

## Summary of JavaBean Rules

### A JavaBean has these characteristics:

- It is a **public** class.
- It has a public **parameterless** constructor (though it may have other constructors as well)
- It implements **Serializable** interface (i.e. it can be made persistent, so its state can be saved)
- It has **properties** with “**getter**” and “**setter**” methods named by following JavaBeans naming patterns
- It has **events** which follow the **standard Java event model** with the **registration** methods named by following the JavaBeans naming patterns
- It may have other methods which do not follow the naming patterns. These methods are not exposed by a builder tool.

### Comments:

A **user** of JavaBean components can build an **application** consisting of JavaBeans, by **customizing**, and **wiring** JavaBeans together at **design time**, by using a **builder** tool in an **IDE** (Integrated Development Environment), such as **JBuilder**.

The builder tool analyses and **exposes** the **properties** and **events** of the JavaBeans at **design time** by means of **introspection**. A builder tool can do this only if proper naming patterns have been followed to name the properties, and event registration methods (we will not talk about the *beanInfo* class in these notes)

### Naming patterns for properties:

Property name: **<propertyName>**  
e.g. temperature

Property type: **<Type>**  
e.g. double

Getter method: **public <Type> get<PropertyName>() { //... }**  
e.g. double getTemperature() { // ... }

Boolean Property Getter Method: **public boolean is<PropertyName>() {**

```
                // ... }  
e.g. public boolean isFull() { // ... }
```

Setter method: **public void set<PropertyName>(<Type> p)** { // ... }  
e.g. void setTemperature(double temp) { // ... }

**... Notes to be continued ...**