Stapmi project

Are we creating JavaBeans™ or Java classes??

Let s define the difference!!

1. WHAT IS A JAVABEAN?

- **Reusable** software components that can be **visually** manipulated in **builder tools** to create applications.

- They can be **simple components,** like a button, or can be more **complex software components** like a calendar:

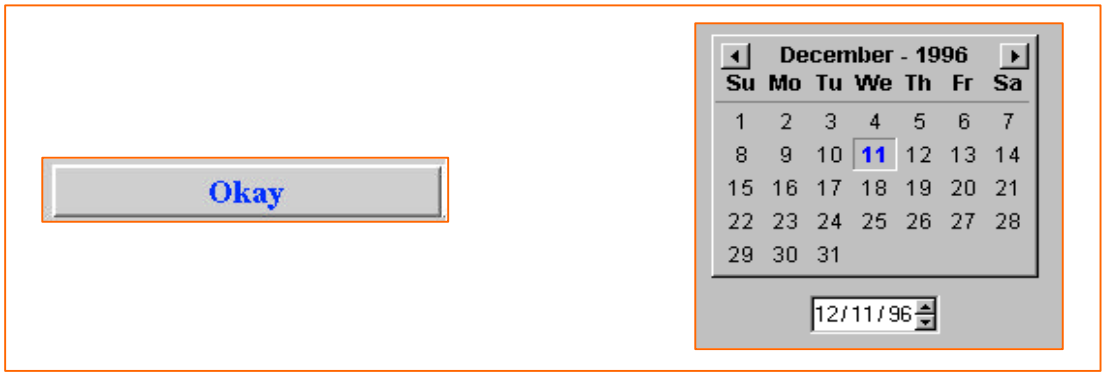

Fig 1. JavaBeans examples

2. WHAT IS THE DIFFERENCE BETWEEN BEANS AND CLASS LIBRARIES?

Stapmi project

- The difference is **INTROSPECTION**.

  - Introspection is what builder tools use to look inside a Bean and determine its properties and behavior.

  - Beans publish their attributes and methods through special method signature patterns that are recognized by beans-aware application construction tools.

- BASICALLY, JAVABEANS ARE WHAT BUILDER TOOLS USE TO CREATE APPLICATIONS.

Stopmi project

3. WHY DOES THE TOOL NEED TO LOOK INSIDE THE BEAN?

- To know which are the methods and the properties defined in the Bean and to create the respective editors to manage those methods and properties.

4. COULD I ADD MY JAVACLASS TO THE COMPONENT PALETTE OF THE TOOL?

- It is true that a Javabean is defined following standard design patterns that the tool can recognize.

- But it is also true that you CAN CREATE your Java classes, and ADD them to the tool, even if they are not designed following the design patterns.

To do so, we need to create a complementary Java class extended from BeanInfo class, defining the methods and properties we have used in our class, and add it to the tool.

5. CONSEQUENCES

Stapmi project

- Two consequences are derived from the above definitions:

1. If we use a Builder tool, it is better to think of creating Beans rather than Java classes, to take advantage of the possibilities that tools provide:
   - Property editors automatically created.
   - Easy management of events and listeners.

2. If we don't know if we are using a tool, we can define our elements as Java classes, in which case:
   - Properties are read from an external file.

Later on, we could add them to the tool palette and specify the corresponding BeanInfo class.

Stopmi project

6. JAVABEANS™ ADDITIONAL INFORMATION

6.1 JAVABEANS FEATURES

1. They can **be instantiated**.

   A Javaclass can be instantiated if it is neither an interface nor an abstract class.
   A class is an interface when it is defined like that, and it is abstract when contains any abstract method or variable.

2. They contain a **constructor method,** even if it is an empty method.

3. They are **persistent** (must implement either a serializable or externalizable interfaz).

4. They follow a **standard design pattern** defined by five rules.

5. They use the **event delegation model**.

6.2 BEANS ARCHITECTURE

Beans are composed of three elements:

1. Properties
2. Methods
3. Events

- Properties define the attributes of the Bean.
- Methods are used to set the Beans properties and to define and receive events.
- Events are the vehicle between Beans, as well as between Beans and containers.

Stapmi project

6.3 PROPERTIES

Property types

- Simple (numbers, characters and strings)
- Boolean
- Array
- Indexed
- Bound
- Constrained

Stapmi project

6.3.1 BOUND AND CONSTRAINED PROPERTIES

- A **Bound property** is a property that notifies any listeners of changes in the property s value.

  The listener notifies other components of changes.
  The listening component then has the opportunity to respond to the change in the property s value.

- A **Constrained property** is a property in which a change to the property value can be vetoed by another component that is listening for changes in the property s value.

Stapmi project

6.3.2 PROPERTY EDITORS

A property editor is an editor for changing property values at design time.

They vary from one IDE to another, but typically appear as a top-level dialog box.

They can edit a single property at a time or the entire component at once (which is called component customizes).

Component customizers present a dialog box or panel that lets the user set many properties at once.

Stapmi project

6.4 METHODS

They are mainly used:

- to **read/write** beans **properties:**
  - get ( ), set ( ), is( ) methods
- to **handle** bean **events:**
  - Notifying the actionlistener that an event has ocurred
  - Implementing the action associated to that event.
  - Notifying other beans.