



e(fx)clipse – JavaFX Tooling

Tom Schindl – BestSolution Systemhaus GmbH

EclipseCon October 2012

(c) Tom Schindl – BestSolution Systemhaus GmbH

About Tom

- ❖ CTO BestSolution Systemhaus GmbH
- ❖ Eclipse Committer
 - ❖ e4
 - ❖ Platform UI
 - ❖ EMF
- ❖ Main developer of e(fx)clipse
- ❖ Twitter: @tomsontom



About e(fx)clipse

About e(fx)clipse

- ❖ Opensource project lead by BestSolution.at

About e(fx)clipse

- ❖ Opensource project lead by BestSolution.at
- ❖ All sources provided under EPL

About e(fx)clipse

- ❖ Opensource project lead by BestSolution.at
- ❖ All sources provided under EPL
- ❖ Started in Summer 2011

About e(fx)clipse

- ❖ Opensource project lead by BestSolution.at
- ❖ All sources provided under EPL
- ❖ Started in Summer 2011
- ❖ Releases every 6-8 weeks (~2500 Downloads/release)

About e(fx)clipse

- ❖ Opensource project lead by BestSolution.at
- ❖ All sources provided under EPL
- ❖ Started in Summer 2011
- ❖ Releases every 6-8 weeks (~2500 Downloads/release)
- ❖ Community Support: www.efxclipse.org

About e(fx)clipse

- ❖ Opensource project lead by BestSolution.at
- ❖ All sources provided under EPL
- ❖ Started in Summer 2011
- ❖ Releases every 6-8 weeks (~2500 Downloads/release)
- ❖ Community Support: www.efxclipse.org
- ❖ Commercial Support: www.bestsolution.at

e(fx)clipse - Tooling

Basic-Tooling



e(fx)clipse - Tooling

Basic-Tooling

- ✧ Java
 - ✧ Classpath-Container for JDT
 - ✧ Bootstrap Wizards



e(fx)clipse - Tooling

Basic-Tooling



- * Java
 - * Classpath-Container for JDT
 - * Bootstrap Wizards
- * CSS
 - * Custom written CSS-Editor

e(fx)clipse - Tooling

Basic-Tooling



- ✧ Java
 - ✧ Classpath-Container for JDT
 - ✧ Bootstrap Wizards
- ✧ CSS
 - ✧ Custom written CSS-Editor
- ✧ FXML
 - ✧ Customized WST-XML-Editor

e(fx)clipse - Tooling

Basic-Tooling



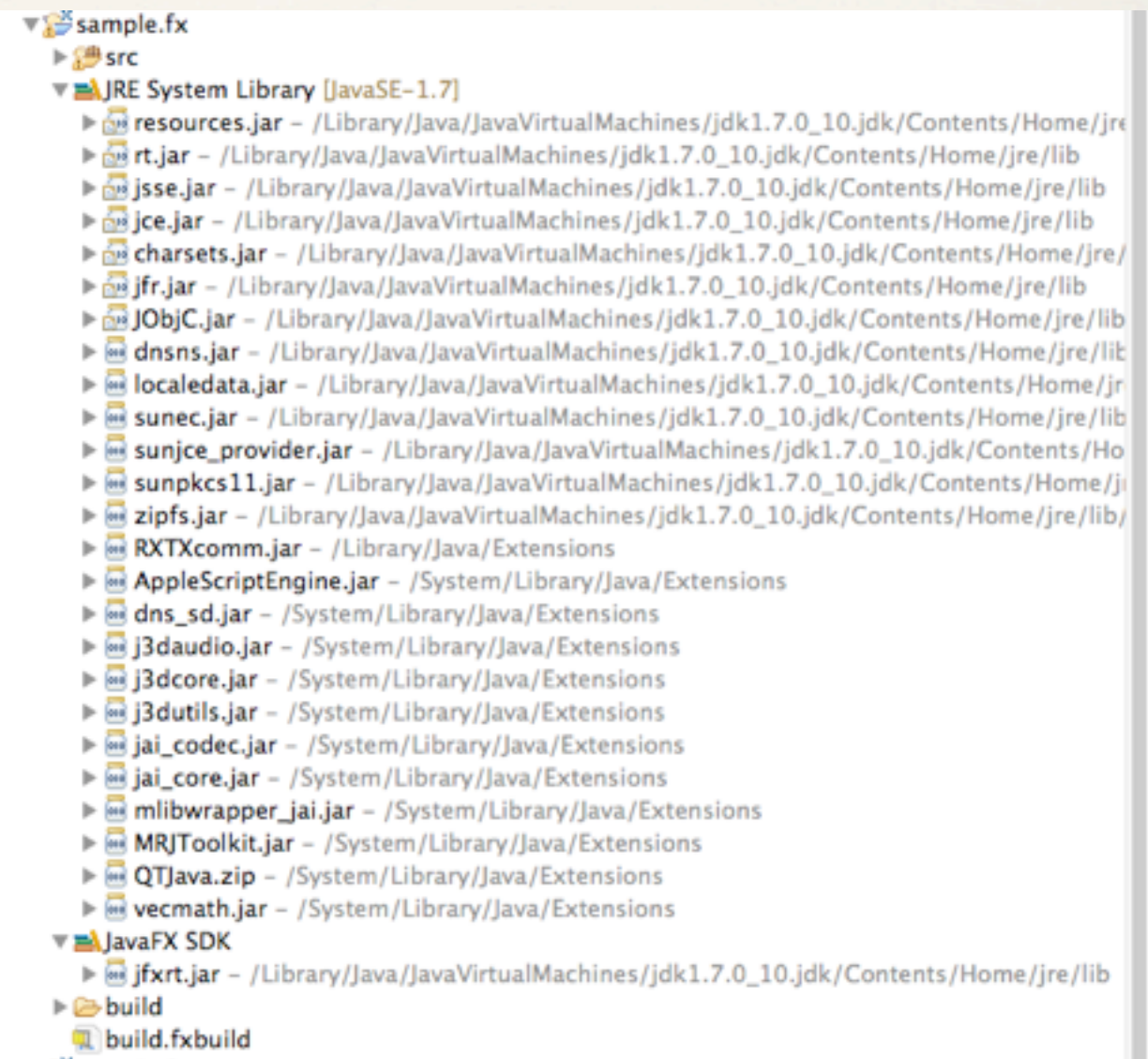
- * Java
 - * Classpath-Container for JDT
 - * Bootstrap Wizards
- * CSS
 - * Custom written CSS-Editor
- * FXML
 - * Customized WST-XML-Editor
- * Export
 - * Support for fx-ant tasks

e(fx)clipse - The classpath

JRE 1.7

Bootclasspath
rt.jar
...

javafxrt.jar

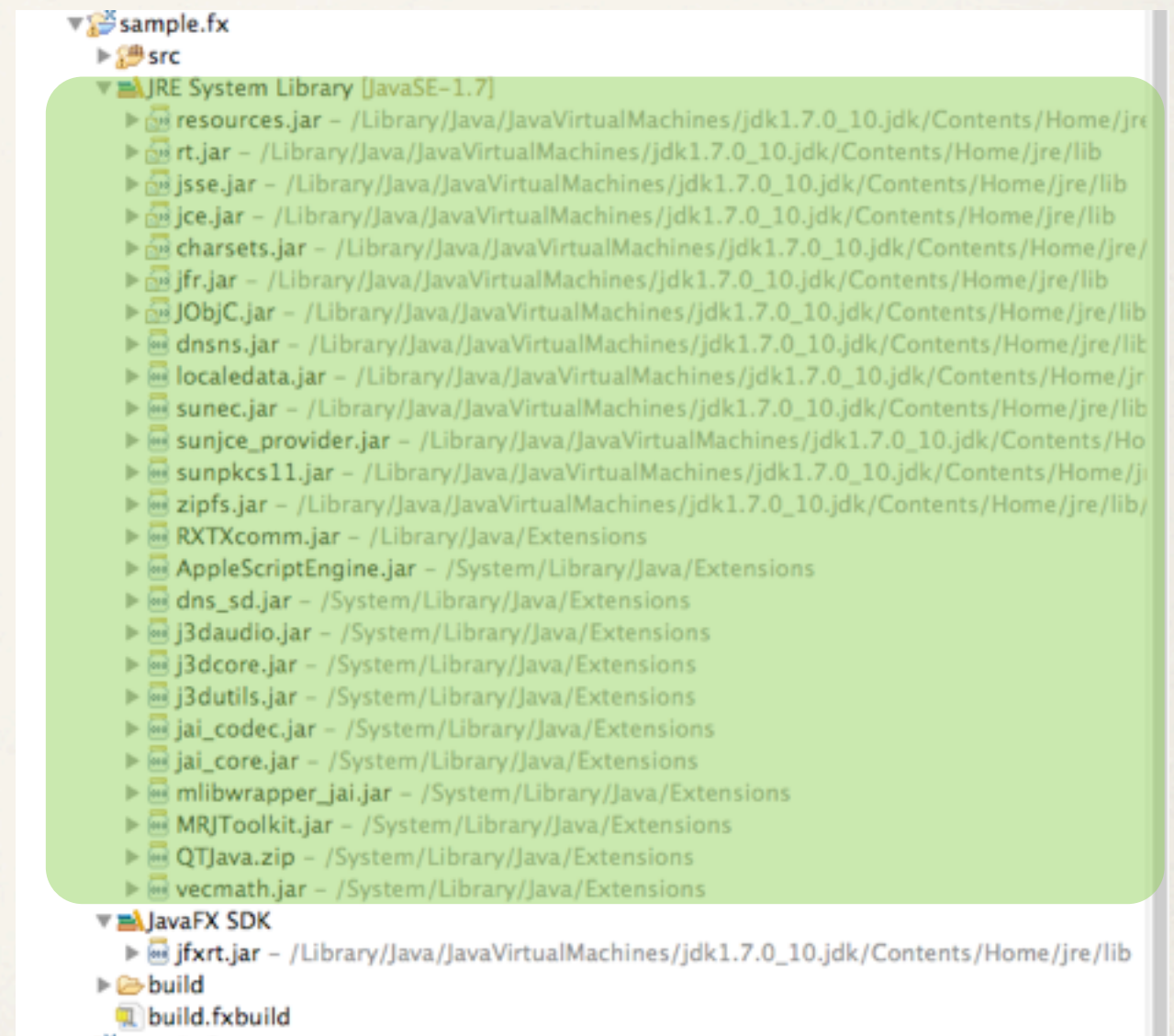


e(fx)clipse - The classpath

JRE 1.7

Bootclasspath
rt.jar
...

javafxrt.jar

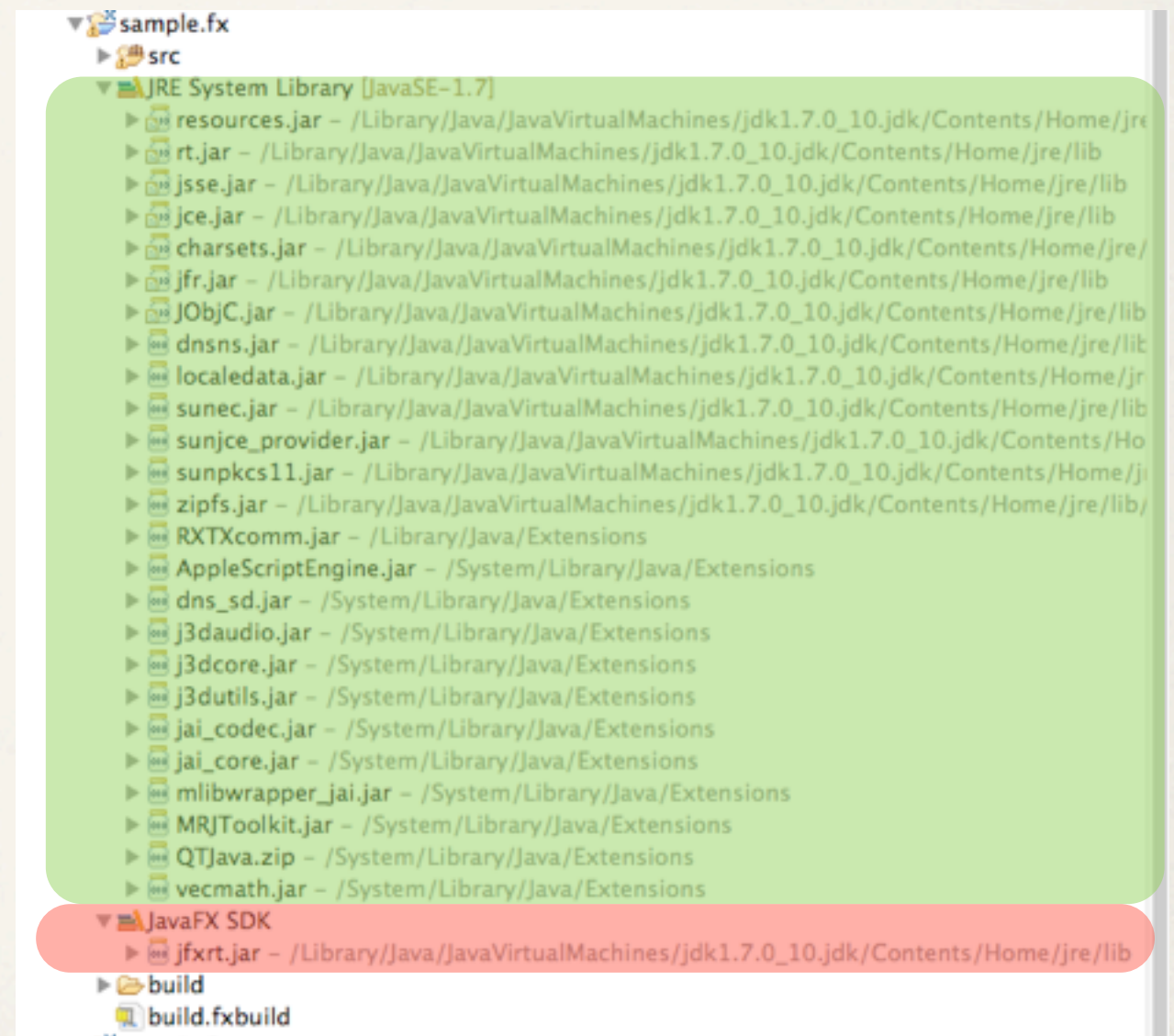


e(fx)clipse - The classpath

JRE 1.7

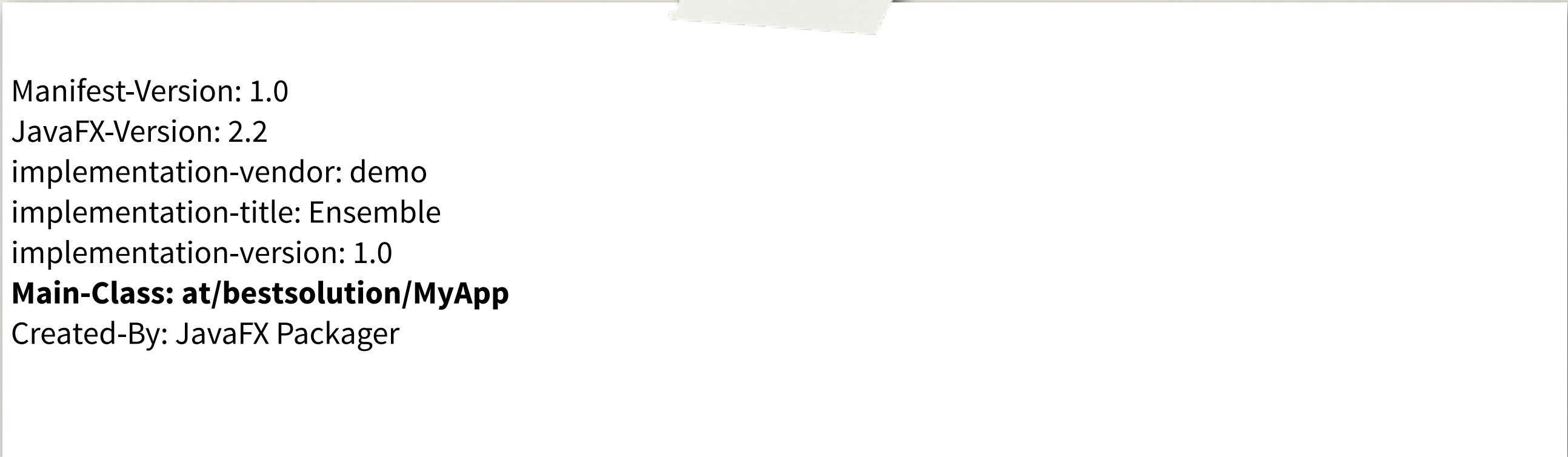
Bootclasspath
rt.jar
...

javafxrt.jar



e(fx)clipse - The classpath

- ✧ JavaFX apps can't be simply launched



Manifest-Version: 1.0
JavaFX-Version: 2.2
implementation-vendor: demo
implementation-title: Ensemble
implementation-version: 1.0
Main-Class: at/bestsolution/MyApp
Created-By: JavaFX Packager

e(fx)clipse - The classpath

- ✧ JavaFX apps can't be simply launched

Manifest-Version: 1.0

JavaFX-Version: 2.2

implementation-vendor: demo

implementation-title: Ensemble

implementation-version: 1.0

JavaFX-Application-Class: at.bestsolution.MyApp

Created-By: JavaFX Packager

Main-Class: com/javafx/main/Main

e(fx)clipse - CSS + FXML + JVM-Language

e(fx)clipse - CSS + FXML + JVM-Language

- ✧ FXML for the structure

e(fx)clipse - CSS + FXML + JVM-Language

- ✧ FXML for the structure
- ✧ CSS for the styling

e(fx)clipse - CSS + FXML + JVM-Language

- ❖ FXML for the structure
- ❖ CSS for the styling
- ❖ Programming language for the logic

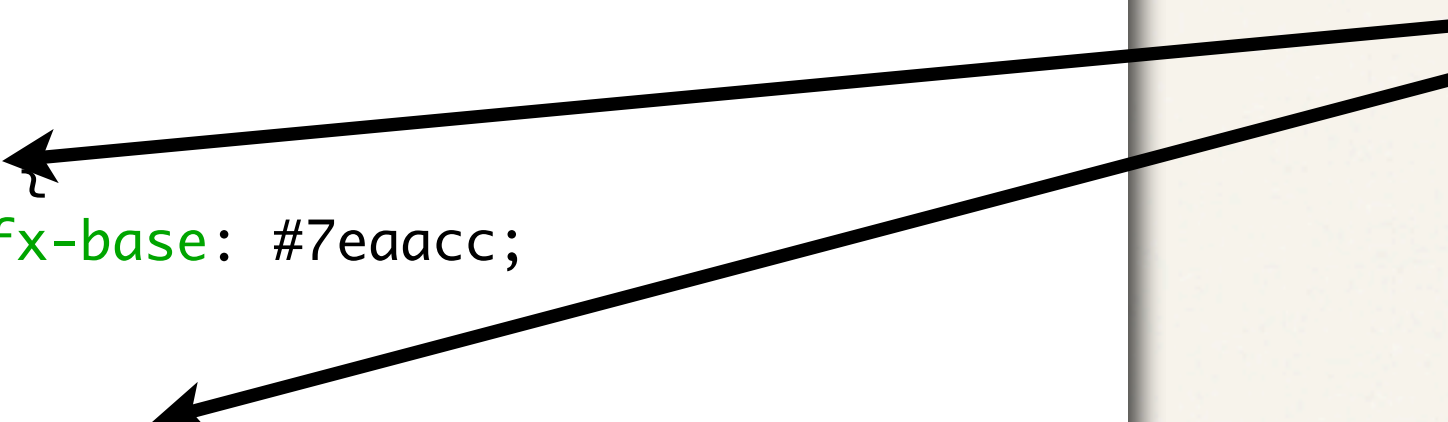
e(fx)clipse - CSS

```
.root {  
  -fx-base: #7eaacc;  
}  
  
#myelement .text:focus {  
  -fx-background-color:  
    linear-gradient(  
      to bottom,  
      derive(-fx-base,-30%),  
      derive(-fx-base,-60%));  
}
```

e(fx)clipse - CSS

CSS 2 selector

```
.root {  
  -fx-base: #7eaacc;  
}  
  
#myelement .text:focus {  
  -fx-background-color:  
    linear-gradient(  
      to bottom,  
      derive(-fx-base,-30%),  
      derive(-fx-base,-60%));  
}
```



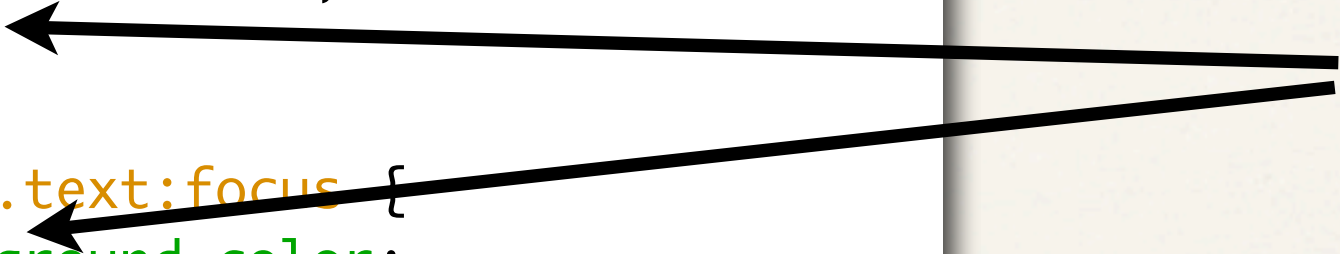
e(fx)clipse - CSS

```
.root {  
  -fx-base: #7eaacc;  
}  
  
#myelement .text:focus {  
  -fx-background-color:  
    linear-gradient(  
      to bottom,  
      derive(-fx-base,-30%),  
      derive(-fx-base,-60%));  
}
```

e(fx)clipse - CSS

```
.root {  
  -fx-base: #7eaacc;  
}  
  
#myelement .text:focus {  
  -fx-background-color:  
    linear-gradient(  
      to bottom,  
      derive(-fx-base,-30%),  
      derive(-fx-base,-60%));  
}
```

FX-Specific
attributes



e(fx)clipse - CSS

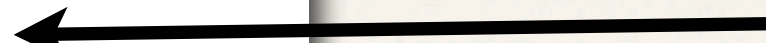
```
.root {  
  -fx-base: #7eaacc;  
}  
  
#myelement .text:focus {  
  -fx-background-color:  
    linear-gradient(  
      to bottom,  
      derive(-fx-base,-30%),  
      derive(-fx-base,-60%));  
}
```

e(fx)clipse - CSS

```
.root {  
  -fx-base: #7eaacc;  
}
```

```
#myelement .text:focus {  
  -fx-background-color:  
    linear-gradient(  
      to bottom,  
      derive(-fx-base,-30%),  
      derive(-fx-base,-60%));  
}
```

referencing
attribute
values



e(fx)clipse – FXML

```
<BorderPane xmlns:fx="http://javafx.com/fxml"
  fx:controller="MyController">
  <center>
    <Button
      text="Hello World"
      onAction="#clicked" />
  </center>
</BorderPane>
```

```
final MyController c = new MyController();
BorderPane p = new BorderPane();
Button b = new Button("Hello World");
b.setOnAction(new EventHandler<ActionEvent>() {
  @Override
  public void handle(ActionEvent event) {
    c.clicked(event);
  }
});
```

e(fx)clipse – FXML

```
<BorderPane xmlns:fx="http://javafx.com/fxml"
  fx:controller="MyController">
  <center>
    <Button
      text="Hello World"
      onAction="#clicked" />
    </center>
  </BorderPane>
```

```
final MyController c = new MyController();
BorderPane p = new BorderPane();
Button b = new Button("Hello World");
b.setOnAction(new EventHandler<ActionEvent>() {
  @Override
  public void handle(ActionEvent event) {
    c.clicked(event);
  }
});
```

- ❖ Declarative Definition enables Tooling

e(fx)clipse – FXML

```
<BorderPane xmlns:fx="http://javafx.com/fxml"
  fx:controller="MyController">
  <center>
    <Button
      text="Hello World"
      onAction="#clicked" />
  </center>
</BorderPane>
```

```
final MyController c = new MyController();
BorderPane p = new BorderPane();
Button b = new Button("Hello World");
b.setOnAction(new EventHandler<ActionEvent>() {
  @Override
  public void handle(ActionEvent event) {
    c.clicked(event);
  }
});
```

- ❖ Declarative Definition enables Tooling
- ❖ Clean separation between UI-Structure and Logic

e(fx)clipse – FXML

```
<BorderPane xmlns:fx="http://javafx.com/fxml"
  fx:controller="MyController">
  <center>
    <Button
      fx:id="mybutton"
      text="Hello World"
      onAction="#clicked" />
    </center>
  </BorderPane>
```

e(fx)clipse – FXML

```
<BorderPane xmlns:fx="http://javafx.com/fxml"
  fx:controller="MyController">
  <center>
    <Button
      fx:id="mybutton"
      text="Hello World"
      onAction="#clicked" />
    </center>
  </BorderPane>
```

```
public class MyController {
  @FXML
  Button mybutton;

  @FXML
  public void clicked() {
    // logic
  }
}
```

e(fx)clipse - FXML

```
<BorderPane xmlns:fx="http://javafx.com/fxml"
  fx:controller="MyController">
  <center>
    <Button
      fx:id="mybutton"
      text="Hello World"
      onAction="#clicked" />
  </center>
</BorderPane>
```

```
public class MyController {
  @FXML
  Button mybutton;

  @FXML
  public void clicked() {
    // logic
  }
}
```

e(fx)clipse - FXML

```
<BorderPane xmlns:fx="http://javafx.com/fxml"
  fx:controller="MyController">
  <center>
    <Button
      fx:id="mybutton"
      text="Hello World"
      onAction="#clicked" />
  </center>
</BorderPane>
```

```
URL location = getClass().getResource("form.fxml");
FXMLLoader fxmLoader = new FXMLLoader(location);

BorderPane root = (BorderPane)fxmLoader.load();
```

```
public class MyController {
  @FXML
  Button mybutton;

  @FXML
  public void clicked() {
    // logic
  }
}
```

e(fx)clipse – Deployment

e(fx)clipse – Deployment

- ✧ Remember the classpath problem!

e(fx)clipse – Deployment

- ✧ Remember the classpath problem!
- ✧ JavaFX ships with packaging tool

e(fx)clipse – Deployment

- ✧ Remember the classpath problem!
- ✧ JavaFX ships with packaging tool
 - ✧ Provides ant-tasks

e(fx)clipse – Deployment

- ✧ Remember the classpath problem!
- ✧ JavaFX ships with packaging tool
 - ✧ Provides ant-tasks
 - ✧ Produces msi, dmg, deb, rpm packages

e(fx)clipse – Deployment

- ✧ Remember the classpath problem!
- ✧ JavaFX ships with packaging tool
 - ✧ Provides ant-tasks
 - ✧ Produces msi, dmg, deb, rpm packages
 - ✧ Allows to package JRE => AppStores!!!!

e(fx)clipse – Deployment

Ensemble

Ensemble provides a gallery of over 100 sample applications that use a wide range of JavaFX features, such as graphics, UI controls, animation, charts, media and WebView. Source code for each sample and links to API documentation are also provided. Ensemble will provide developers with an interactive reference tool for what they can build with JavaFX.

[Gratis](#)

JavaFX Ensemble

HIGHLIGHTS **NEW**

Search samples and docs here!

Adv Candle Stick Chart

Custom Candle Stick Chart

Description
A custom candlestick chart.

API Documentation
`javafx.scene.chart.Axis`
`javafx.scene.chart.Chart`
`javafx.scene.chart.NumberAxis`
`javafx.scene.chart.XYChart`

Informationen

Kategorie: Entwickler-Tools
Erschienen: 29.09.2012
Version: 1.0
Preis: Kostenlos
Größe: 61.3 MB
Sprache: Englisch
Entwickler: Scott Kovatch
© Copyright (c) 2008, 2012 Oracle and/or its affiliates.
Kennzeichnung: 4+
Voraussetzungen: OS X 10.7.3 oder neuer, 64-Bit-Prozessor

(c) Tom Schindl – BestSolution Systemhaus GmbH

e(fx)clipse - Tooling

Basic Tooling Demo

- ✧ Create application
- ✧ Create and edit FXML
- ✧ Connect to controller
- ✧ Edit CSS-Stylesheet
- ✧ Export application

e(fx)clipse – CSS Impl

e(fx)clipse – CSS Impl

- ✧ Based on Xtext

e(fx)clipse – CSS Impl

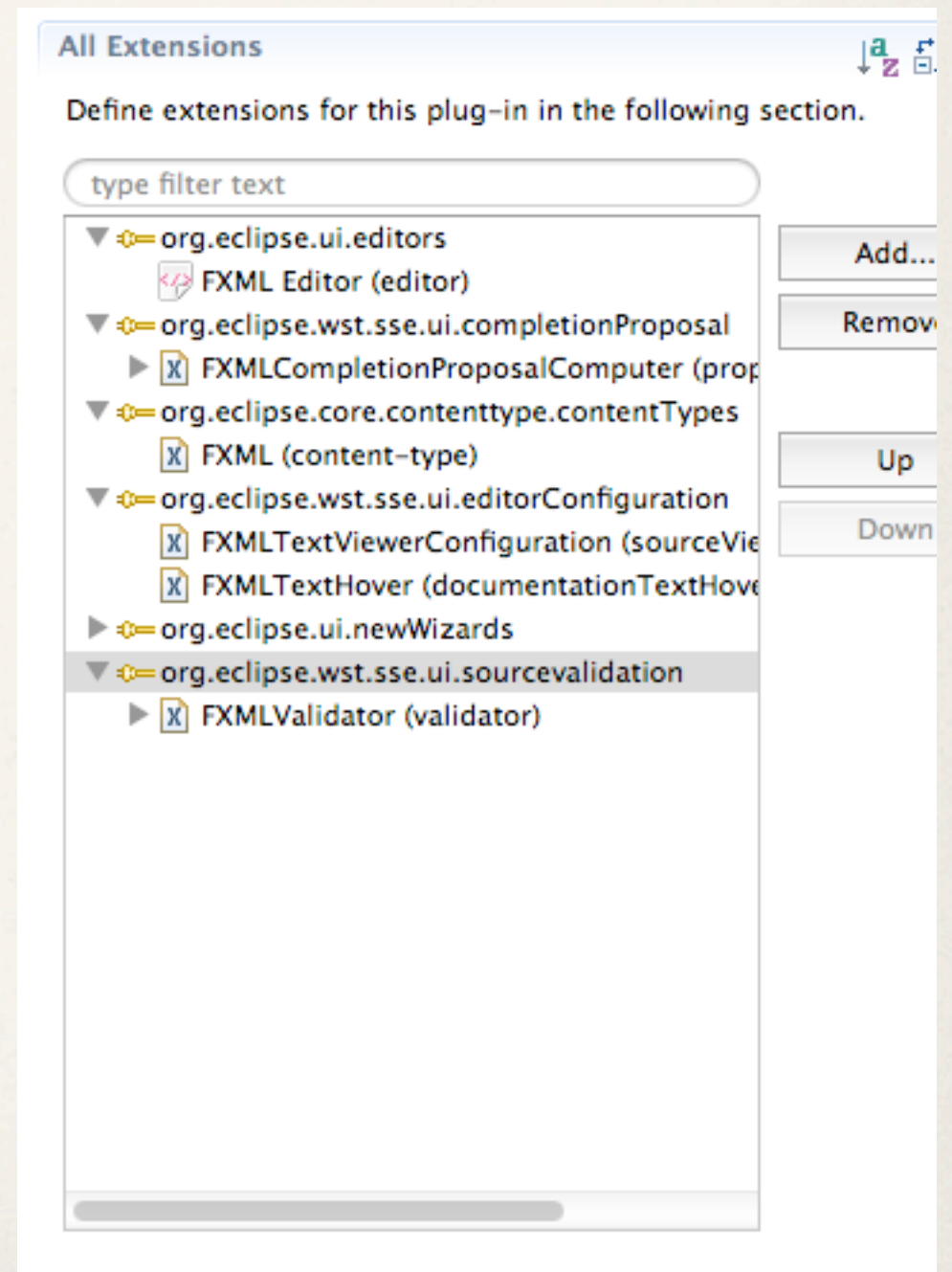
- ✧ Based on Xtext
- ✧ Editor can be customized using 2nd DSL (cssext)

e(fx)clipse – CSS Impl

- ✧ Based on Xtext
- ✧ Editor can be customized using 2nd DSL (cssext)

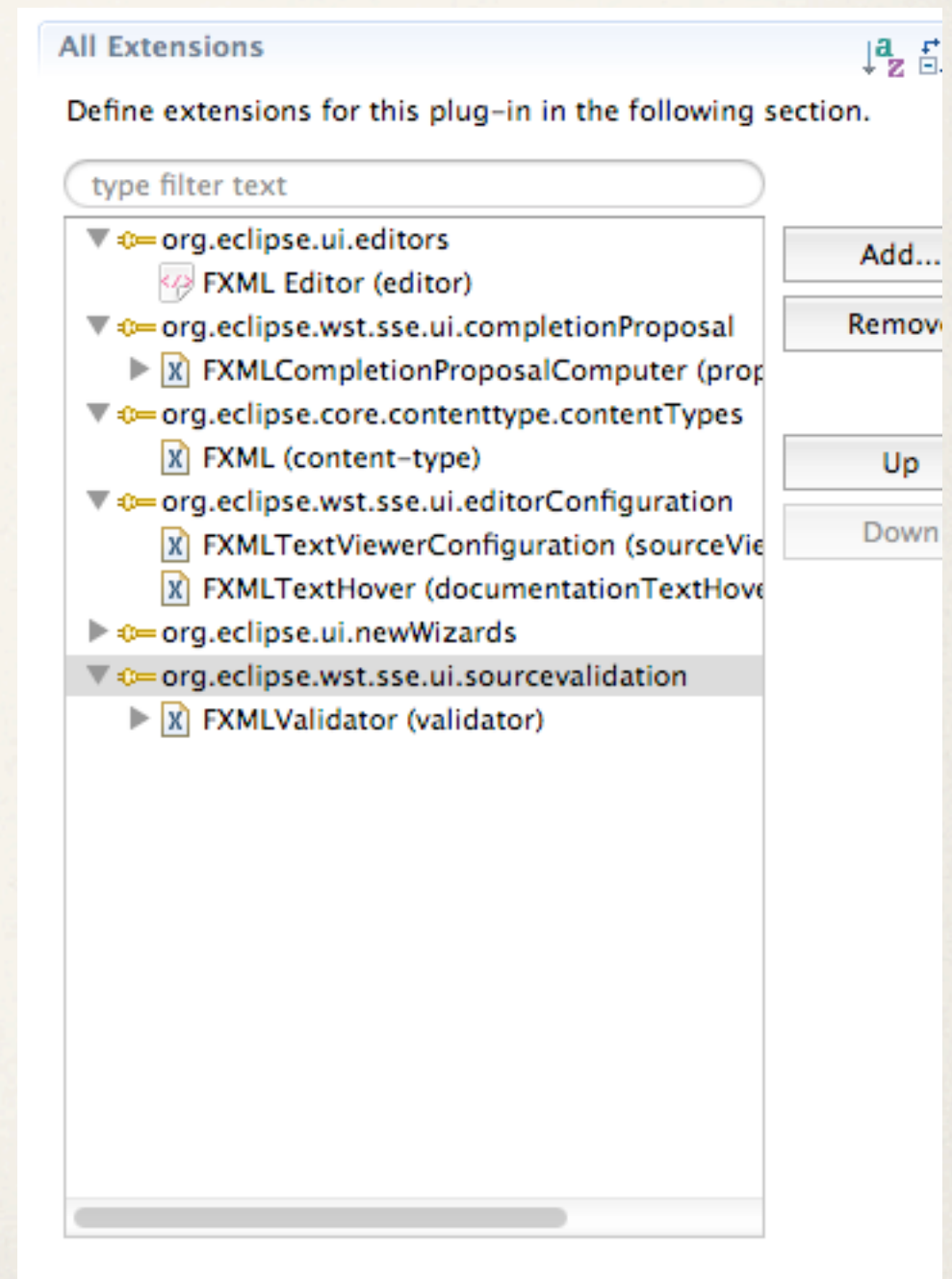
```
Node {  
    -fx-blend-mode <blend-mode> default: null;  
    -fx-cursor <cursor> default: null;  
    -fx-focus-traversable <boolean> default: null;  
    -fx-opacity double(.0->1.0) default: 1.0;  
    -fx-rotate double default: 0;  
    :disabled  
    :focused  
    :hover  
    :pressed  
    :show-mnemonic  
}
```

e(fx)clipse – FXML Impl



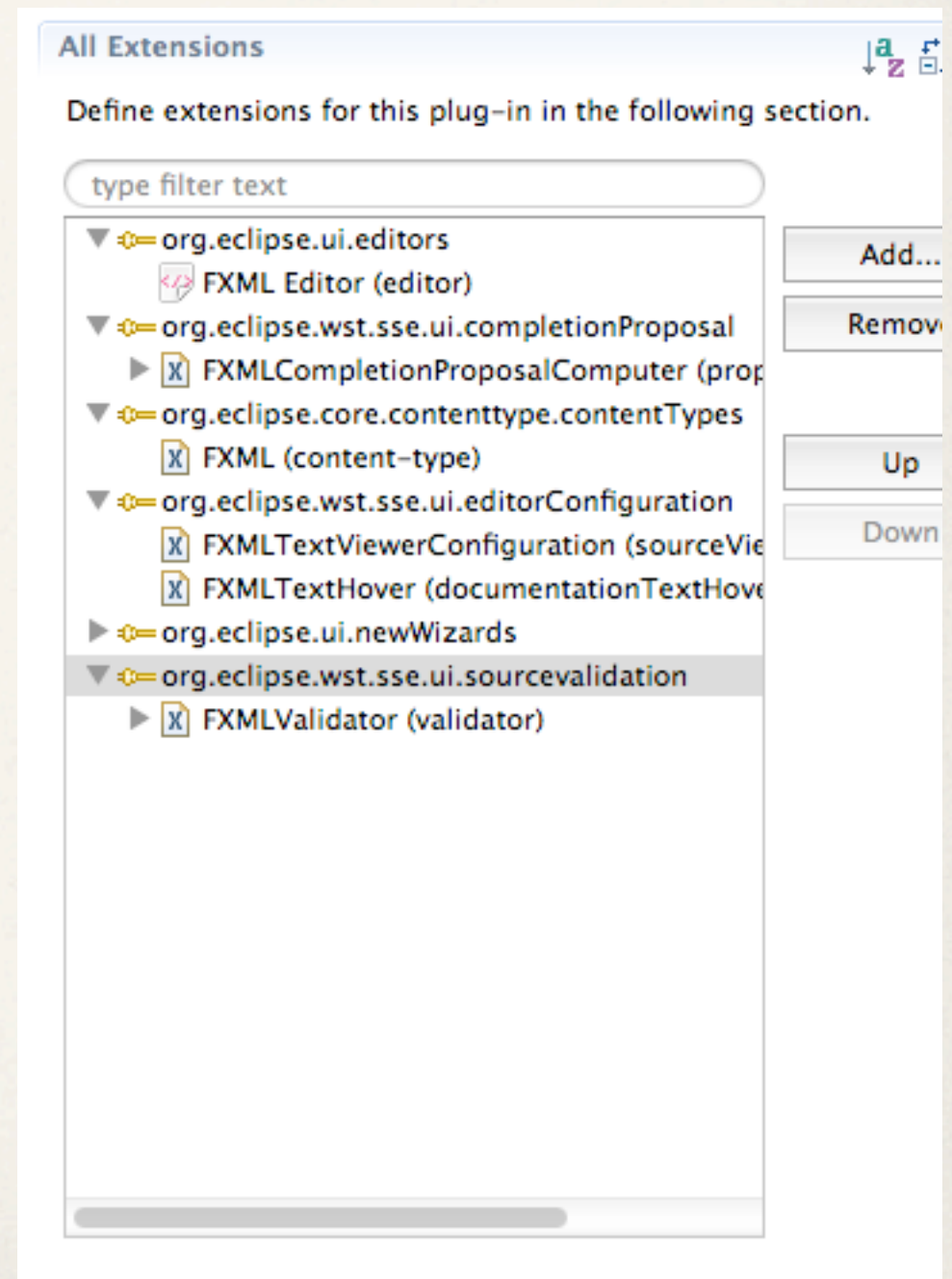
e(fx)clipse – FXML Impl

- ❖ Reuse WST XML-Editor Infrastructure



e(fx)clipse – FXML Impl

- ❖ Reuse WST XML-Editor Infrastructure
- ❖ Contributing through extension points



e(fx)clipse – FXML Impl

```
@SuppressWarnings("restriction")
public class FXMLCompletionProposalComputer extends AbstractXMLCompletionProposalComputer
implements ICompletionProposalComputer {

    protected void addTagNameProposals(ContentAssistRequest contentAssistRequest,
    int childPosition, CompletionProposalInvocationContext context) {
        Node parent = contentAssistRequest.getParent();

        // .....
        IType type;
        // .....

        IFXClass fxClass = FXPlugin.getClassmodel().findClass(type.getJavaProject(), type);

        if (fxClass != null) {
            for (IFXProperty property : fxClass.getAllProperties().values()) {
                createPropertyElementNameProposal(contentAssistRequest, context, property);
            }
        }
    }
}
```

About e(fx)clipse

Advanced-Tooling



About e(fx)clipse

Advanced-Tooling



- ✧ FXGraph
 - ✧ JSON like Language that „compiles“ to FXML

About e(fx)clipse

Advanced-Tooling



- ✧ FXGraph
 - ✧ JSON like Language that „compiles“ to FXML
- ✧ OSGi-Tooling
 - ✧ PDE-Classpath container
 - ✧ Wizards

About e(fx)clipse

Advanced-Tooling



- ✧ FXGraph
 - ✧ JSON like Language that „compiles“ to FXML
- ✧ OSGi-Tooling
 - ✧ PDE-Classpath container
 - ✧ Wizards
- ✧ SVG
 - ✧ Converter for SVG to FXML

e(fx)clipse – FX+OSGi

`my.app.bundle`

```
import java.lang  
import javafx.scene
```

e(fx)clipse – FX+OSGi

`my.app.bundle`

```
import java.lang  
import javafx.scene
```

JRE 1.7

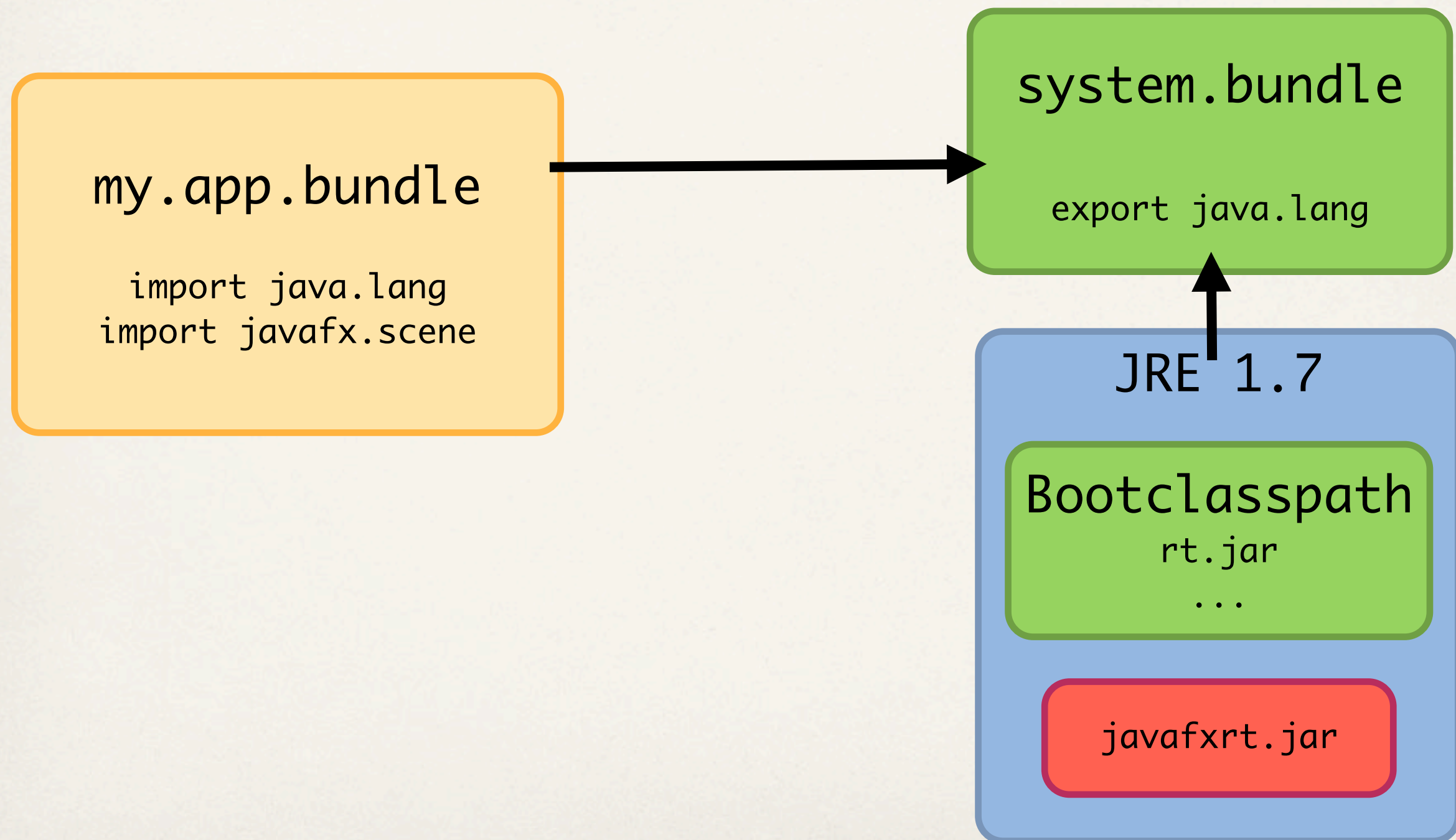
Bootclasspath

`rt.jar`

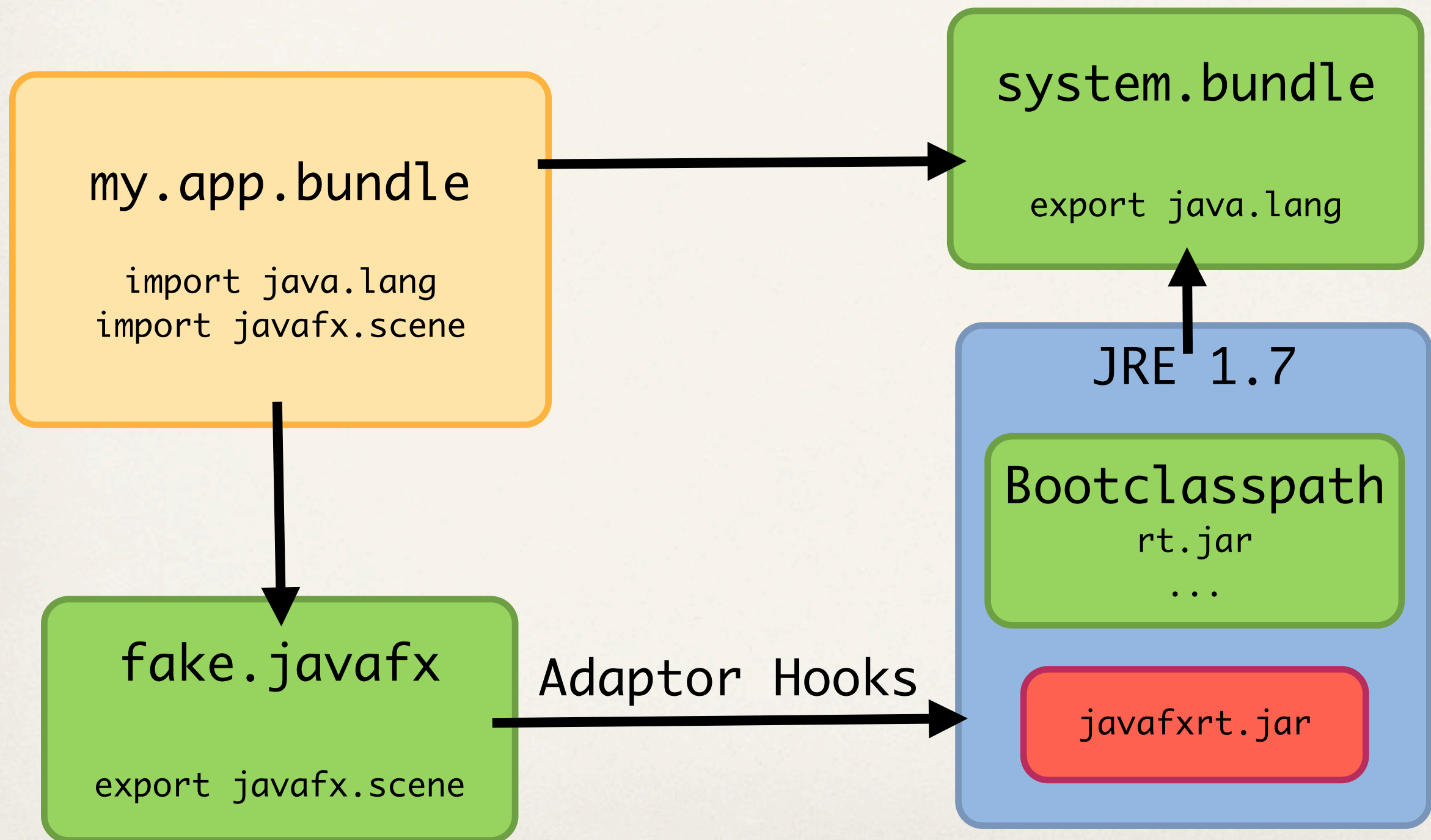
...

`javafxrt.jar`

e(fx)clipse - FX+OSGi

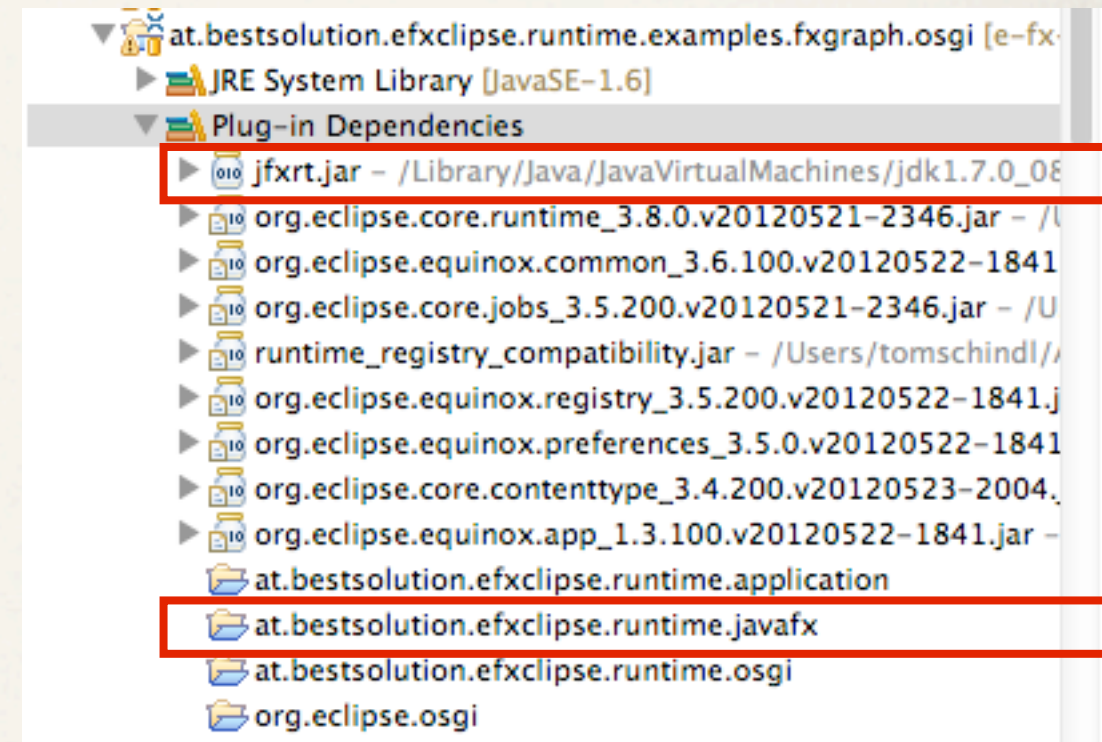


e(fx)clipse - FX+OSGi



e(fx)clipse – FXML Impl

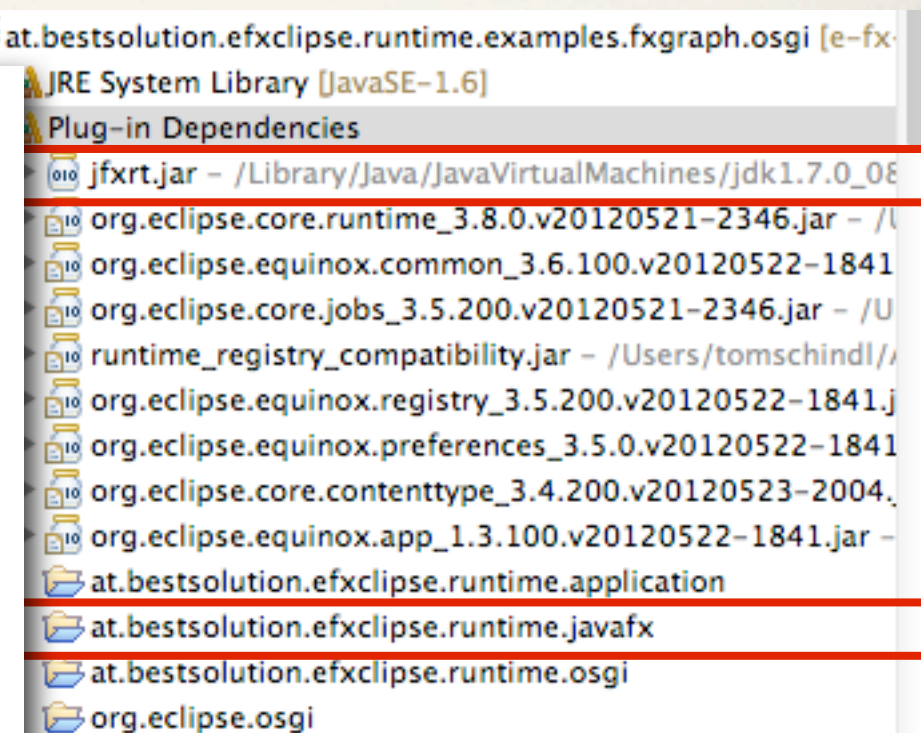
- ❖ Patching PDE using AdapterHooks (StorageHook) on the fly



e(fx)clipse – FXML Impl

- ❖ Patching PDE using AdapterHooks (StorageHook) on the fly

```
public class PFStorageHook implements StorageHook {  
    public static final String BUNDLE_TYPE_PATCH = "fxpatch.fragment";  
    public static final String BUNDLE_TYPE_HEADER = "Equinox-BundleType";  
  
    public void initialize(Dictionary manifest) {  
        // make sure this is a fragment manifest  
        if (manifest.get(Constants.FRAGMENT_HOST) == null)  
            return; // not a fragment;  
  
        String type = (String) manifest.get(BUNDLE_TYPE_HEADER);  
        patchFragment = BUNDLE_TYPE_PATCH.equals(type);  
    }  
}
```



e(fx)clipse – FXML Impl

- ❖ Patching PDE using AdapterHooks (StorageHook) on the fly

```
public class PFStorageHook {
    public static final String BUNDLE_NAME = "org.eclipse.pde.core";
    public static final String BUNDLE_SYMBOLIC_NAME = "org.eclipse.pde.core";

    public void initialize(Dictionary manifest) {
        // make sure this is a fragment
        if (manifest.get(Constants.FRAGMENT_SYMBOLIC_NAME) != null)
            return; // not a fragment

        String type = (String) manifest.get(Constants.BUNDLE_TYPE);
        patchFragment = BUNDLE_NAME;
    }
}
```

```
Manifest-Version: 1.0
Bundle-ManifestVersion: 2
Bundle-Name: %Bundle-Name
Bundle-SymbolicName: at.bestsolution.efxclipse.tooling.pde.fragment;singleton:=true
Bundle-Version: 0.1.1.qualifier
Fragment-Host: org.eclipse.pde.core;bundle-version="[3.7.0,3.9.0)"
Bundle-RequiredExecutionEnvironment: JavaSE-1.6
Equinox-BundleType: fxpatch.fragment
Require-Bundle: at.bestsolution.efxclipse.tooling.jdt.core;bundle-version="0.1.1",
at.bestsolution.efxclipse.tooling.pde.adaptor;bundle-version="0.1.1"
Bundle-Vendor: %Bundle-Vendor
```

e(fx)clipse – FXGraph

```
package sample.fxgraph

import javafx.scene.layout.BorderPane
import javafx.scene.control.Button

component MySample controlledby MyController {
  BorderPane {
    center : Button {
      text : "Hello World",
      onAction : controllermethod helloWorld
    }
  }
}
```

e(fx)clipse – FXGraph

- ✧ JSON like syntax

```
package sample.fxgraph

import javafx.scene.layout.BorderPane
import javafx.scene.control.Button

component MySample controlledby MyController {
    BorderPane {
        center : Button {
            text : "Hello World",
            onAction : controllermethod helloWorld
        }
    }
}
```

e(fx)clipse – FXGraph

```
package sample.fxgraph

import javafx.scene.layout.BorderPane
import javafx.scene.control.Button

component MySample controlledby MyController {
    BorderPane {
        center : Button {
            text : "Hello World",
            onAction : controllermethod helloWorld
        }
    }
}
```

- ✧ JSON like syntax
- ✧ „Compiles“ to FXML

e(fx)clipse – FXGraph

```
package sample.fxgraph

import javafx.scene.layout.BorderPane
import javafx.scene.control.Button

component MySample controlledby MyController {
  BorderPane {
    center : Button {
      text : "Hello World",
      onAction : controllermethod helloWorld
    }
  }
}
```

- ✧ JSON like syntax
- ✧ „Compiles“ to FXML
- ✧ more expressive as FXML

e(fx)clipse - Tooling

Advanced Tooling Demo

- ❖ Create an FXGraph-File
- ❖ Connect to a controller
- ❖ Convert a SVG-File

e(fx)clipse - Tooling

Advanced Tooling Demo

e(fx)clipse - Tooling

Advanced Tooling Demo

- ✧ Create OSGi-e4 Application

e(fx)clipse - Tooling

Advanced Tooling Demo

- ✧ Create OSGi-e4 Application
- ✧ Create a Jemmy-test suite

e(fx)clipse – Future

e(fx)clipse – Future

✦ CSS

e(fx)clipse – Future

- ✧ CSS

- ✧ Extend usage of cssexp – validation and complex auto complete

e(fx)clipse – Future

- ✧ CSS

- ✧ Extend usage of cssexp - validation and complex auto complete

- ✧ FXML

e(fx)clipse – Future

- ✧ CSS

- ✧ Extend usage of cssexp – validation and complex auto complete

- ✧ FXML

- ✧ Tighter integration with JDT (refactoring)

e(fx)clipse – Future

- ✧ CSS

- ✧ Extend usage of cssexp - validation and complex auto complete

- ✧ FXML

- ✧ Tighter integration with JDT (refactoring)

- ✧ FXGraph

e(fx)clipse – Future

- ✧ CSS

- ✧ Extend usage of cssexp - validation and complex auto complete

- ✧ FXML

- ✧ Tighter integration with JDT (refactoring)

- ✧ FXGraph

- ✧ Full FXGraph => FXML => FXGraph cycle

e(fx)clipse - Future

- ✧ CSS

- ✧ Extend usage of cssexp - validation and complex auto complete

- ✧ FXML

- ✧ Tighter integration with JDT (refactoring)

- ✧ FXGraph

- ✧ Full FXGraph => FXML => FXGraph cycle

- ✧ OSGi

e(fx)clipse – Future

- ✧ CSS

- ✧ Extend usage of cssexp - validation and complex auto complete

- ✧ FXML

- ✧ Tighter integration with JDT (refactoring)

- ✧ FXGraph

- ✧ Full FXGraph => FXML => FXGraph cycle

- ✧ OSGi

- ✧ tycho enabled project skeleton wizard

e(fx)clipse – Resources

e(fx)clipse – Resources

- ✧ Homepage: <http://www.efxclipse.org>

e(fx)clipse – Resources

- ✧ Homepage: <http://www.efxclipse.org>
- ✧ Source-Code:

e(fx)clipse – Resources

- ✧ Homepage: <http://www.efxclipse.org>
- ✧ Source-Code:
 - ✧ <https://github.com/tomsontom/e-fx-clipse>

e(fx)clipse – Resources

- ✧ Homepage: <http://www.efxclipse.org>
- ✧ Source-Code:
 - ✧ <https://github.com/tomsontom/e-fx-clipse>
 - ✧ <https://github.com/tomsontom/fx-ide>

e(fx)clipse – Resources

- ❖ Homepage: <http://www.efxclipse.org>
- ❖ Source-Code:
 - ❖ <https://github.com/tomsontom/e-fx-clipse>
 - ❖ <https://github.com/tomsontom/fx-ide>
 - ❖ <https://github.com/tomsontom/fx43>

e(fx)clipse – Resources

- ❖ Homepage: <http://www.efxclipse.org>
- ❖ Source-Code:
 - ❖ <https://github.com/tomsontom/e-fx-clipse>
 - ❖ <https://github.com/tomsontom/fx-ide>
 - ❖ <https://github.com/tomsontom/fx43>
- ❖ Twitter: @tomsontom

e(fx)clipse – Resources

- ✧ Homepage: <http://www.efxclipse.org>
- ✧ Source-Code:
 - ✧ <https://github.com/tomsontom/e-fx-clipse>
 - ✧ <https://github.com/tomsontom/fx-ide>
 - ✧ <https://github.com/tomsontom/fx43>
- ✧ Twitter: @tomsontom
- ✧ Commercial Support: <http://www.bestsolution.at>