

The Eclipse IDE logo, which consists of a large circle containing a stylized, teardrop-shaped element that resembles a leaf or a drop. The text "Eclipse + Html: A Journey" is centered within this logo.

Eclipse + Html: A Journey

Kris De Volder <kdevolder@gopivotal.com> , Pivotal Software
Martin Lippert <mlippert@gopivotal.com>, Pivotal Software

Outline

- Goal
- Motivation
- Case Studies
- The Journey
- API Comparison
- Lessons Learned
- What's next?
- Questions?

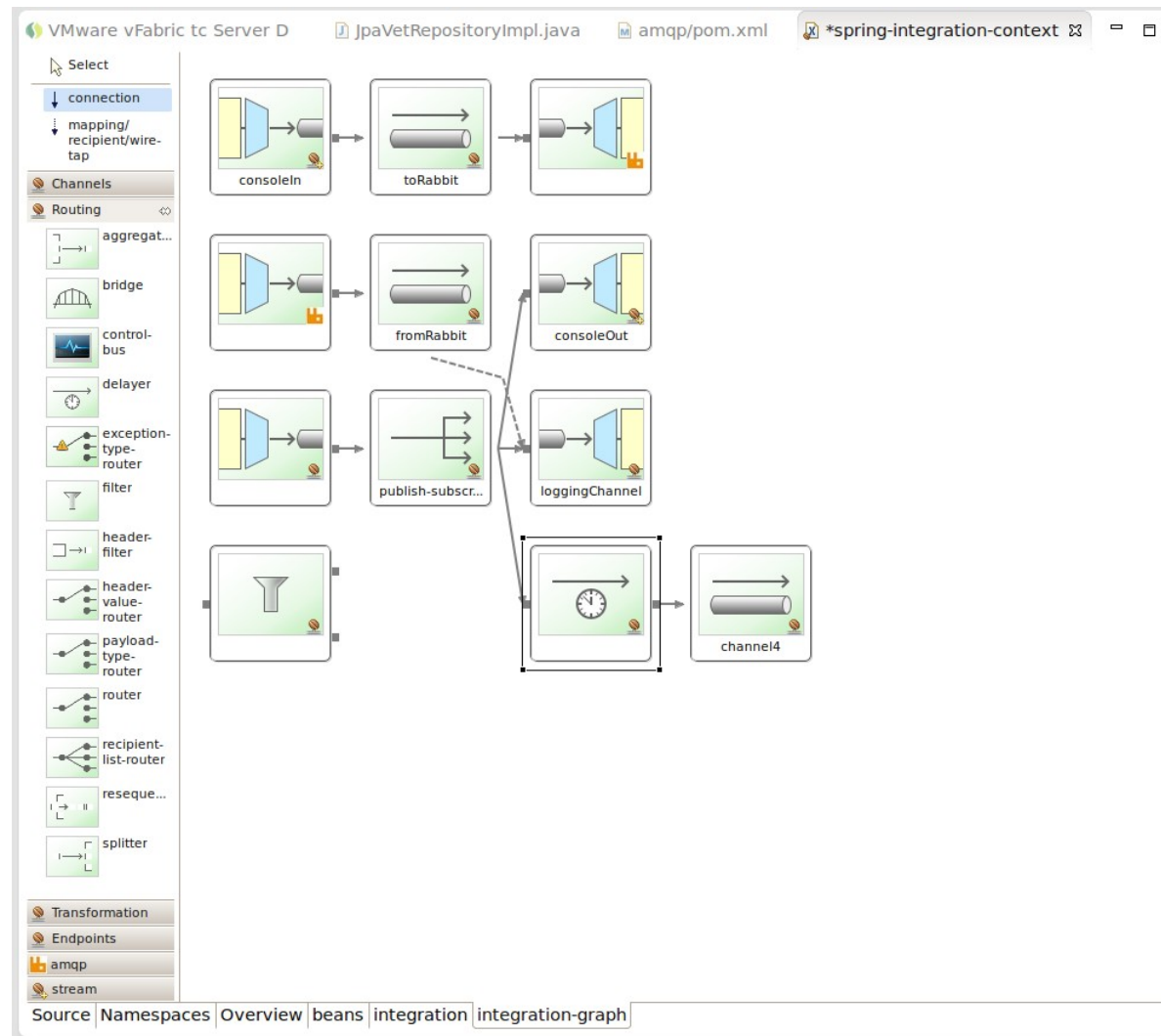
Goals

- Embed a **Rich HTML5 UI** into Eclipse.
- **Seamless Integration**
 - Eclipse Code Drives the Web UI
 - Web UI Triggers Eclipse Functionality
 - Web UI Looks like it Belongs
- **Cross Platform**
 - **OS:** Windows / Mac / Linux
 - **Eclipse** 3.8 ... 4.x
 - **Java** 6, 7, (8)

Why?

- It is Cool
 - HTML 5 is all the rage
 - More 'Sexy' UI
 - Stuff that is hard with standard SWT widget
- It is Economical
 - Build once
 - Reuse for Web & Desktop

Why? An Example



Case Study 1: STS Dashboard

Before:

The screenshot shows the Spring Dashboard website. The browser tabs include 'build.propertie', 'org.springframe', 'org.springsourc', and 'Dashboard'. The page title is 'Spring Dashboard'. There is a search bar for 'spring.io' and a 'Subscribe' button. The main content is divided into three sections: 'Create', 'Updates', and 'Feeds'. The 'Create' section has links for 'Java Project', 'Spring Project', and 'Spring Roo Project'. The 'Updates' section features a prominent announcement for 'Cloud Foundry Integration for Eclipse 1.6.0 released' by Pivotal, dated 24-Feb-2014, with several related links. Below this is a 'Help and Documentation' section with links to 'Community Support Forums', 'New and Noteworthy', 'Issue and Bug Tracker', 'Extensions', 'SpringSource Commercial Support', and 'Product Page'. The 'Feeds' section lists various articles and webinars, such as 'Webinar Replay: Spring Data Repositories - Best Practices' and 'This Week in Spring - March 4th, 2014'. The bottom of the page has a navigation bar with 'Dashboard' and 'Extensions' tabs.

Case Study 1: STS Dashboard

After:

build.propertie org.springframe org.springsourc Dashboard »4

spring DOCS GUIDES ISSUES BLOG FORUM

Tooling Updates

- ! **Cloud Foundry Integration for Eclipse 1.6.0 released**
- ! **Thanks for installing STS/GGTS 3.5.0.M2**

News

Webinar Replay: Spring Data Repositories - Best Practices
Speakers: Oliver Gierke and Thomas Darimont Slides:
<https://speakerdeck.com/olivergierke/spring-data-repositories-best-practices> The repository abstraction layer is one of the core pieces of the Spring ... Pieter Humphrey 5-Mar-2014

This Week in Spring - March 4th, 2014
Welcome to another installment of This Week in Spring. As usual, we've got a lot to cover so let's get to it! Spring Batch and Boot co-founder Dr. Dave Syer has announced that Spring Boot RC4 is now available ... Josh Long 4-Mar-2014

Spring Data Redis 1.2 GA Released
Spring Data Redis 1.2.0 has been released and is now available from Maven Central. This release sums up the fixes and enhancements from 1.1.1 and RC1 plus an updated documentation. The release has been ... Christoph Strobl 4-Mar-2014

SpringOne2GX 2013 Replay: Spring Integration Internals
Recorded SpringOne2GX 2013 in Santa Clara, CA Speaker: Gary Russell A comprehensive review of message routing within a flow - including exactly how and when replyChannel and errorChannel headers are used ... Pieter Humphrey 4-Mar-2014

Get Started!

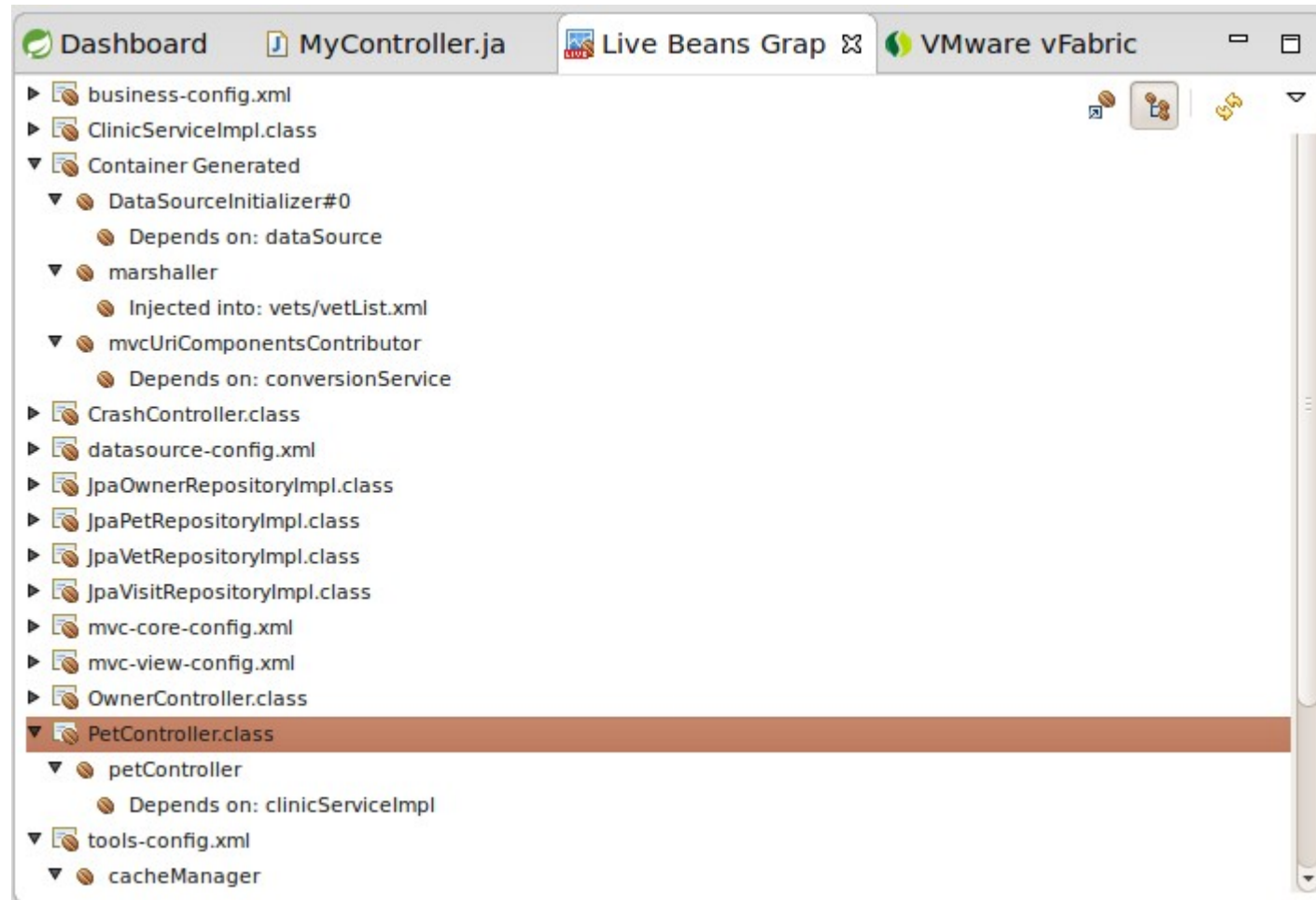
- IMPORT GETTING STARTED GUIDE
- IMPORT TUTORIAL GUIDE
- IMPORT REFERENCE APP
- CREATE JAVA PROJECT
- CREATE SPRING STARTER PROJECT
- CREATE SPRING ROO PROJECT

Manage

- IDE EXTENSIONS

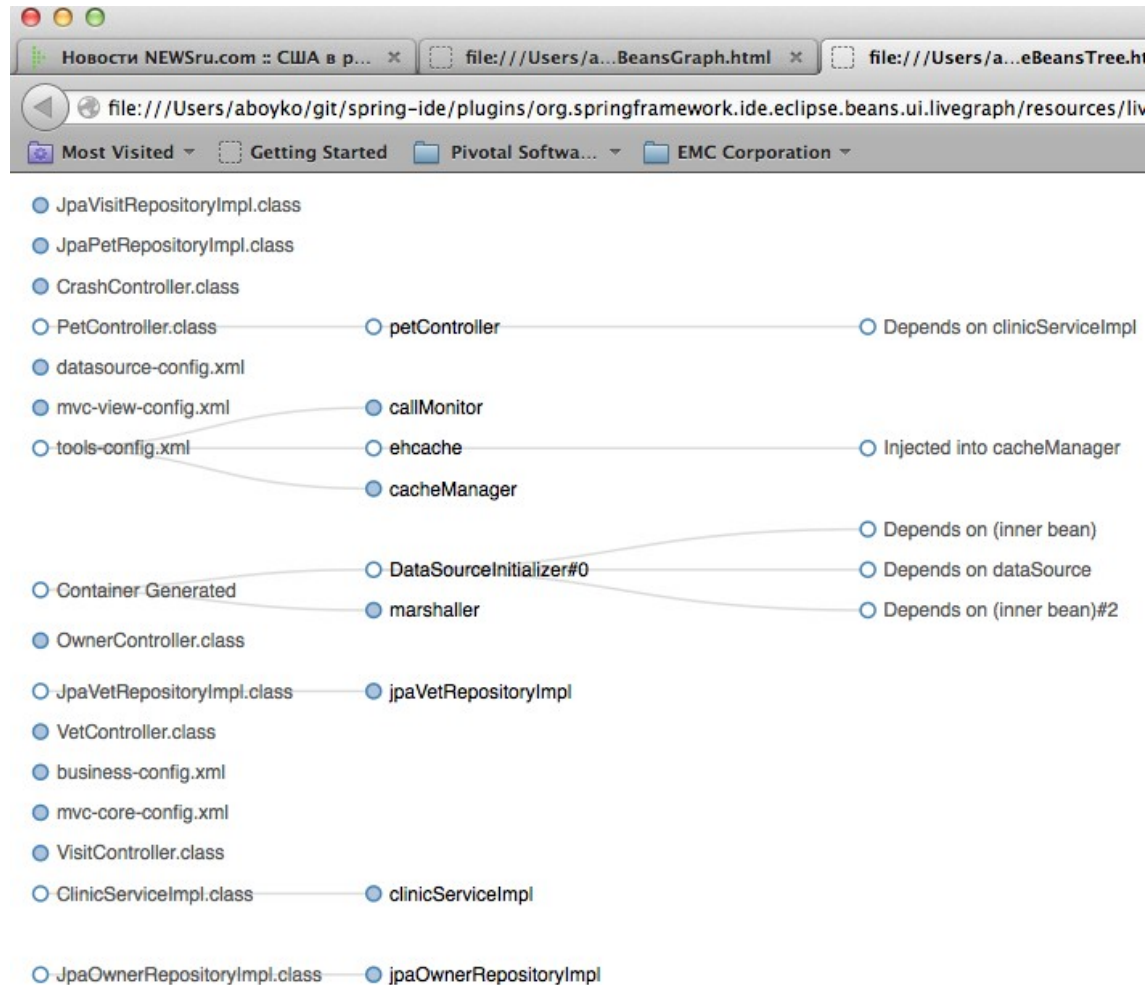
Case Study 2: Spring Live Beans Graph

Before:



Case Study 2: Spring Live Beans Graph

After:



Outline

- Goal
- Motivation
- The Case Studies
- **The Journey**
- API Comparison
- Lessons Learned
- What's next?
- Questions?

The Journey Begins

SWT Browser Widget as '**Advertised**'

- Since Eclipse 3.0
- Uses Platform's native Browser.
- Embedded into a SWT Widget.
- Can be placed inside UI amongst other widgets.
- Linux / Mac and Windows Supported
- Good API to Integrate With Eclipse
 - listeners
 - inject code and functions

The Journey Begins

SWT Browser Widget as '**Advertised**'

- Since Eclipse 3.0
- Uses
- Embe
- Can l
- Linux
- Good
 - list
 - inject code and functions

The Ideal Solution!
But is it?

SWT Browser on **Windows / Mac**

- Uses Native Browser Support (IE / WebKit)
 - Works out of the box.
 - Minor issues experienced on Windows:
 - IE Browser Quirks
 - SWT Browser forced lower IE <9 mode. Even if user has IE 10 installed.
 - Can be dealt with by setting extra system property:
-Dorg.eclipse.swt.browser.IEVersion=10001

SWT Browser Linux Support

- Linux is Supported But...
 - Requires Specific Native Libraries
 - Webkit for GTK2 or ...
 - XULRunner (ancient version)
 - Depending on Distro / Version and installed Third Party Components
 - Libraries may be missing
 - Libraries may cause frequent JVM crashes

SWT Browser Linux Support

- Personal Experience on Ubuntu (*)
 - Ubuntu 10.04 (now obsolete)
 - Great: Works out of the box
 - Ubuntu 12.04 (current LTS)
 - Install non-default libWebKit for GTK2.
 - causes JVM crashes in combination with Google Talk plugin (**)
 - Downloading ancient XULRunner and set some system props => works.
 - ...

(*) = https://bugs.eclipse.org/bugs/show_bug.cgi?id=420030

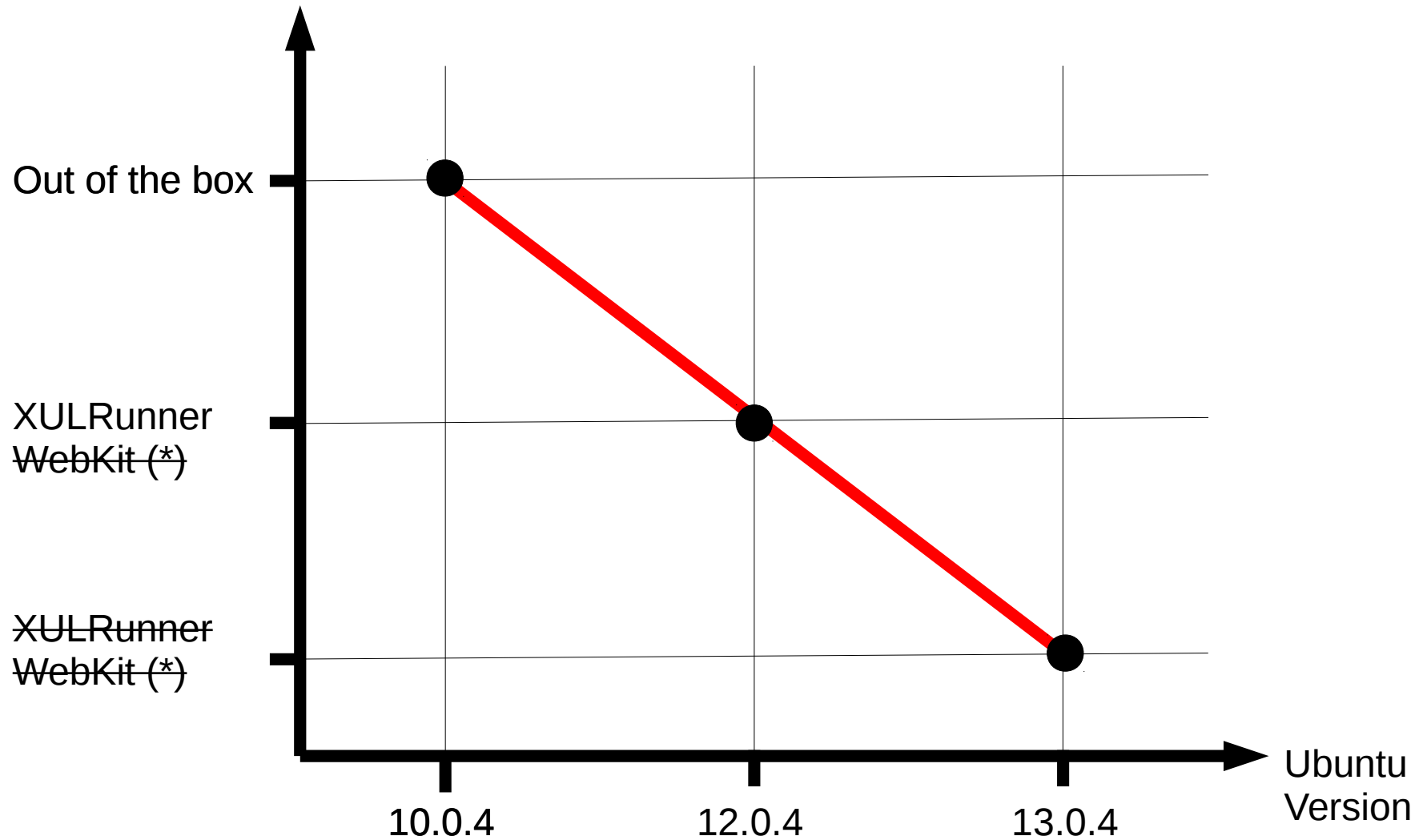
(**) = https://bugs.eclipse.org/bugs/show_bug.cgi?id=334466

SWT Browser Linux Support

- Personal Experience on Ubuntu (cont)
 - Ubuntu 13.04
 - Ancient XULRunner no longer works (crashes immediately)
 - WebKitGTK solution works but also affected by JVM Crash bug (*)

(*) = https://bugs.eclipse.org/bugs/show_bug.cgi?id=334466

SWT Browser: Linux



(*) = https://bugs.eclipse.org/bugs/show_bug.cgi?id=334466

The Journey Continues

PLAN B: JavaFX “WebView”

- JavaFX ships with JDK 7
- Embedded Copy of WebKit Included
- e(fx)clipse: tooling and runtime for Eclipse / OSGI.

JavaFX Experience

APIs / Integration

- Comparable to SWT widget but no 1-1 mapping.
 - Inject Code / Functions
 - Some listener support

=> Viable substitute for SWT Widget.

- Similar Capabilities
- Refactoring existing code requires effort.

JavaFX Experience

APIs

- Comparable to SWT widget but no 1-1 mapping.
 - Inject Code / Functions
 - Some listener support

**Viable substitute for SWT Widget.
Refactoring existing code requires some effort.**

JavaFX Experience

Availability / Cross Platform

- Sun JDK 7 or better required
- Eclipse 4.x

**Viable substitute for SWT Widget.
Cross Platform not Ideal
OpenJDK on Linux?
Expected to improve over time (newer JVM = better)**

Outline

- Goal
- Motivation
- The Case Studies
- The Journey
- **API Comparison**
- Lessons Learned
- What's next?
- Questions?

API comparison

- How to
 - Embed in SWT
 - Call Java function from Browser

SWT: Embed Browser in SWT

- How? => Relatively Trivial

The SWTBrowserWidget is an SWT widget. Just instantiate and use like any other widget.

```
import org.eclipse.swt.browser.Browser;
```

```
Composite parent = ...
```

```
...
```

```
Browser browser = new Browser(parent, SWT.NONE);
```

```
    //Note: style constant can also specify Mozilla vs Webkit
```


JavaFX: Embed Browser in SWT

- How? Slightly harder
 - Use EFX JavaFX runtime
 - Wrap JavaFX component in a FXCanvas

From the JavaDocs:

```
public class FXCanvas  
extends org.eclipse.swt.widgets.Canvas
```

FXCanvas is a component to embed JavaFX content into SWT applications. The content to be displayed is specified with the `setScene(javafx.scene.Scene)` method that accepts an instance of JavaFX Scene. After the scene is assigned, it gets repainted automatically. All the input and focus events are forwarded to the scene transparently to the developer.

JavaFX: Embed Browser in SWT

```
public class JavaFxBrowserViewer extends Composite {  
    ...  
  
    public JavaFxBrowserViewer(Composite parent, int style) {  
        ...  
        final FXCanvas fxCanvas = new FXCanvas(this, SWT.NONE);  
        fxCanvas.setLayoutData(...);  
        fxCanvas.setLayout(GridLayoutFactory.fillDefaults().create());  
        browser = new WebView();  
        BorderPane border = new BorderPane();  
        Scene scene = new Scene(border);  
        border.setCenter(browser);  
        fxCanvas.setScene(scene);  
        ...  
    }  
}
```

API comparison

- How to
 - Embed in SWT
 - **Call Java function from Browser**

Calling Java from The Browser

Html/JavaScript:

```
<a href=""  
  onclick="ide.call('openWizard', 'org.springframework.newSpringProject')">  
  Spring Project  
</a>
```

Adding a Browser Function

- Both JavaFX and SWT Browser each provide a mechanism to define a function in Java and inject it into your page(s).
- The main mechanism to integrate between Eclipse & Browser functionality
- Capabilities similar, APIs slightly different.

SWT Browser: Injecting A Function

```
import org.eclipse.swt.browser.Browser;
import org.eclipse.swt.browser.BrowserFunction;

public class MyJSFunction extends BrowserFunction {

    public ImportJSFunction(Browser browser) {
        super(browser, "ide");
    }

    @Override
    public Object function(Object[] arguments) {
        ...do your thing...
    }
}
```

JavaFX: Injecting a Function

```
import javafx.scene.web.WebEngine;
import javafx.scene.web.WebView;

/**
 * @author Miles Parker
 */
public class JavaFxBrowserManager {

    private WebEngine engine;
    private WebView view;

    ...

    public void setClient(WebView view) {
        ...
        this.engine = view.getEngine();
    }
}
```

JavaFX: Injecting a Function

```
public class JavaFxBrowserManager {
    ...

    public void setClient(WebView view) {
        ...
        this.engine = view.getEngine();
        JSObject window = (JSObject)engine.executeScript("window");
        window.setMember("ide", this);
        ...
    }

    /**
     * Called via reflection by JavaFx.
     */
    public void call(String functionId, String argument) {
        ...insert your code here...
    }
}
```


Outline

- Goal
- Motivation
- The Case Studies
- The Journey
- API Comparison
- **Lessons Learned**
- What's next?
- Questions?

Lessons Learned

			SWT	JavaFX	
API/Integration			✓	✓	
X-Platform	Mac		✓	✓	
	Windows		✓	✓	
	Linux	OpenJDK		✗	✗
		SunJDK		✗	✓
	Java 6		✓	✗	
	Eclipse 3.x		✓	✗	
	Recent (J7 E4)		✓	✓	

Other Lessons Learned

- Debugging Complications
 - Embedded widget may behave differently
 - No 'Chrome Dev Tools' available
 - => Complicates debugging embedded app
- Solutions
 - console.log (but doesn't work with JavaFX webkit)
 - create your own console.log by adding messages to the dom.

Outline

- Goal
- Motivation
- The Case Studies
- The Journey
- API Comparison
- Lessons Learned
- **What's next?**
- Questions?

What's Next?

- Neither JXF or SWT cover all cases we care about.
 - More Linux Trouble Ahead?
 - SWT Switch on GTK3 by Default
=> Incompatible with JFX at this time.
 - Ironically... SWT Widget will probably work better in GTK3 on most Linux Systems!
- => Refactor to use either SWT or JavaFX WebView interchangeably**

Thanks

Kris De Volder <kdevolder @ gopivotal.com>

Martin Lippert <mlippert @ gopivotal.com>

Credit to:

- Alex Boyko <aboyko @ gopivotal.com>
- Miles Parker <miles.parker @ tasktop.com>