JAVA BEANS INTRODUCTION

The JavaBeans Standard allows reusable software components to be modelled in Java so that these components can be assembled to create sophisticated applications.In particular, *builder tools* can take advantage of how these components are specified, in order to build new applications based on these components. The Java-Beans specification specifies the rules for defining such components (called *Java-Beans*). The interested reader is encouraged to consult this documentation (see http://java.sun.com/javase/technologies/desktop/javabeans/docs/spec.html) for details since we only cover the basic fundamentals for creating JavaBeans .

Naming Patterns for Properties

The rules of the JavaBean specification stipulate *naming patterns* for declaring *properties* of JavaBeans. A naming pattern defines a standard naming convention. A property of an object is normally defined as a field in the object, which is usually not directly accessible by clients (see Example 3.1). A JavaBean should adhere to the following naming patterns when specifying its properties:

• The properties are assumed to be private, and their names start with a lowercase letter. Example shows that the JavaBean class Light has three properties.

• In order to retrieve and change values of its properties, a JavaBean provides *getter* and *setter* methods for them. Example shows a JavaBean with three getter and three setter methods for its properties.

• For a property, the setter method starts with the prefix set. The rest of the method name is assumed to be a property name, where the first letter of the property name has been converted to uppercase. In Example, the value of the property noOfWatts can be changed by the setter method setNoOfWatts(). Setter methods are public and void, having a parameter of the same type as that of the property.

• For a property, the getter method starts with the prefix get. The rest of the method name is assumed to be a property name, where the first letter of the property name has been converted to uppercase. In Example, the value of the property noOfWatts can be retrieved by the getter method getNoOfWatts(). For a boolean property, the getter method can start with the prefix get or is. In Example 3.1, the value of the boolean property indicator can be retrieved by the getter method isIndicator(). Getter methods are no-argument public methods that return a value of the same type as the parameter of the corresponding setter method.

Example *A JavaBean*

```
public class Light {
        // Properties:
        private int noOfWatts; // wattage
        private String location; // placement
        private boolean indicator; // on or off
        // Setters
        public void setNoOfWatts(int noOfWatts) { this.noOfWatts = noOfWatts; }
        public void setLocation(String location) { this.location = location; }
        public void setIndicator(boolean indicator) { this.indicator = indicator; }
        // Getters
        public int getNoOfWatts() { return noOfWatts; }
        public String getLocation() { return location; }
        public boolean isIndicator() { return indicator; }
}
```