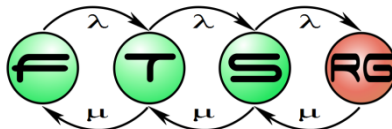


Web Application Development



UI Development Trends

Desktop applications

- Operating system integration
- Rich set of available widgets
- Performance and security

Web applications

- Web 2.0 standards
- No installation required
- Collaboration and communication

What is RAP?

- Component-oriented development
- Based on Eclipse Workbench (+ SWT, JFace)
- Based on Eclipse plug-ins
 - Dependency management
 - Extension points
 - Lifecycle management
 - Modular UI (similar to Web 2.0 mashups)

The Goals of RAP

- Support Java
 - Using the full Java API (compare with GWT)
 - Based on Java-based widget toolkit
- Uses component-oriented model (OSGi)
 - Modular (Eclipse extension points)
 - Can be used without web/Javascript coding

Why RAP?

- Skill preservation
 - Eclipse RCP knowledge reusable
- Single sourcing
 - Common technology for client and web application

How well does it work?

- Eclipse/RCP code...
 - 70-90% reusable
 - RAP implements a(n increasing) subset of RCP
 - But incompatibilities

 - **Important:** multi-user support

Sample application

Workbench Demo

File Window Help

Selection View Browser

Vehicle Type	Revenue (in Millions)
Classic Cars	3.85
Motorcycles	0.66
Planes	0.19
Ships	0.95
Trains	0.66
Trucks and Buses	1.80
Vintage Cars	1.02

Revenue (in Millions)

Column0	Column1	Column2	Column3	Column4	Column5	Column6
Item0-0	Item0-1	Item0-2	Item0-3	Item0-4	Item0-5	Item0-6
Item1-0	Item1-1	Item1-2	Item1-3	Item1-4	Item1-5	Item1-6
Item2-0	Item2-1	Item2-2	Item2-3	Item2-4	Item2-5	Item2-6
Item3-0	Item3-1	Item3-2	Item3-3	Item3-4	Item3-5	Item3-6

Not only RCP appearance...

The screenshot shows a web browser window with the address bar containing `http://127.0.0.1:2869/rap?startup=mail`. The page features a blue header with the word "Banner" and three buttons labeled "click me 0", "click me 1", and "click me 2". Below the header is a green navigation bar with tabs for "In", "Out", "Over", "Under", and "Through", and two buttons for "Perspective 1" and "Perspective 2". The main content area is titled "Content" and "The content goes here". On the left, there is a list of four items, each with a blue circular icon and the text "This is Text for chapter 0.", "This is Text for chapter 1.", "This is Text for chapter 2.", and "This is Text for chapter 3.". On the right, there are three video thumbnails, each with a yellow "0" icon, a "rap it" button, and a video player. The first video is titled "recep ivedik fragman" with ID "X0_7uUQKER0" and a description "Recep ?vedik'in beklenen filminin fragman?". The second video is titled "amazing guitar player" with ID "m3qMqK7h-BA" and a description "...one of those that will make you say...holy %\$#^". The third video is titled "Peanut and Jeff # 2" with ID "EpRW8jh8AqY" and a description "Ventriliquist". Each video player includes a "2y" icon and a "youtube.com" link.

But Still an Eclipse Workbench

Banner
commands

click me 0 | click me 1 | click me 2

perspectives

In Out Over Under Through Perspective 1 Perspective 2

Content

- This is Text for chapter 0.
- This is Text for chapter 1.
- selection service**
- This is Text for chapter 2.
- This is Text for chapter 3.
- This is Text for chapter 4.

The content goes here

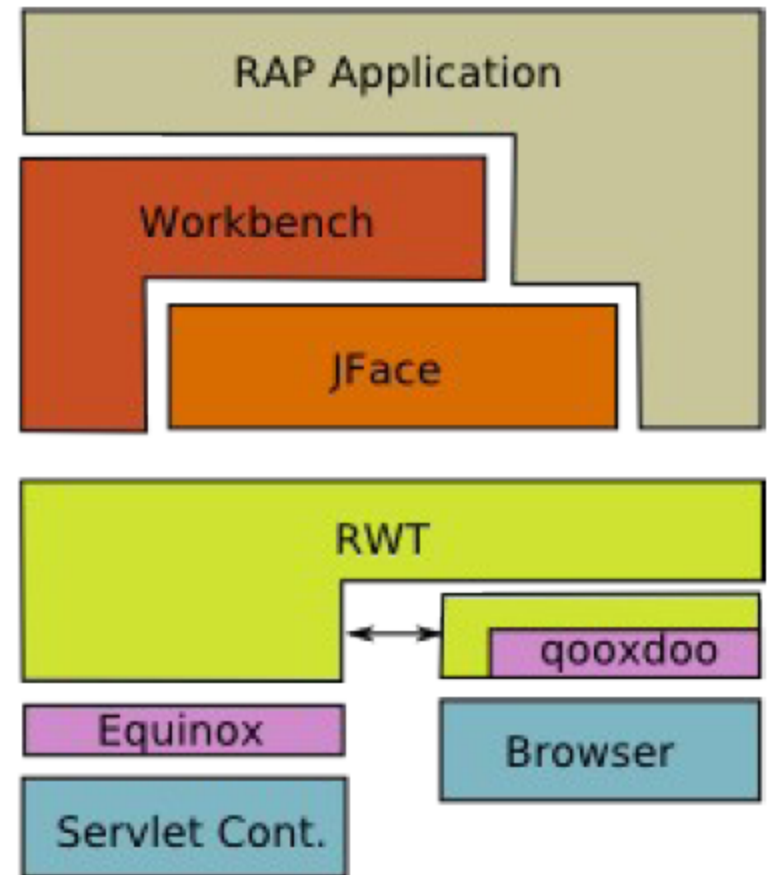
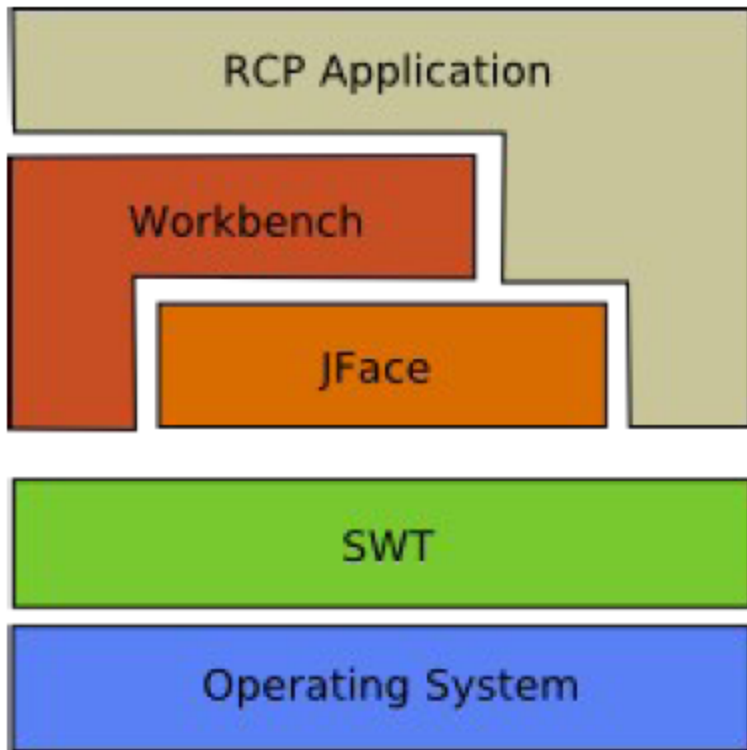
- 0 **recep ivedik fragman**
X0 7uJOKERD
Recep ?vedik'in beklenen filminin fragman?.
- 0 **amazing guitar player**
m3qMqk7h-BA
...one of those that will make you say...holy %\$#^
- 0 **Peanut and Jeff # 2**
EpRW8ih8AaY
Ventriliquist

Views, Editors

views

editors

Architecture



Architecture

- RWT (RAP Widget Toolkit)
 - Web-based implementation of SWT
 - Uses qooxdoo Javascript library
- Most SWT-based library works (mostly)
- Important difference: multiuser behaviour
 - Application bundles are shared between sessions

Internals

- Server centric
 - Entire business logic in server
- Browser
 - Javascript based rendering
 - Since RAP 2.0 uses a JSON for communication
- User event results in state change
 - → evaluated by the server
 - State delta traverses on the network

For the developer...

```
org.eclipse.rap.demo  *DemoTreeViewPart.java X
+ * Copyright (c) 2002-2006 Innoopract Informationssysteme GmbH.

package org.eclipse.rap.demo;

+ import java.util.ArrayList;

public class DemoTreeViewPart extends ViewPart implements IDoubleClickListener {

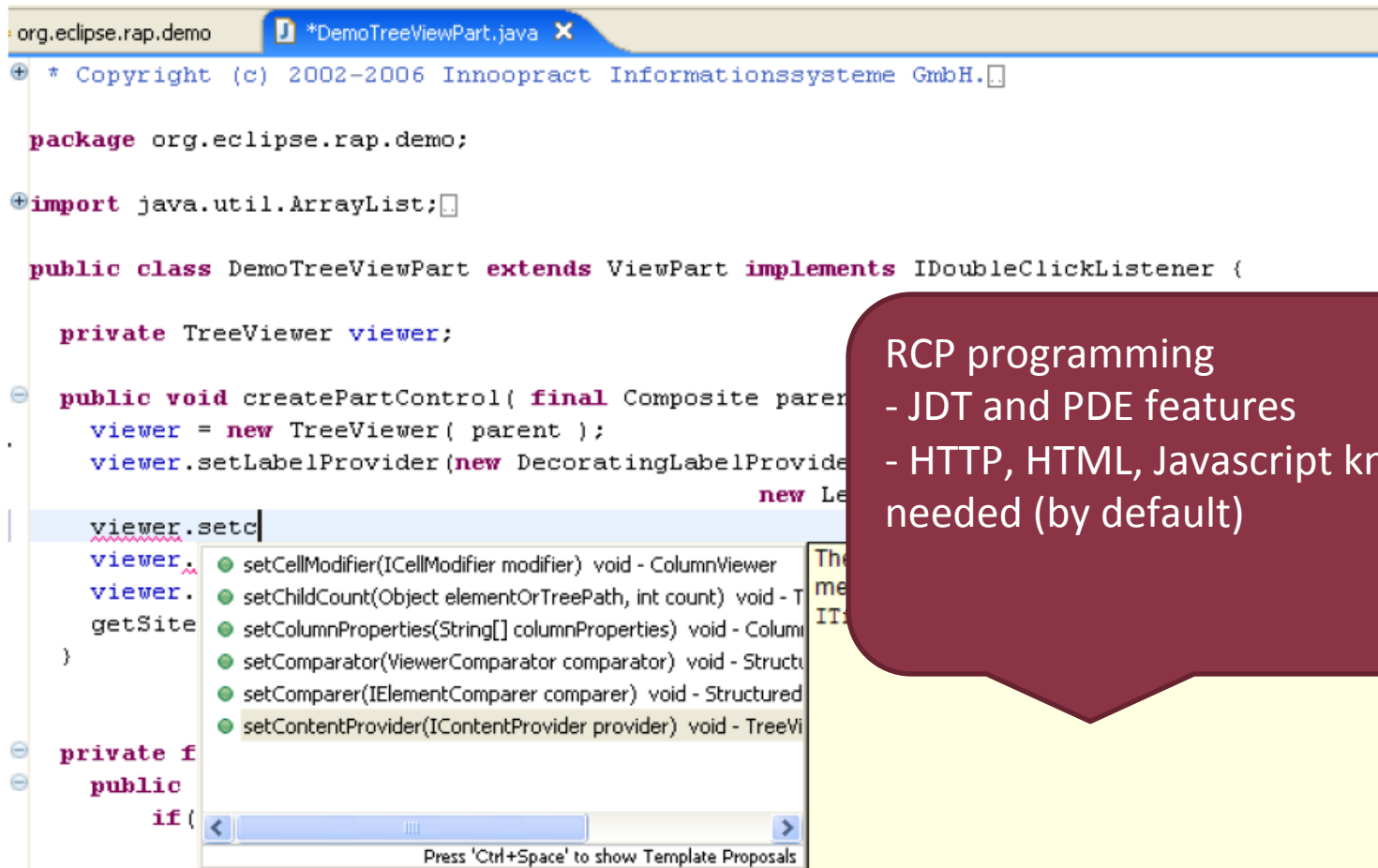
    private TreeViewer viewer;

- public void createPartControl( final Composite parent ) {
    viewer = new TreeViewer( parent );
    viewer.setLabelProvider(new DecoratingLabelProvider(new LabelProvider(),
                                                         new LeafStarLabelDecorator()));
viewer.setC
viewer.
viewer.
getSite
)
- private f
- public
    if (
< >
Press 'Ctrl+Space' to show Template Proposals
```

The AbstractTreeViewer implementation of this method checks to ensure that the content provider is an ITreeContentProvider.

For the developer...

```
org.eclipse.rap.demo  *DemoTreeViewPart.java x
+ * Copyright (c) 2002-2006 Innoopract Informationssysteme GmbH.
package org.eclipse.rap.demo;
+ import java.util.ArrayList;
public class DemoTreeViewPart extends ViewPart implements IDoubleClickListener {
    private TreeViewer viewer;
- public void createPartControl( final Composite parent
    viewer = new TreeViewer( parent );
    viewer.setLabelProvider(new DecoratingLabelProvider
                                new Le
viewer.setc
viewer.
viewer.
getSite
)
- private f
- public
    if (
Press 'Ctrl+Space' to show Template Proposals
```



RCP programming

- JDT and PDE features
- HTTP, HTML, Javascript knowledge not needed (by default)

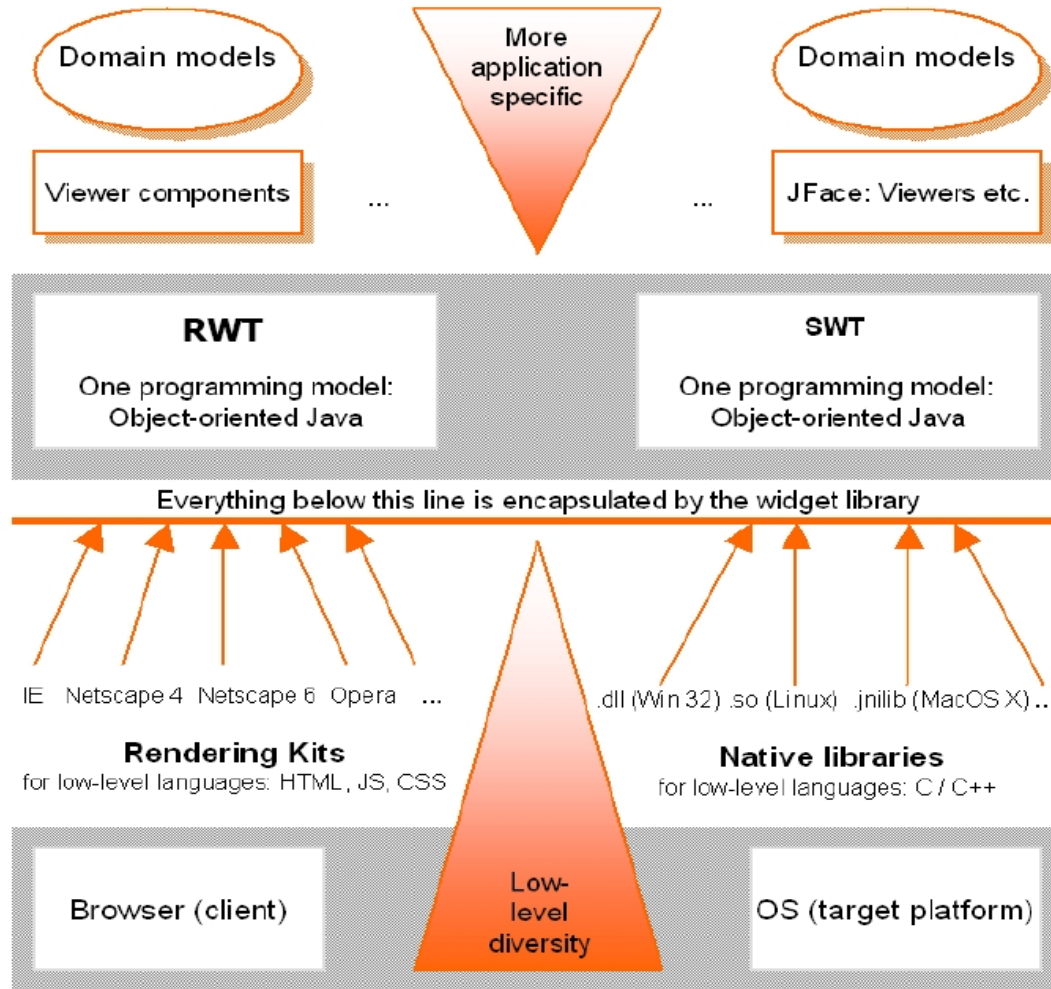
RAP Application Execution / Debug

The screenshot displays the Eclipse IDE interface with the 'Run' dialog box open. The dialog is titled 'Create, manage, and run configurations' and is used to launch a RAP application. The 'Name' field is set to 'RMS'. The 'Framework' is set to 'Equinox', the 'Default Start level' is '4', and 'Default Auto-Start' is 'true'. The 'Bundles' tab is active, showing a list of bundles with their start levels. The 'Workspace' section is expanded, showing the following bundles:

Bundle	Start Level
org.eclipse.equinox.http.servletbridge (1.0.0)	default
org.eclipse.equinox.servletbridge (1.0.0.qualif	default
org.eclipse.rap.custom.viewers (1.0.0)	default
org.eclipse.rap.demo (1.0.0.20071008-1834)	
org.eclipse.rap.demo.gmaps (1.0.0)	
org.eclipse.rap.malldemo (1.0.0)	default
org.eclipse.rap.pde.runtime (3.3.0.v20070606	default
org.eclipse.rap.rms.data (1.0.0)	default
org.eclipse.rap.rms.ui (1.0.0)	default
org.eclipse.rap.rwt.custom (1.0.0)	default
org.eclipse.update.configurator (3.2.100.qual	default

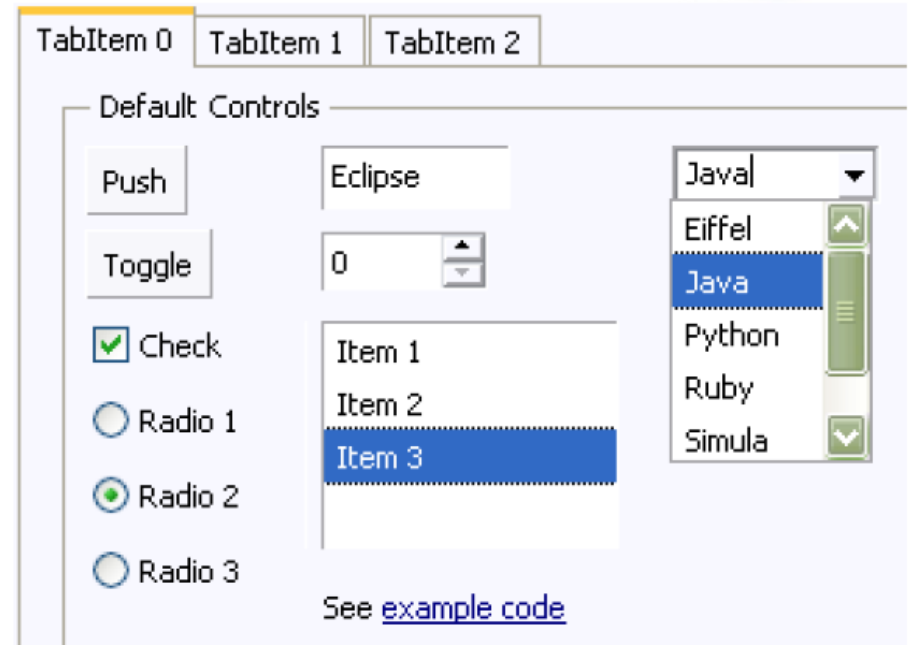
At the bottom of the dialog, there are two checked options: 'Include optional dependencies when computing required bundles' and 'Add new workspace bundles to this launch configuration automatically'. The 'Outline' view is visible in the top right corner, and the 'Extensions' view is visible in the top left corner.

RWT Architecture



RWT

- Most basic widgets available
- Workbench widgets (CTabFolder, CBanner)
- Browser widget ☺
- Tree, Table

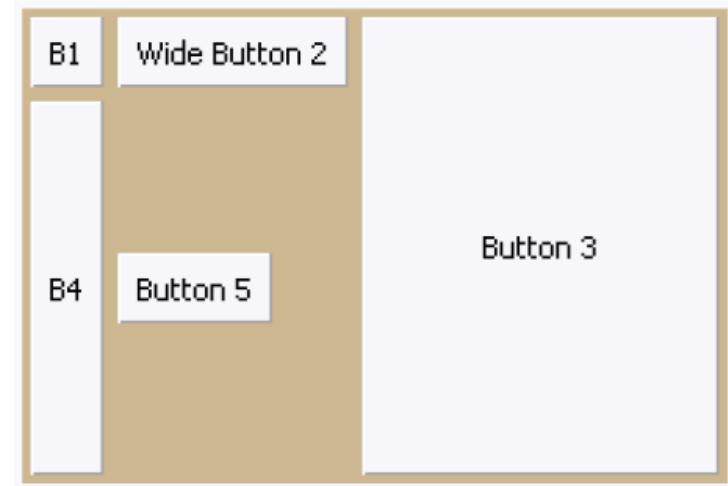


Column 1	Column 2
<input type="checkbox"/> Node_1	abc
<input checked="" type="checkbox"/> Node_2	jklmnop
<input type="checkbox"/> Subnode	jklmnop
<input type="checkbox"/> Node_3	jklmnop
<input type="checkbox"/> Node_4	jklmnop

Col 0	Col 1	Col 2	Col 3	Col 4
<input checked="" type="checkbox"/> Item0-0	Item0-1	Item0-2	Item0-3	Item0-4
<input type="checkbox"/> Item1-0	Item1-1	Item1-2	Item1-3	Item1-4
<input type="checkbox"/> Item2-0	Item2-1	Item2-2	Item2-3	Item2-4
<input type="checkbox"/> Item3-0	Item3-1	Item3-2	Item3-3	Item3-4
<input type="checkbox"/> Item4-0	Item4-1	Item4-2	Item4-3	Item4-4

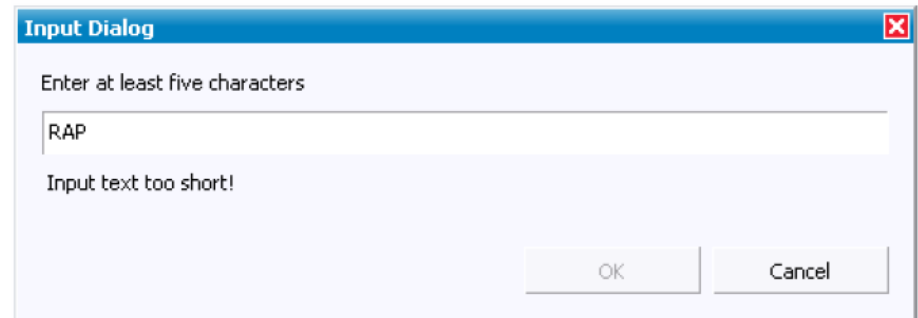
RWT Layouts

- All layouts available:
 - Grid, Row, Fill, Form, Stack
 - Basically the same as SWT Layouts
- Behaviour
 - Server calculates layout with pixel precision



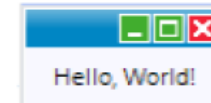
Event Handling

- All SWT Listeners supported
 - SelectionListener
 - ControlListener
 - ShellListener
 - MenuListener
 - ...



- JFace data binding, validation also works

RAP Hello World



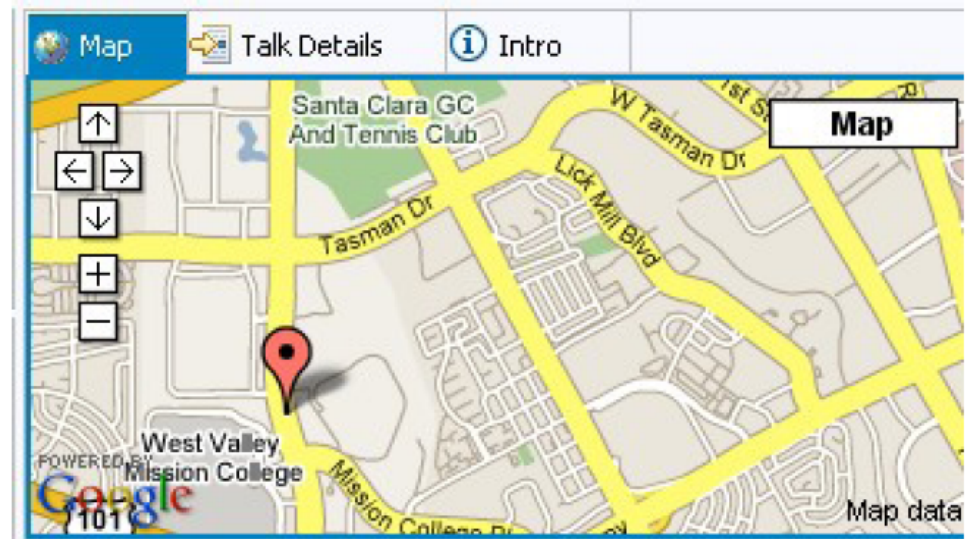
```
public class RWTHello implements IEntryPoint {

    public Display createUI() {
        Display result = new Display();
        final Shell shell = new Shell( result );
        RowLayout layout = new RowLayout();
        layout.justify = true;
        layout.pack = true;
        shell.setLayout( layout );
        Label label = new Label( shell, SWT.CENTER );
        label.setText( "Hello, World!" );
        shell.pack();
        shell.open();
        return result;
    }
}

<extension
    id="org.eclipse.rap.demo.demointerpoint"
    point="org.eclipse.rap.ui.workbench.entrypoint">
    <entrypoint
        class="org.eclipse.rap.tutorial.HelloWorld"
        parameter="default"/>
</extension>
```

Extensibility (Custom Widgets)

- Possible, but requires deep understanding
 - Component developer: Javascript, qooxdoo, RAP
 - Application developer: only Java API
- Example
 - Google Maps integration: RAP Help

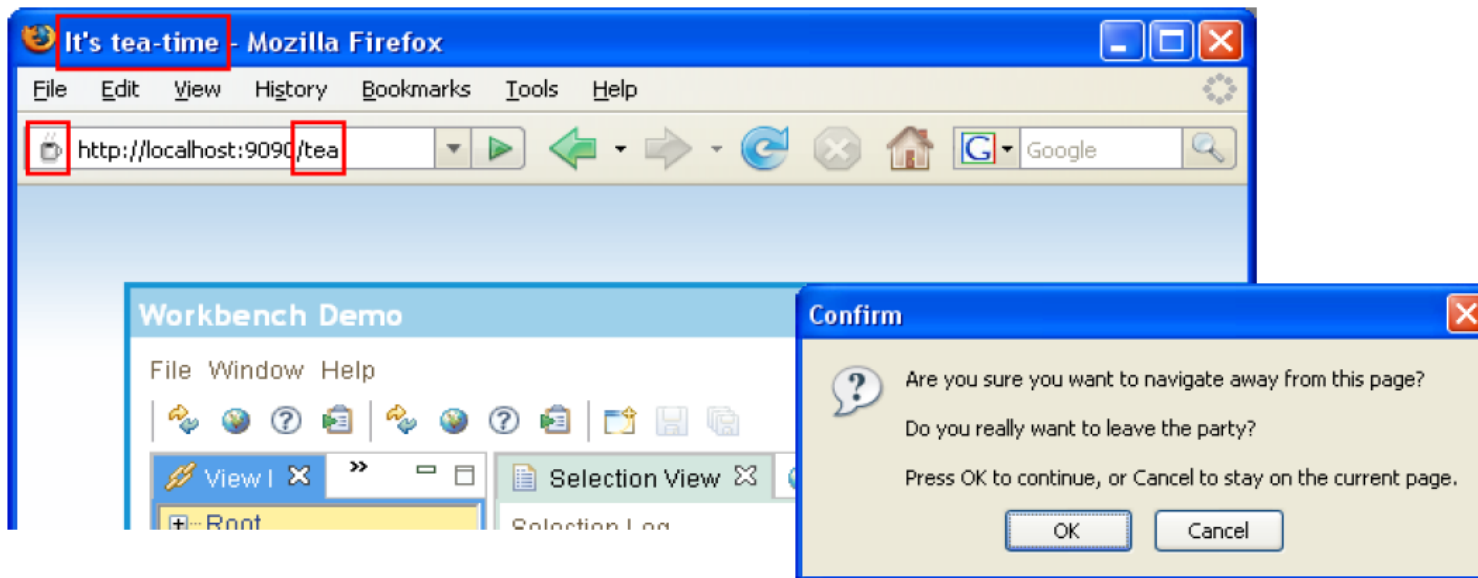


RWT Customization (Themes)

- Unique description
- CSS-based theming
 - More powerful than Eclipse 4 CSS
 - Widget settings
 - Size
 - Colors
 - Fonts
 - ...
 - Works for custom widgets (if implemented)



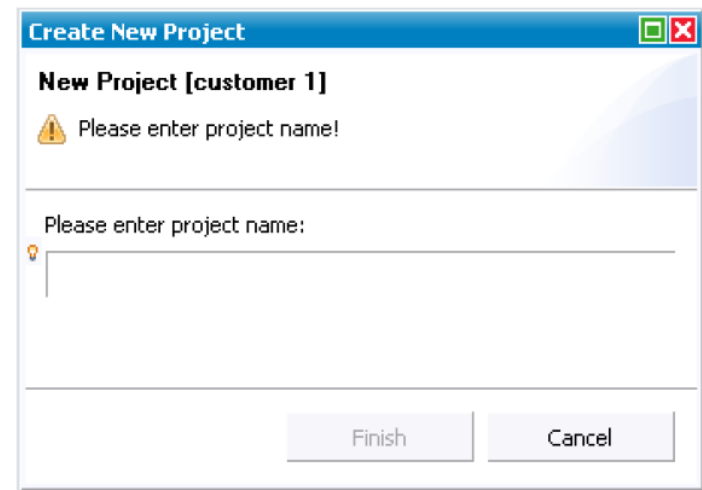
Branding



```
<extension
  point="org.eclipse.rap.ui.branding">
  <branding
    id="org.eclipse.rap.demo.branding1"
    servletName="tea"
    defaultEntrypointId="org.eclipse.rap.demo.entrypoint1"
    themeId="org.eclipse.rap.demo.alttheme"
    title="It's tea-time"
    favicon="icons/favicon2.ico"
    body="body.html"
    exitConfirmation="Do you really want to leave the party?">
  </branding>
</extension>
```


RAP and JFace

- JFace viewer framework works as expected
 - TableView, TreeViewer
 - Provider, Sorter, Filter, Decorator
 - Cell Editor
- Dialogs
 - Standard dialogs (MessageDialog, ErrorDialog)
 - Modal dialogs
 - Custom dialogs
- Wizards



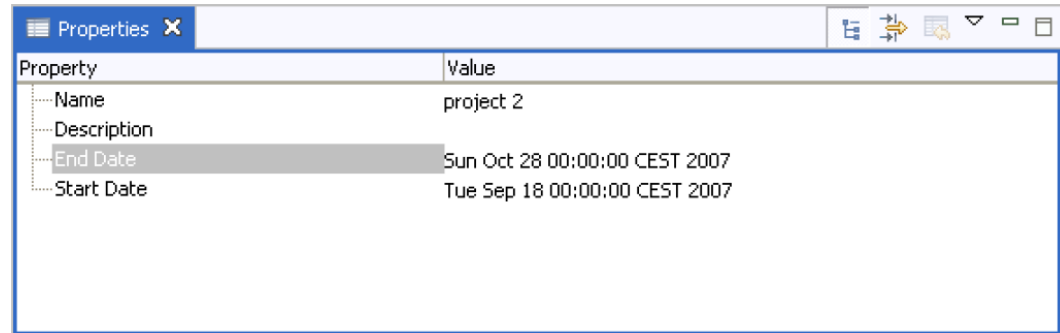
RAP and Workbench

The screenshot displays the RAP Showcase application interface with several components highlighted by colored boxes:

- Menu Bar:** Located at the top left, containing 'User' and 'Help' options.
- Workbench Window:** The main application window, outlined in blue.
- Workbench Parts:** A red box highlights the top navigation area, including 'Map', 'Talk Details', and 'Intro' tabs.
- Workbench Page:** A purple box highlights the main content area, which features a map of Karlsruhe, Germany, with navigation controls like 'Karte', 'Satellit', and 'Hybrid'.
- Cool Bar:** A yellow box highlights the left sidebar, which contains a 'Me: Appel Frank' profile and a list of contacts and tracks.
- Dialog:** A green box highlights a 'Change User Data' dialog box. It contains a message: 'Please correct the input data which is marked red.' and fields for 'Firstname: Frank', 'Lastname: Appel', 'Street:', 'City: Karlsruhe', 'Country: Deutschland', 'Username: fa', 'Password: ..', and 'Confirm Password: ...'. A red error message at the bottom states: 'The password and its confirmation are not the same.'

RAP and Workbench

- View
 - Full Support
 - Outline, Properties
- Editor
 - Multi-page editors as well
- Perspective
 - Built-in perspective switcher
- Actions, Commands

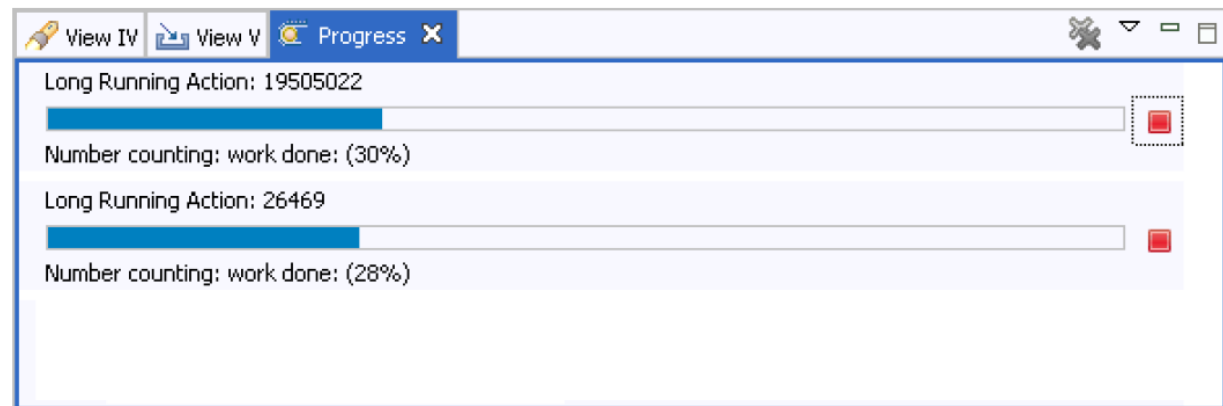


The screenshot shows a 'Properties' window with a table of project details. The table has two columns: 'Property' and 'Value'. The 'End Date' row is highlighted.

Property	Value
Name	project 2
Description	
End Date	Sun Oct 28 00:00:00 CEST 2007
Start Date	Tue Sep 18 00:00:00 CEST 2007

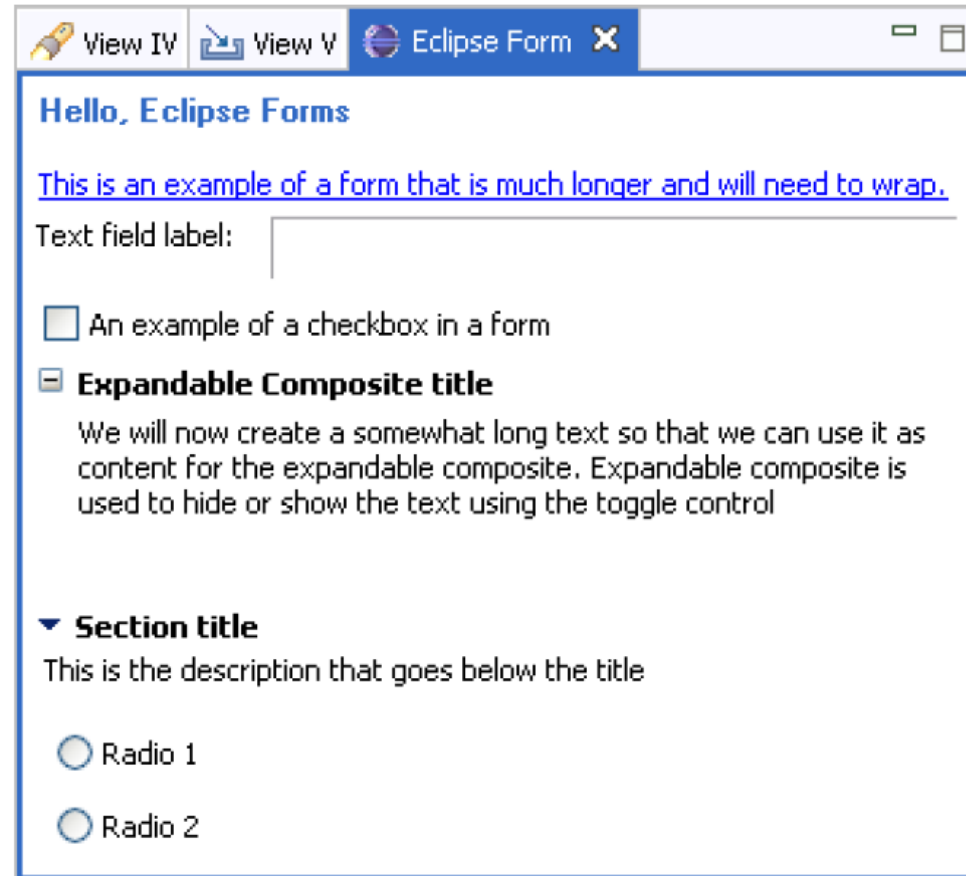
Workbench jobs, tasks

- Support for Job API
 - Run in background
 - Progress Monitors



Forms, Data binding

- Data binding works
- Eclipse UI Forms works



The screenshot shows an Eclipse IDE window titled "Eclipse Form" with three tabs: "View IV", "View V", and "Eclipse Form". The form content is as follows:

Hello, Eclipse Forms

[This is an example of a form that is much longer and will need to wrap.](#)

Text field label:

An example of a checkbox in a form

Expandable Composite title

We will now create a somewhat long text so that we can use it as content for the expandable composite. Expandable composite is used to hide or show the text using the toggle control

▼ **Section title**

This is the description that goes below the title

Radio 1

Radio 2

RAP Deployment

- An RAP application can be deployed to
 - JEE Server konténerbe (WAR)
 - Uses OSGi Servlet Bridge
 - OSGi standalone (default)
 - Uses Jetty as web server

RAP – More information

- <http://www.eclipse.org/rap/>
- <http://rapblog.innoopract.com/>
- <http://www.ibm.com/developerworks/library/os-eclipse-richajax1/>
- [http://www.infoq.com/news/2008/06/eclipse-ganymede-
rap;jsessionid=A0CC55BBF7B9C7CB44D51CD26D6
FB6EE](http://www.infoq.com/news/2008/06/eclipse-ganymede-
rap;jsessionid=A0CC55BBF7B9C7CB44D51CD26D6
FB6EE)

RAP and RCP Together: Single Sourcing in Practice

Main Differences

RCP	RAP
File System	File Upload
Graphical Canvas	Limited Canvas
One User	Multiple Users



Session Management

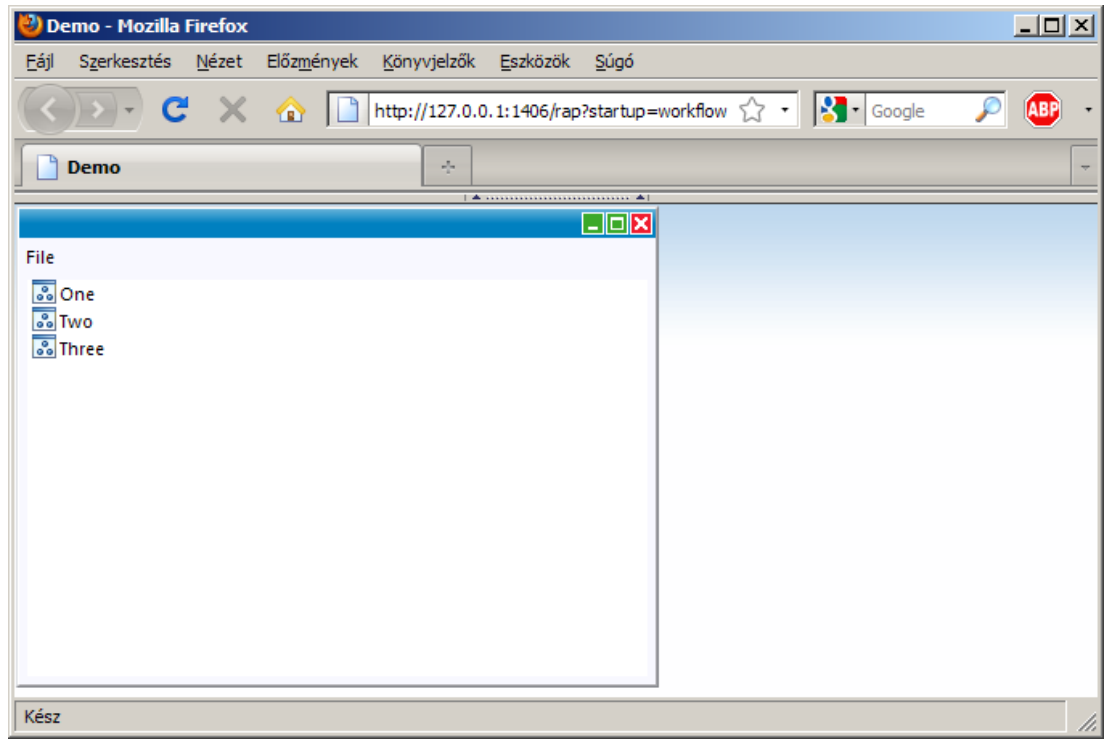
- Multiple users => session management
 - No static/singleton states!
 - RWT session store
 - `RWT.getSessionStore()`

RWT Session Management

- Querying Session id:
 - `RWT.getSessionStore().getId();`
- User-specific session info
 - `RWT.getSessionStore().getHttpSession().put(key, value);`
- Session listener
 - `RWT.getSessionStore().addSessionStoreListener(myListener);`
- `HttpServletRequest` object also available
 - `RWT.getRequest();`

Appearance

- Different UI paradigm
- Window in window
 - Problematic
 - Closable



Appearance

■ Solution - advisor:

```
@Override
public void preWindowOpen() {
    IWorkbenchWindowConfigurer configurer =
        getWindowConfigurer();
    configurer.setShellStyle(SWT.NO_TRIM);
}
```

No border

```
@Override
public void postWindowOpen() {
    final IWorkbenchWindow window =
        getWindowConfigurer().getWindow();
    Shell shell = window.getShell();
    shell.setMaximized(true);
}
```

Maximalized windows

Single Sourcing

- Common functionality between RAP and RCP
 - Minimal costs
 - Common codebase
 - Few code duplication
- Identify UI-specific code
 - Use target platforms
 - Use plug-in fragments

Plug-in fragments

- Not a standalone plug-in
 - Defines a host
 - No Activator (no separate lifecycle)
 - No unique classloader for fragments
 - Reuses host classloader
- Typical uses
 - Multilanguage support
 - Multiple platform support

Single Sourcing with RAP fragment

- Host plugin: RCP
 - Optional dependencies:
 - org.eclipse.ui
 - org.eclipse.rap.ui
- Fragment: RAP
 - Every RAP-specific code goes here
 - Entry point
 - Branding

Hiding API differences

■ Use common base classes

```
public abstract class AboutActionHelper {  
  
    private static final AboutActionHelper IMPL;  
  
    static {  
        Class clazz = AboutActionHelper.class;  
        Object newInstance = ImplementationLoader.newInstance( clazz );  
        IMPL = ( AboutActionHelper )newInstance;  
    }  
  
    public static IWorkbenchAction create( IWorkbenchWindow window ) {  
        return IMPL.createInternal( window );  
    }  
  
    protected abstract IWorkbenchAction createInternal( IWorkbenchWindow  
        window );  
  
}
```

Hiding API differences

■ Load implementation using reflective API

```
public final class ImplementationLoader {  
  
    public static Object newInstance( Class type ) {  
        String name = type.getName();  
        Object result = null;  
        ClassLoader loader = type.getClassLoader();  
        try {  
            Class clazz = loader.loadClass( name + "Impl" );  
            result = clazz.newInstance();  
        } catch( Throwable t ) {  
            // ...  
        }  
        return result;  
    }  
  
    private ImplementationLoader() {}  
  
}
```

Hiding API differences

■ RAP-specific implementation

```
public class AboutActionHelperImpl extends AboutActionHelper {
    private static final class AboutAction extends Action
        implements IWorkbenchAction {
        private AboutAction() {
            setText( "About" );
            setId( "aboutAction" );
        }
        public void run() {
            MessageDialog.openInformation( null, "About", "About" );
        }
        public void dispose() {}
    }
    protected IWorkbenchAction createInternal( IWorkbenchWindow
        win ) {
        return new AboutAction();
    }
}
```

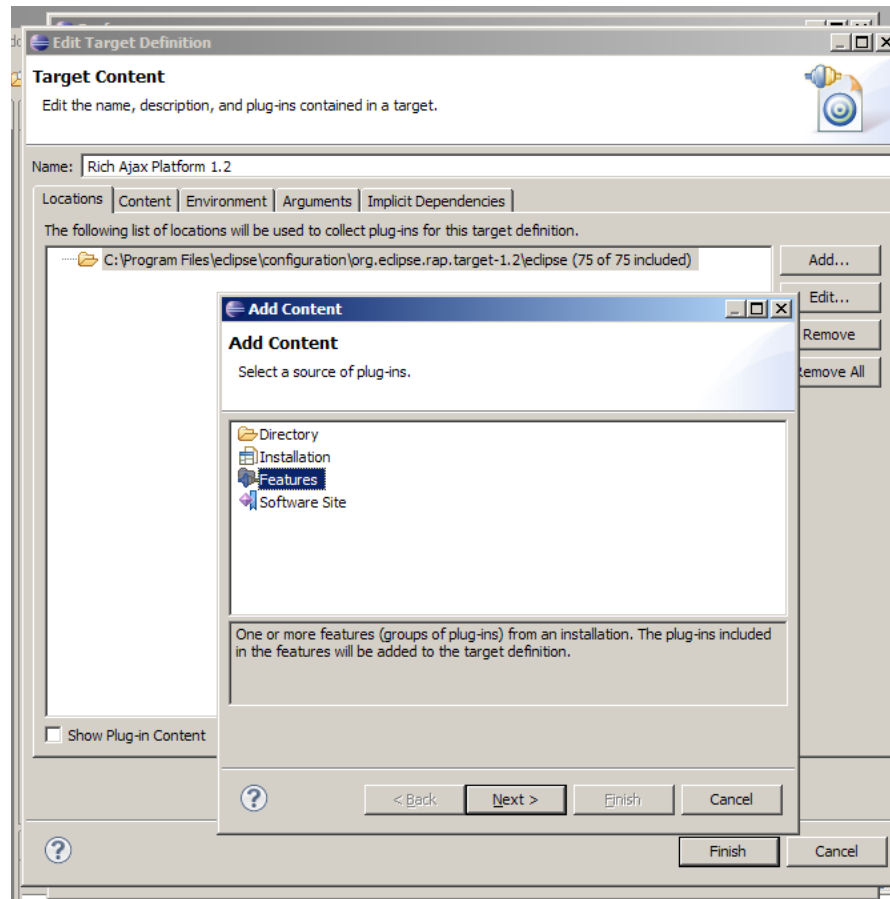
Hiding API differences

■ RCP-specific implementation

```
public class AboutActionHelperImpl extends AboutActionHelper {  
    protected IWorkbenchAction createInternal( IWorkbenchWindow  
        win ) {  
        return ActionFactory.ABOUT.create(win);  
    }  
}
```

Target Platform Management

- No custom workspace required
- Add required plug-ins to the target platform



RAP Branding

- Different options than in RCP (extension point)
 - servletName
 - For the URL (default: rap)
 - defaultEntrypointId
 - Application entry point
 - themeld
 - (CSS) Theme identifier
 - title
 - For the HTML title
 - favicon
 - body
 - Additional HTML code for the page body
 - additionalHeaders gyermekelemek
 - Additional HTML code for page header

RWT Theming

- Widgetek appearance
- CSS-like definition

```
* {  
  font: 12px Tahoma;  
}
```



Changes default
font for all
elements

```
TableItem:even {  
  background-color: #eef7ff;  
}
```



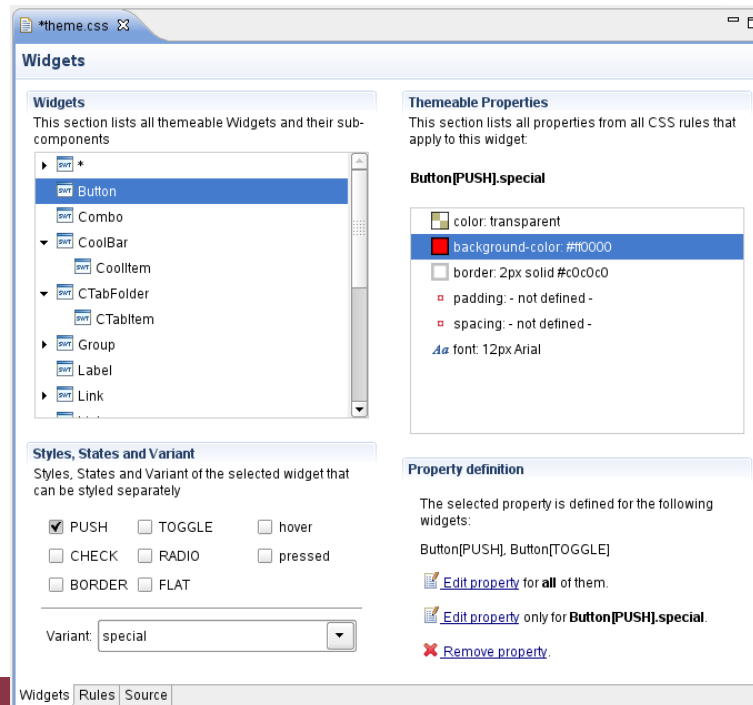
Background for
table even rows

RWT Theming

- Selectors:
 - Widget (Button, List...)
 - Style (PUSH, TOGGLE, CHECK, RADIO, BORDER, FLAT)
 - State (hover, pressed, disabled, selected)
- Properties:
 - color
 - padding
 - border
 - font
- Reference:
 - <http://eclipse.org/rap/developers-guide/devguide.php?topic=theming.html&version=2.2>

RWT Theme Editor

- RAP Theme Editor
 - Form-based editor
 - Preview functionality
 - [http://wiki.eclipse.org/RAP Theme Editor](http://wiki.eclipse.org/RAP_Theme_Editor)



Custom RWT Widget

- Required steps
 - Java: Widget-interface creation
 - JavaScript: Widget creation with qooxdoo framework
 - Combining client and server-side
 - Javascript registration in plugin.xml

- More details:
<http://eclipse.org/rap/developers-guide/devguide.php?topic=custom-widget.html&version=2.2>

Theming Custom Widgets

- org.eclipse.rap.ui.themeableWidgets extension point
- Required:
 - Theme adapter:
org.eclipse.rwt.theme.IControlThemeAdapter
 - Theme definition: XML fájl
 - Appearance fragment: CSS template
- In more details:
[http://dev.eclipse.org/viewcvs/index.cgi/
org.eclipse.rap/org.eclipse.rap.help/help/html/
advanced/theming-custom.html?
root=Technology Project&view=co](http://dev.eclipse.org/viewcvs/index.cgi/org.eclipse.rap/org.eclipse.rap.help/help/html/advanced/theming-custom.html?root=Technology+Project&view=co)

Additional features – 1.

- Theme contribution:
 - Extending existing themes
 - Uses `org.eclipse.rap.ui.themes` extension point

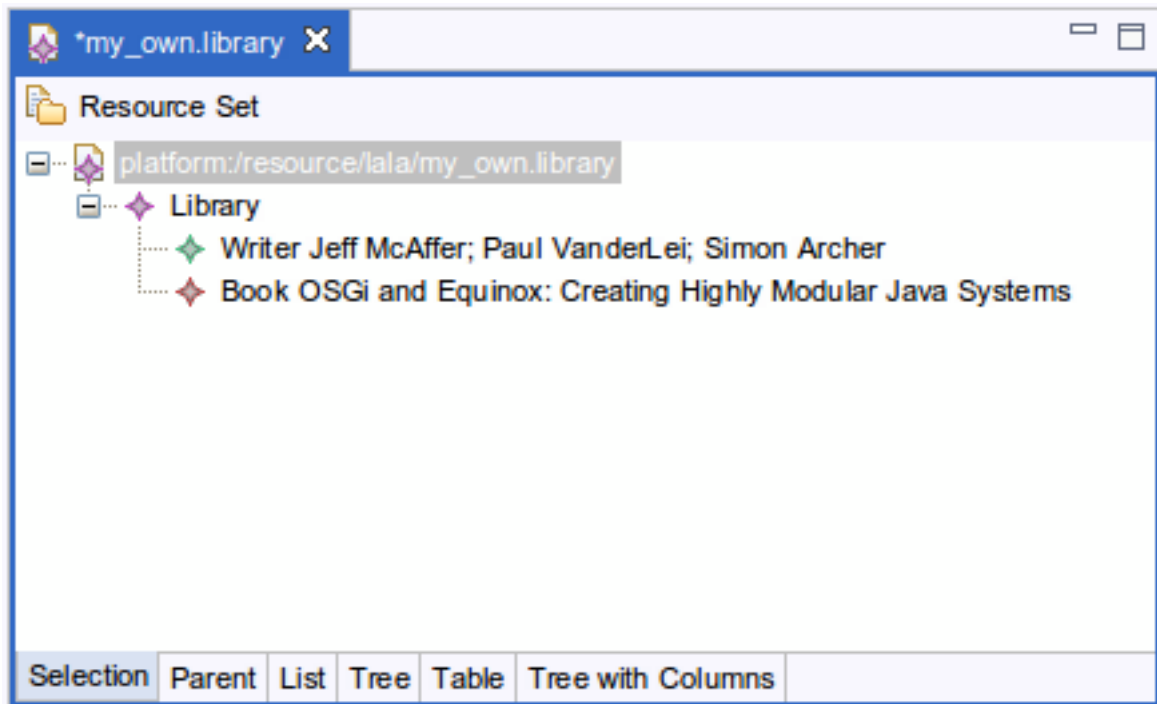
Additional Features – 2.

- Graphics Context:
 - SWT Canvas
 - Subset of SWT options



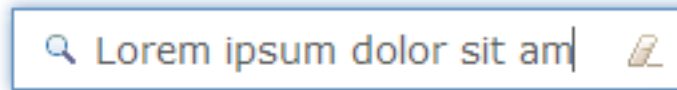
Additional Features – 3.

- EMF on RAP
 - EMF tree editor in RAP
 - Can be generated from EMF genmodel

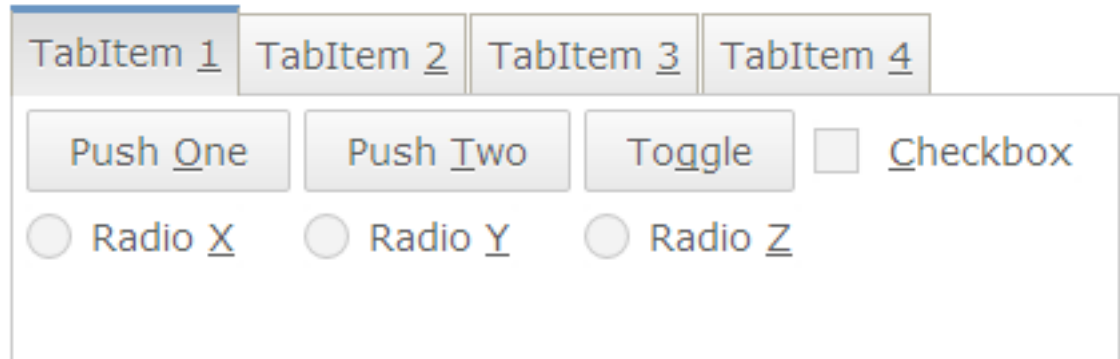
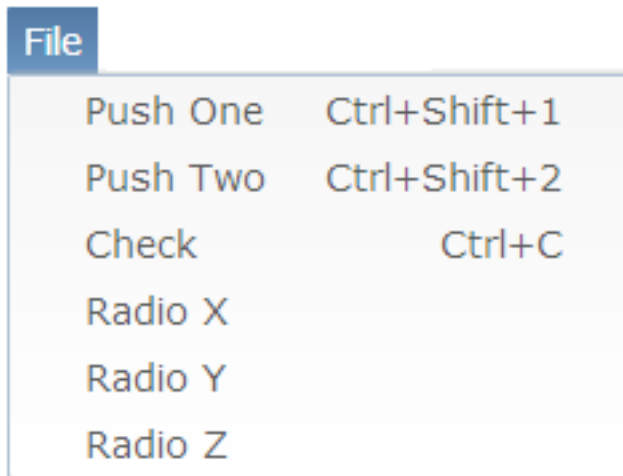


Additional Features – 4.

- SWT.ICON_SEARCH and SWT.ICON_CANCEL support for Text widget









- Keyboard accelerators and mnemonics



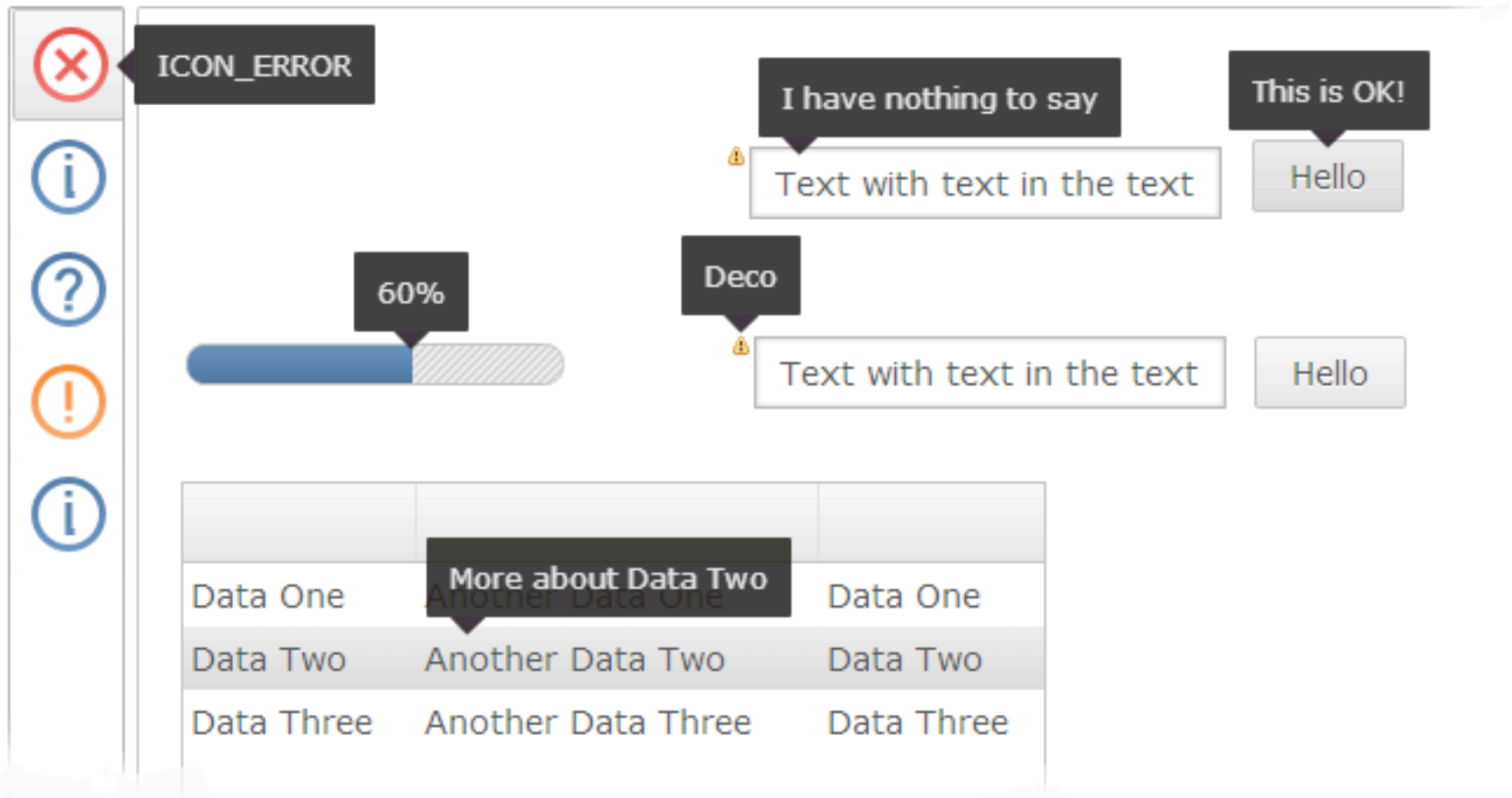
Additional Features – 5.

■ Row Templates

	Adam Archer	E-Mail: adam@mail.domain Phone: 555 123456	>
	Barabara Baker	E-Mail: barbara@mail.do... Phone: 555 123456	>
	Casper Carter	E-Mail: casper@mail.dom... Phone: 555 123456	>
	Damien Dyer	E-Mail: damien@mail.dom... Phone: 555 123456	>
	Edward Evans	E-Mail: edward@mail.dom... Phone: 555 123456	>
	Frank	E-Mail: frank@mail domain	

Additional Features – 6.

■ Tooltips



Additional Features – 7.

- Other important parts
 - Browser History
 - DateTime widget
 - Drag and drop
 - Eclipse Help, Cheat Sheet