

Eclipse 4 and me

Tom Schindl - BestSolution Systemhaus GmbH

Eclipse Democamp 2011

(c) Tom Schindl – BestSolution Systemhaus GmbH – Licensed under Creative Commons Attribution–Noncommercial–No Derivative Works 3.0 Germany License.

About Me

- ❖ CEO BestSolution Systemhaus GmbH
- ❖ Eclipse Committer
 - ❖ e4
 - ❖ Platform UI
 - ❖ EMF
- ❖ Projectlead: UFaceKit, Nebula
- ❖ Member of the Architectural Council



Agenda

Agenda

- ❖ The Compability layer for 3.x code

Agenda

- ❖ The Compability layer for 3.x code
- ❖ Use the Eclipse DI framework even in 3.x

Agenda

- ❖ The Compability layer for 3.x code
- ❖ Use the Eclipse DI framework even in 3.x
- ❖ Eclipse Application Platform without SWT

API Compability

- ❖ What are the actions you need to take to make your plugin or RCP application work on Eclipse 4.x?

API Compability

- ❖ What are the actions you need to take to make your plugin or RCP application work on Eclipse 4.x?

NONE

Compat case study Eclipse SDK

Eclipse 3.x

Eclipse 4.x

(c) Tom Schindl – BestSolution Systemhaus GmbH – Licensed under Creative Commons Attribution–Noncommercial–No Derivative Works 3.0 Germany License.

Compat case study Eclipse SDK

PDE

JDT

Eclipse 3.x

Eclipse 4.x

Compat case study Eclipse SDK

PDE

JDT

Platform 3.x

Eclipse 3.x

Eclipse 4.x

Compat case study Eclipse SDK

PDE

JDT

Platform 3.x

Equinox

Eclipse 3.x

Eclipse 4.x

Compat case study Eclipse SDK

PDE

JDT

Platform 3.x

Equinox

Eclipse 3.x

PDE

JDT

Platform 4.x

Equinox

Eclipse 4.x

Compat case study Eclipse SDK

PDE

JDT

Platform 3.x

Equinox

Eclipse 3.x

PDE

JDT

Platform 4.x

Eclipse 4.x Application
Platform (RCP 2.0)

Equinox

Eclipse 4.x

Eclipse SDK 3.x vs 4.x

Platform 3.x

jface

core.resources

...

ui.workbench 3.x

Platform 4.x

jface

core.resources

...

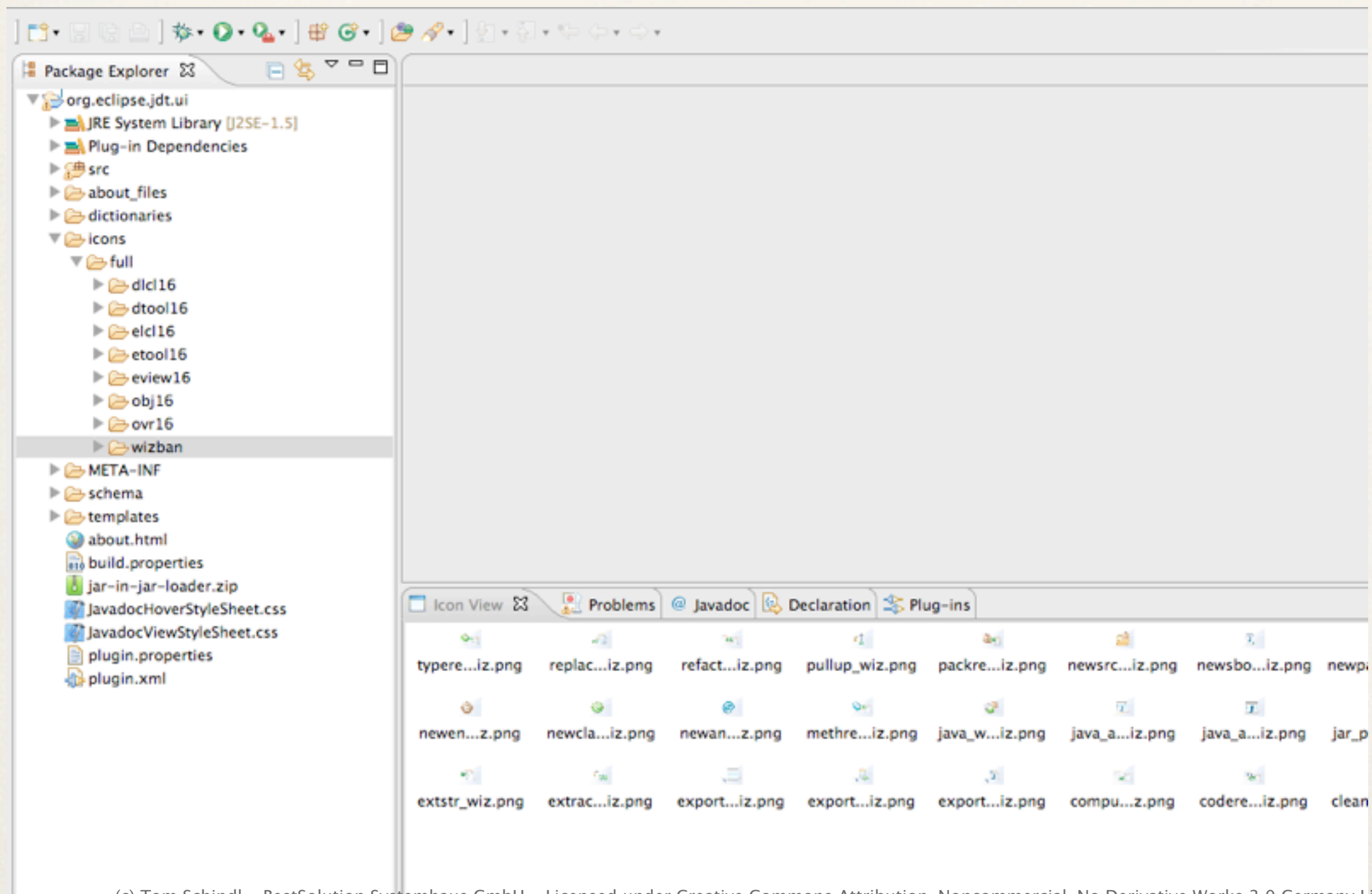
ui.workbench 4.x

Eclipse 4
Application Platform

How to make use of e4 DI in 3.x

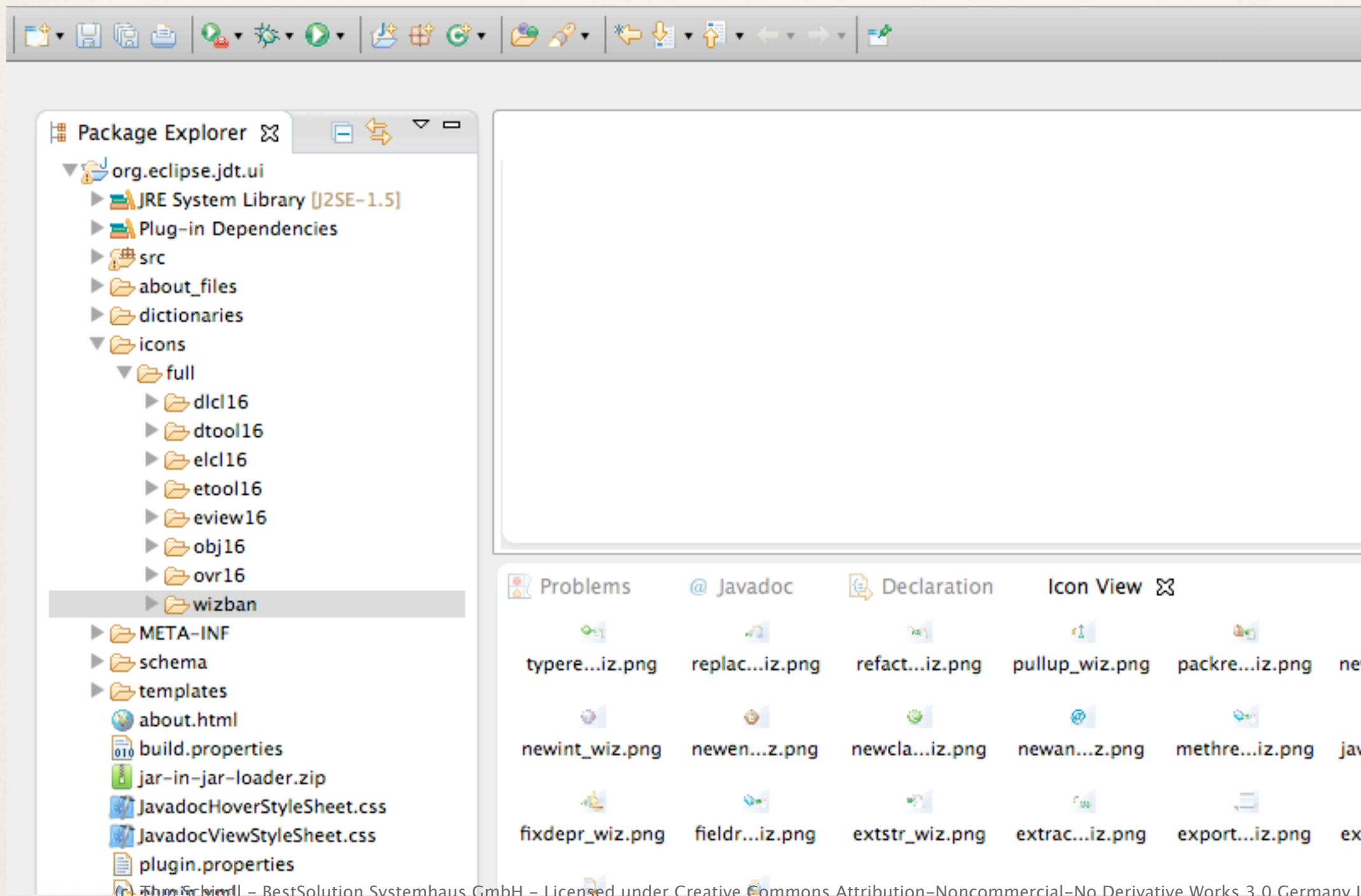
- ❖ I'd like to use the new programming model but want to deliver on Eclipse 3.x and Eclipse 4.x. Can I do that?

Example - IconView in 3.x



(c) Tom Schindl – BestSolution Systemhaus GmbH – Licensed under Creative Commons Attribution-Noncommercial-No Derivative Works 3.0 Germany License.

Example - IconView in 4.x



(c) Tom Schindl - BestSolution Systemhaus GmbH - Licensed under Creative Commons Attribution-Noncommercial-No Derivative Works 3.0 Germany License.

Analyze 3.x source code

```
public class IconViewPart extends ViewPart {  
  
    public void createPartControl(Composite parent) {  
        parent.setLayout(new FillLayout());  
        this.viewer = new GalleryTreeViewer(parent, SWT.MULTI | SWT.V_SCROLL);  
  
        this.workspace = ResourcesPlugin.getWorkspace();  
        this.workspace.addResourceChangeListener(resourceListener);  
  
        this.selectionListener = new ISelectionListener() {  
            public void selectionChanged(IWorkbenchPart part, ISelection selection) {  
                setFolder(selection);  
            }  
        };  
        getSite().getWorkbenchWindow().getSelectionService().addSelectionListener(selectionListener);  
    }  
  
    public void dispose() {  
        getSite().getWorkbenchWindow().getSelectionService().removeSelectionListener(selectionListener);  
        this.workspace.removeResourceChangeListener(resourceListener);  
        super.dispose();  
    }  
  
    public void setFocus() {  
        viewer.getControl().setFocus();  
    }  
}
```

(c) Tom Schindl – BestSolution Systemhaus GmbH – Licensed under Creative Commons Attribution-Noncommercial-No Derivative Works 3.0 Germany License.

Analyze 3.x source code

```
public class IconViewPart extends ViewPart {  
  
    public void createPartControl(Composite parent) {  
        parent.setLayout(new FillLayout());  
        this.viewer = new GalleryTreeViewer(parent, SWT.MULTI | SWT.V_SCROLL);  
  
        this.workspace = ResourcesPlugin.getWorkspace();  
        this.workspace.addResourceChangeListener(resourceListener);  
  
        this.selectionListener = new ISelectionListener() {  
            public void selectionChanged(IWorkbenchPart part, ISelection selection) {  
                setFolder(selection);  
            }  
        };  
        getSite().getWorkbenchWindow().getSelectionService().addSelectionListener(selectionListener);  
    }  
  
    public void dispose() {  
        getSite().getWorkbenchWindow().getSelectionService().removeSelectionListener(selectionListener);  
        this.workspace.removeResourceChangeListener(resourceListener);  
        super.dispose();  
    }  
  
    public void setFocus() {  
        viewer.getControl().setFocus();  
    }  
}
```

Split your 3.x source code

```
public class IconView {
    private IWorkspace workspace;

    public IconView(IWorkspace workspace) {
        this.workspace = workspace;
        this.workspace.addResourceChangeListener(resourceListener);
    }

    public void createPartControl(Composite parent) {
        parent.setLayout(new FillLayout());
        this.viewer = new GalleryTreeViewer(parent, SWT.MULTI | SWT.V_SCROLL);
    }

    public void dispose() {
        this.workspace.removeResourceChangeListener(resourceListener);
    }

    public void setFocus() {
        viewer.getControl().setFocus();
    }

    public void setFolder(Object selection) {
    }
}
```

Split your 3.x source code

```
public class IconViewPart extends ViewPart {
    public IconView view;

    public void createPartControl(Composite parent) {
        this.view = new IconView(ResourcesPlugin.getWorkspace());

        this.selectionListener = new ISelectionListener() {
            public void selectionChanged(IWorkbenchPart part, ISelection selection) {
                view.setFolder(selection);
            }
        };
        getSite().getWorkbenchWindow().getSelectionService().addSelectionListener(selectionListener);
    }

    public void dispose() {
        getSite().getWorkbenchWindow().getSelectionService().removeSelectionListener(selectionListener);
        this.view.dispose();
        super.dispose();
    }

    public void setFocus() {
        view.setFocus();
    }
}
```

Split up your bundles

Split up your bundles

- ❖ Split up 3 bundles

Split up your bundles

- ✧ Split up 3 bundles
 - ✧ UI-Bundle: Depends only on SWT, JFace, if needed Resources

Split up your bundles

- ❖ Split up 3 bundles
 - ❖ UI-Bundle: Depends only on SWT, JFace, if needed Resources
 - ❖ Framework-Integration-Bundle: Integration into 3.x framework (depends on org.eclipse.ui)

Split up your bundles

- ❖ Split up 3 bundles
 - ❖ UI-Bundle: Depends only on SWT, JFace, if needed Resources
 - ❖ Framework-Integration-Bundle: Integration into 3.x framework (depends on org.eclipse.ui)
 - ❖ Broker-Bundle: Custom Service brokers to seal off from framework services

Summary

- ❖ Decoupled UI Code from Eclipse Framework
 - ❖ Easier to test
 - ❖ UI-Code can be used in multiple contexts (e.g. in a Dialog)

Summary

- ❖ Decoupled UI Code from Eclipse Framework
 - ❖ Easier to test
 - ❖ UI-Code can be used in multiple contexts (e.g. in a Dialog)
- ❖ ... but we still need to write a lot of glue code because we are not yet using dependency injection

Too much glue, what can we do?

```
public class IconViewPart extends ViewPart {  
    public IconView view;  
  
    public void createPartControl(Composite parent) {  
        this.view = new IconView(ResourcesPlugin.getWorkspace());  
  
        this.selectionListener = new ISelectionListener() {  
            public void selectionChanged(IWorkbenchPart part, ISelection selection) {  
                view.setFolder(selection);  
            }  
        };  
        getSite().getWorkbenchWindow().getSelectionService().addSelectionListener(selectionListener);  
    }  
  
    public void dispose() {  
        getSite().getWorkbenchWindow().getSelectionService().addSelectionListener(selectionListener);  
        this.view.dispose();  
        super.dispose();  
    }  
  
    public void setFocus() {  
        view.setFocus();  
    }  
}
```

Let's reduce the glue!

```
public class IconViewPart extends ViewPart {  
    public IconView view;  
  
    public void createPartControl(Composite parent) {  
        this.view = new IconView(ResourcesPlugin.getWorkspace());  
  
        this.selectionListener = new ISelectionListener() {  
            public void selectionChanged(IWorkbenchPart part, ISelection selection) {  
                view.setFolder(selection);  
            }  
        };  
        getSite().getWorkbenchWindow().getSelectionService().addSelectionListener(selectionListener);  
    }  
  
    public void dispose() {  
        getSite().getWorkbenchWindow().getSelectionService().addSelectionListener(selectionListener);  
        this.view.dispose();  
        super.dispose();  
    }  
  
    public void setFocus() {  
        view.setFocus();  
    }  
}
```

Let's reduce the glue!

```
public class IconViewPart extends DUIViewPart<IconView> {  
  
    public IconViewPart() {  
        super(IconView.class);  
    }  
}
```

Let's reduce the glue!

```
public class IconViewPart extends DUIViewPart<IconView> {  
  
    public IconViewPart() {  
        super(IconView.class);  
    }  
}
```

How the hell does that work?

We use some annotations

```
public class IconView {
    private IWorkspace workspace;

    public IconView(IWorkspace workspace) {
        this.workspace = workspace;
        this.workspace.addResourceChangeListener(resourceListener);
    }

    public void createPartControl(Composite parent) {
        parent.setLayout(new FillLayout());
        this.viewer = new GalleryTreeViewer(parent, SWT.MULTI | SWT.V_SCROLL);
    }

    public void dispose() {
        this.workspace.removeResourceChangeListener(resourceListener);
    }

    public void setFocus() {
        viewer.getControl().setFocus();
    }

    public void setFolder(@Named(IServiceConstants.ACTIVE_SELECTION) @Optional Object selection) {
    }
}
```

(c) Tom Schindl – BestSolution Systemhaus GmbH – Licensed under Creative Commons Attribution-Noncommercial-No Derivative Works 3.0 Germany License.

We use some annotations

```
public class IconView {
    private IWorkspace workspace;

    @Inject
    public IconView(IWorkspace workspace) {
        this.workspace = workspace;
        this.workspace.addResourceChangeListener(resourceListener);
    }

    @PostConstruct
    public void createPartControl(Composite parent) {
        parent.setLayout(new FillLayout());
        this.viewer = new GalleryTreeViewer(parent, SWT.MULTI | SWT.V_SCROLL);
    }

    @PreDestroy
    public void dispose() {
        this.workspace.removeResourceChangeListener(resourceListener);
    }

    @Focus
    public void setFocus() {
        viewer.getControl().setFocus();
    }

    @Inject
    public void setFolder(@Named(IServiceConstants.ACTIVE_SELECTION) @Optional Object selection) {
    }
}
```

(c) Tom Schindl – BestSolution Systemhaus GmbH – Licensed under Creative Commons Attribution-Noncommercial-No Derivative Works 3.0 Germany License.

e4 bridge DI container

- ✧ Provides fairly the same features default e4 DI-Container does
 - ✧ Injection of OSGi-Services
 - ✧ Support for @Focus, @Persist
 - ✧ Support for @EventTopic, @UIEventTopic and IEventBroker
 - ✧ Support for @Preference

e4 bridge DI container

- ✧ Provides fairly the same features default e4 DI-Container does
 - ✧ Injection of OSGi-Services
 - ✧ Support for @Focus, @Persist
 - ✧ Support for @EventTopic, @UIEventTopic and IEventBroker
 - ✧ Support for @Preference
- ✧ ... and it goes beyond that and adds new concepts for pure e4 and 3.x integration

e4 bridge DI extension

- ❖ Resource Management - Image-Management in particular

```
public class ModelEditor extends EditorPart {  
  
    public void createPartControl(Composite composite) {  
        Label l = new Label(composite, SWT.NONE);  
        l.setImage(  
            Activator.imageDescriptorFromPlugin(Activator.PLUGIN_ID, "/resource/myimage.png").createImage()  
        );  
    }  
}
```

e4 bridge DI extension

- ❖ Resource Management - Image-Management in particular

```
public class ModelEditor {  
  
    @Inject  
    private final IResourcePool resourcePool;  
  
    @PostConstruct  
    void init(Composite composite) {  
        Label l = new Label(composite, SWT.NONE);  
        l.setImage(resourcePool.getImageUnchecked(ResourceProvider.IMG_Obj16_cross));  
    }  
}
```

e4 bridge DI extension

❖ NLS / Externalized String support

```
public class Messages {  
    public static String MyLabel;  
  
    NLS.initializeMessages(Messages.class.getName(), Messages.class);  
}
```

```
public class ModelEditor {  
  
    public void createPartControl(Composite composite) {  
        Label l = new Label(composite, SWT.NONE);  
        l.setText(Messages.MyLabel);  
    }  
}
```

e4 bridge DI extension

❖ NLS / Externalized String support

```
public class Messages {  
    public String MyLabel;  
}
```

```
public class ModelEditor extends EditorPart {  
  
    @Inject  
    @Translation  
    private Messages messages;  
  
    @PostConstruct  
    void createPartControl(Composite composite) {  
        Label l = new Label(composite, SWT.NONE);  
        l.setText(messages.MyLabel);  
    }  
}
```

EAP 4.x without SWT?

EAP 4.x without SWT?

Framework

Eclipse 4 Application Platform

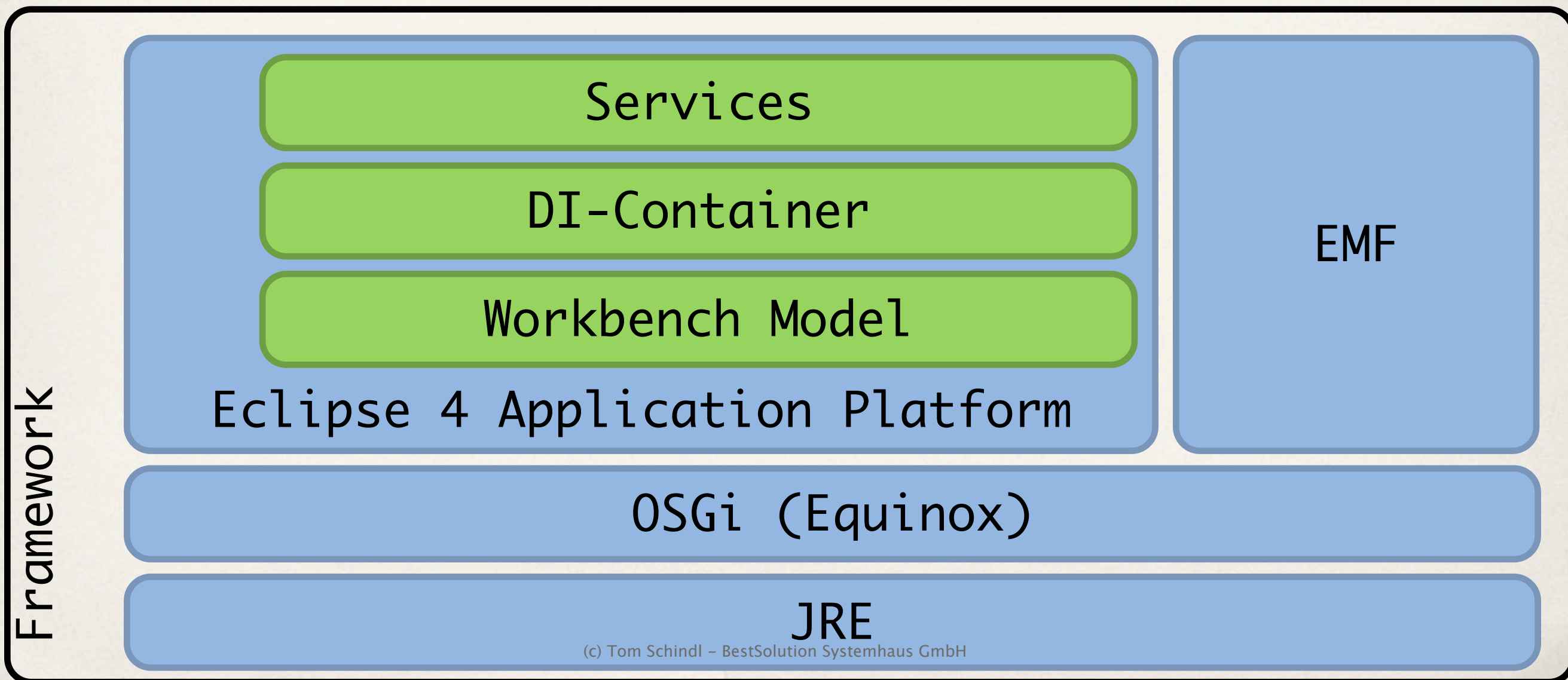
EMF

OSGi (Equinox)

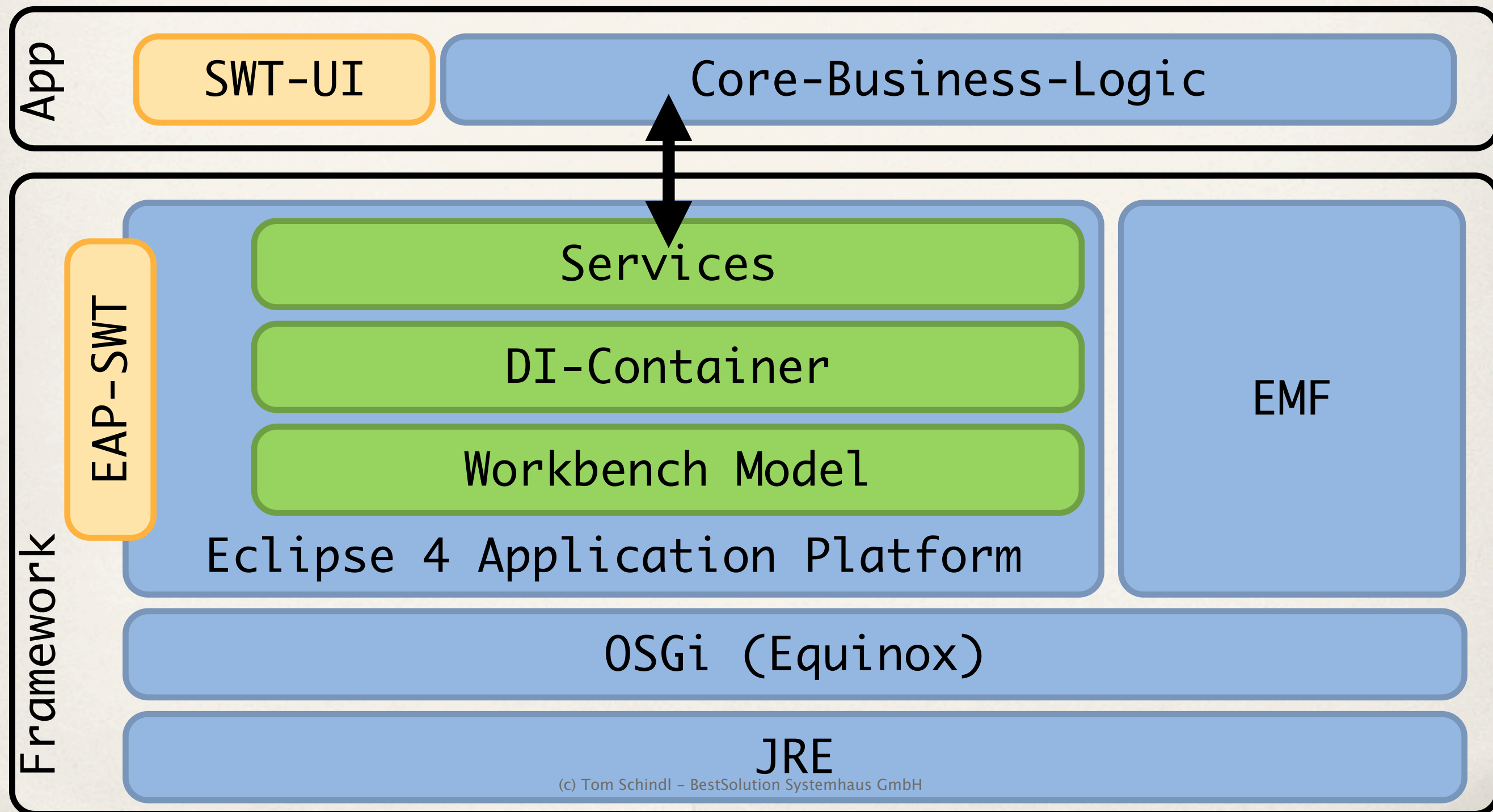
JRE

(c) Tom Schindl – BestSolution Systemhaus GmbH

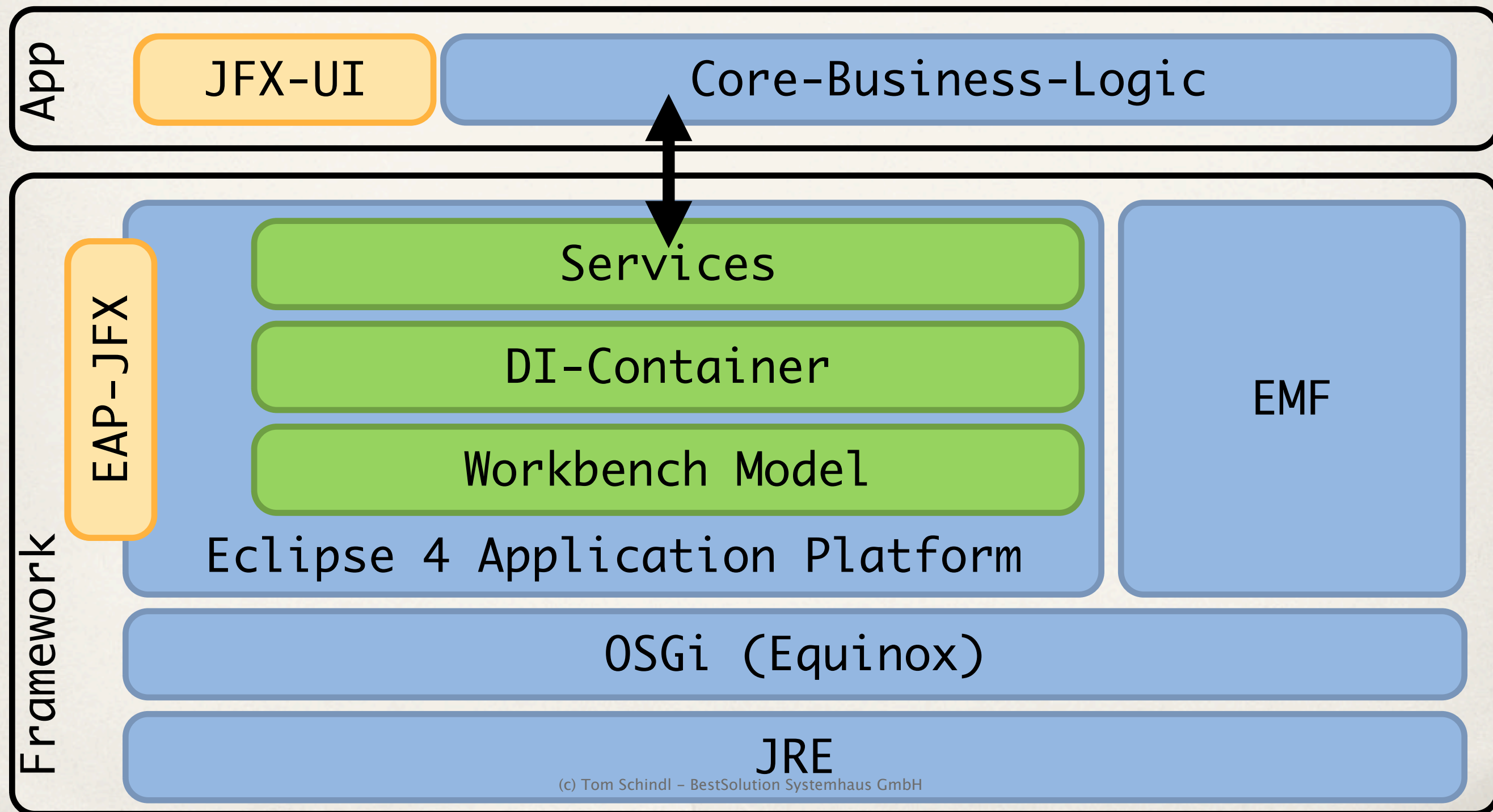
EAP 4.x without SWT?



EAP 4.x without SWT?



EAP 4.x without SWT?



e(fx)clipse, from BestSolution.at

JavaFX 2.0 Tooling and Runtime for Eclipse and OSGi

About a short intro

Tooling

e(fx)clipse provides JavaFX tooling for the Eclipse IDE. We are currently supporting Eclipse 3.7.1 and the currently developed Eclipse 4.2.

- JDT Support
- PDE Support
- UI DSL to author FXML
- ...

[View details](#)

Runtime

It's not all about tooling in the e(fx)clipse project but we are also providing a runtime platform and libraries you can use in your JavaFX 2.0 applications.

- Support for OSGi
- Support for the Eclipse 4 Application Platform
- ...

[View details](#)

Platforms

Because JavaFX 2.0 is currently only available on Windows (GA) and Mac OS X (Beta) the tooling is only supported on those 2 platforms. Once Oracle provides at least a beta builds for Linux support for this platform will be added immediately.

Tooling It's all about your favorite IDE

By integrating into Eclipse, JDT etc. by providing a specialized classpath

Additional Information

- ❖ Blog: <http://tomsondev.bestsolution.at>
- ❖ e4 Newsgroup: eclipse.e4
- ❖ e4 Mailingliste: e4-dev@eclipse.org
- ❖ e4 wiki: <http://wiki.eclipse.org/e4>
- ❖ sources: <https://github.com/tomsontom/iconview>
- ❖ twitter: @tomsontom
- ❖ JavaFX for e4: <http://efxclipse.org>