Getting Started with RAP Development

Ralf Sternberg
rsternberg@innoopract.com

Rüdiger Herrmann
rherrmann@innoopract.com

Outline

- RAP Introduction and Overview
  - How to create your first RAP application
  - Lab I: RAP tooling, create application from template
- Differences between RCP and RAP
  - Support for creating web applications
  - What RAP (currently) lacks and why
  - Lab II: Make application session-aware
- Customize Look and Feel
  - Apply a custom theming and branding to your application
  - Lab III: Application styling

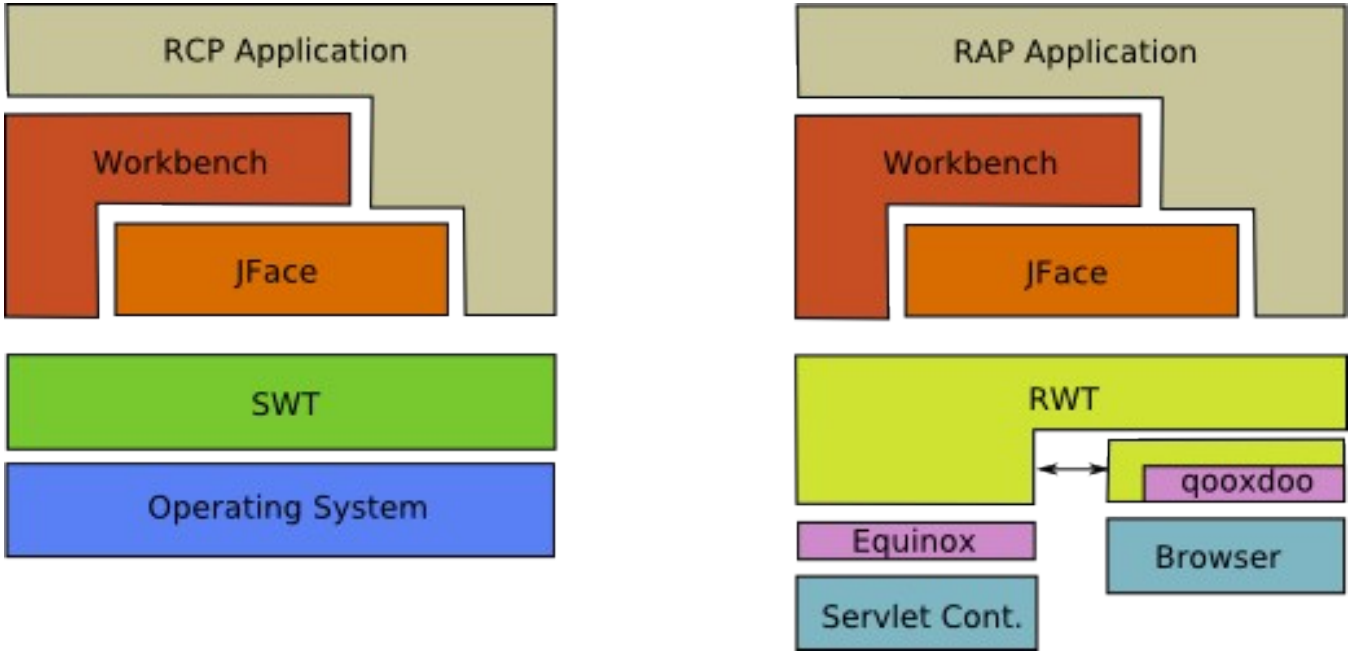
What is RAP?

The Rich Ajax Platform

enables developers to build rich, Ajax-enabled web applications by using the Eclipse development model, plug-ins and a Java-only API:

- coding in Java, developing the UI with SWT, JFace, Workbench and extension points
- using the Eclipse extension point mechanism
- using the Eclipse development tools
- running the application on the server
- and accessing it with a web browser

RAP Architecture Overview

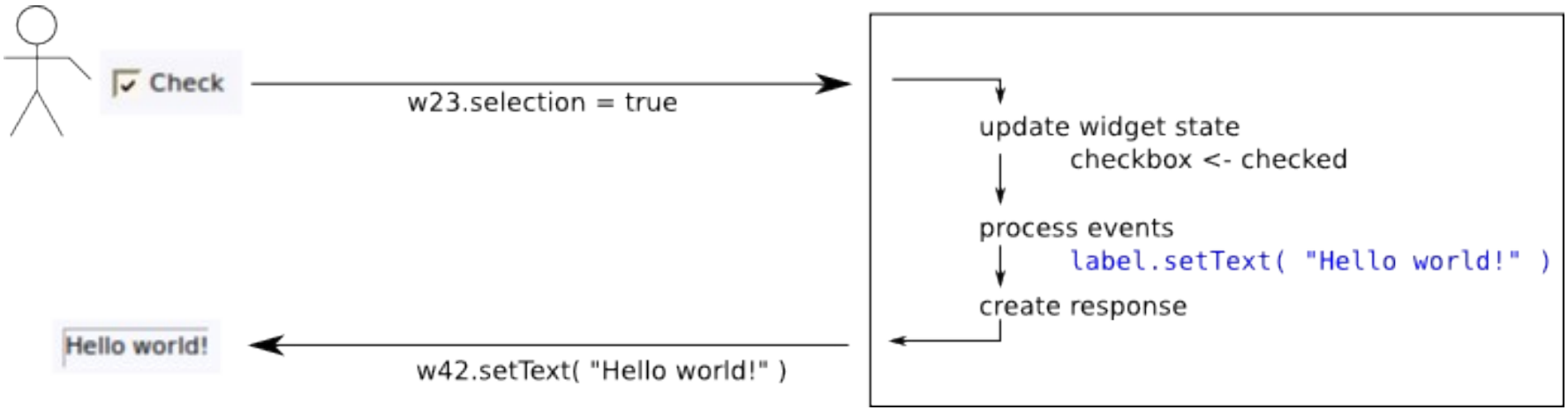


RAP implements a subset of SWT, JFace, Workbench APIs

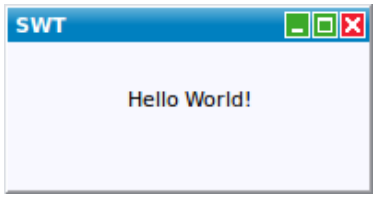
- is built on top of Equinox, running in server environments
- based on JEE Servlet technology
- uses the Qooxdoo JavaScript library for client side rendering in the browser

How does it work?

- Widgets have a server-side and a client-side part that communicate
- Events trigger a request
- Request is processed server-side
  - update server-side widget states
  - process events
  - generate response that contains updates for client-side widgets
- UI changes are rendered in the response (delta)



The RAP Hello World

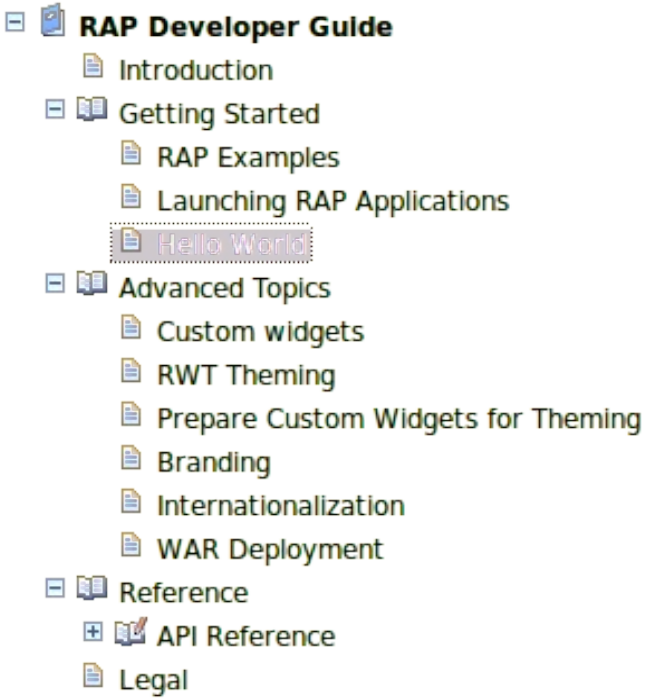


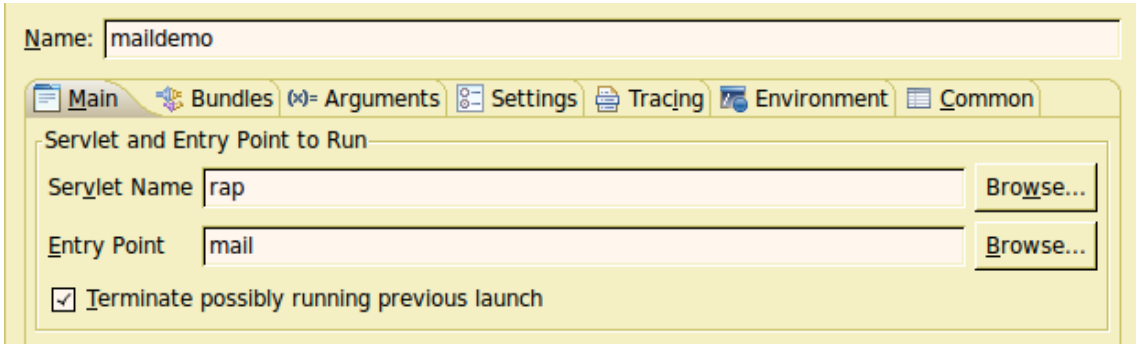

```java
public class HelloSnippet {

  public static void main( String[] args ) {
    Display display = new Display();
    Shell shell = new Shell( display );
    shell.setText( "SWT" );

    FillLayout layout = new FillLayout();
    layout.marginHeight = 20;
    shell.setLayout( layout );

    Label label = new Label( shell, SWT.CENTER );
    label.setText( "Hello World!" );

    shell.setSize( 200, 100 );
    shell.open();
    while( !shell.isDisposed() ) {
      if( !display.readAndDispatch() )
        display.sleep();
    }
    display.dispose();
  }
}
```

```java
public class HelloSnippet implements IEntryPoint {

  public int createUI() {
    Display display = new Display();
    Shell shell = new Shell( display );
    shell.setText( "SWT" );

    FillLayout layout = new FillLayout();
    layout.marginHeight = 20;
    shell.setLayout( layout );

    Label label = new Label( shell, SWT.CENTER );
    label.setText( "Hello World!" );

    shell.setSize( 200, 100 );
    shell.open();
    while( !shell.isDisposed() ) {
      if( !display.readAndDispatch() )
        display.sleep();
    }
    display.dispose();
    return 0;
  }
}
```

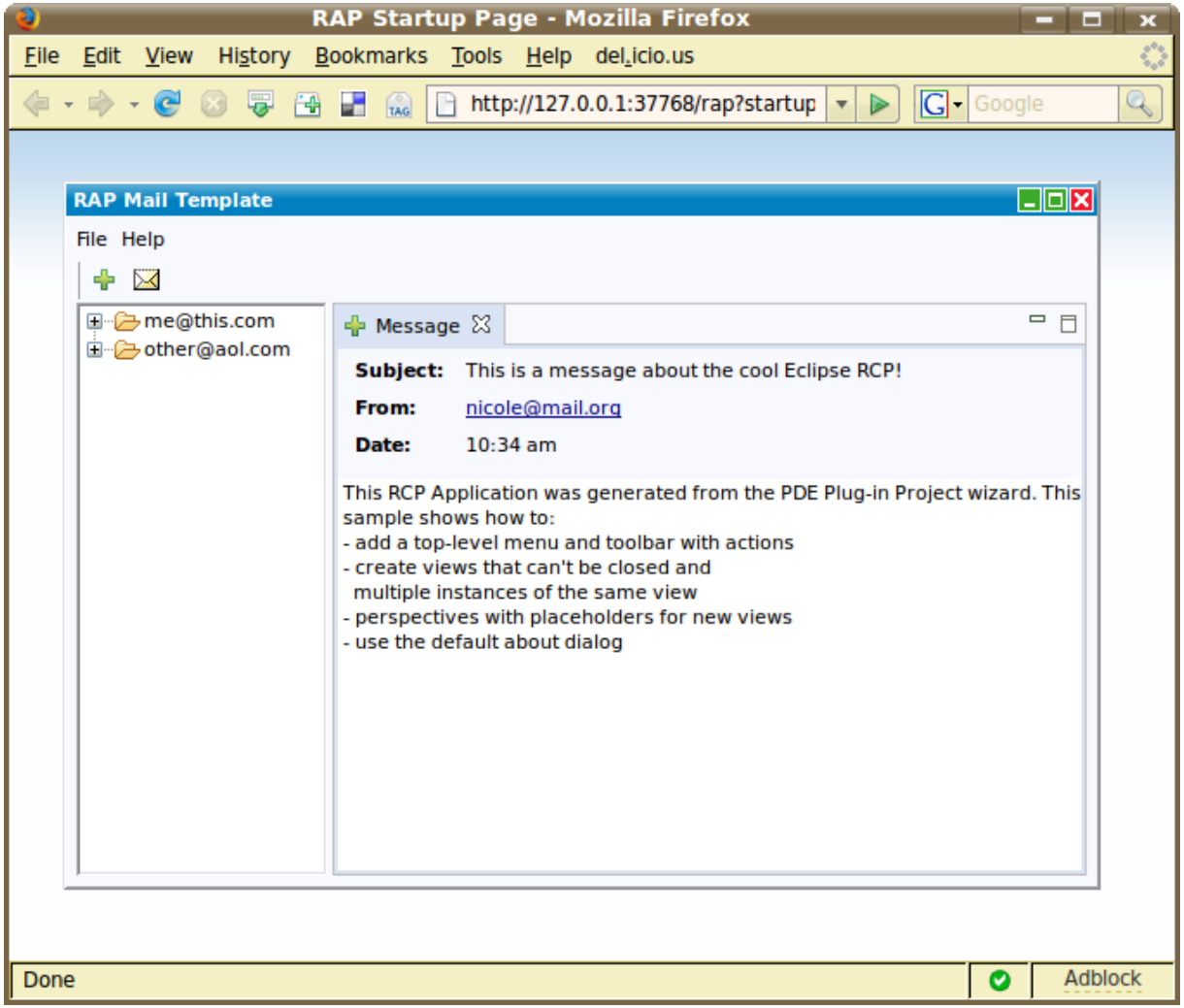
What You Need to Get Started

- Target Platform provides Runtime
  - RAP plug-ins (org.eclipse.rap.rwt, org.eclipse.rap.ui, ...)
  - Equinox (org.eclipse.equinox.http, ...)
  - Eclipse platform core plug-ins (org.eclipse.core.runtime, ...)
- Tooling eases development
  - Templates
  - Help, cheat sheet
  - Launcher
  - Comes with the most recent milestone target



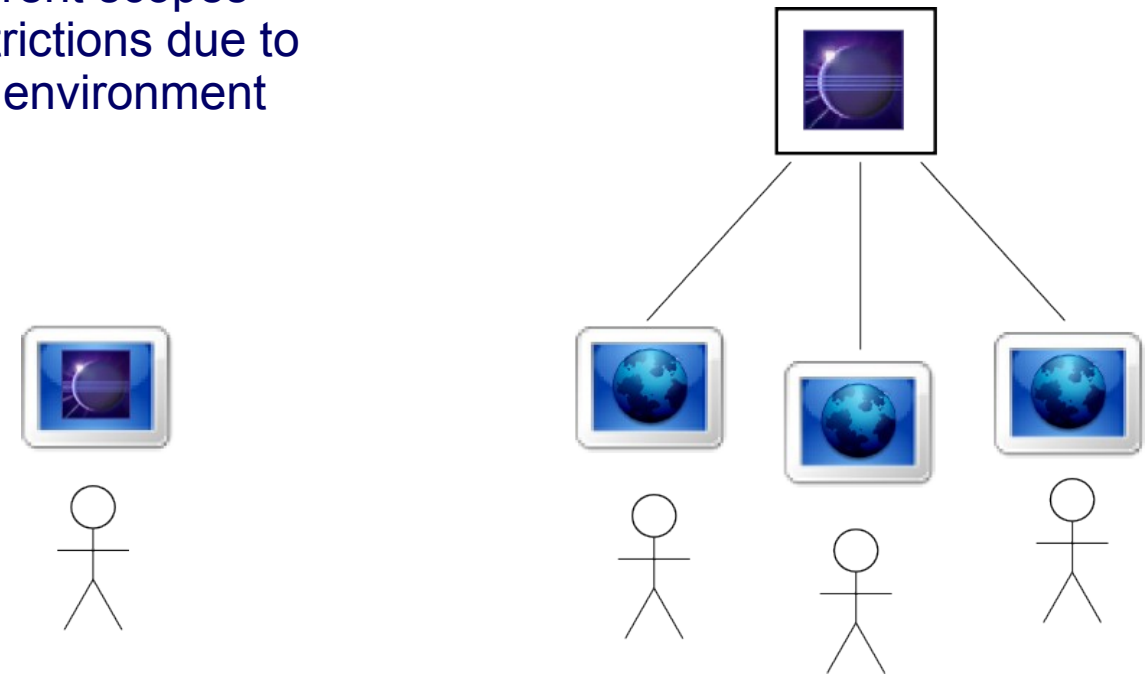


Lab I: The Legendary Mail Template



Differences between RAP and RCP

- Client/Server
- Shared OSGi bundles
- Different scopes
- Restrictions due to web environment



Web Application Support

RAP applications are web applications (distributed environment)

Some additional requirements that are not covered by RCP

Additional API and functionality is provided by the RWT Core:

- Lifecycle management of requests
- API that addresses the different scopes in distributed environments
- Resource-management in case of Javascript libraries
- Service-Handler
- HTTP Session, SessionStore, Request parameters (deep links)
- Shared Images, Fonts, Colors
- Theming and Branding (see later)

Missing / Challenging Features

- Some Events
  Due to the limitations of the distributed environment it will not be possible to provide the same variety of event types as SWT does
  - Modify events behave slightly different, they collect consecutive key-strokes and submit them in chunks
  - Mouse move events not feasible
- Graphics Context
  - Challenge: no high-perfomance drawing engine available for all browsers
  - Research is in progress with different appraces (Flash-based Draw2D, image-based, ...)
- StyledText
  - Extremely complex widget, diff transfer while editing, ...
  - There are efforts in the RAP community

The Interface IEntryPoint

- RAP applications are shared between user sessions
- IEntryPoint defines the startup point of a RAP application instance
- There can be multiple IEntryPoint definitions per applications
  But only one is active per user session
- Users can select them by request parameters

- IEntryPoints are registered via extensions of
  org.eclipse.rap.ui.entrypoint

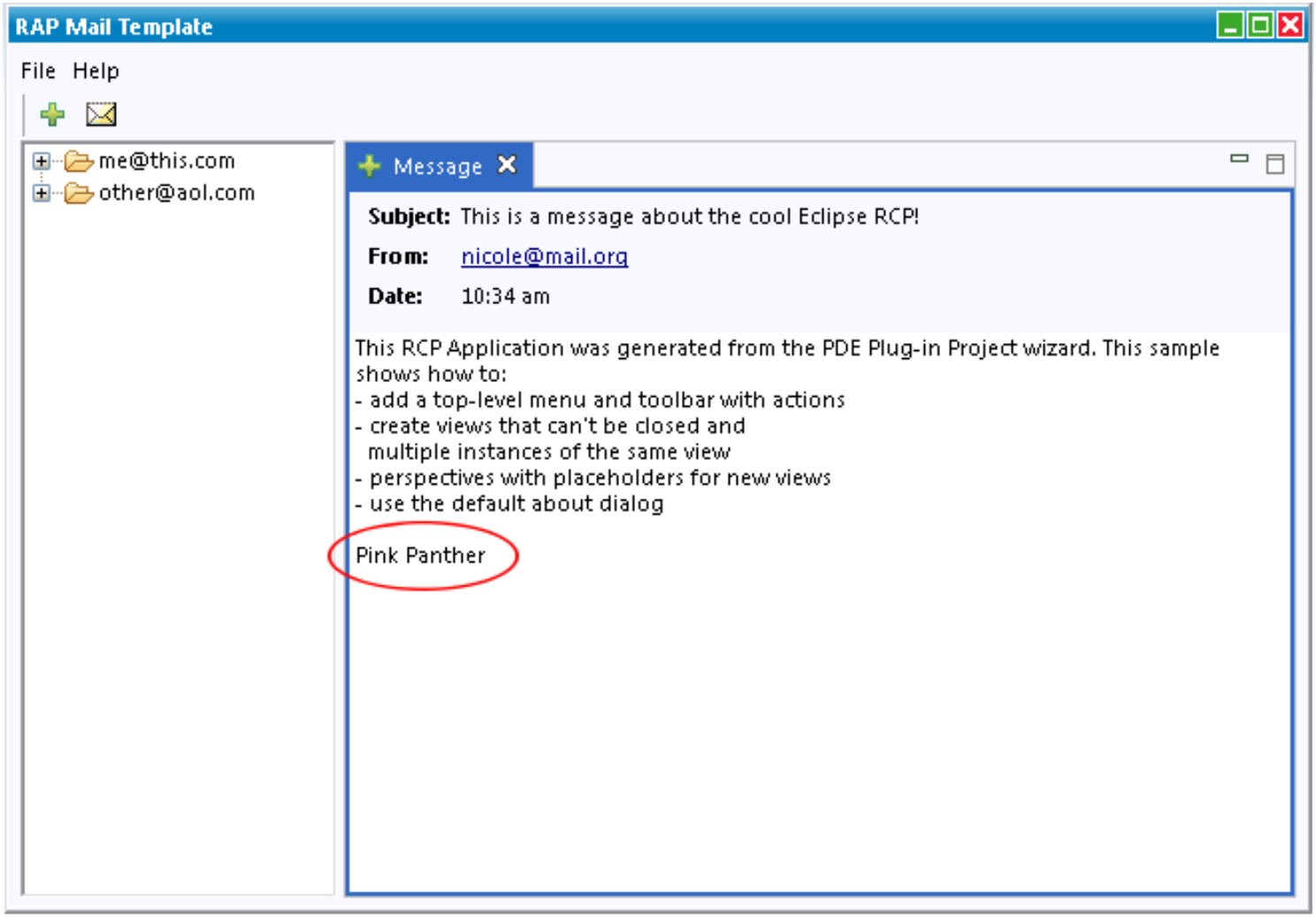
```
<extension point="org.eclipse.rap.ui.entrypoint">
   <entrypoint
         class="org.eclipsecon.demo.Startup"
         parameter="demo"
         id="org.eclipsecon.demo.startup">
   </entrypoint>
</extension>
```

Singletons with Session Scope

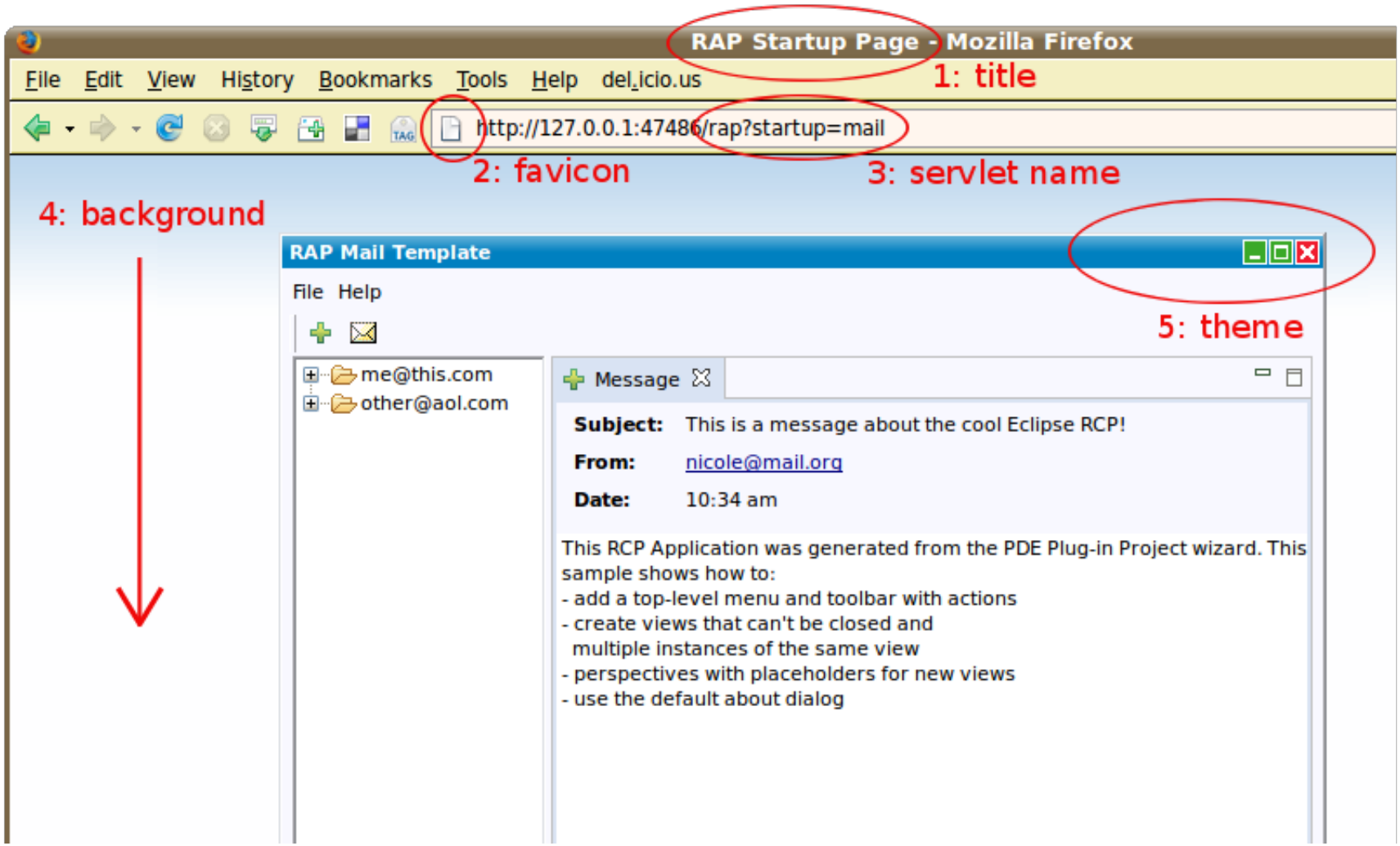
- Singleton pattern widely used
- Implementation of RCP workbench is a classical singleton
- Within RAP this would cause the workbench to run in application scope
- The RAP workbench holds state-information which belongs to the user session
- RWT provides *SessionSingletonBase* to create singletons with session scope easily

```
public class Workbench extends SessionSingletonBase implements IWorkbench {

  private Workbench() {
  }

  public static Workbench getInstance() {
    return ( Workbench )getInstance( Workbench.class );
  }
```

Lab II: Making the Application Session-Aware



Customize Look and Feel



Branding

- RAP provides branding extension point

```
<extension
      point="org.eclipse.rap.ui.branding">
   <branding
         id="org.eclipse.rap.demo.branding1"
         defaultEntrypointId="org.eclipse.rap.demo.entrypoint1"

         title="It&apos;s tea-time"

         favicon="icons/favicon2.ico"

         servletName="tea"

         body="body.html"

         themeId="org.eclipse.rap.demo.alttheme"

         exitConfirmation="Do you really want to leave the party?">
   </branding>
</extension>
```

1. title
2. favicon
3. servlet name
4. background
5. theme

Define and Register a Custom Theme

- RAP themes are simple Java properties file
- Contain definitions for colors, borders, fonts, images, dimensions

```
# button colors
button.foreground: black
button.background: #9dd0ea
button.hover.background: white
...

# button borders
button.border: 1px solid blue
button.BORDER.border: 2px outset
...
```

- Registered with theme extension point

```
<extension
      point="org.eclipse.rap.ui.themes">
   <theme
        id="org.eclipse.rap.demo.alttheme"
        name="Alternative Demo Theme"
        file="theme1/theme.properties"/>
</extension>
```

Widget Variants

- Allow separate styling of certain widgets
- Similar to classes in CSS
- Variants are set in widget user data, single sourcing friendly

SWT Code:

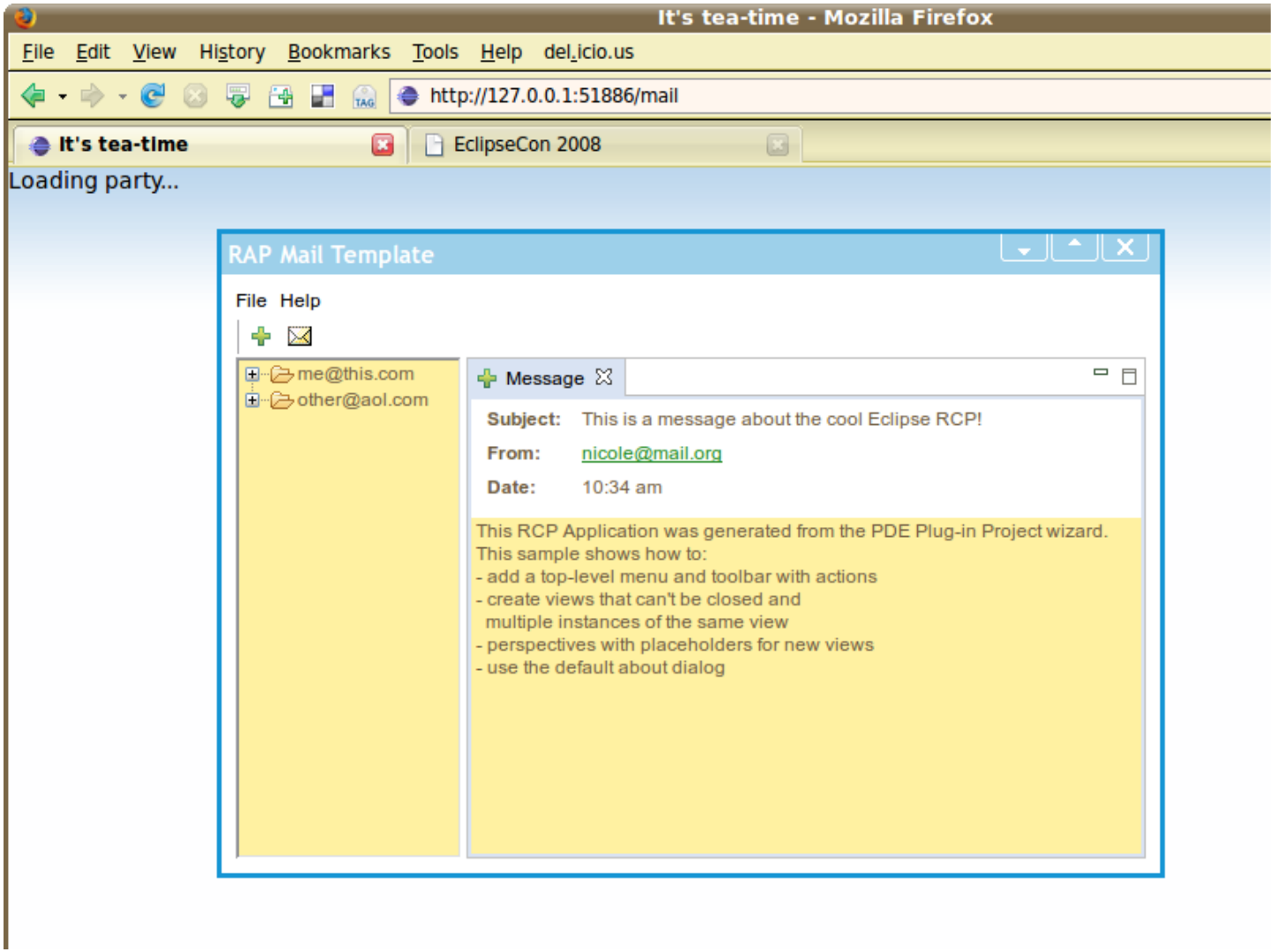
```
button1.setData( WidgetUtil.CUSTOM_VARIANT, "mybutton" );
```

Theme file:

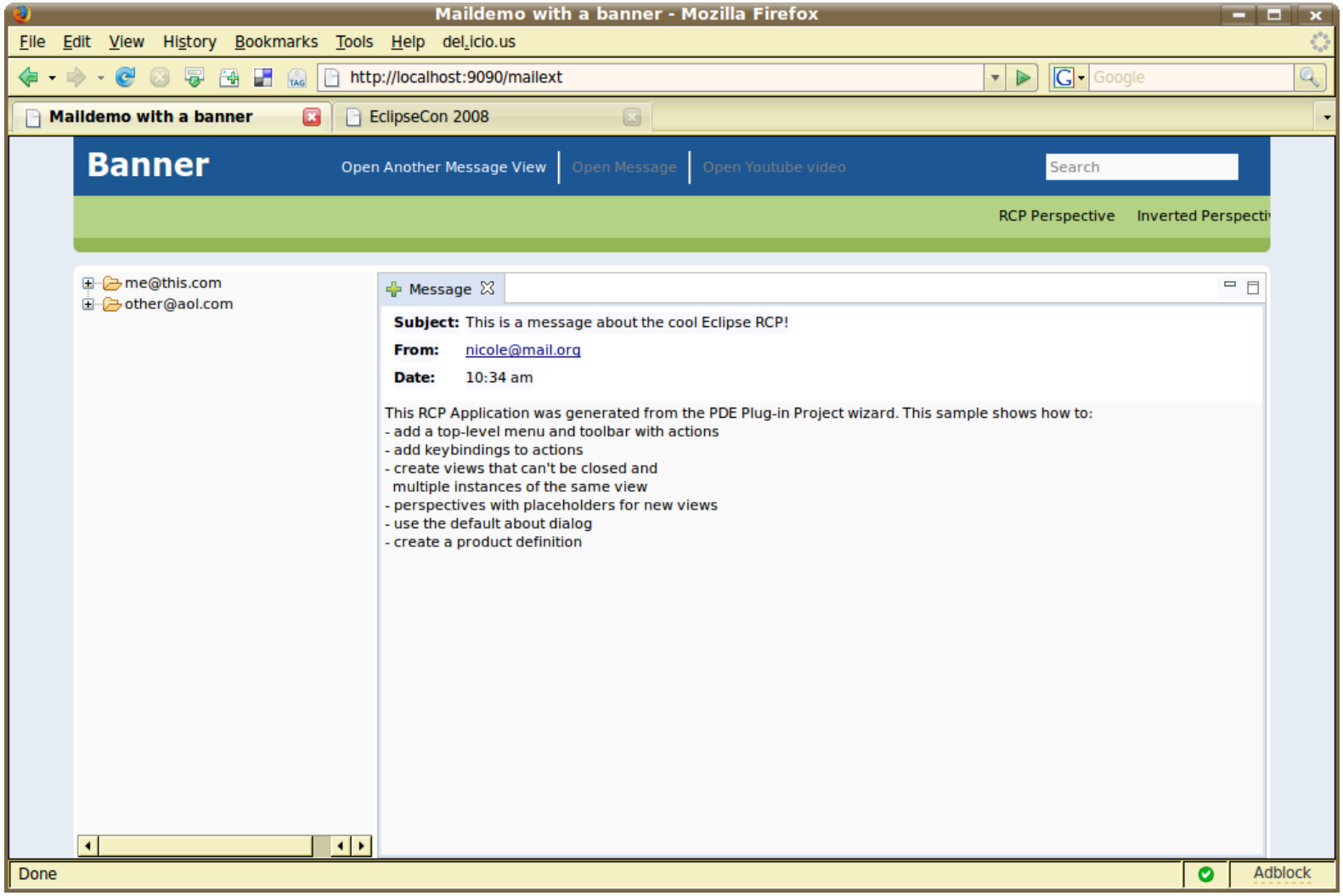
```
mybutton/button.border: 2px #169531
mybutton/button.background: #9dd044
```

Result:

Lab III: Apply Custom Branding and Theme



More Ways to Make it More “Webbish”



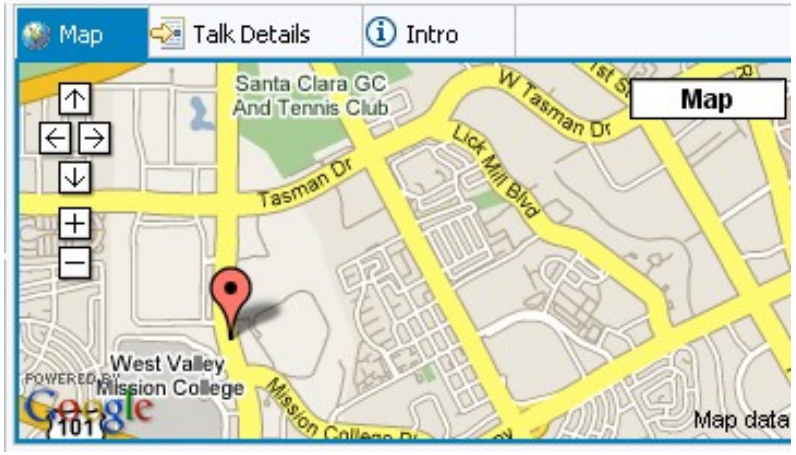
Custom Widgets

Composed Custom Widgets

- Made out of basic widgets like labels and buttons
- No difference to SWT

Native Custom Widgets

- Requires platform knowledge
  - Javascript
  - qooxdoo
  - RWT



Single Sourcing

- Develop desktop and Web clients with a single code base
- RAP provides a good basis for single sourcing
- We strive for best single sourcing support
- Eclipse supports this approach through fragments
- Details: Tutorial about single sourcing strategies

Get in contact with RAP

- Project Home
  http://eclipse.org/rap
- news group
  eclipse.technology.rap
- bugzilla
  https://bugs.eclipse.org/bugs/
- FAQ (work in progress)
  http://wiki.eclipse.org/RapFaq
- CVS / get team project set
  http://eclipse.org/rap/cvs.php
- RAP development plan
  http://wiki.eclipse.org/RapPlan
  - 1.1 Jun 2008 (Ganymede)
  - 1.2 end of Sep 2008