6. Object Oriented Concepts and Techniques

Lesson 1:
Basics of Object Orientation – Part 1

6. Object Oriented Concepts and Techniques

- The central idea of Object Oriented Programming is to organize your programs in a way that mirrors the way objects are organized in the real world
- Object oriented programming is, at its root, a way to conceptualize a computer program.
- This is a way to look at a program is as a set of objects that work together in predefined ways to accomplish tasks.
- Using OOP, your overall program is made up of different objects.

6.1. Objects

- An object is a self-contained element of a computer program that represents a related group of features and is designed to accomplish specific tasks.

- Also known as instances.

- Each object has a specific role in a program, and all objects can work with other objects in specifically defined ways.

6.1. Objects

6.1.2.More definitions for an Object

- A ‘thing’ may have a physical presence such as a ‘table’ ,‘chair’ or an abstract concept such as ‘a job’ .
- An object is an abstract representation of a ‘thing’ in the real world.
- We simulate a problem domain in the real - world through objects.
- An object has a unique identity, attributes (What it knows or data about it), and behavior (What it can do).

6.1. Objects

6.1.2. More definitions for an Object Cont…

- For example, an Employee object (say employee1) will have the following attributes (what it knows):
  - name
  - age
  - salary

- It will also have the following behavior (what it can do):
  - set salary
  - get salary
  - set name
  - set age

6.1. Objects
6.1.3. Employee Object Example

- A ‘thing’ may have a physical presence such as a ‘table’ ,‘chair’ or an abstract concept such as ‘a job’ .
- An object is an abstract representation of a ‘thing’ in the real world.
- We simulate a problem domain in the real - world through objects.
- An object has a unique identity, attributes (What it knows or data about it), and behavior (What it can do).

6.2. Classes

- A Class is a template used to create multiple objects with similar features.
- When we write a program in an OO language, we don't need to define individual objects. Instead, we define classes of objects. Classes embody all features of a particular set of objects.
- For example, a Class "Tree" describes the features of all trees:
  - Has leaves & roots
  - Grows
  - Creates chlorophyll
- The Tree Class serves as an abstract model for the concept of a tree.
- Every class written in java made up of two components: Attributes (what it knows) and Behavior (what it can do)

6.2. Classes
6.2.1. Attributes

- Individual things that differentiate one class of objects from another and determine the appearance, state and other qualities of that class.
  - E.g: Color orange, raw umber, lemon yellow, maize

- Attributes of a class of objects also can include information about an object's state.

- In a class, attributes are defined by variables.

- Each object can have different values for its variables. These are called instance variables.

6.2. Classes

6.2.2. Instance variables and Class variables

- Instance variables
  - An instance variable is an item of information that defines an attribute of one particular object.
  - The object's class defines what kind of attribute it is, and each instance object stores its own value for that attribute.
  - Instance variables also known as object variables.
- Class variables
  - A Class variables is an item of information that defines an attribute of an entire class.
  - The variable applies to the class itself and to all of its instances.

6.2. Classes
6.2.3. Behavior

- Behavior is the way that a class of objects can do to change their attributes, and also what they do when ask them to do something.

- Examples for behavior
  - Get angry
  - Calm down
  - Eat a peasant

- Behavior for a class of objects is done by using methods.

6.2. Classes
6.2.4. Creating a Class

- Open the text editor to create Java programs. Start with class definition.

```
class Jabberwock
{
}
```

- Then create instance variables

```
String color;
boolean hungry;
```

6.2. Classes

6.2.4. Creating a Class (continued)

- After that the programmer can add behavior to the class by adding methods.

```
void feedJabberwock()
{
        if (hungry == true)
        {
                System.out.println("yum – a peasant");
                hungry = false;
        }
        else
                System.out.println("No, thanks – already ate");
}
```

- Use one of the following procedures to compile the program, depending on the system you're using.

```
javac Jabberwock.java
```

6.2. Classes

6.2.5. Creating Objects

- Objects are created by instantiating classes. To use a class in a program, you must first create an instance of it. Objects of a class can be created using the new operator.

```
Employee newEmp = new Employee();
```

- Object References following declaration will create an Object reference

```
Employee newEmp2;
```

- You can create multiple References to the same object

```
Employee newEmp;
newEmp = new Employee();
newEmp2 = newEmp;
```