ENGINEER SERIES

IN JAVA LANGUAGE

1
CH

OBJECTS AND CLASSES

ENG : Qusay Khudair

Creativity and Accuracy in Work

AUG PLUS .Edu

Objects and Classes

- **Class: A blueprint for creating objects. It defines a datatype by grouping together data and methods that work on the data into one single unit.**

- **Object: An instance of a class. Represents a real-world entity with state (attributes) and behavior (methods).**

Example:

```
public class Car {
    String color;
    int year;

    public void displayDetails() {
        System.out.println("Color: " + color + ",
Year: " + year);
    }
}

public class Main {
    public static void main(String[] args) {
        Car myCar = new Car();
        myCar.color = "Red";
        myCar.year = 2020;
        myCar.displayDetails();  // Output: Color:
Red, Year: 2020
    }
}
```

Constructors

- **Constructor**: A special method called when an object is instantiated. It initializes the object.
- **Types**: Default constructor (no arguments) and parameterized constructor (with arguments).
- **Rules**: Constructors have the same name as the class and no return type.

Example:

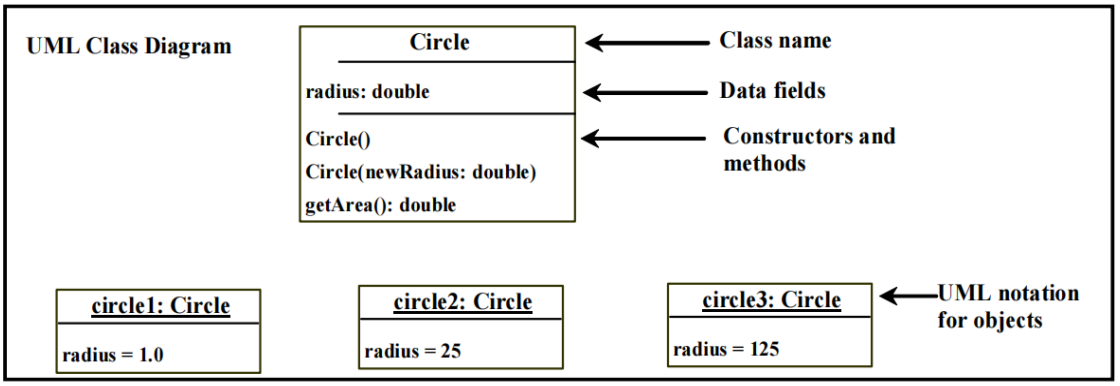
```
public class Car {
    String color;
    int year;

    // Default Constructor
    public Car() {
        color = "Unknown";
        year = 0;
    }

    // Parameterized Constructor
    public Car(String c, int y) {
        color = c;
        year = y;
    }
}

public class Main {
    public static void main(String[] args) {
        Car defaultCar = new Car();
        Car myCar = new Car("Red", 2020);
        System.out.println(defaultCar.color + " " +
defaultCar.year);  // Output: Unknown 0
        System.out.println(myCar.color + " " +
myCar.year);  // Output: Red 2020
    }}
```

UML Class Diagram



Declaring Object Reference Variables

To reference an object, assign the object to a reference variable.

To declare a reference variable, use the syntax:

ClassName objectRefVar ;

Example:

Circle myCircle;

```
ClassName objectRefVar = new ClassName();
```

Example:

Assign object reference — Create an object

```
Circle myCircle = new Circle();
```

Reference Data Fields

- **Definition**: Variables that store the reference (address) to an object, rather than the object itself.
- **Behavior**: Changes made to the object through one reference will reflect in other references pointing to the same object.

Example:

```
public class Car {
    String color;
    int year;
}
public class Main {
    public static void main(String[] args) {
        Car car1 = new Car();
        Car car2 = car1; // car2 references the same
object as car1
        car2.color = "Blue";
        System.out.println(car1.color); // Output:
Blue  }}
```

The null Value

If a data field of a reference type does not reference any object, the data field holds a special literal value, null.

Default Value for a Data Field

null for a reference type.

0 for a numeric type.

false for a boolean type.

'\u0000' for a char type.

- **However, Java assigns no default value to a local variable inside a method (Compilation error)**

Copying Variables of Primitive Data Types and Object Types

Copying Variables

Primitive Data Types: Copying creates a new value.

```
int x = 5;
int y = x; // y = 5
y = 10;
System.out.println(x); // Output: 5
```

Object Types: Copying references the same object.

```
Car car1 = new Car();
Car car2 = car1; // car2 references the same object
as car1
car2.color = "Blue";
System.out.println(car1.color); // Output: Blue
```

Garbage Collection

- **Definition**: Automatic process of reclaiming memory by deleting objects that are no longer reachable in the program.
- **How it Works**: Managed by the Java Virtual Machine (JVM). No explicit deletion of objects is required.
- **Benefits**: Prevents memory leaks, simplifies memory management.

Example:

```
public class Main {
public static void main(String[] args) {
Car car1 = new Car();
```

```
Car car2 =new Car();
car1=car2; // car2 references the same object as car1
car2.color = "Blue";
System.out.println(car1.color); \\Output : Blue
}
}
```

Displaying GUI Components

When you develop programs to create graphical user interfaces, you will use Java classes such as JFrame, JButton, JRadioButton, JComboBox, and JList to create frames, buttons, radio buttons, combo boxes, lists, and so on.

Ex :

```
import javax.swing.*;

public class GUIComponents {

 public static void main(String[] args) {

 // Create a button with text OK

 JButton jbtOK = new JButton("OK");

 // Create a button with text Cancel

 JButton jbtCancel = new JButton("Cancel");

 // Create a label with text "Enter your name: "

 JLabel jlblName = new JLabel("Enter your name: ");

 // Create a text field with text "Type Name Here"

 JTextField jtfName = new JTextField("Type Name Here");

 JTextField jtfName = JTextField( );

 // Create a check box with text bold
```

```
JCheckBox jchkBold = new JCheckBox("Bold");

// Create a check box with text italic

JCheckBox jchkItalic = new JCheckBox("Italic");

// Create a radio button with text red

JRadioButton jrbRed = new JRadioButton("Red");

// Create a radio button with text yellow

JRadioButton jrbYellow = new JRadioButton("Yellow");

// Create a combo box with several choices

JComboBox jcboColor = new JComboBox(new
String[]{"Freshman",

"Sophomore", "Junior", "Senior"});

// Create a panel to group components

JPanel panel = new JPanel();

panel.add(jbtOK); // Add the OK button to the panel

panel.add(jbtCancel); // Add the Cancel button to the panel

panel.add(jlblName); // Add the label to the panel

panel.add(jtfName); // Add the text field to the panel

panel.add(jchkBold); // Add the check box to the panel

panel.add(jchkItalic); // Add the check box to the panel

panel.add(jrbRed); // Add the radio button to the panel

panel.add(jrbYellow); // Add the radio button to the panel

panel.add(jcboColor); // Add the combo box to the panel

panel.add(jcboColor); // Add the combo box to the panel
```

```
JFrame frame = new JFrame(); // Create a frame
frame.add(panel); // Add the panel to the frame
frame.setTitle("Show GUI Components");
frame.setSize(450, 100);
frame.setLocation(200, 100);
frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
frame.setVisible(true); } }
```

Output >>>>>>

Instance Variables and Methods

- **Instance Variables**: Attributes unique to each instance of a class.
- **Instance Methods**: Behaviors that operate on instance variables of the object.

Example:

```
public class Car {
    String color;
    int year;

    // Instance method to display car details
    public void displayDetails() {
        System.out.println("Color: " + color + ",
Year: " + year);
    }
}
```

```
public class Main {
    public static void main(String[] args) {
        Car myCar = new Car();
        myCar.color = "Red";
        myCar.year = 2020;
        myCar.displayDetails();  // Output: Color:
Red, Year: 2020}}
```

Static Variables, Constants, and Methods

Static Variables: Shared among all instances of a class.

```
public class Car {
    static int count = 0;
    public Car() {
        count++;
    }}

public static void main(String[] args) {
Car car1 = new Car();
Car car2 = new Car();
 Car car3 = new Car();
 System.out.println(Car.count); // This will print 3
} }
```

Constants: Declared as `final`, value cannot be changed.

```
public class Car {
    public static final int MAX_SPEED = 200;
}
```

Static Methods: Can be called without creating an instance of the class.

```
public class Car {
    public static void displayCount() {
        System.out.println("Number of cars: " +
count);
    }
}
```

Visibility Modifiers and Accessor/Mutator Methods

Visibility Modifiers

- **Public**: Accessible from any other class.
- **Private**: Accessible only within the declared class.
- **Protected**: Accessible within the same package and subclasses.
- **Default (Package-Private)**: Accessible only within the same package.

Accessor/Mutator Methods

- **Getters and Setters**: Methods to access and update private variables.

Example:

```
public class Car {
    private String color;

    // Getter method
    public String getColor() {
        return color;
    }

    // Setter method
```

```
    public void setColor(String color) {
        this.color = color;
    }
}

public class Main {
    public static void main(String[] args) {
        Car myCar = new Car();
        myCar.setColor("Red");
        System.out.println(myCar.getColor()); //
Output: Red
    }
}
```

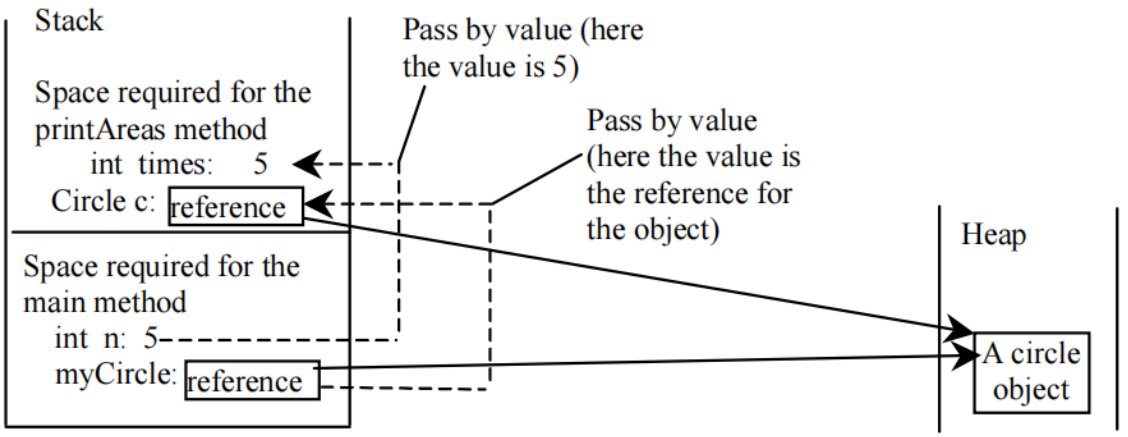
Passing Objects to Methods

- **Passing by value for primitive type value(the value is passed to the parameter)**
- **Passing by value for reference type value (the value is the reference to the object)**

Example:

```
public class TestPassObject {
 public static void main(String[] args) {
// Create a Circle object with radius 1
 CircleWithPrivateDataFields myCircle = new
CircleWithPrivateDataFields(1);
 // Print areas for radius 1, 2, 3, 4, and 5.
 int n = 5;
 printAreas(myCircle, n);
// See myCircle.radius and times
 System.out.println("\n" + "Radius is " + myCircle.getRadius());
System.out.println("\n" + "Radius is " + myCircle.getRadius());
System.out.println("n is " + n); }
```

```
/** Print a table of areas for radius */
public static void printAreas( CircleWithPrivateDataFields c, int times) {
System.out.println("Radius \t\tArea");
while (times >= 1) {
System.out.println(c.getRadius() + "\t\t" + c.getArea());
c.setRadius(c.getRadius() + 1);
times--; } } }
```



Array of Objects

- **Definition**: An array that holds references to objects.
- **Behavior**: Each element of the array can hold a reference to an object.

Example:

```
public class Car {
    String color;
    int year;

    public Car(String color, int year) {
        this.color = color;
        this.year = year;
    }
}

public class Main {
    public static void main(String[] args) {
        Car[] cars = new Car[3];
        cars[0] = new Car("Red", 2020);
        cars[1] = new Car("Blue", 2021);
        cars[2] = new Car("Green", 2022);

        for (Car car : cars) {
            System.out.println(car.color + " " +
car.year);
        }
    }
}
```

- **For More … Follow Me :**

  Eng.Qusay Khudair

- **For Business :**

  Linkedin

  Behance

  Github

- **For My Files and Slides :**

  studocu.com

- **Whatsapp : +972567166452**