ThoughtWorks

JavaOne

Advanced Enterprise Debugging

Neal Ford

ThoughtWorker/Meme Wrangler
ThoughtWorks
www.thoughtworks.com

TS-4588

What This Session Covers

- Forensic debugging
- Debugging web applications
- Debugging class loaders
- Groovy enterprise debugging
- Automating common tasks

Effective Debugging

- One of the keys to effective debugging is to not use brute force
  - `System.out.println()`
  - Line breakpoints
- There is always a sense of urgency when debugging—stop and think!
  - I know that you are behind schedule, but you're just making it worse!
  - Step away from the keyboard
  - Don't ever say: "But that's impossible!"
  - Or, "But it works on **my** machine!"

Forensic Debugging Using Loggers

- Loggers are the best way to do forensic debugging
- Logging setup is well documented just about everywhere
- Use the levels wisely
- If you do need to get down to the “debug” level, use tools
  - grep will help you find just what you are looking for
  - Don’t go scroll blind
  - Use ChainSaw

Aspects and Loggers

- One of the best uses of Aspects is to inject logging code
- Logging tests are cheap, but there is still overhead
- There's no overhead if the code isn't there!
- Aspects allow you to specifically target code only when you need to

Aspects

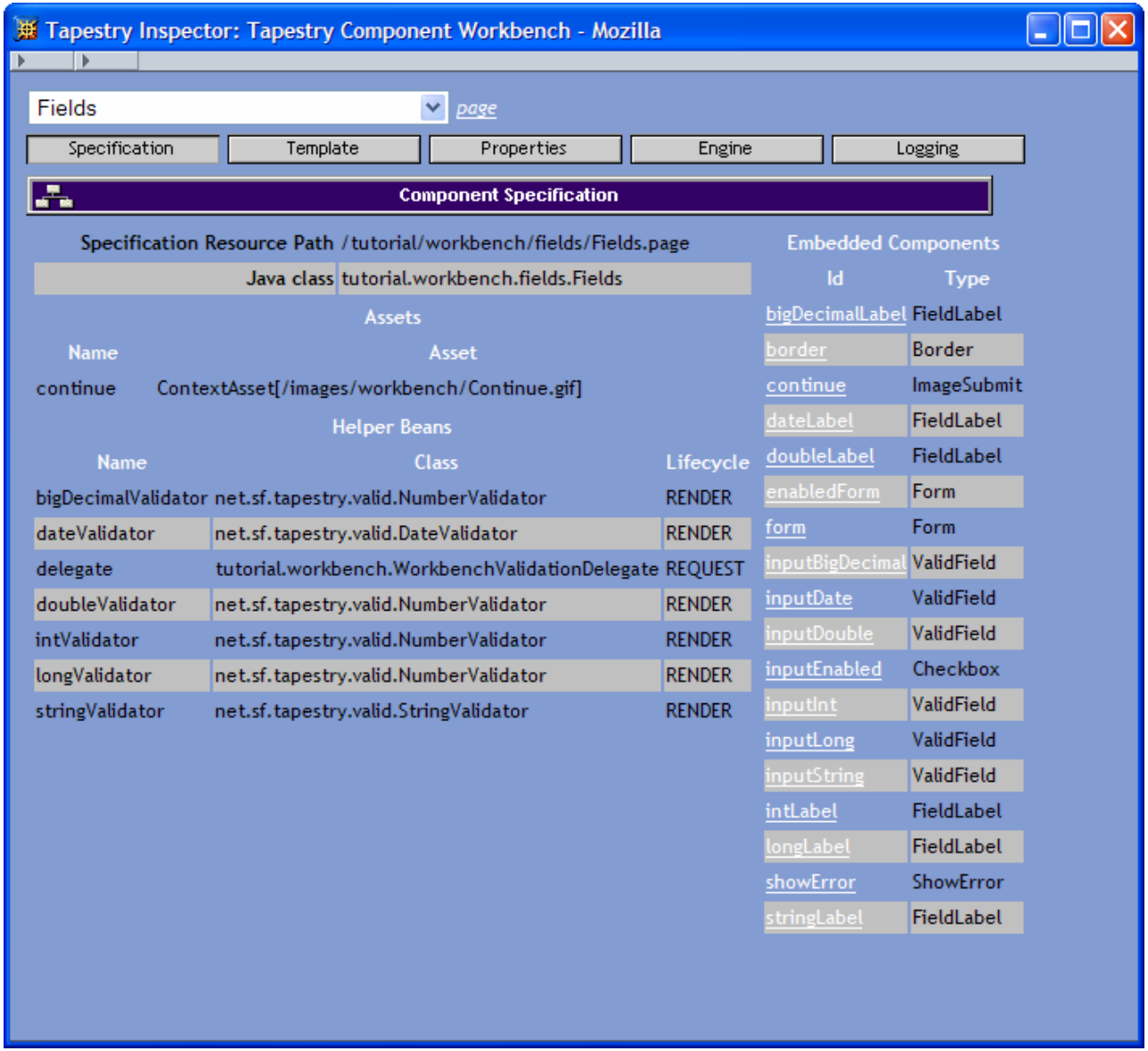
```
aspect Logging {

    pointcut populate() :
        call(public void OrderDb.setDbPool(DBPool)) ||
        call(public void ProductDb.setDbPool(DBPool));

    before(): populate() {
        log.debug("Setting the DbPool");
    }
}
```

Debugging Web Applications

- Firefox developer's toolbar
  - http://www.chrispederick.com/work/firefox/webdeveloper/
- Bookmarklets
  - Bookmarks (generally JavaScript™ technology) that expose information about the page
  - It is sometimes amazing how much you can find out
  - Search for Bookmarklets
    - www.**bookmarklets**.com/
    - www.squarefree.com/**bookmarklets**/

Tapestry

- Debugging JavaServer Pages™ (JSP™) technology is tough
- Mixed HTML, JavaScript technology, JSP technology custom tags, and JSP technology
- Tapestry has the best debugging aid of any web framework
- Inspector

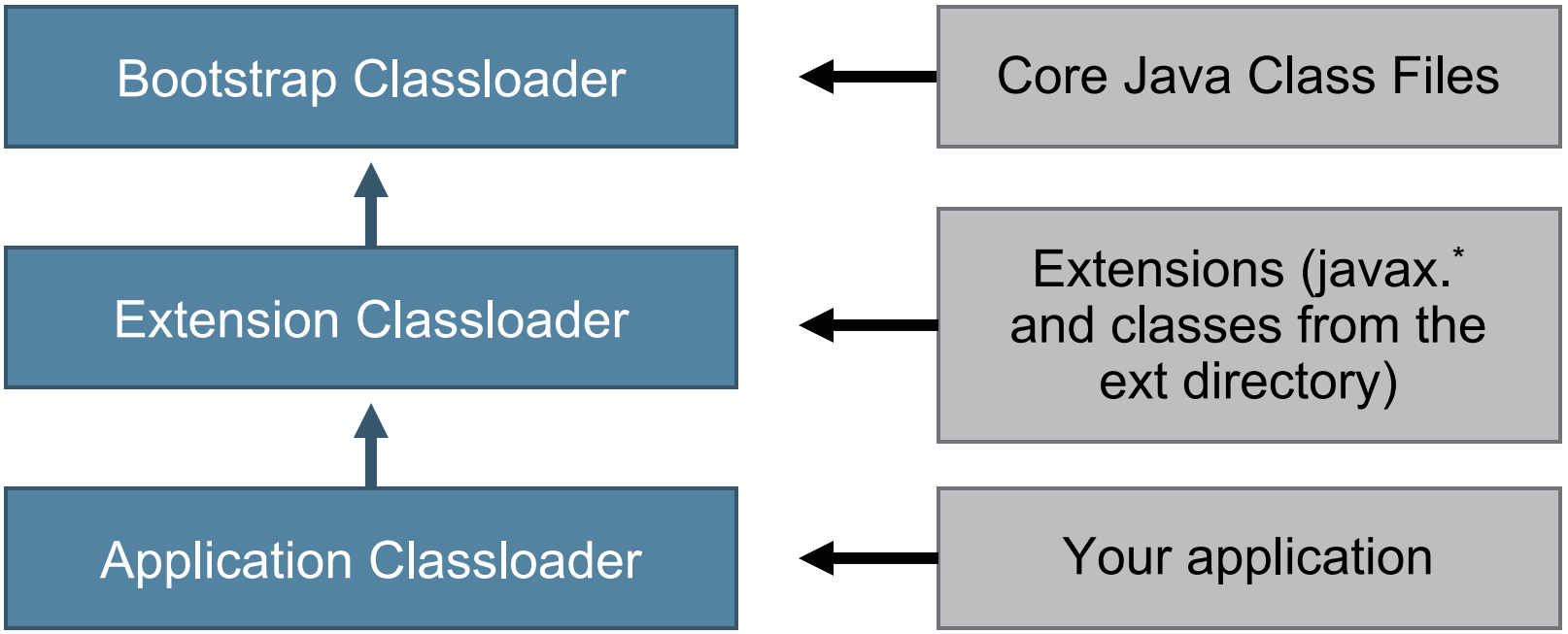

Tapestry Inspector: Tapestry Component Workbench - Mozilla
Fields
page
Specification
Template
Properties
Engine
Logging
Component Specification
Specification Resource Path /tutorial/workbench/fields/Fields.page
Java class tutorial.workbench.fields.Fields
Assets
Name Asset
continue ContextAsset[/images/workbench/Continue.gif]
Helper Beans
Name Class Lifecycle
bigDecimalValidator net.sf.tapestry.valid.NumberValidator RENDER
dateValidator net.sf.tapestry.valid.DateValidator RENDER
delegate tutorial.workbench.WorkbenchValidationDelegate REQUEST
doubleValidator net.sf.tapestry.valid.NumberValidator RENDER
intValidator net.sf.tapestry.valid.NumberValidator RENDER
longValidator net.sf.tapestry.valid.NumberValidator RENDER
stringValidator net.sf.tapestry.valid.StringValidator RENDER
Embedded Components
Id Type
bigDecimalLabel FieldLabel
border Border
continue ImageSubmit
dateLabel FieldLabel
doubleLabel FieldLabel
enabledForm Form
form Form
inputBigDecimal ValidField
inputDate ValidField
inputDouble ValidField
inputEnabled Checkbox
inputInt ValidField
inputLong ValidField
inputString ValidField
intLabel FieldLabel
longLabel FieldLabel
showError ShowError
stringLabel FieldLabel

Debugging Class Loaders

- One of the bane's of application servers are class loader issues
- Class loaders are responsible for loading classes within the VM
- When you execute a Java™ application, the native Java platform class loader loads (the bootstrap class loader)
- The Virtual Machine for the Java platform (JVM™ machine) loads 2 other class loaders by default
  - Extension class loader
  - Application class loader

The terms "Java Virtual Machine" and "JVM" mean a Virtual Machine for the Java™ platform.

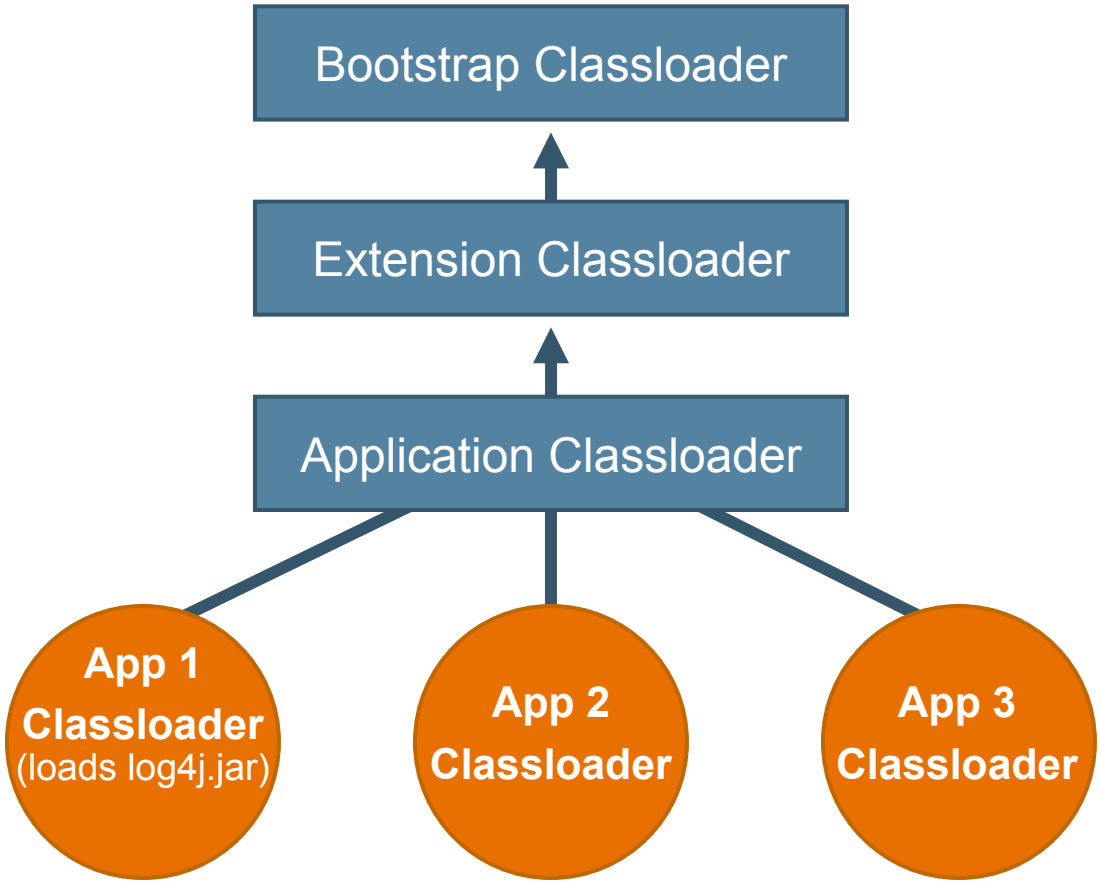
Class Loaders



Class Loaders

- Class loaders use the delegation model to load classes
  - Each class loader defers to its parent
- In application servers, each deployed application generally gets its own class loader
  - EAR files get a single class loader for both web and Enterprise JavaBeans™ (EJB™) architecture
- These class loaders are responsible for loading **and unloading** classes

Application Server Class Loaders



Why Should You Care?

- Shared libraries
  - Logger packages (like Log4J)
  - XML parsers
- You have several options:
  - Deploy the shared library on the application server's class loader
    - Move it to the app server's lib directory
    - Can cause problems

Class Loader Recipe

- If the utility class is required only by your web application, move it to WEB-INF/lib
- If you have a web application and EJB architecture that don't share a class loader
  - Make an entry in your manifest file for the EJB architecture
  - Specify all the utility classes using the `Classpath` entry
  - Keep this utility Java Archive (JAR) file either in a shared location where both web and EJB architecture can see it (but off the app server's boot classpath)

Debugging Class Loader Problems

- You usually don't see a problem when the class loads (or fails to load)
- The `toString()` methods of the Java Development Kit (JDK™) software class loaders don't provide much information
- What to do when you get the dreaded `ClassNotFound` exception
  1. Look for the line where your class loader's `loadClass()` call is involved
  2. Figure out what class loader is used there
  3. Figure out the parent class loaders recursively until you find the bootstrap class loader
  4. Figure out why the class cannot be found by any of these class loaders

Debugging Class Loader Problems

- Some causes:
  - An archive, directory, or other source for the classes was not added to the class loader asked to load the class, or to its parent
  - A class loader's parent is not set correctly
  - The wrong class loader is used to load the class in question

Debugging Class Loader Problems

- The 3rd option happens very easily
- In Java 2 Platform, Enterprise Edition (J2EE™ platform), an EJB bean is required to use the thread context class loader (`Thread.currentThread.getContextClassLoader()`) to load classes
- Some developers use `Class.forName()` to load a class without specifying the thread context class loader as the current class loader

Symbolic-Link Class Not Found

- If a symbolic link cannot be found, a `NoClassDefFoundError` occurs
  - This means that the code has been successfully compiled but can't find the referenced class at runtime
- How to investigate this problem:
  1. Find the `NoClassDefFoundError` and print out its stack trace
  2. Find the topmost line in the stack trace that is not class-loader-related; This most likely indicates the class of the symbolic link that was not found

Symbolic-Link Class Not Found

- How to investigate this problem:
  3. Figure out the class loader of the class containing the offending symbolic link
  4. List the parent class loaders recursively
  5. Figure out why the class is not available to any of these class loaders
- Potential causes:
  - An archive, directory, or other source for the classes was not added to the class loader asked to load the class, or to its parent
  - A class loader's parent is not set correctly
  - Symbolic links in a class are unaccessible by the containing class's class loader

Last Resort to Solve CL Issues

- Patch the JDK's Software Class Loader with an improved `toString()` method
  1. Take the source of `java.lang.ClassLoader`, `java.security.SecureClassLoader` and `java.net.URLClassLoader` and copy them
  2. Add the desired `toString()` method and make sure that every class loader prints out its own address by using `super.toString()` so that you can test if class loaders from the same class are identical or not
  3. JAR them up
  4. Add `-Xbootclasspath/p:`*`<the archive you just have created>`* to your script that starts Java platform

Last Resort

- We need three pieces of information from the class loader
  - An indicator of the instance
  - What archives are available to this class loader
  - Information about the parent class loader

Sample `toString()` for ClassLoader

```
public String toString() {
    if( getParent() != null ) {
       return "java.net.URLClassLoader:\n"
            + "hashcode: " + hashCode() + "\n"
            + "URLs: " + java.util.Arrays.asList(getURLs())
            + + "\n parent { " + getParent() + " }\n";
    } else {
        return "java.net.URLClassLoader:\n"
            + "hashcode: " + hashCode() + "\n"
            + "URLs: " + java.util.Arrays.asList(
           getURLs() ) + "\n";
    }
}
```

Investigating Class Loader Info

- Three solutions
  - Use a debugger
    - Find the parent(s) right away
    - Not persistent
  - Re-write the classloader code to create better toString() method
    - Invasive
    - Persistent
  - Use aspects
    - Inject your logging code
  - Remove it when done

Dynamic Server-Side Debugging

- Sometimes you need a little snippet of information from the application server that is very difficult to get from a debugger without Herculean effort
- Fiddle
  - A servlet that includes Groovy and a way to dynamically execute Groovy scripts
  - Groovy allows you to ferret out all sorts of server-side information

Fiddle Scripts

- Need to know which XML parser is being loaded?

```
import javax.xml.parsers.DocumentBuilderFactory as dbf
clazz = dbf.newInstance().newDocumentBuilder().class
clazz.protectionDomain.codeSource.location
```

- Interested in a particular Java Naming and Directory Interface™ (J.N.D.I.) API resource?

```
import javax.naming.*
ctx = new InitialContext()
ctx.lookup("java:/DefaultDS")
```

Fiddle Scripts

- You can use JavaScript technology as well
- Need to get a loader path?

```
importPackage(net.java.dev.fiddle);

clazz = new FiddleServlet().getClass();
clazz.getProtectionDomain().getCodeSource().getLocation();
```

Fiddle Scripts

- Using JavaScript technology to get session info

```
output = "";

keys = session.getAttributeNames();

while(keys.hasMoreElements()) {
  key = keys .nextElement();
  output = output + key + "=" + session.getAttribute(key) + "<br>";
}

output;
```

Fiddle Scripts

- Want to see all session variables?

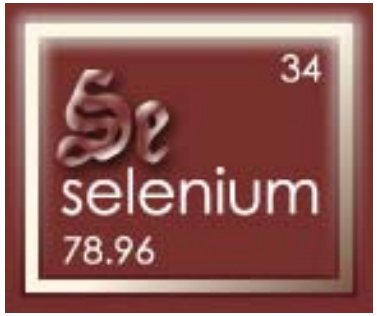
```
output = ""
sessionVars = [:]
for (n in session.attributeNames) {
  sessionVars[n] = session.getAttribute(n)
}
sessionVars.each { s |
  output = output + "${s.key} = ${s.value} <br>"
}
output
```

Automating Debugging Tasks

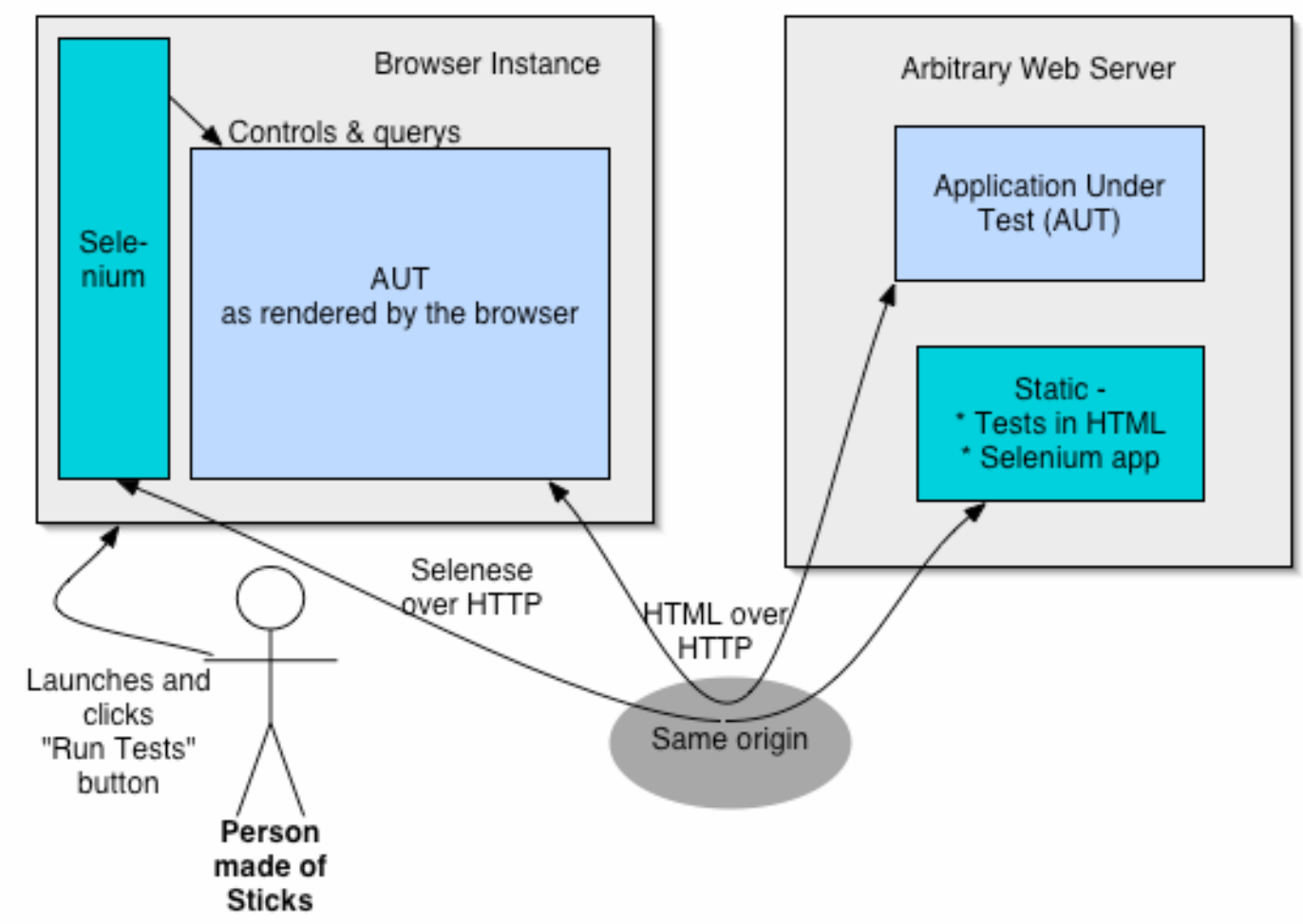
- How many times have you walked the same path through a web application to get to the point that you want to debug?
- Stop working so hard for your computer!
- Set up jWebUnit to “walk through” the application for you to a certain point
- jWebUnit is designed for unit testing but it has great automation for driving web applications
- While you are there, go ahead and write a unit test!

Automating Debugging Tasks

- What if you need to interact with the application once you've gotten to the critical point?
- Selenium
- A testing tool for web applications
- Selenium uses JavaScript technology and IFrames to embed a test automation engine in your browser
- This technique works with any JavaScript technology-enabled browser
- Developed by ThoughtWorks for internal use
- It was so useful that we've open-sourced it
- Version 1.0



How Does Selenium Work?



DEMO

Selenium

Selenium as a Debugging Aid

- Test cases are trivial to write
- Use Selenium to “walk” your web application to the point where you want to debug it
- Either:
  - Take over by hand
  - Use Selenium’s step behavior to check individual behavior
- Selenium now has an IDE to record tests

Words of Debugging Wisdom

- From **The Pragmatic Programmer**:
  - "Embrace the fact that debugging is just **problem solving**, and attack it as such"
  - Tip #24: Fix the problem, not the blame
  - Tip #25: Don't Panic
- **The Pragmatic Programmer** has lots of information about the debugging mindset
- Don't use brute force!
- Work the problem
- Apply as much ingenuity to debugging as to the design that got you here

Questions? Samples and Slides at www.nealford.com

Neal Ford
www.nealford.com
nford@thoughtworks.com
memeagora.blogspot.com

ThoughtWorks

JavaOne

Advanced Enterprise Debugging

Neal Ford

ThoughtWorker/Meme Wrangler
ThoughtWorks
www.thoughtworks.com

TS-4588