport 10 minutes from my house
- E1 connection to my house
- Closer to customers than Cape Town
- Great lifestyle, good food, clean air
- Super friendly citizens
- Wife and children are Greek citizens
- And now for the *real reason ...*

Javaspecialists.eu

USA Trainer for Java Master Course

- **Last year I travelled 150 days**
- **We are looking for a Java expert who likes teaching**
  - **4 courses presented a year, 4 days each**
  - **Master or PhD in Computer Science**
  - **Practical experience with programming Java, including:**
    - **Threads, Java NIO, reflection, etc. all the advanced topics**
  - **Teaching experience**
  - **You will be trained on how to teach the course**
    - **But should already know most of the topics**

Javaspecialists.eu

The Java Specialists' Newsletter

- **Some Statistics**
  - **Served more than 1 million newsletters in past 12 months**
  - **50 000 readers in 121 countries**

- **But let us go back 10 years ...**

Javaspecialists.eu

History - The Java Specialists' Newsletter

- **"Java Argot, Things your Dad never told you"**
  - **James Pereira suggested title for my book in April 2000**
  - **Argot - slang or jargon peculiar to a particular group, esp. (formerly) a group of thieves (Collins English Dictionary)**
- **Philip Greenspun**
  - **Explained the pain of writing a computer book**
  - **http://philip.greenspun.com/wtr/dead-trees/story.html**
- **Peter Carruthers "Pete's Weekly" Style**
  - **Newsletter for small business owners, started in 1999**
    - **www.petercarruthers.com**
  - **Short ideas with a personal anecdote**

Javaspecialists.eu

Dot-Com Crash

- Short 3 month visit to Germany in 2000
  - Customers bankrupt or cancelled projects
- "When fate hands us a lemon, let's try to make lemonade." - Dale Carnegie
  - Lemon: No work, but lots of time
  - Lemonade: The Java Specialists' Newsletter
- Wrote first Java article on Deadlocks
  - November 2000
  - Sent to 75 friends, colleagues and customers
  - http://www.javaspecialists.eu/archive/Issue001.html

Javaspecialists.eu

Not Everyone Loves You

- First newsletter got a lot of positive feedback
  - Thank you for your newsletter - it is definitely very welcome!
  - always interested in the tricks of the trade. keep 'em coming heinz.
  - Thanks for the newsletter. Great, keep it up.
- But one "neutral" comment almost made me give up
  - Unfortunately, I'm not interested in your newsletter, because Java is not the world I live in - Lizette G.
- In 10 years, I received some unprintable emails
  - Remember - they are having a bad day!
    - Maybe mother-in-law announced she is visiting?

Javaspecialists.eu

Issue 188 - Cool Threaded Code

- **One response from a reader:**
  - **Hey, Dr. No offense but this is very boring. Take a look at app development for Droids. Very cool and big money. I am currently developing apps using eclipse. Best, Dan**
- **My response:**
  - **No offence taken - to me it was extremely interesting :-D**
  - **But you are right, I should also do some Android topics. Would you like to write a newsletter for me on the topic?**

Javaspecialists.eu

#002 Anonymous Inner Classes (2000-12-07)

- **In Java 1.0, we could initialize arrays like this:**

```
String[] temp_names = new String[3];
temp_names[0] = "Heinz";
temp_names[1] = "John";
temp_names[2] = "Anton";
universityRegistration.addNames(temp_names);
```

- **Or**

```
String[] temp_names = { "Heinz", "John", "Anton" };
universityRegistration.addNames(temp_names);
```

- **In Java 1.1, we had a new way of creating arrays**

```
universityRegistration.addNames(
  new String[] { "Heinz", "John", "Anton" });
```

Javaspecialists.eu

Constructing Collections using Anon Classes

- **Instead of doing it in several steps**

```
Collection temp_names = new Vector(3);
temp_names.add("Heinz");
temp_names.add("John");
temp_names.add("Anton");
universityRegistration.addNames(temp_names);
```

- **Java 1.1 introduced anonymous inner classes**

```
universityRegistration.addNames(new Vector(3)
  {{ add("Heinz"); add("John"); add("Anton"); }});
```

- **How does this work?**

Javaspecialists.eu

- Imagine we had just a simple class named MyVector

```
public class MyVector extends Vector {
  public MyVector() {
    super(3);
    add("Heinz"); add("John"); add("Anton");
  }
}
```

- Move "add()" method calls to initializer block

```
public class MyVector extends Vector {
  { // initializer block
    add("Heinz"); add("John"); add("Anton");
  }
  public MyVector() {
    super(3); // to initialise it with a size of 3
  }
}
```

- Make the class anonymous, but keep initializer block

```
Vector myVector =
  new Vector(3) { // defining anonymous inner class
  {
    add("Heinz"); add("John"); add("Anton");
  }
};
```

Javaspecialists.eu

Java 5 Varargs

- **In Java 5 you probably want to do**

```
universityRegistration.addNames(
  Arrays.asList("Heinz","John","Anton")
);
```

Javaspecialists.eu

Soft Reference Based Map (#015 and #098)

- SoftReferences are not a reliable caching mechanism
  - "Poor man's cache"
  - Rather do not use
- WeakHashMap uses weak references for the keys
  - Not useful as a cache

Javaspecialists.eu

Measuring Object Memory Usage

- **Various approaches for measuring memory usage:**
  - **Experimental (from Runtime memory functions)**
    - **http://www.javaspecialists.eu/archive/Issue029.html**
    - **http://www.javaspecialists.eu/archive/Issue078.html**
  - **Agents**
    - **http://www.javaspecialists.eu/archive/Issue142.html**
  - **Neither is necessarily 100% accurate**
  - **Both approaches use reflection to resolve object graphs**
- **You can then determine what uses more memory:**
  - **ArrayList, HashSet, LinkedList or TreeSet**
  - **String or byte[ ]**

Javaspecialists.eu

How Much Memory Does "Joe" Use in Java?

- In C, "Joe" is a char[], zero terminated
- Each char in C takes one byte
- Thus, "Joe" takes 4 bytes
- In Java, each char uses two bytes
- How much memory do you think that "Joe" uses in Java?

Javaspecialists.eu

32-bit "Joe" uses 48 bytes

- **In Java, "Joe" is a String object:**
  - **8 byte object descriptor**
  - **int length (4 bytes)**
  - **int offset (4 bytes)**
  - **int cachedHashCode (4 bytes)**
  - **Pointer to char[ ] (4 bytes)**
  - **For String object (excluding char[ ]) we use 24 bytes**
  - **char[3] uses 8 + 4 + 3*2 ≈ 24 bytes**

Javaspecialists.eu

64-bit "Joe" uses 72 bytes!

- Larger pointers
  - 16 byte object descriptor
  - int length (4 bytes)
  - int offset (4 bytes)
  - int cachedHashCode (4 bytes)
  - Pointer to char[ ] (8 bytes)
  - For String object (excluding char[ ]) we use 40 bytes
    - Rounded up to nearest 8 bytes
  - char[3] uses 16 + 4 + 3*2 ≈ 32 bytes
- Some special XX flags can reduce pointer size

Javaspecialists.eu

Memory Usage of Fields in Java

- **Primitive fields**
  - Java 1.0 to 1.3, each field used 4 bytes minimum
  - Java 1.4 onwards, each field uses a minimum of 1 byte
    - boolean, byte: 1 byte
    - char, short: 2 bytes
    - int, float: 4 bytes
    - double, long: 8 bytes
- **Pointers to objects**
  - 32-bit use 4 bytes
  - 64-bit use 8 bytes

Javaspecialists.eu

GUI Newsletters

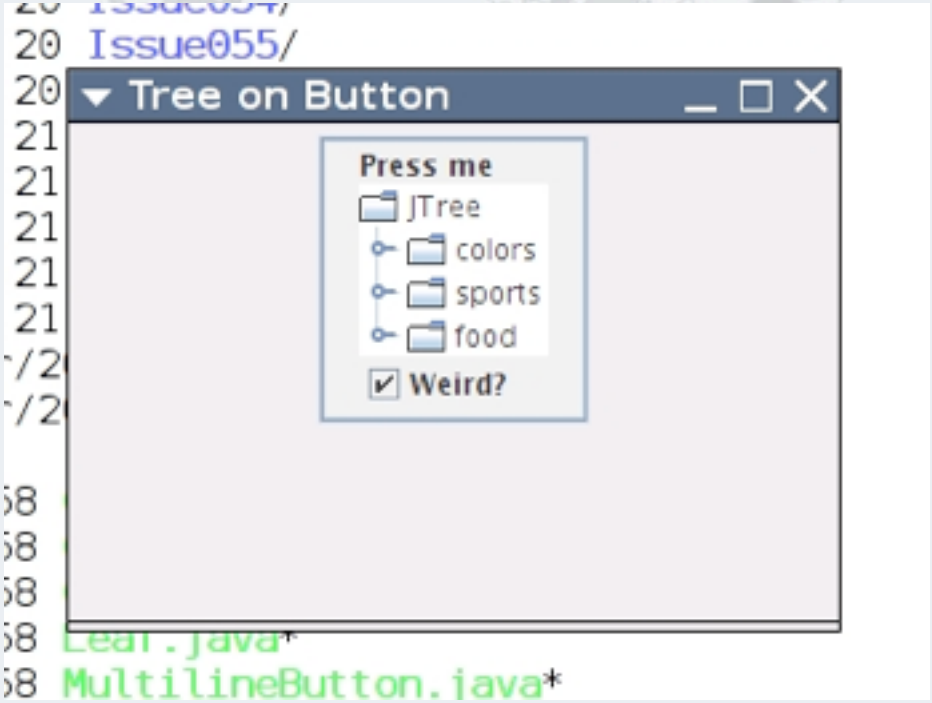
- **Global Hot Key Manager (#007)**

```
public class GlobalHotkeyManager extends EventQueue {
  static {
    Toolkit.getDefaultToolkit().
      getSystemEventQueue().push(instance);
  }
  protected void dispatchEvent(AWTEvent event) {...}
}
```

Javaspecialists.eu

GUI Newsletters

- **Placing components on each other**
  - **Newsletter #041**

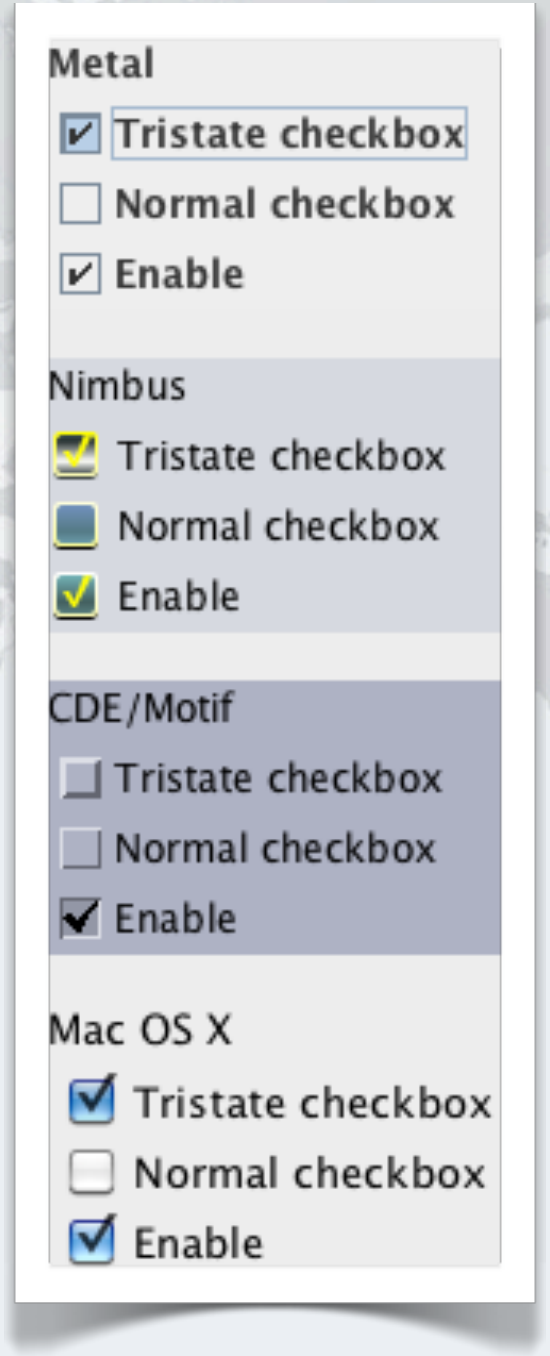


  - **(Real-life use case was multi-line button in JDK 1.2)**

Javaspecialists.eu

GUI Newsletters

- **Tristate Checkbox**
  - **Newsletters #082 and #145**
  - **Used by several commercial systems**



Javaspecialists.eu

Generating Classes in Java 6 (#180 & #181)

- **Instead of using dynamic proxies, we can also generate code**
  - **In our approach, we use Java 6 javax.tools.JavaCompiler**
  - **We compile the Java String in-memory to a byte[]**
  - **We inject the class into any class loader**
    - **With dynamic proxy class hack**

Javaspecialists.eu

Enum Inversion Problem (#113)

- Answers the question: How can we persist enums?
  - Using the ordinal is brittle
    - Modifying the order of enums changes ordinal value
  - Using the String representation is also risky
    - Change in case would break backwards compatibility
- We introduce an interface EnumConverter and
  - ReverseEnumMap<V extends Enum<V> & EnumConverter>

Javaspecialists.eu

Creating Objects Without Calling Constructors (#175)

- **Using the deserialization mechanism to make objects**

```
public class SilentObjectCreator {
  public static <T> T create(Class<T> clazz) {
    return create(clazz, Object.class);
  }
  public static <T> T create(Class<T> clazz,
                             Class<? super T> parent) {
    try {
      ReflectionFactory rf =
          ReflectionFactory.getReflectionFactory();
      Constructor objDef = parent.getDeclaredConstructor();
      Constructor intConstr =
        rf.newConstructorForSerialization(clazz, objDef);
      return clazz.cast(intConstr.newInstance());
    } catch (Exception e) {
      throw new IllegalStateException(e);
    }
  }
}
```

Javaspecialists.eu

The Laws of Concurrency

- **With colourful names to help us remember**
  - **The Law of the Sabotaged Doorbell (#146)**
  - **The Law of the Distracted Spearfisherman (#147)**
  - **The Law of the Overstocked Haberdashery (#149)**
  - **The Law of the Blind Spot (#150)**
  - **The Law of the Leaked Memo (#151)**
  - **The Law of the Corrupt Politician (#152)**
  - **The Law of the Micromanager (#155)**
  - **The Law of Cretan Driving (#156)**
  - **The Law of Sudden Riches (#159)**
  - **The Law of the Uneaten Lutefisk (#160)**
  - **The Law of the Xerox Copier (#176)**

Javaspecialists.eu

8. The Law of Cretan Driving

The JVM does not enforce all the rules.
Your code is probably wrong, even if it works.

*** Don't *stop* at a stop sign if you treasure your car!**

Javaspecialists.eu

Law 8: The Law of Cretan Driving

- **Learn the JVM Rules !**
- **Example from JSR 133 – Java Memory Model**
  - **VM implementers are encouraged to avoid splitting their 64-bit values where possible. Programmers are encouraged to declare shared 64-bit values as volatile or synchronize their programs correctly to avoid this.**

Javaspecialists.eu

JSR 133 allows this – NOT a Bug

- **Method set() called by two threads with**
  - **0x12345678ABCD0000L**
  - **0x1111111111111111L**

```
public class LongFields {
  private long value;
  public void set(long v) { value = v; }
  public long get()       { return value; }
}
```

- **Besides obvious answers, "value" could now also be**
  - **0x11111111ABCD0000L or 0x1234567811111111L**

Javaspecialists.eu

Tips on Writing Articles

- Keep it short
- Make it funny
- One thought
- Be brief
- Have fun!
- Keep it short
  - We are all far too busy to read long articles and complicated prose that elucidate the advantages of the author's most excellent perspective versus the terrible spouting of the lesser mortals who try to contradict his viewpoint.
    - Fortunately I am too illiterate to write like that :-)

The Future?

- **We are becoming multi-media junkies**
  - **Webinars, podcasts, etc.**
    - **The Youtube generation**
  - **Magazines and books contain old information**
- **Java Specialist Club**
  - **Weekly mentoring and training sessions**
    - **Gym contract for the mind**
    - **www.javaspecialists.eu/club**
  - **Developed an open source set of annotations for patterns**
    - **www.jpatterns.org**
- **The Java Specialists' Newsletter**
  - **Will continue ... http://www.javaspecialists.eu**

Javaspecialists.eu

Questions?

Javaspecialists.eu
java training

10 Years of The Java Specialists' Newsletter

Dr Heinz M. Kabutz
heinz@javaspecialists.eu
http://www.javaspecialists.eu

Javaspecialists.eu
java training

© 2010 Heinz Kabutz – All Rights Reserved