Component-Based Software Engineering

Introduction to Java Beans
Paul Krause
Lecture 7 -
Java Beans
Lecture 7 - Java Beans

Contents
Lecture 7 - Java Beans

Contents

- Definition
Lecture 7 - Java Beans

Contents

- Definition
- Bean Basics
What is a Java Bean?

“A Java Bean is a reusable software component that can be manipulated visually in a builder tool”

JavaSoft
Sources of Builder Tools
Sources of Builder Tools

- Bean Builder from Sun:
Sources of Builder Tools

- Bean Builder from Sun:
  http://java.sun.com/products/javabeans/beanbuilder/
Sources of Builder Tools

- Bean Builder from Sun:
  http://java.sun.com/products/javabeans/beanbuilder/

- NetBeans:
Sources of Builder Tools

- Bean Builder from Sun:
  http://java.sun.com/products/javabeans/beanbuilder/

- NetBeans:
  http://www.netbeans.org
Sources of Builder Tools

- Bean Builder from Sun: http://java.sun.com/products/javabeans/beanbuilder/
- NetBeans: http://www.netbeans.org
- JBuilder:
Sources of Builder Tools

- Bean Builder from Sun:

- NetBeans:
  [http://www.netbeans.org](http://www.netbeans.org)

- JBuilder:
Sources of Builder Tools

- Bean Builder from Sun: http://java.sun.com/products/javabeans/beanbuilder/
- NetBeans: http://www.netbeans.org
- JBuilder: http://www.borland.com/jbuilder/
Sources of Builder Tools

- Bean Builder from Sun:
  [Java Beans beanbuilder](http://java.sun.com/products/javabeans/beanbuilder/)

- NetBeans:
  [NetBeans](http://www.netbeans.org)

- JBuilder:
  [JBuilder](http://www.borland.com/jbuilder/)

- Note, you also need to have the Java SDK installed.
```java
public ColorSwitcher() {
    initComponents();
}

/** This method is called from within the constructor to
 * initialize the form.
 * WARNING: Do NOT modify this code. The content of this method is
 * always regenerated by the Form Editor. */

private void initComponents() {
    jButton1 = new javax.swing.JButton();

    getContentPane().setLayout(new java.awt.GridBagLayout());

    setDefaultCloseOperation(javax.swing.WindowConstants.EXIT_ON_CLOSE);
    jButton1.setText("jButton1");
    getContentPane().add(jButton1, new java.awt.GridBagConstraints());
    pack();
}

/**
 * @param args the command line arguments
 */

public static void main(String args[]) {
    java.awt.EventQueue.invokeLater(new Runnable() {
        public void run() {
            new ColorSwitcher().setVisible(true);
        }
    });
}

// Variables declaration - do not modify
private javax.swing.JButton jButton1;
// End of variables declaration
```
<table>
<thead>
<tr>
<th>Title</th>
<th>Author</th>
<th>Publisher</th>
<th>Recommended</th>
</tr>
</thead>
<tbody>
<tr>
<td>Developing Java Beans</td>
<td>Robert Englander</td>
<td>O’Reilly</td>
<td>Recommended</td>
</tr>
<tr>
<td>JavaBeans by Example</td>
<td>Henri Jubin</td>
<td>Prentice Hall</td>
<td>Simpler than the above, but with useful examples</td>
</tr>
<tr>
<td>NetBeans – the Definitive Guide</td>
<td>Boudreau, Glick, Greene, Spurlin, Woehr</td>
<td>O’Reilly</td>
<td>Not so good on Beans. More about the IDE.</td>
</tr>
</tbody>
</table>
Lecture 2 - Java Beans
Lecture 2 - Java Beans

Contents

- Definition
Lecture 2 - 
Java Beans

Contents

- Definition
- Bean Basics
The Component Model
The Component Model

- Discovery and Registration
The Component Model

- Discovery and Registration
- Raising and Handling of Events
The Component Model

- Discovery and Registration
- Raising and Handling of Events
- Persistence
The Component Model

- Discovery and Registration
- Raising and Handling of Events
- Persistence
- Visual Presentation
The Component Model

- Discovery and Registration
- Raising and Handling of Events
- Persistence
- Visual Presentation
- Support for Visual Programming
Properties, Methods & Events

By following specific naming conventions, the properties of a Bean that are “revealed” to the world can be identified:

```java
public class SignalGenerator {
    private double frequency;
    // ... constructor methods here
    public double getFrequency() {
        return frequency;
    }
    public void setFrequency(double frequency) {
        this.frequency = frequency;
    }
}
```
Conventions for Access Methods
Conventions for Access Methods

- Simple Properties:
  - For a property of type Type and name Name:
  - public Type getName();
  - public void setName(Type value);
Conventions for Access Methods

- Simple Properties:
  - For a property of type `Type` and name `Name`:
    - `public Type getName();`
    - `public void setName(Type value);`

- Boolean Properties:
  - `public boolean boolean isName();`
  - `public void setName(boolean boolean value);`
Bean Methods
Bean Methods

- A Bean may be implemented by a Java Class
Bean Methods

- A Bean may be implemented by a Java Class
- That Class contains a number of methods that may be used to access and control the component
Bean Methods

- A Bean may be implemented by a Java Class
- That Class contains a number of methods that may be used to access and control the component
- These are generally all the public methods of the Class that implements the Bean
Events
Events

- Bean components interact by generating “Events”
Events

- Bean components interact by generating “Events”
- Several components may register an interest in an Event that is generated by a specific component
Events

- Bean components interact by generating "Events"
- Several components may register an interest in an Event that is generated by a specific component
- Occurrence of the Event triggers methods to be called in all the components that are "listening" for it
Introspection
Introspection

- Exposes the properties, methods and events supported by a Java Bean
Introspection

➢ Exposes the properties, methods and events supported by a Java Bean

➢ Low-level reflection:
  • Follow Bean coding style (we have seen)
  • Analysis of the Bean’s class can then reveal properties and methods
Introspection

- Exposes the properties, methods and events supported by a Java Bean

- Low-level reflection:
  - Follow Bean coding style (we have seen)
  - Analysis of the Bean’s class can then reveal properties and methods

- Revealing complex properties:
  - Implement a “BeanInfo” class
Customisation
Customisation

Simple properties

- Development tool will build property sheets dynamically
- User may then edit the properties to customise the Bean
Customisation

- **Simple properties**
  - Development tool will build property sheets dynamically
  - User may then edit the properties to customise the Bean

- **For the Advanced User**
  - Create a specific customiser for a Bean
  - This is kept separate to the Bean Class, as with a BeanInfo Class
Further Features
Further Features

➢ Persistence
Further Features

- Persistence
- Visibility
Further Features

➢ Persistence
➢ Visibility
➢ Multithreading
Further Features

- Persistence
- Visibility
- Multithreading
- Security
Summary
Summary

- Beans build on Java features that already exist
Summary

- Beans build on Java features that already exist
- We add a Builder Tool
Summary

- Beans build on Java features that already exist
- We add a Builder Tool
- We use design patterns
Summary

- Beans build on Java features that already exist
- We add a Builder Tool
- We use design patterns
- We record information about the Classes that implement Beans