codecentric

SWT to JavaFX Migration_

JFS - 09.07.2015

Speaker_

Dirk Fauth

codecentric AG
Curiestraße 2
70563 Stuttgart

dirk.fauth@codecentric.de
www.codecentric.de
blog.codecentric.de
blog.vogella.com/author/fipro/
Twitter: fipro78

SWT to JavaFX Migration

- **Terminologie**
- **SWT vs. JavaFX**
  - Nachteile SWT
  - Vorteile JavaFX
- **Migration Basis**
- **Migration UI**
  - Controls
  - Layout
  - Interaktionen & Event Handling
  - CSS Styling

Terminologie

SWT

- **Standard Widget Toolkit**

JavaFX

- **Oracle Standard Java UI Toolkit**

SWTonJavaFX

- **SWT Implementierung mit JavaFX**

Eclipse RCP

- Rich Client Platform

e(fx)clipse

- Tooling (IDE)
- Runtime (JRE, OSGi, RCP, EMF)

SWT vs. JavaFX

Nachteile von SWT_

- OSS getrieben
- Plattformabhängigkeit
- Verwendung nativer Widgets
- CSS nicht vollständig
- Resource Management

Vorteile von JavaFX_

- **Bestandteil von Java**
- **Multimedia-Plattform**
- **Verwendung Graphics Engine**
- **Unterstützung von Transitions**
- **Vollständige CSS Unterstützung**
- **Graph anstelle von “Leinwand”**
- **Deklarative UI**

Migration Basis

Plain SWT -> Plain JavaFX

```
public class Main {

    public static void main(String[] args) {
        // build UI

        while (!shell.isDisposed()) {
            if (!display.readAndDispatch()) {
                display.sleep();
            }
        }

        // shutdown
    }
}
```

Eclipse RCP

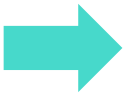
- **Target Platform**
- **Product**
  - Feature Dependencies
  - Product / Application
  - Classloader-Delegation
- **Application Model**
  - SWT Addons
- **Plug-in Dependencies**

Migration UI

Controls

- **SWT Controls mit JavaFX Controls austauschen**

```
org.eclipse.swt.widgets.Label
org.eclipse.swt.widgets.Text
org.eclipse.swt.widgets.Button
```



```
javafx.scene.control.Label
javafx.scene.control.TextField
javafx.scene.control.Button
```

- **Konstruktor-Parameter beachten**

```
Label label =
    new Label(parent, SWT.NONE);
label.setText("My Label");
```

```
Label label =
    new Label("My Label");
parent.getChildren().add(label);
```

Layout

```
@PostConstruct
public void postConstruct(Composite parent) {
        parent.setLayout(new GridLayout(3, true));
        ...
        GridDataFactory.fillDefaults().applyTo(label);
```

```
@PostConstruct
public void postConstruct(GridPane parent) {
        ...
        GridPane.setConstraints(label, 0, 0);
        GridPane.setMargin(label, new Insets(5.0));
```

Interaktionen & Event Handling

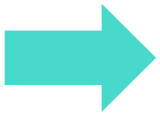
```
button.addSelectionListener(new SelectionAdapter() {
        @Override
        public void widgetSelected(SelectionEvent e) {
                ...
        }
});
```

```
button.setOnAction(event -> ...);
```

CSS Styling

- **Kein direktes CSS Styling (*applicationCSS*)**
- **Nur Theme-Styling via CSS**
- **Themes als OSGi declarative services**

```
Text {
    color: green;
}
```



```
.text-field {
        -fx-text-fill: green;
}
```

Weiterführende Informationen_

- **SWT to JavaFX migration**
  Dirk Fauth
  - https://blog.codecentric.de/en/2015/02/swt-javafx-migration-eclipse-rcp-cookbook/

- **Migrate E4 RCP from SWT to JavaFX using SWTonJavaFX**
  Christoph Keimel
  - http://www.kware.net/?p=64

“Things are getting really, really well now with JavaFX”

-Tom Schindl