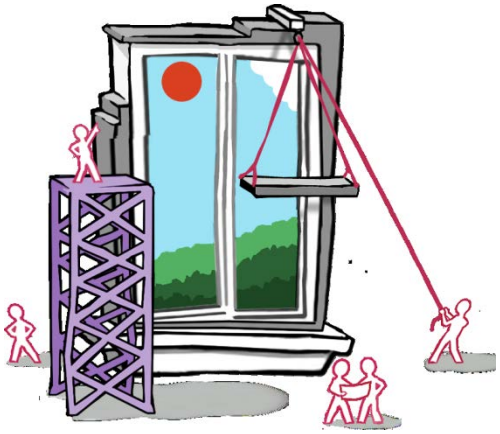


Google™

Building Amazing UIs with WindowBuilder

Eclipse Day at the Googleplex 2012
December 12, 2012

Eric Clayberg
Software Engineering Manager
Google, Inc.

History

Google

WindowBuilder has a very long history spanning multiple technologies and companies

Smalltalk

- 1991 Original release for Smalltalk/V by Cooper & Peters
- 1993 VisualSmalltalk release by ObjectShare
- 1994 VisualAge Smalltalk release by ObjectShare
- 1996 Briefly owned by ParcPlace-Digitalk
- 1997 VisualAge Smalltalk release by Instantiations

Eclipse/Java

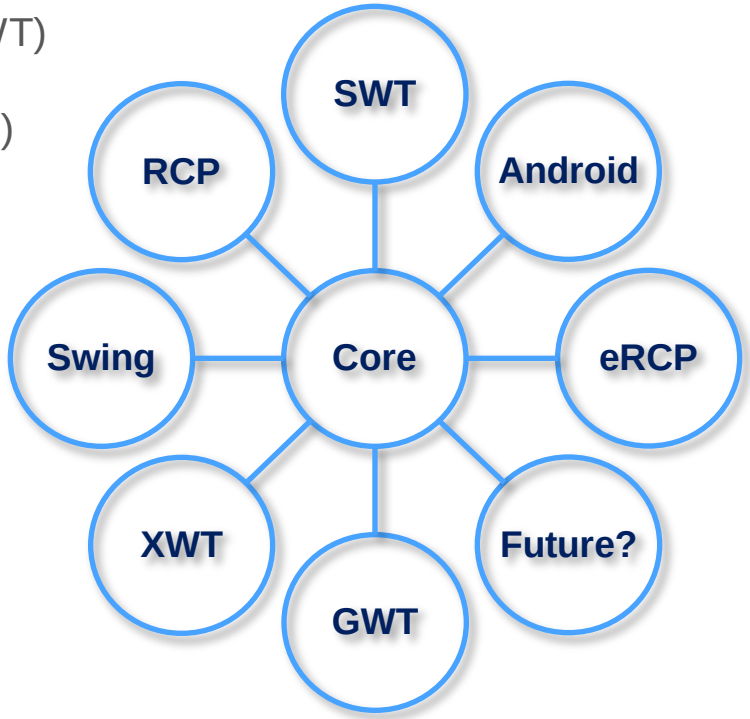
- 2003 New Eclipse/Java version for SWT/RCP (SWT Designer)
- 2004 Swing support added (Swing Designer)
- 2006 Google Web Toolkit (GWT) support added (GWT Designer)
- 2009 Eclipse community award for Best Commercial Add-on
- 2010 Acquired by Google and released free to the world
- 2011+ Contributed to Eclipse.org as new open-source project; Part of Indigo & Juno release trains (Eclipse 3.7, 3.8 & 4.2)

Same Team

Overview

Google

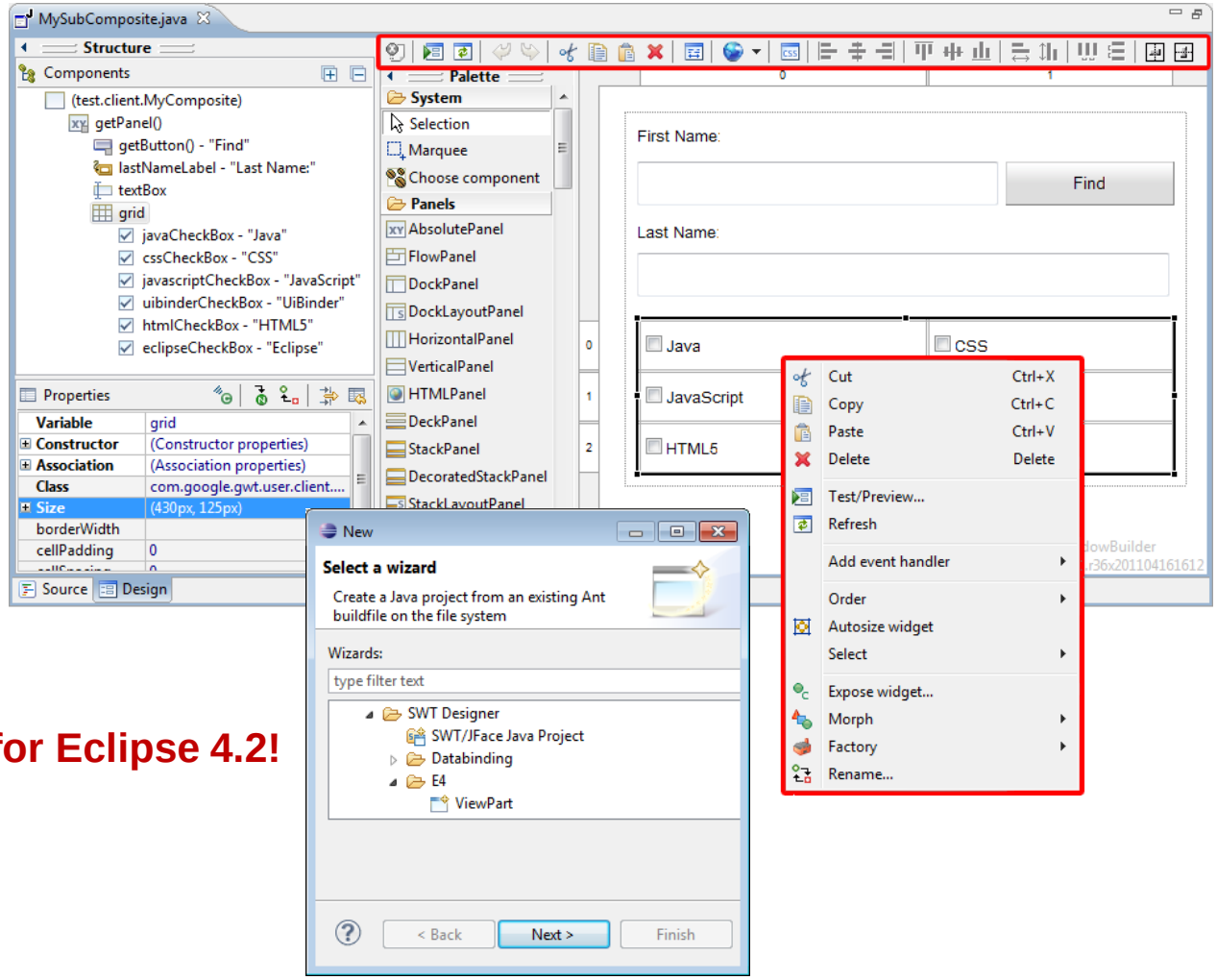
- Available now from http://www.eclipse.org/windowbuilder
- Composed of WindowBuilder Engine, SWT, eRCP, XWT & Swing Designer
- WindowBuilder Engine provides a rich API for creating UI designers
  - Very modular with dozens of extension points
  - Pluggable support for different languages and parsers
  - Java-based UI frameworks (e.g., Swing, SWT/RCP, eRCP, GWT)
  - XML-based UI frameworks (e.g., XWT, GWT UiBinder, Android)
- Exemplary tool examples:
  - SWT Designer
  - Swing Designer
  - eRCP Designer
  - XWT Designer
- 3rd Party Tools
  - JBuilder Swing Designer
  - GWT Designer
  - Android Designer



User Interface

WindowBuilder is composed of the following major components

- Source View
- Design View
- Component Tree
- Property Pane
- Palette
- Wizards
- Toolbars & Context Menus



New e4 ViewPart wizard for Eclipse 4.2!

Features

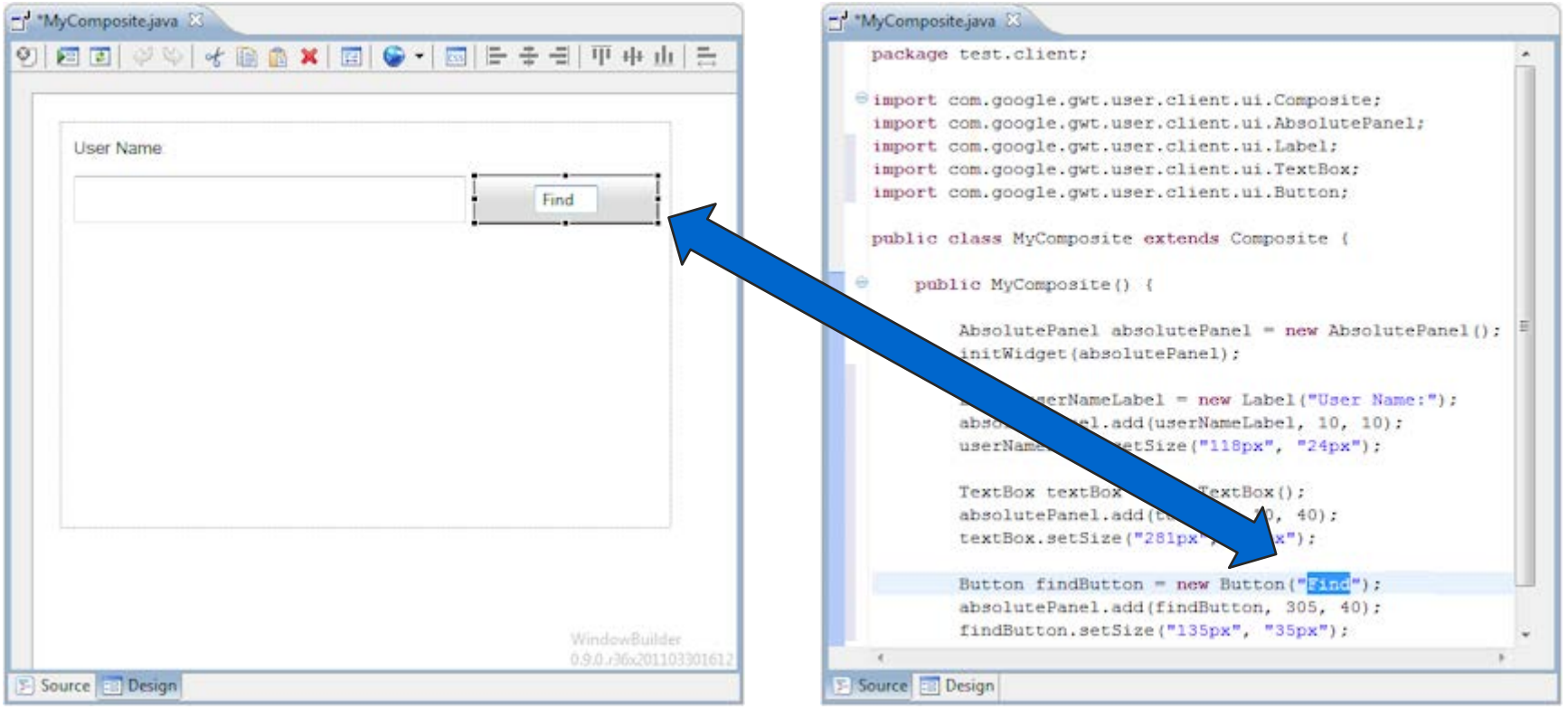
Google

WindowBuilder supports many state-of-the-art features

- WYSIWYG & Bi-directional Code Generation
- Powerful & Flexible Code Parser
- Read & Write Any Format or Style
- Internationalization
- Visual Inheritance
- UI Factories
- Morphing
- Widgets & Layout Managers
- Graphical Menu Editing
- Data Binding

WYSIWYG & Bi-directional Code Generation

- WYSIWYG editing in design view
- Bi-directional Code Generation
- Micro edits result in smallest possible code change



Powerful & Flexible Code Parser

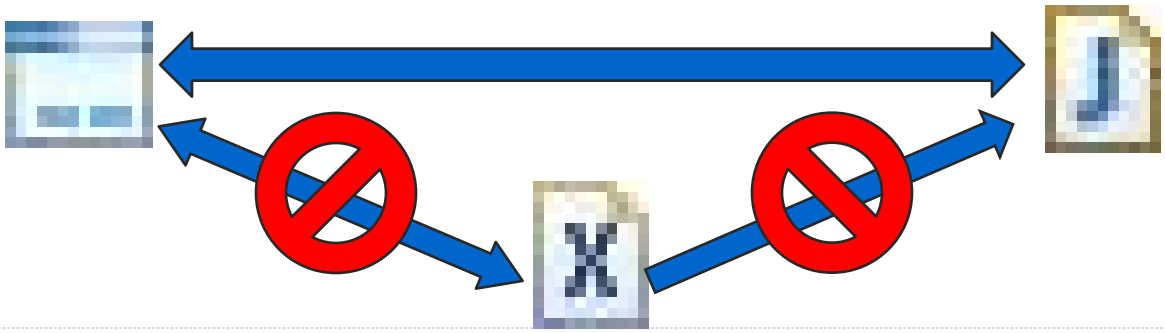
- Can parse its own code and code written by hand
- No protected code blocks
- Understands data flow
- Ignores & preserves non-UI code
- Refactoring friendly and resilient to hand-made changes

*MyComposite.java

```
package test.client;

import com.google.gwt.user.client.ui.Composite;
import com.google.gwt.user.client.ui.AbsolutePanel;
import com.google.gwt.user.client.ui.Label;
import com.google.gwt.user.client.ui.TextBox;
import com.google.gwt.user.client.ui.Button;

public class MyComposite extends Composite {

    public MyComposite() {

        AbsolutePanel absolutePanel = new AbsolutePanel();
        initWidget(absolutePanel);

        Label userNameLabel = new Label("User Name:");
        absolutePanel.add(userNameLabel, 10, 10);
        userNameLabel.setSize("118px", "24px");

        TextBox textBox = new TextBox();
        absolutePanel.add(textBox, 10, 40);
        textBox.setSize("281px", "27px");

        Button findButton = new Button("Find");
        absolutePanel.add(findButton, 305, 40);
        findButton.setSize("135px", "35px");
```

Source | Design

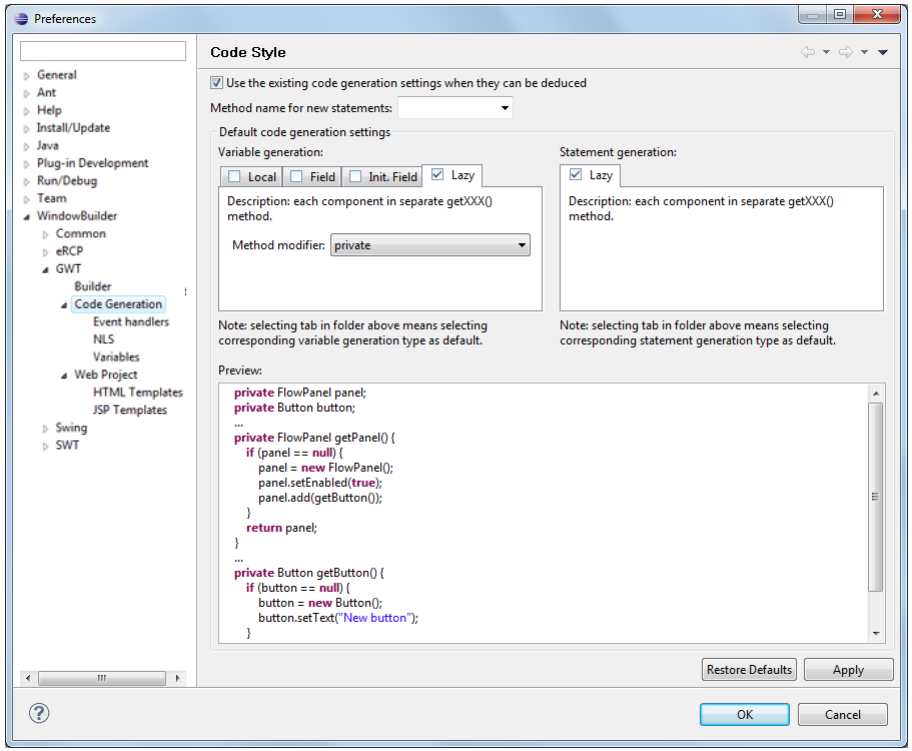
- One-to-one relationship between UI and Java/XML code
- No intermediate metadata file to get lost or out of sync



Read & Write Any Format or Style

Window > Preferences > WindowBuilder > GWT | Swing | SWT > Code Generation

- Local variables vs. Fields
- Flat vs. Block
- Initialized fields
- Lazy declaration




```
...
FlowPanel panel = new FlowPanel();
panel.setEnabled(true);

Button button = new Button();
button.setText("Add customer...");
panel.add(button);
...
```

```
private FlowPanel panel;
private Button button;
...
private FlowPanel getPanel() {
   if (panel == null) {
      panel = new FlowPanel();
      panel.setEnabled(true);
      panel.add(getButton());
   }
   return panel;
}
...
private Button getButton() {
   if (button == null) {
      button = new Button();
      button.setText("New button");
   }
   return button;
}
```

```
...
{
   FlowPanel panel = new FlowPanel();
   panel.setEnabled(true);
   {
      Button button = new Button();
      button.setText("Add customer...");
      panel.add(button);
   }
   ...
}
...
```

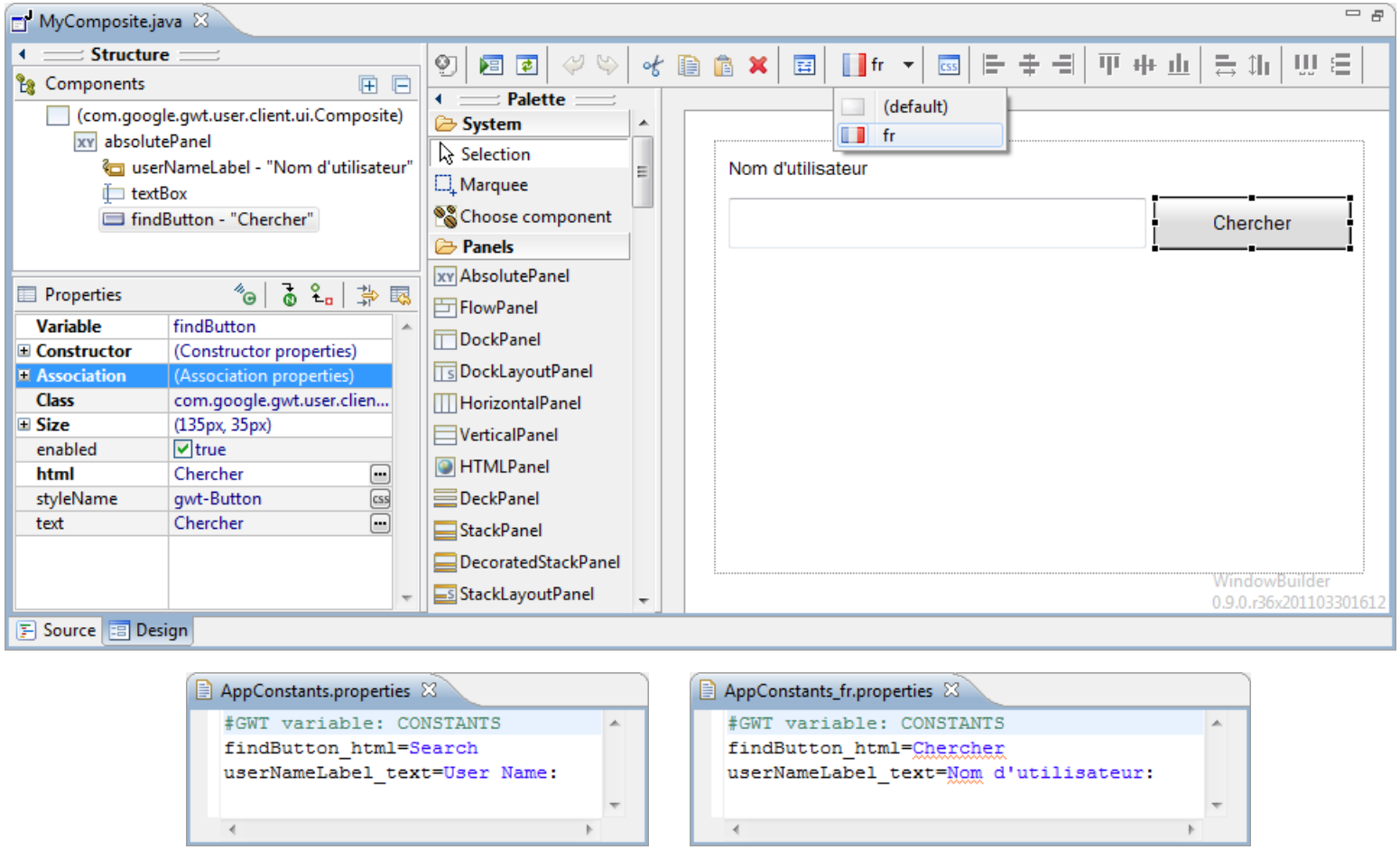
```
private final FlowPanel panel = new FlowPanel();
private final Button button = new Button();
...
panel.setEnabled(true);

panel.add(button);
button.setText("Add customer...");
...
```

```
private FlowPanel panel;
private Button button;
...
panel = new FlowPanel();
panel.setEnabled(true);

button = new Button();
button.setText("Add customer...");
panel.add(button);
...
```

Internationalization

Google

Offers easy-to-use Internationalization and Localization tools

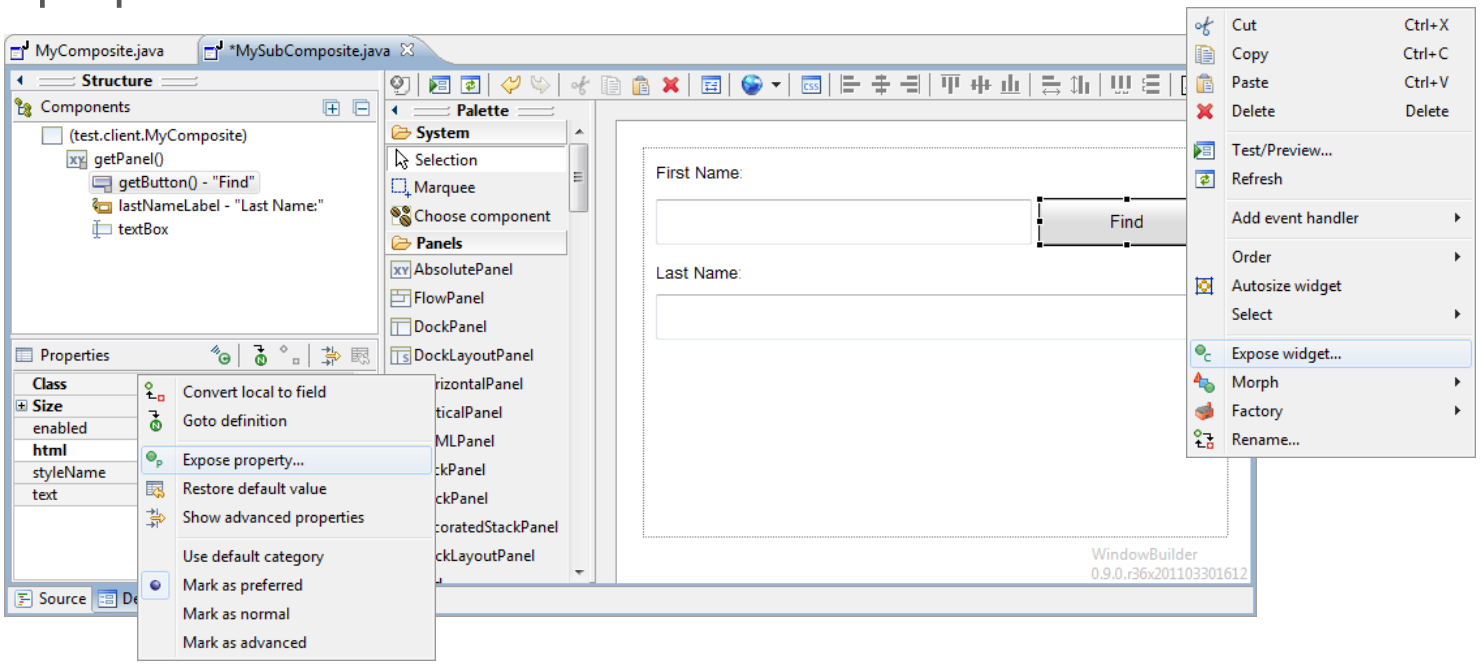


Visual Inheritance

Google

Provides visual inheritance so that code features can be easily inherited from a parent – child hierarchy

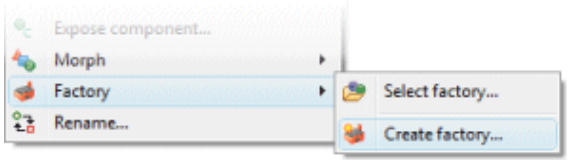
- Easily expose fields and properties
- Add components & event handlers to inherited fields
- Change public properties of inherited components
- Change properties of inherited fields

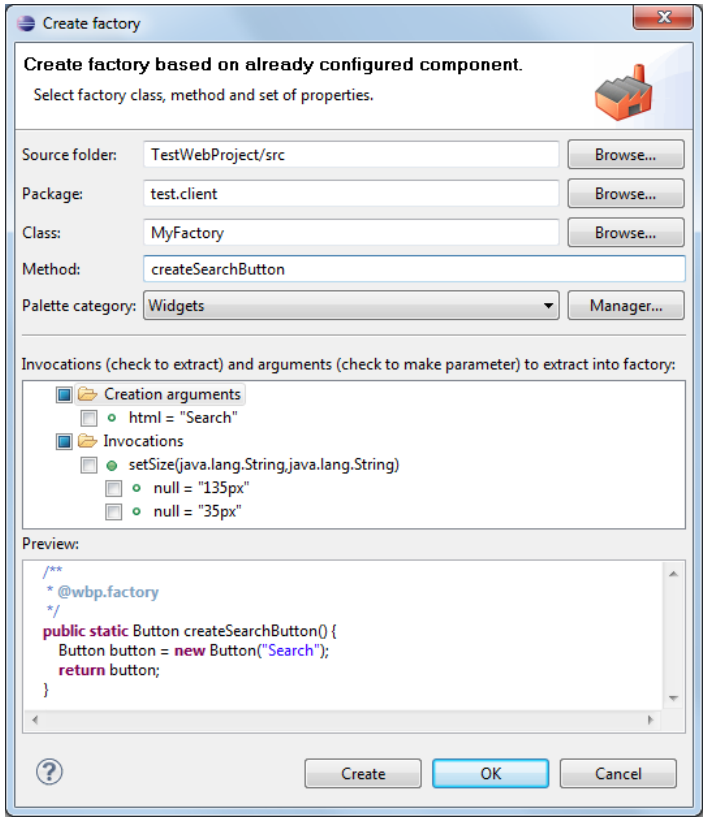


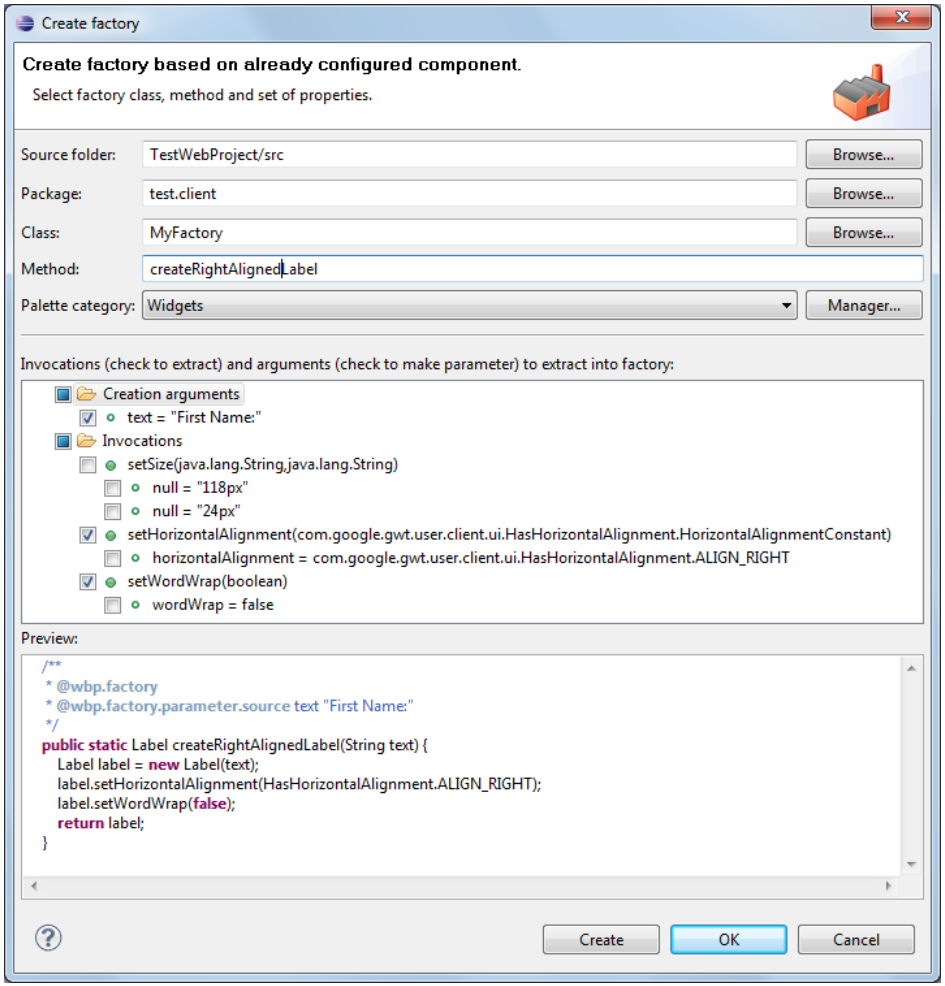
UI Factories

Google

Support for UI Factories and reusable customized GUI elements



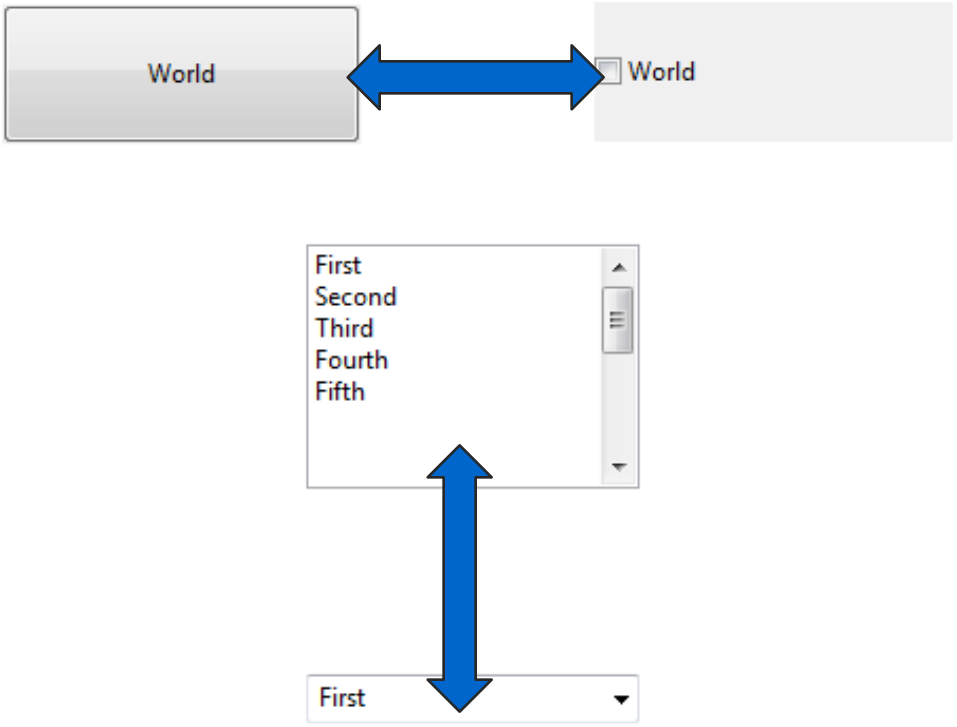




Morphing

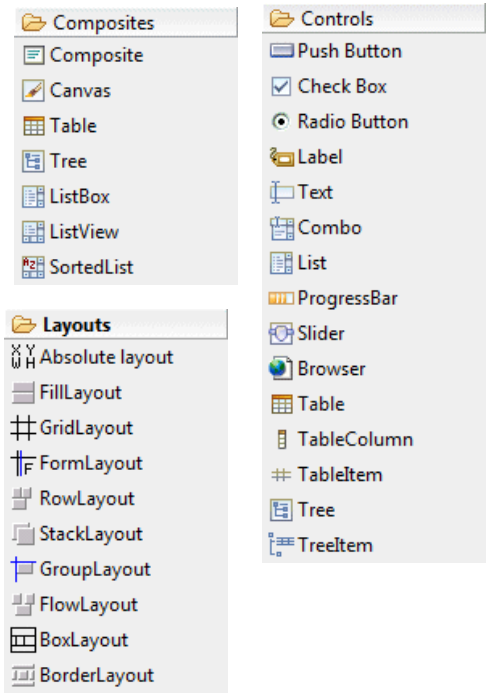
Google

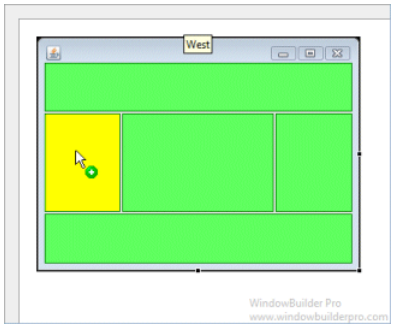
Provides a Morphing tool to easily change one widget type into another

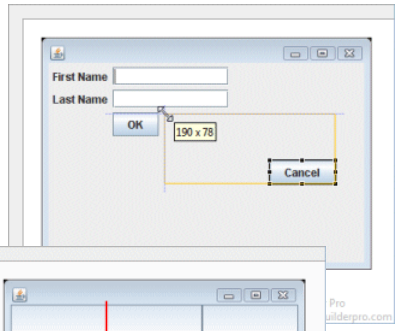


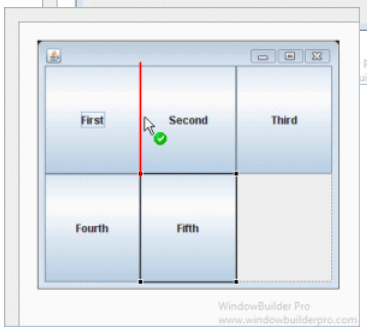
Widgets & Layout Managers

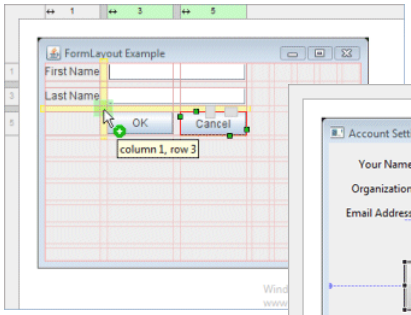
Fully supports all standard widgets and layout managers as well as select third-party widgets and layout managers

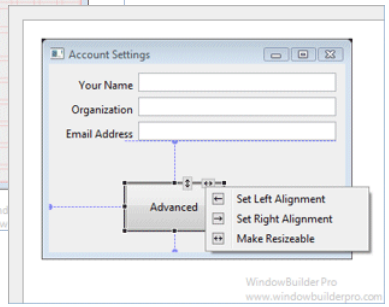


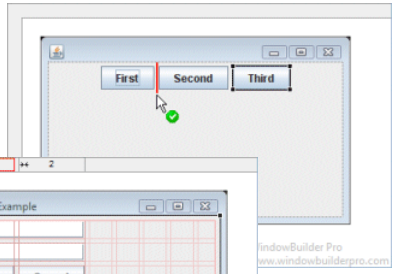


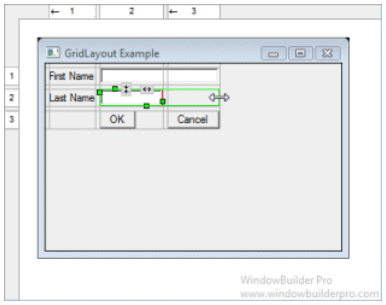


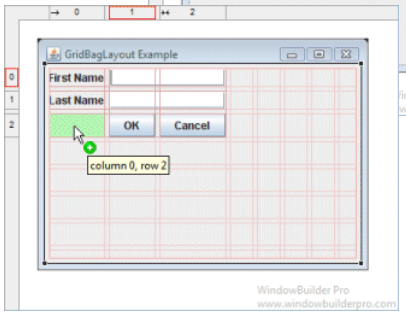








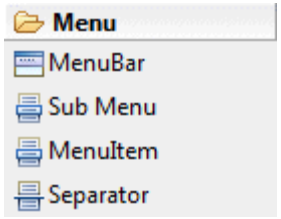


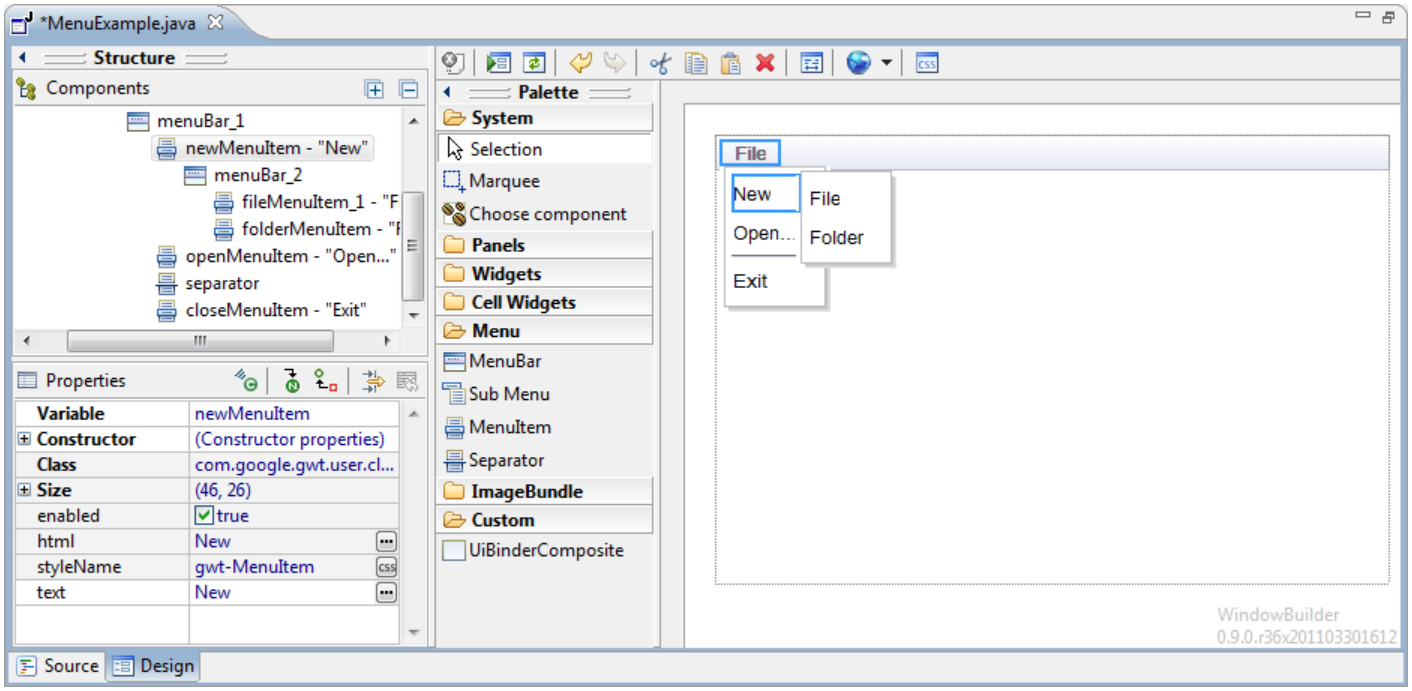


Graphical Menu Editing

Supports WYSIWYG Graphical Menu Editing

- Graphical edit menubars and menuitems
- Use drag/drop to rearrange menus
- Direct edit menu labels

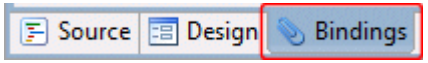




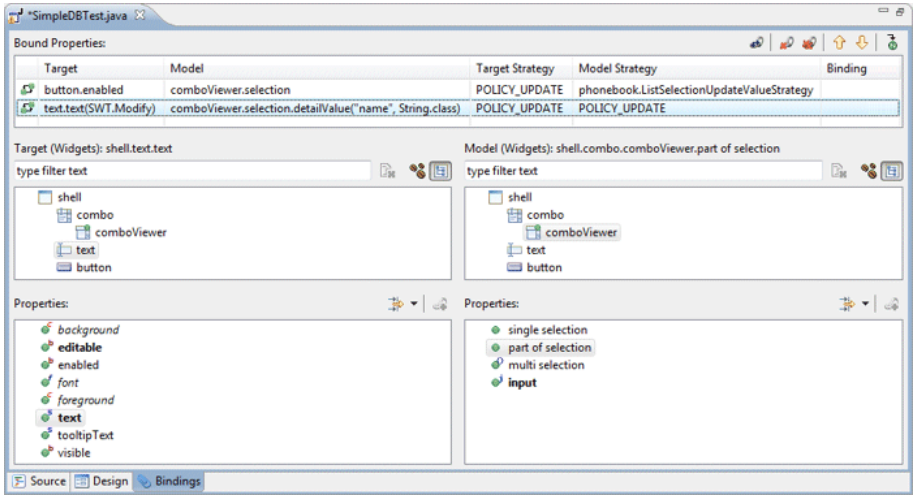
Data Binding

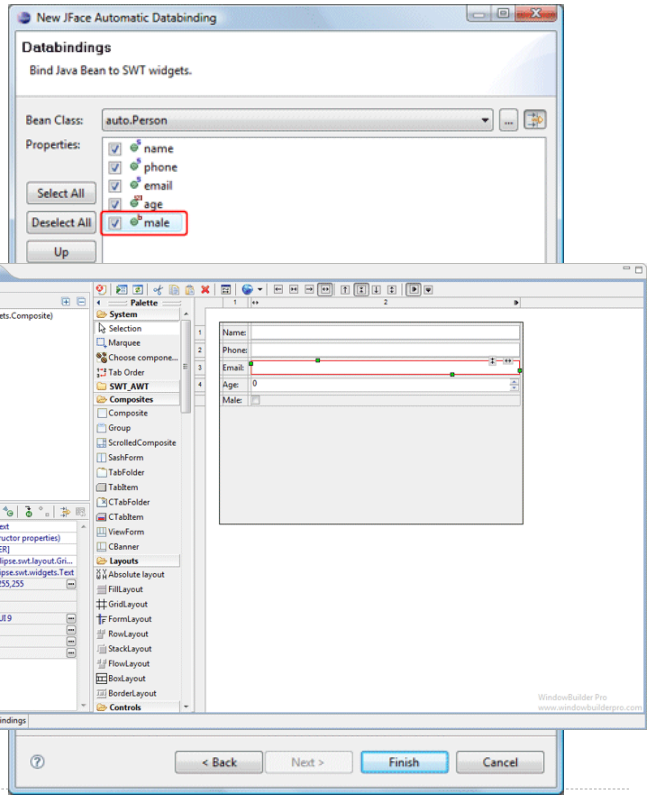
Google

Supports binding UI elements and model elements



- No need to write and register listeners
- Common infrastructure for validation and conversion
- Connect data sources to widgets such as text fields, combos, tables and trees, for viewing and editing
- Manual and automatic tools provided

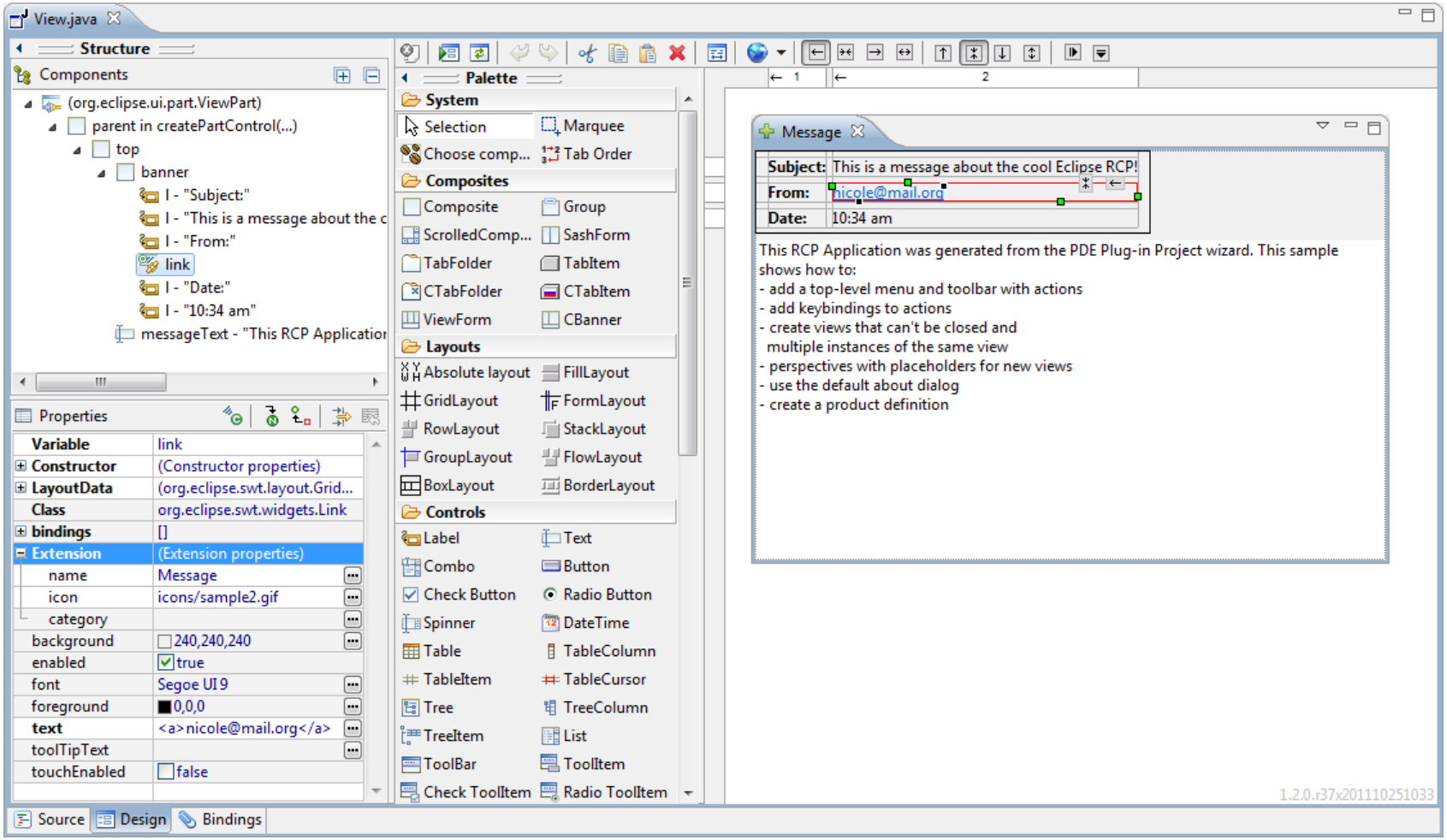




Gallery – SWT Designer

Google

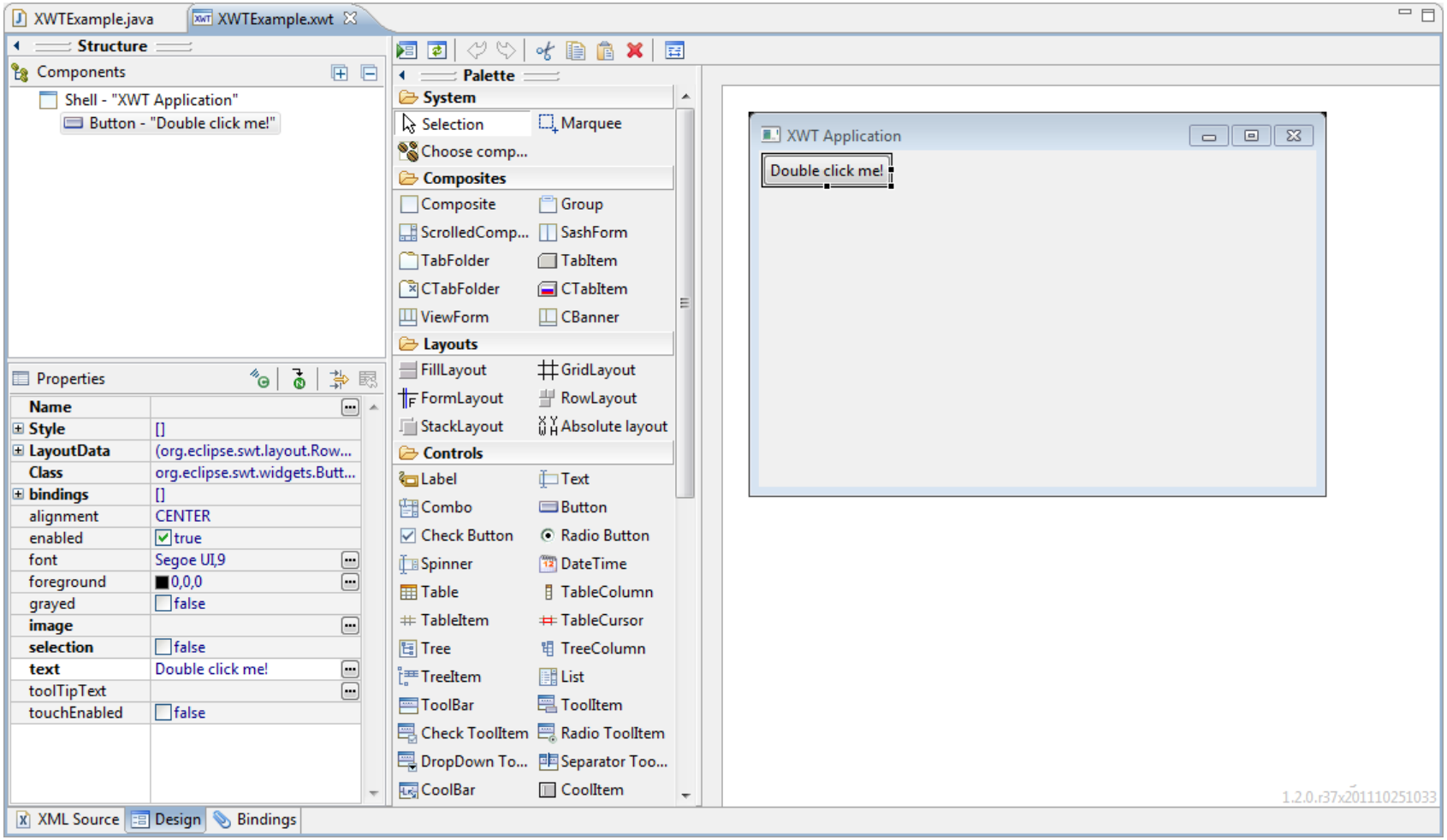
http://eclipse.org/windowbuilder/



Gallery – XWT Designer

Google

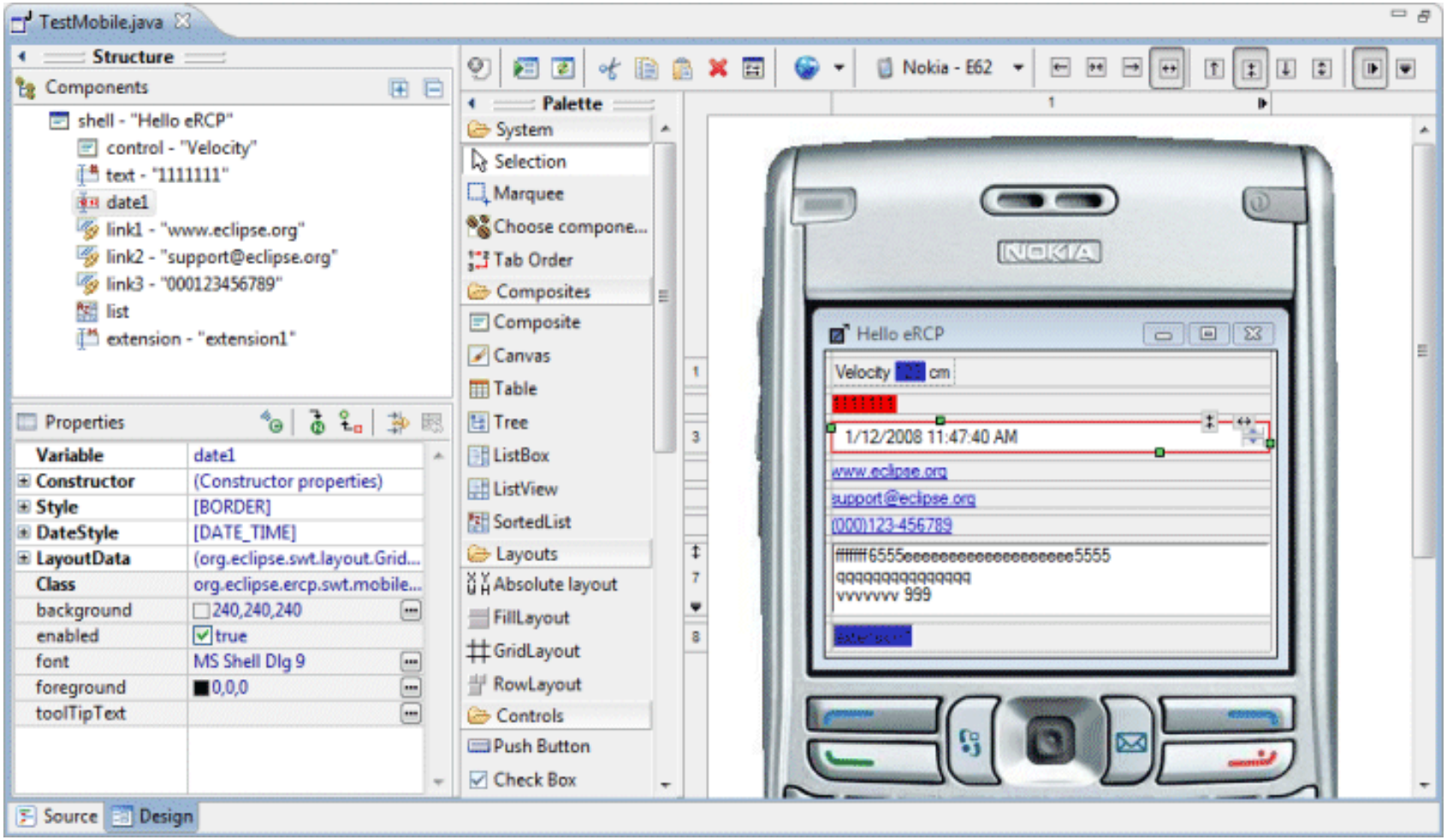
http://eclipse.org/windowbuilder/



Gallery – eRCP Designer

Google

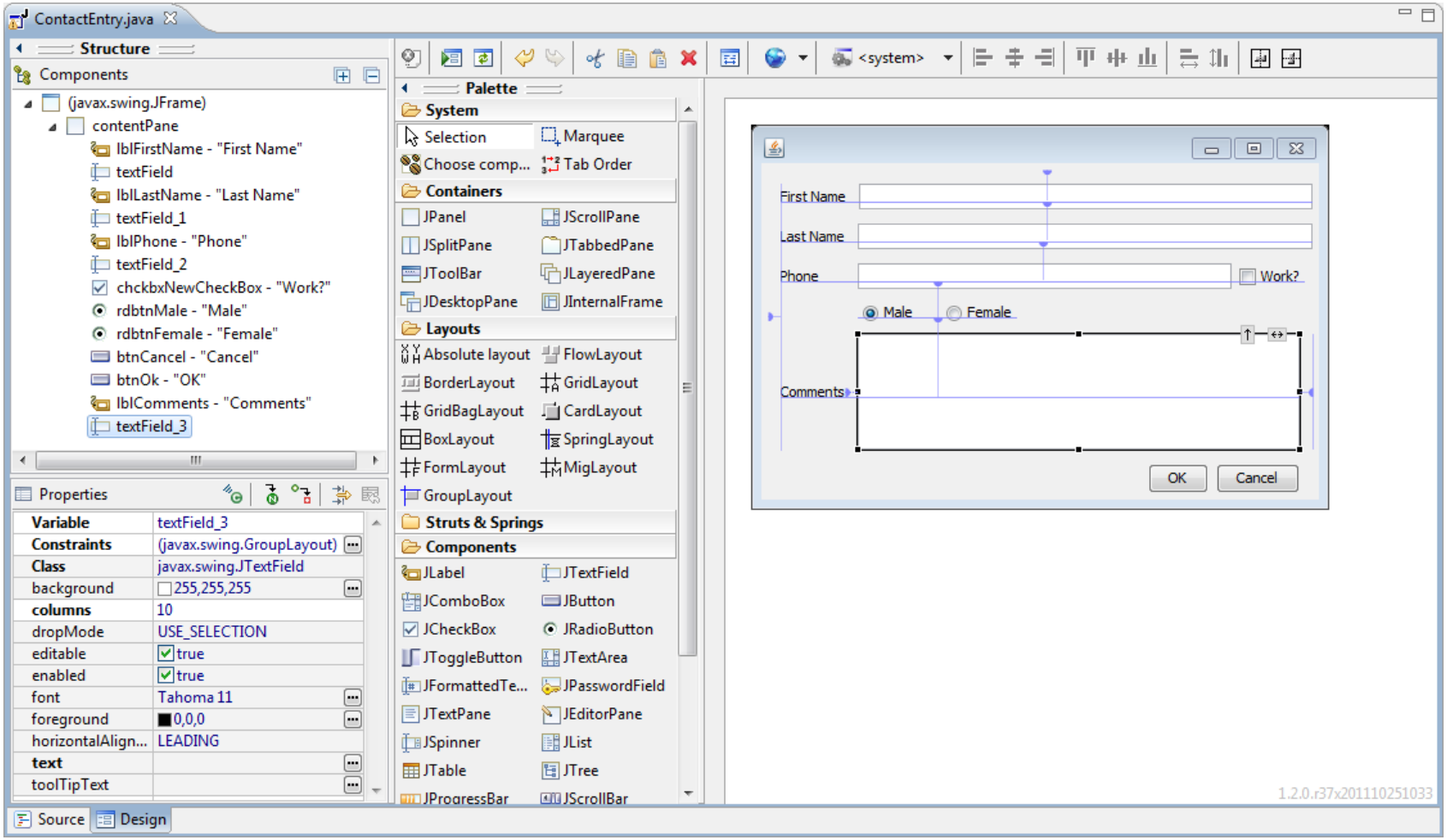
http://eclipse.org/windowbuilder/



Gallery – Swing Designer

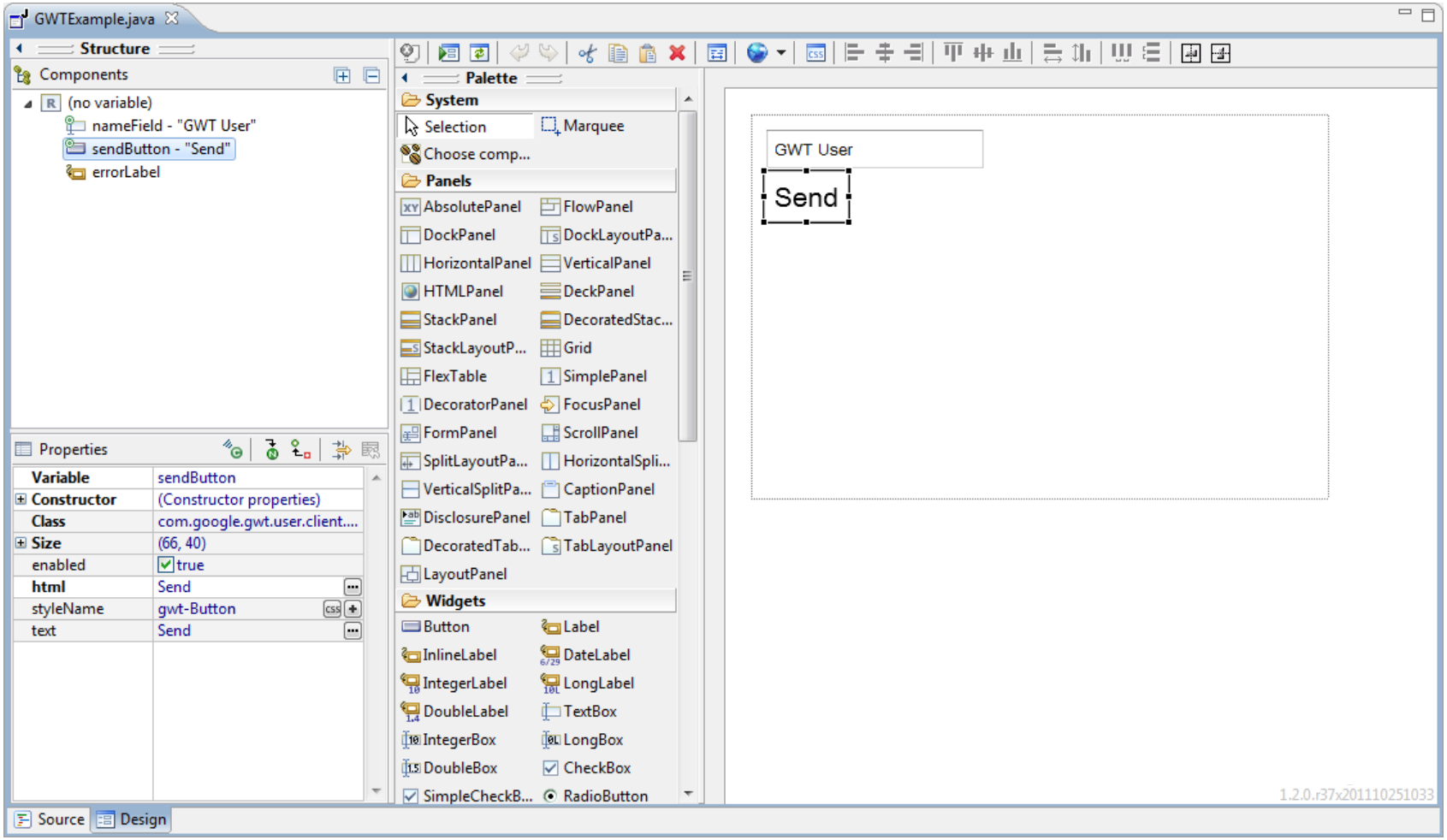
Google

http://eclipse.org/windowbuilder/



Gallery – GWT Designer

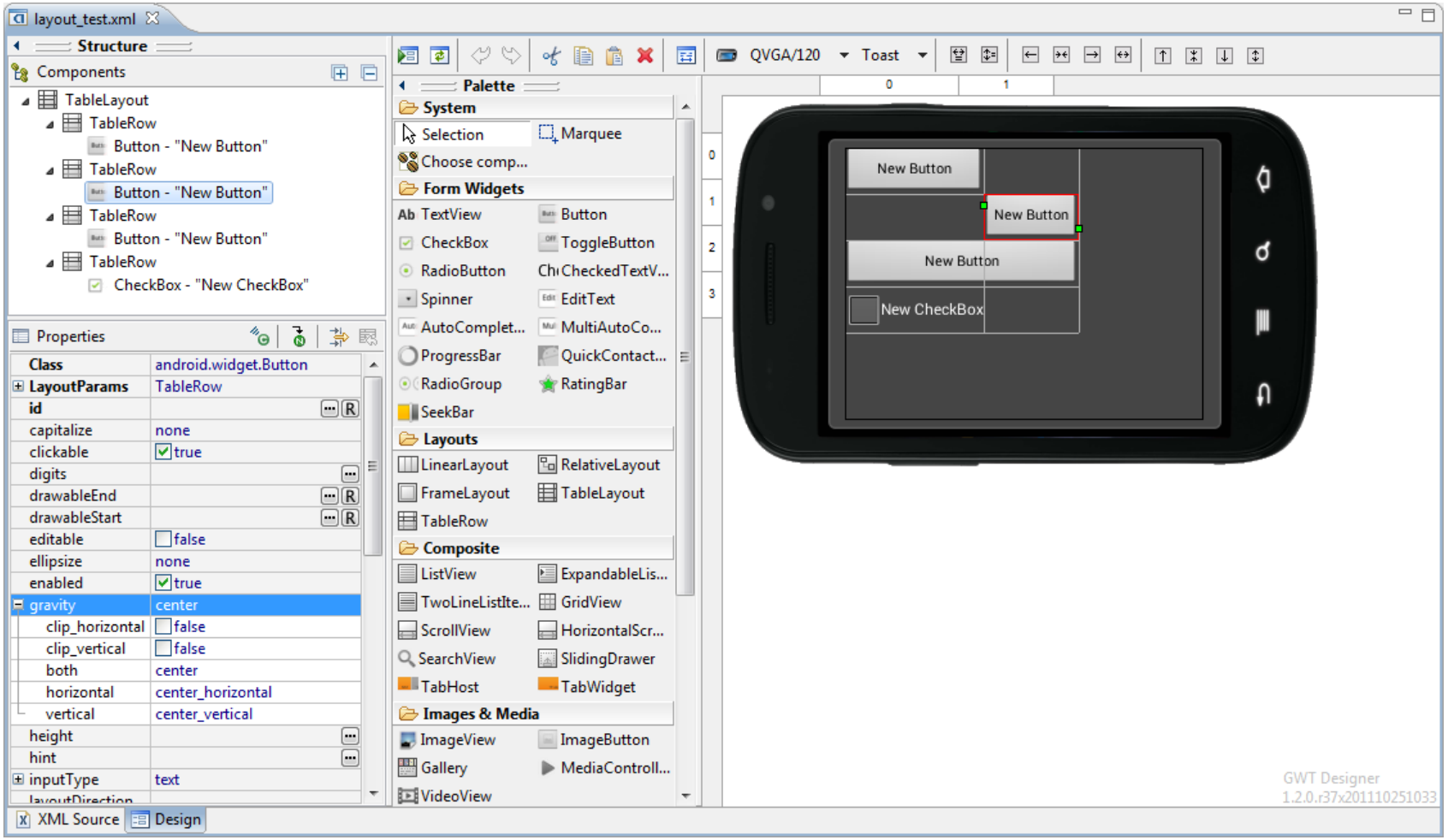
http://code.google.com/p/gwt-designer/



Gallery – Android Designer

Google

http://code.google.com/a/eclipselabs.org/p/android-designer



Get More Info

Where to get it:

http://eclipse.org/windowbuilder/download.php
https://developers.google.com/java-dev-tools/download
https://developers.google.com/web-toolkit/tools/download-gwtdesigner
http://code.google.com/a/eclipselabs.org/p/android-designer/downloads

Documentation:

https://developers.google.com/java-dev-tools/wbpro/

Issue tracker, source:

https://bugs.eclipse.org/bugs/ → Tools > WindowBuilder
http://dev.eclipse.org/svnroot/tools/org.eclipse.windowbuilder
https://svn.codespot.com/a/eclipselabs.org/windowbuilder-extras/trunk

Forum:

http://eclipse.org/forums/index.php?t=thread&frm_id=214