Unit 3

Java Beans

Beans allows to build complex system by reusing software component. These components are provided by certain vendors. Java beans is a software component that has been designed to be reusable in a variety of different environment. Java beans can be a program performing simple operation or can be complex program like obtaining a online report of stock, inventory etc.

A bean can be visible to user like button in GUI or cannot be visible to user. A bean may be designed to work autonomously on a user' s workstation or to work in cooperation with a set of other distribute component.

Advantages:

- Helps in obtaining "write once run anywhere" paradigm
- The properties, events and methods of a Bean that can be exposed to another application and can be controlled
- The state of bean can be saved in persistent storage and restored at a later time.
- A bean may register to receive events from other object and can generate events that are sent to another object

Introspection:

It is the process of analyzing a Bean to determine its capability. Introspection allows another application like design tool, to obtain information about components. There are two ways from which we can indicate which of the properties (bean state), event and method should be exposed and hide. First method is to use simple naming convention. Second method is to used additional class that extends BeanInfo interface

Property of the beans determine the behavior and appearance of that component. A property is set through set method and value of property is retrieve through get method.

There are two types of properties: simple and indexed

Simple properties:

It has a single value and can be identified by the following design pattern:

public T getN()

public void setN(T arg)

for read/write property both patterns are used. For read only property, first pattern is used and for write only property only second pattern is used.

```
Class Demo{

Private int length;

Private int breadth;

public void setLength( int l){

Length = l;

}

public void setBreadth( int b){

breadth = b;

}

Public int getLength(){

return length;

}
```

```
Public int getBreadth(){
return breadth;
}
}
```

Indexed properties:

It consists multiple value and can be identified using following design pattern

```
public T getN(int index);
public void setN(int index, T value);
public T[ ] getN();
public void setN(T values[])
```

for example

```
private int data[ ];
public int getData(int index){
return data[index];
}
public void setData(int index, int value){
data[index] = value;
}
public int [ ] getData(){
return data;
}
```

```
public void setData(int values[ ]){


}
```

Design Patterns for Event:

Java beans use delegation event model (using listener) to generate and send events. Beans can both generate and sends event to another object. Following pattern are used for generating and sending events:

Public void addTListener(TListener eventListener): used for multicast an event i.e. more than one listener can be register for event notification.

Public void addTListener(TListener eventListener) throws java.util.TooManyListenersException: used for providing restriction on registering listener.

Public void removeTListener(TListener eventListener): used to remove the listerner

Using BeanInfo Interface to set the beans property

BeanInfo interface enables to explicitly control the information available. The BeanInfo interface defines several methods:

PropertyDescriptior[] getPropertyDescriptor()

EventSetDescriptor[] getEventSetDescriptor()

MethodDescriptor[] getMethodDescriptor()

All the three methods are defined within java.beans package. They return array of objects that provide information about properties, events and method.

Box implements BeanInfo{

}

BoxBeanInfo

When creating a class that implements BeanInfo, that class should called by using bnameBeanInfo syntax. For example, if class Box implements BeanInfo then class Box should be called using BoxBeanInfo

JavaBeans supplies SimpleBeanInfo class that provide default implementation of the BeanInfo interface including three methods of interface. We can extend this SimpleBeanInfo class and overrides one or more methods to explicitly control feature of beans that needs to be exposed.

Bound and Constrained Properties

Bound property generates a event when the property is changed. The event is PropertyChangeEvent() and send to object that previously registered an interest in receiving such notification. A class should use PropertyChangeListener to handle PropertyChangeEvent().

Constrained property generates an event when attempt is made to change its value.

Peristance:

It is the ability to save the current state of bean, including property and instance variables to nonvolatile storage. It is done by using java.io.Serializable interface.

Rules:

When Using java beans:

- Serializable should implemented
- To handle event, listener (delegation method) should be used
- To set values of instance variable, getter and setter method should be used

- Only default constructor is allowed

Java beans API

Four classes of java beans are commonly used :

Introspector, PropertyDiscriptor, EventSetDescriptor, MethodDescriptor

Inrospector:

This class provides several static methods that support introspection. getBeanInfo() is used to return BeanInfo object which is used to obtain information about bean.

Syntax of getBeanInfo():

Static BeanInfo getBeanInfo(Class<?> bean) throws IntrospectionException

PropertyDescriptor

This class describes the characteristics of a Bean property. isBound() method is used to determine whether the property is bound or not. isConstrained() is used to determine whether the property is constrained or not. getName() method is used to obtained name of the property.

EventSetDescriptor

This class represent Bean event. This class provides several methods that helps to find out what events are uses by Bean and what event listener are remove by beans. To obtain the method used to add listener, getAddListenerMethod() is used, to obtain the method used to remove listener, getRemoveListenerMethod() is used. To obtain the type of a Listener, getListenerType() is used. To obtain the name of an event getName() method can be called or used.

MethodDescriptor:

This class represents a Bean method. To obtain the name of method getName() method is called. getMethod() is used to obtain information about the method.

Method getMethod()

An object of type Method that describe method is returned