

Design Fragments Make Framework Use Easier

Motivation

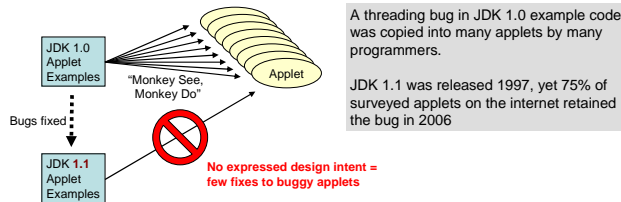
Frameworks like EJB, .Net, and Swing are large-grained reuse components that reduce development risk. Developers "fill in the blanks" to create applications. But frameworks:

- Are hard to learn
- Require coordination of non-local parts
- Constrain the solution space
- Involve heavy bureaucracy

Design fragments make framework use easier by:

- Expressing previously **implicit framework constraints**
- Expressing previously **implicit programmer design intent**
- Encoding **known-good solutions**
- Ensuring **conformance**

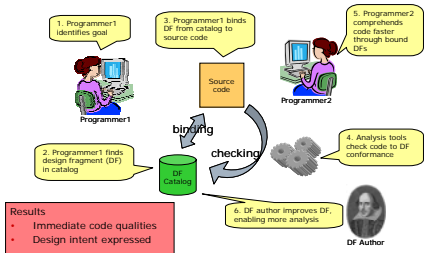
Problems with Implicit Design Intent



A threading bug in JDK 1.0 example code was copied into many applets by many programmers.

JDK 1.1 was released 1997, yet 75% of surveyed applets on the internet retained the bug in 2006

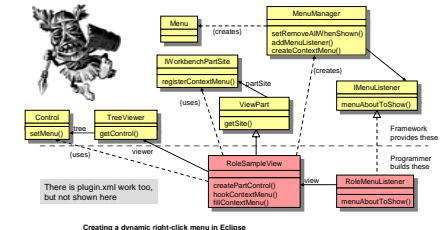
Design Fragments in the Software Lifecycle



What's in a Design Fragment?

Design Fragment: On-Demand Right Click Menu in the Eclipse Framework

Design Intent: This applet listens to mouse events, registering and deregistering in framework lifecycle callback methods

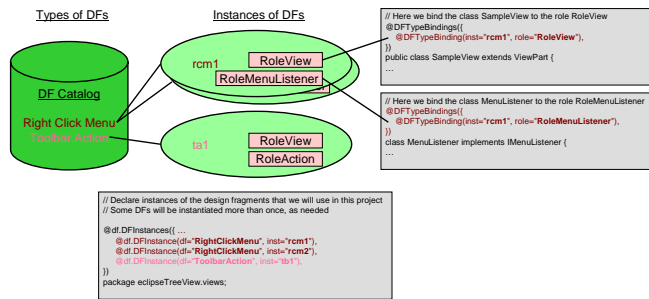


Creating a dynamic right-click menu in Eclipse

Constraints not expressible in UML, above:

- Fully qualified class name must appear in plugin.xml, at:
 - "xpath:///plugin/extension[@point='org.eclipse.ui.views']/view/@class"
- In plugin.xml, the following two IDs must match exactly:
 - p1="xpath:///plugin/extension[@point='org.eclipse.ui.perspectiveExtensions']/view/@id"
 - p2="xpath:///plugin/extension[@point='org.eclipse.ui.views']/view/@id" />
- In menuAboutToShow() in MenuItemListener
 - must call fillContextMenu(manager) on SampleView
- In createPartControl() in SampleView,
 - must create a new instance of DrillDownAdapter
 - must call hookRightClickAction on this
- In hookRightClickAction(),
 - must call addNavigationActions(manager) on drillDownAdapter
 - must call add(action) on manager

How Are Design Fragments Bound to the Code?



A Design Fragment Slice Through a Program

```

File: package-info.java
// Declare instances of the design fragments that we will use in this project
@df.DFInstances({
  @df.DFInstance(id="RightClickMenu", inst="rcm1"),
  ...
})
package eclipse.TreeView.views;

File: SampleView.java
package eclipse.TreeView.views;

@DFTypeBinding({ ...
  @DFTypeBinding(inst="rcm1", role="RoleView"),
  ...
})
public class SampleView extends ViewPart {
  ...
}

File: MenuItemListener.java
package eclipse.TreeView.views;

@DFTypeBinding({
  @DFTypeBinding(inst="rcm1", role="RoleMenuItemListener"),
})
class MenuItemListener implements MenuItemListener {
  ...
}

File: plugin.xml
<?xml version="1.0" encoding="UTF-8"?>
<?eclipse version="3.0"?>
<!--
  point="org.eclipse.ui.views">
  <category
    name="Sample Category"
    id="Test" />
  <view
    name="Sample View"
    icon="icons/sample.gif"
    category="Test"
    class="eclipse.TreeView.views.SampleView"
    id="eclipse.TreeView.views.SampleView">
  </view>
  </extension>
  <extension
    point="org.eclipse.ui.perspectiveExtensions">
    <perspectiveExtension
      targetId="org.eclipse.ui.resource.Perspective">
    <view
      ratio="0.5"
      relative="org.eclipse.ui.views.TaskList"
      relationship="right"
      id="eclipse.TreeView.views.SampleView">
    </view>
  </perspectiveExtension>
  </extension>
</!--
  
```

Ensuring Conformance with Tools

