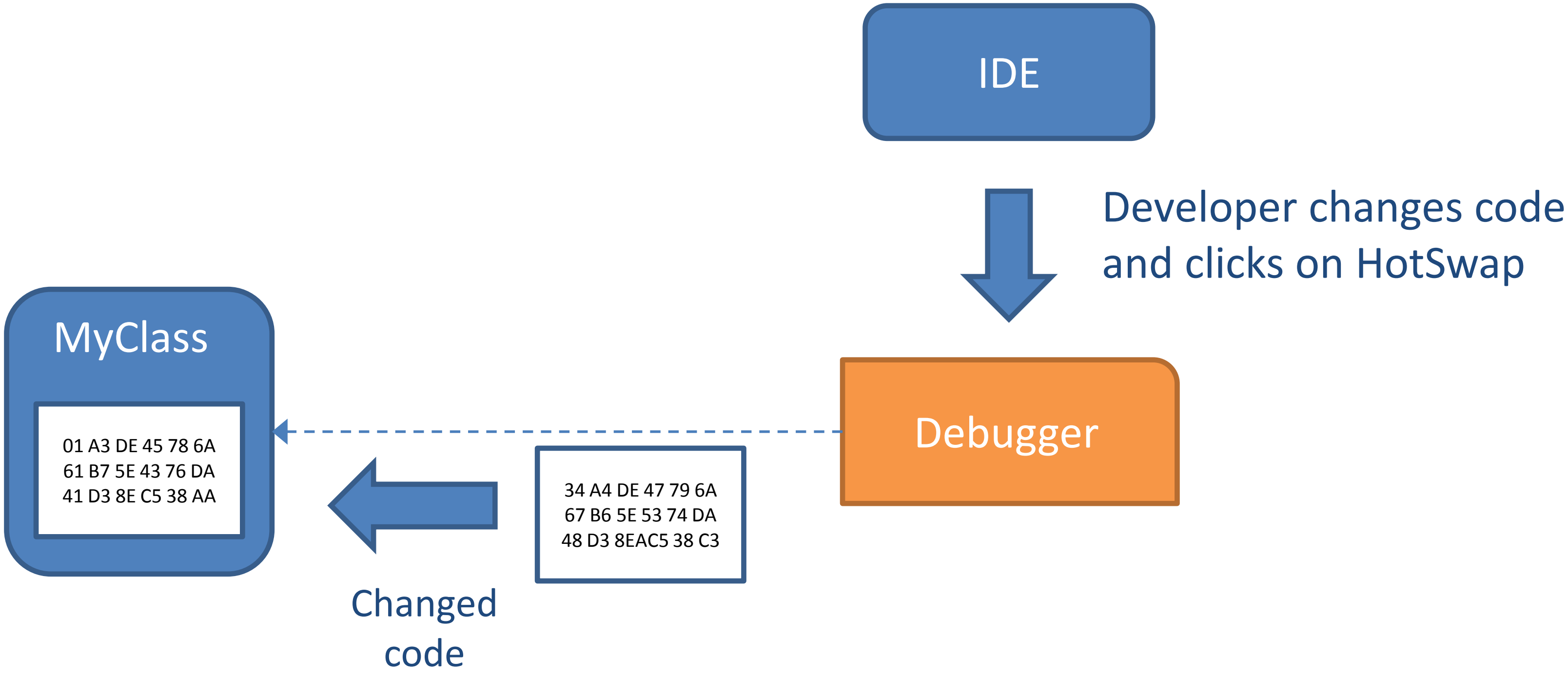
Dynamic Class Reloading in the Wild with Javeleon

Allan Gregersen
Founder & Chief Architect
Javeleon

René Jahn
Head of Research and Development
SIB Visions in Austria

Geertjan Wielenga
Principal Product Manager
Oracle Developer Tools group

HotSwap



IDE

Developer changes code and clicks on HotSwap

Debugger

MyClass

01 A3 DE 45 78 6A
61 B7 5E 43 76 DA
41 D3 8E C5 38 AA

34 A4 DE 47 79 6A
67 B6 5E 53 74 DA
48 D3 8EAC5 38 C3

Changed code

What‘s Javeleon all about?

```java
public class Student extends Person implements Printable {

    private String givenName;
    private String surName;

    public Student(String givenName, String surName){

        this.givenName = givenName;

        this.surName = surName;

    }


    public String getName() {
        return givenName;
    }
    @Override
    public void print(OutputStream out)
    {
        // print Student
    }
}
```

What can Javeleon do?

- Change Java classes at runtime – no restrictions!
- Unique features
  - Automatic field initialization for new fields
  - Replace super class
  - Change implemented interfaces
  - Different operation modes: reloading on file change vs. manual trigger

demo

Examples? Let‘s go into demo mode!

Javeleon in the wild

- Who can use Javeleon?
  - Java SE developers
  - NetBeans Platform developers
  - Wait: there‘s more in the new Javeleon 3.0
    - Java EE on Tomcat, Jetty & JBoss
    - Spring MVC, Bean Validation, RestEasy etc.

Let‘s look at a couple of examples live on stage!

René Jahn

Head of Research and Development

SIB VISIONS simplify IT business

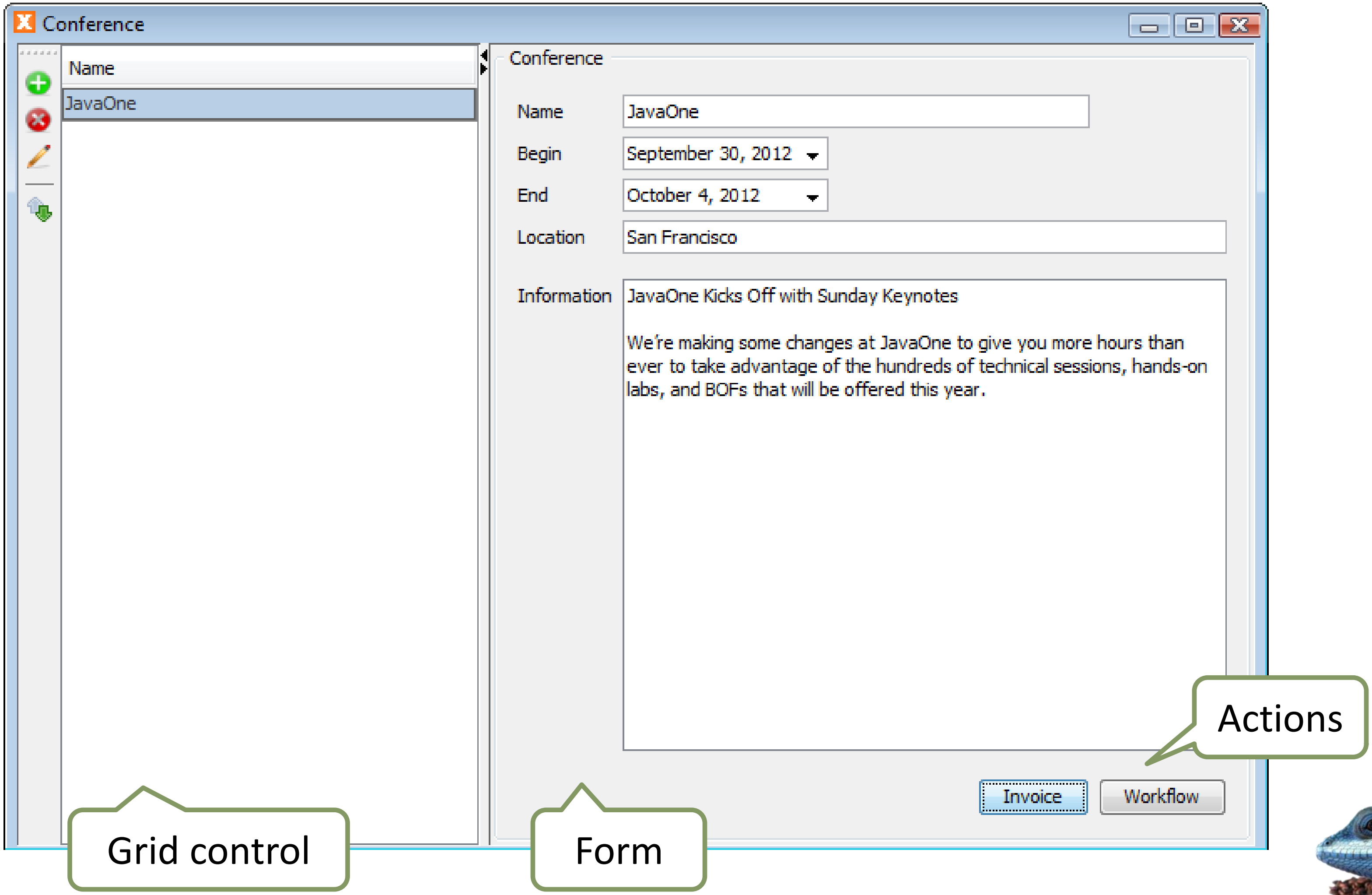
Why dynamic class reloading?

- Increase productivity and save development time.

Our main goals?

- Reduce application restarts to a bare minimum
- Reduction of manual UI tests
- View code changes immediately
- View translation and/or resource file changes immediately

A simple application screen



Conference

Name	JavaOne
Begin	September 30, 2012
End	October 4, 2012
Location	San Francisco
Information	JavaOne Kicks Off with Sunday Keynotes We're making some changes at JavaOne to give you more hours than ever to take advantage of the hundreds of technical sessions, hands-on labs, and BOFs that will be offered this year.

Actions

Grid control

Form

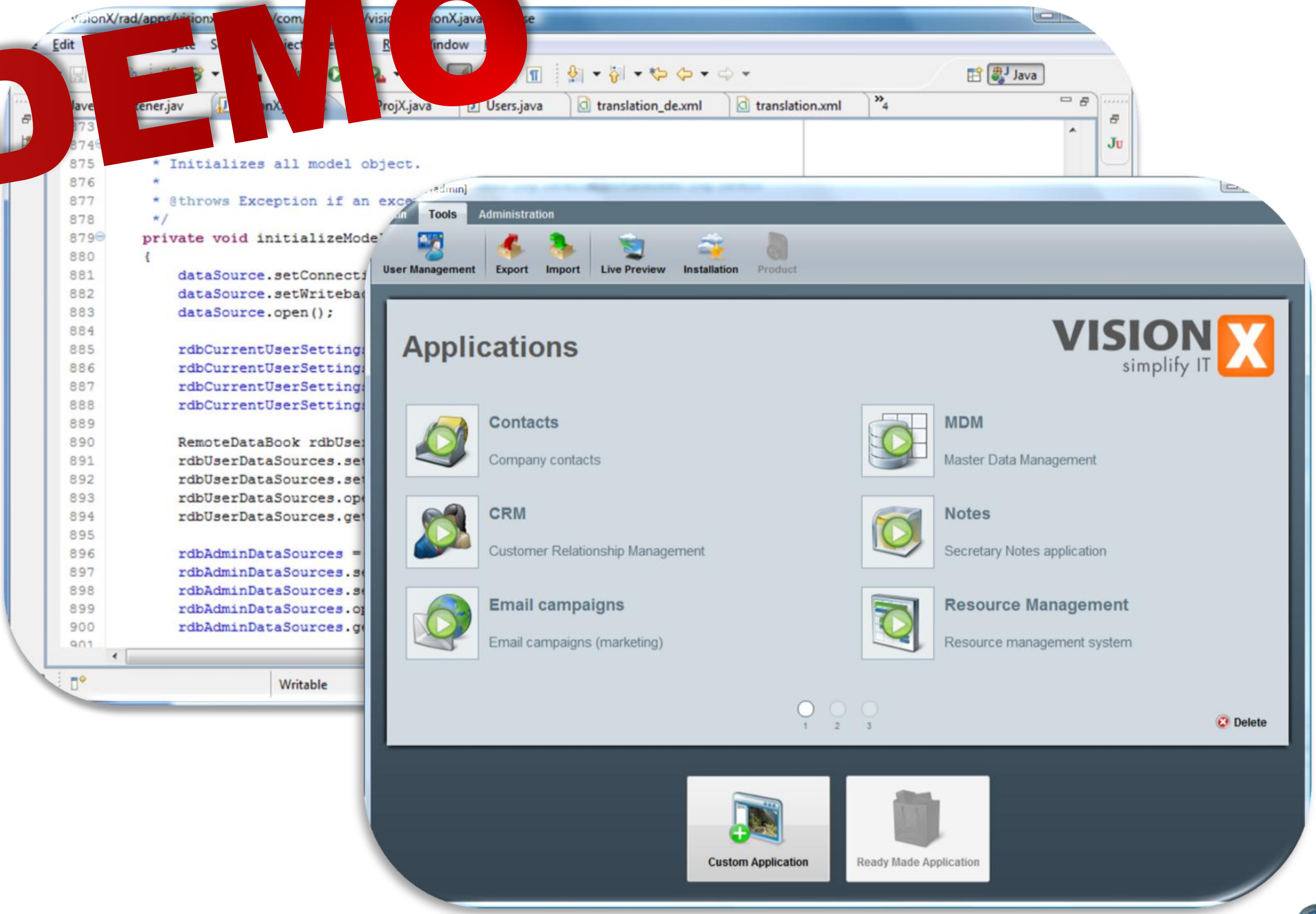
Why Javeleon?

- A research project from the University of Southern Denmark
- It supports all relevant features
- Only one jar file
- No runtime dependencies
- Easy to integrate in our development process
- Works without IDE plugins
- Great support from the development team

How do we use Javeleon?

- With Eclipse IDE, jar-only version (Java SE variant)
- Open Source and Product development team
- Development time only – not released with our products
- Live screen layouting and wizard layouting
- Live resource changes (translation, images, …)
- Live business logic changes

DEMO



```
     * Initializes all model object.
     *
     * @throws Exception if an exce
     */
    private void initializeMode
    {
        dataSource.setConnect
        dataSource.setWriteba
        dataSource.open();

        rdbCurrentUserSetting
        rdbCurrentUserSetting
        rdbCurrentUserSetting
        rdbCurrentUserSetting

        RemoteDataBook rdbUse
        rdbUserDataSources.se
        rdbUserDataSources.se
        rdbUserDataSources.op
        rdbUserDataSources.ge

        rdbAdminDataSources =
        rdbAdminDataSources.s
        rdbAdminDataSources.s
        rdbAdminDataSources.o
        rdbAdminDataSources.g
```

Applications

Contacts
Company contacts

MDM
Master Data Management

CRM
Customer Relationship Management

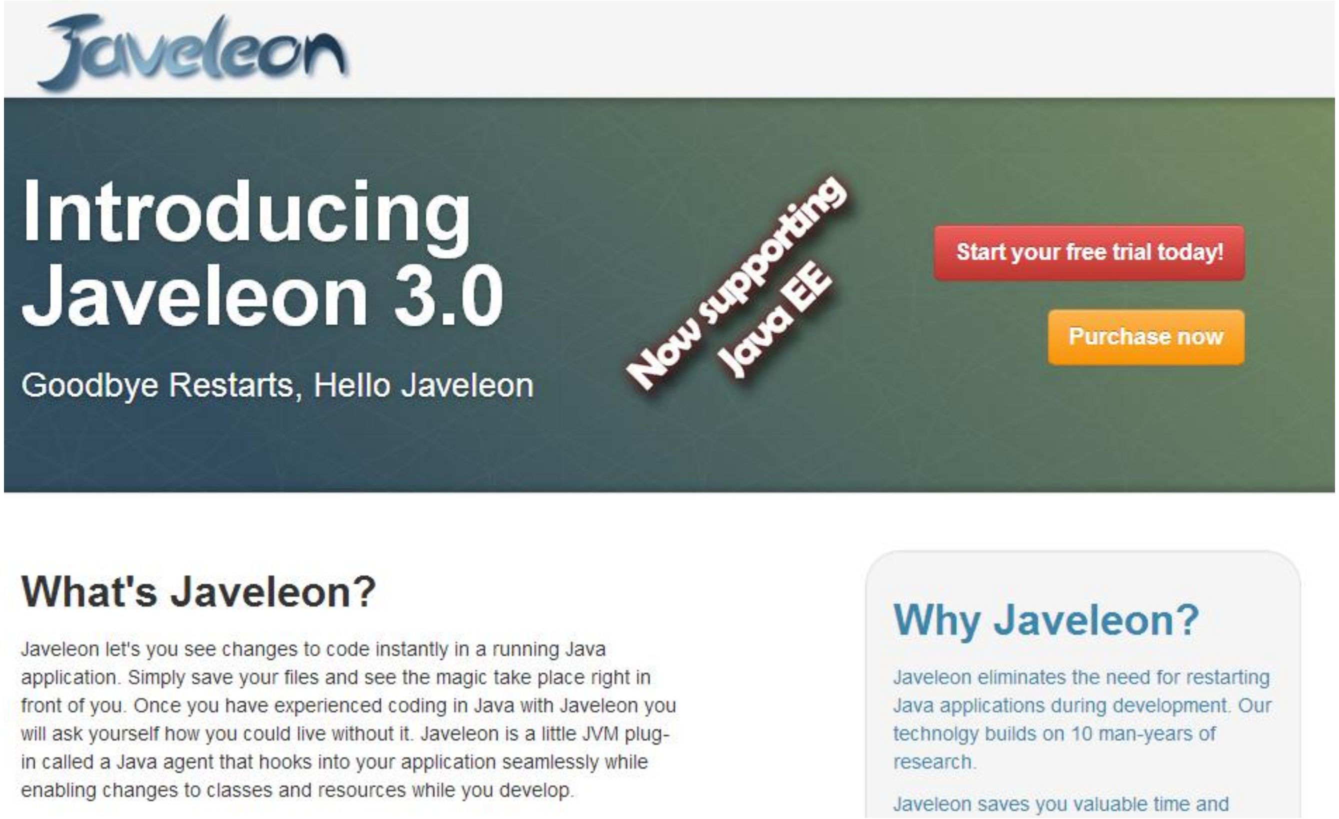
Notes
Secretary Notes application

Email campaigns
Email campaigns (marketing)

Resource Management
Resource management system

Javeleon integration?

1) Download Javeleon from http://www.javeleon.com



Introducing Javeleon 3.0

Goodbye Restarts, Hello Javeleon

What's Javeleon?

Javeleon let's you see changes to code instantly in a running Java application. Simply save your files and see the magic take place right in front of you. Once you have experienced coding in Java with Javeleon you will ask yourself how you could live without it. Javeleon is a little JVM plug-in called a Java agent that hooks into your application seamlessly while enabling changes to classes and resources while you develop.

Why Javeleon?

Javeleon eliminates the need for restarting Java applications during development. Our technolgy builds on 10 man-years of research.

Javeleon saves you valuable time and

Javeleon Reload Listener?

- Javeleon notifies its listeners after reload
- Update or repaint your GUI/Application
- A simple javax.swing.event.ChangeListener
- Configured as JVM argument

```
-Djaveleon.reload.listener=com.sibvisions.visionx.JaveleonListener
```

An example?

```
public class JaveleonListener implements ChangeListener
{
    private VisionX visionx;

    public JaveleonListener()
    {
        visionx = getApplication();
    }

    public void stateChanged(final ChangeEvent event)
    {
       UIComponent.invokeLater(new Runnable()
       {
           public void run()
           {
               visionx.updateUI();
           }
       });
    }
}
```

We use custom classloaders!

- to load user-defined libraries at runtime
- load dynamically compiled application classes
- load classes through a custom communication protocol

Javeleon supports custom classloaders!

- not automatically
- register/unregister your classloader

The Javeleon „API“

- Programmatic access to optional features
- NO dependency to Javeleon
- No difference between runtime and development mode
- Supports custom classloader registration

The Javeleon Paste Idea

```java
public class JaveleonReloadManager
{
    public static void register(ClassLoader ldr, ClassLoaderListener lis)
    {}

    public static void unregister(ClassLoader ldr)
    {}

    public static void reload()
    {}

    public static void monitorFolder(URI uri)
    {}

    public static interface ClassLoaderListener
    {
        public ClassLoader createModuleClassLoader(ClassLoader parent);
    }
}
```

Custom Classloader registration (original)

```
private ClassLoader loader;

public void init()
{
    AbstractConnection con = visionx.getConnection();
    DirectClassLoader dcloader = new DirectClassLoader(
                                          VisionX.class.getClassLoader());
    dcloader.setConnection(con);

    loader = dcloader;
}
```

Custom Classloader registration (Javeleon)

```
private ClassLoader loader;

public void init()
{
    ClassLoaderListener lis = new ClassLoaderListener()
    {
        public ClassLoader createModuleClassLoader(ClassLoader parent)
        {
            AbstractConnection con  = visionx.getConnection();
            DirectClassLoader dcloader = new DirectClassLoader(parent);
            dcloader.setConnection(con);

            loader = dcloader;

            return loader;
        }
    };

    loader = lis.createModuleClassLoader(VisionX.class.getClassLoader());

    JaveleonReloadManager.register(loader, lis);
}
```

Javeleon Eclipse Plugin

- Trigger Javeleon reload manually
- Automatic reload if resource files were changed
- Configured as JVM argument

```
-javaagent:…javeleon.jar=port=7999;
```

- OpenSource
- First version few features (needs helping hands)

http://sourceforge.net/projects/javilionplugin/

So what about Java EE?

Java EE

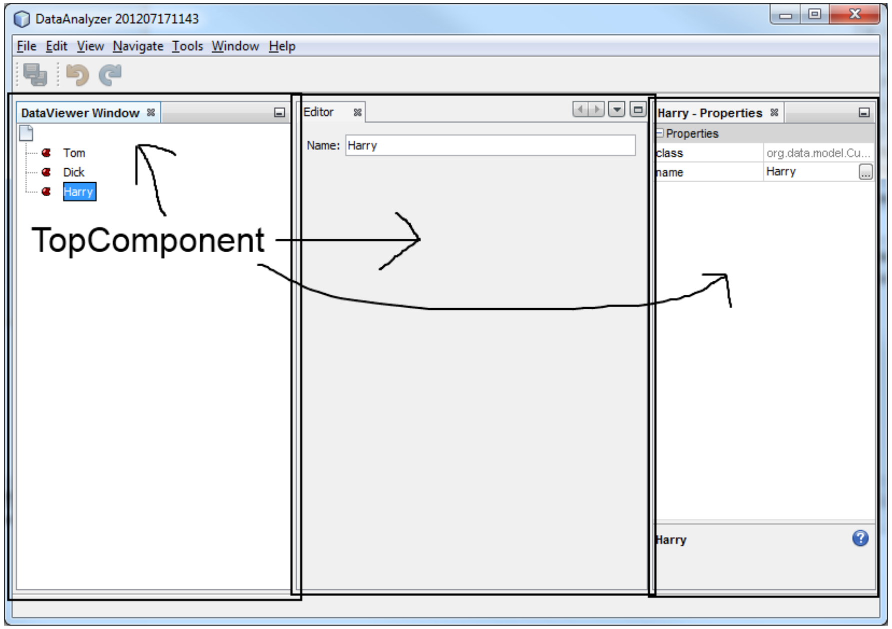
Application servers & Web frameworks

- Web applications use a broad variety of containers and frameworks
  - Many of them cache class metadata about classes
  - Meaning changes to classes won't show up at all!
  - Or you get a NoSuchMethodException because the cache did not contain up-to-date information!
- Specific integration for each server and framework required
  - Already, we have Tomcat 6 & 7, Jetty 7 & 8 and JBoss 7.1 integration
  - Bean validation, Spring MVC, Weld (CDI), RestEasy and more

demo

Another example from Geertjan Wielenga

NetBeans Application development?



Name: Harry

class	org.data.model.Cu...
name	Harry

NetBeans Platform

- NetBeans Platform Windowing System
  - Declarative registration
  - Compiled with annotation pre-processor
  - Stored in layer.xml file
- Meta data NOT picked up by class reloading system automatically!

- Good news, Javeleon has support!

demo

The Road Ahead

More Java EE, remote & more

- Development of core features
  - More Java EE support
  - Help us to prioritize development
- Remoting
  - Enable Javeleon to push changes to a remote server
- Transplanting our standalone Java SE & Java EE ‘reload on save’ to the NetBeans Platform
  - Especially beneficial for Maven based NetBeans Platform development

Thank You!

Q & A