Programming Languages - II
User Defined Variables (Classes)

Özgür Koray ŞAHİNGÖZ
Prof.Dr.

Biruni University
Computer Engineering Department

Java Classes/Objects

- Java is an object-oriented programming language.

- Everything in Java is associated with classes and objects, along with its attributes and methods. For example: in real life, a car is an object. The car has attributes, such as weight and color, and methods, such as drive and brake.

OO Programming Concepts

- Object-oriented programming (OOP) involves programming using objects. An object represents an entity in the real world that can be distinctly identified. For example, a student, a desk, a circle, a button, and even a loan can all be viewed as objects. An object has a unique identity, state, and behaviors. The state of an object consists of a set of data fields (also known as properties) with their current values. The behavior of an object is defined by a set of methods.

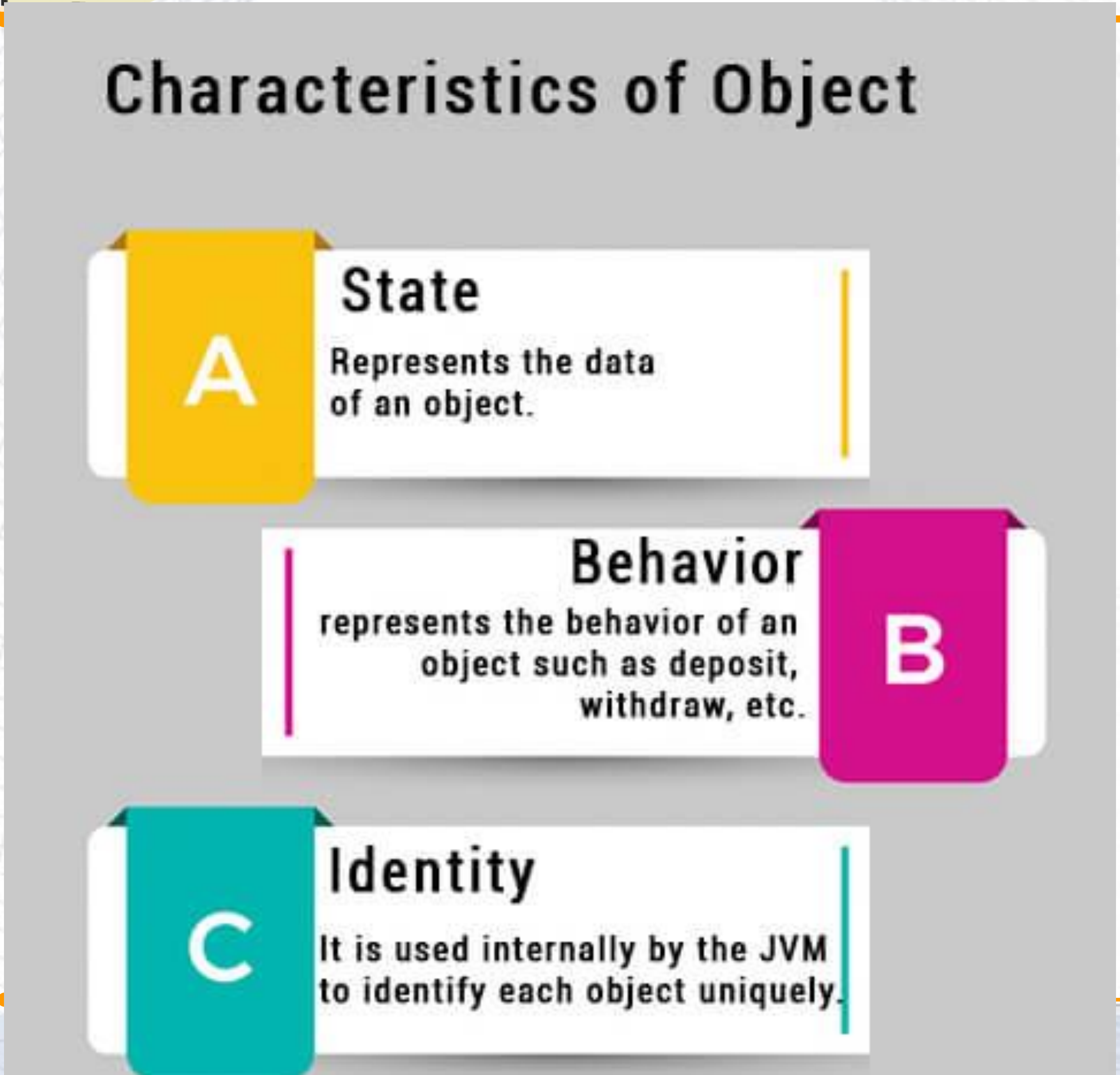
What Does Java Object Mean?

- A Java object is a member (also called an instance) of a Java class. Each object has an identity, a behavior and a state.

- The state of an object is stored in fields (variables), while methods (functions) display the object's behavior. Objects are created at runtime from templates, which are also known as classes.

- In Java, an object is created using the keyword **"new".**

Objects



Characteristics of Object

A — State

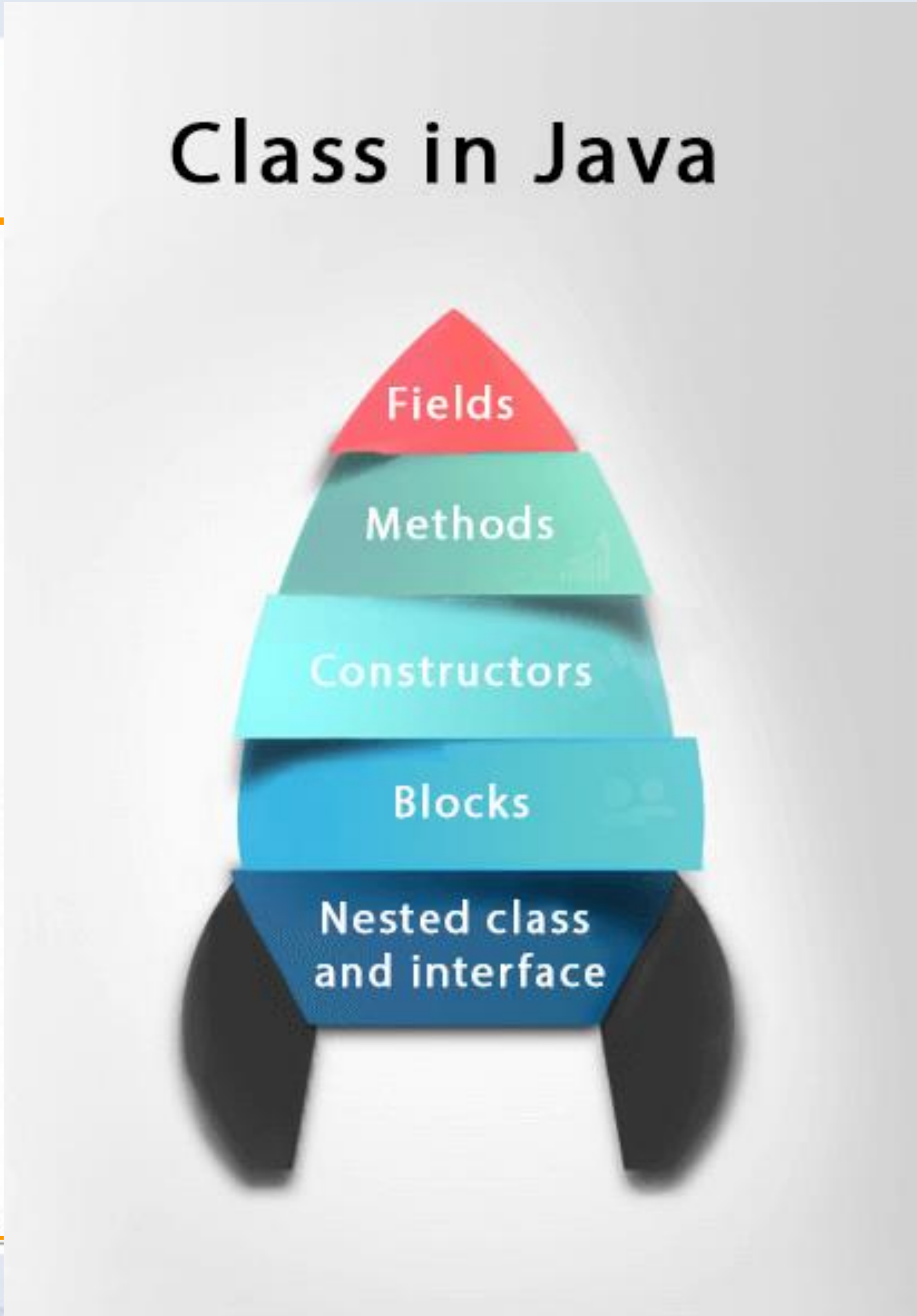
Represents the data of an object.

B — Behavior

represents the behavior of an object such as deposit, withdraw, etc.

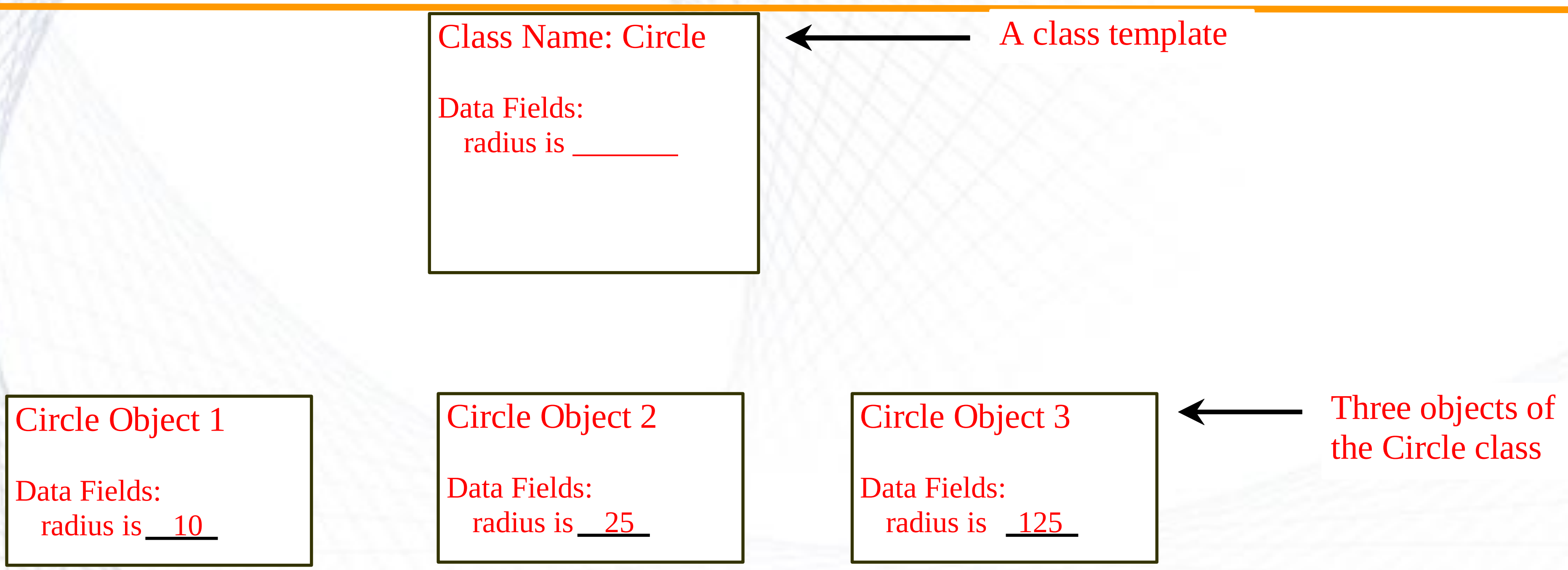
C — Identity

It is used internally by the JVM to identify each object uniquely.



Class in Java

Objects vs Class



An object has both a state and behavior. The state defines the object, and the behavior defines what the object does.

Classes

- Classes are constructs that define objects of the same type. A Java class uses variables to define data fields and methods to define behaviors. Additionally, a class provides a special type of methods, known as constructors, which are invoked to construct objects from the class.

Circle Classes

```
class Circle {                            // This is a Class
   /** The radius of this circle */
   double radius = 1.0;
}
```

Creating a variable

```
class Circle {                           // This is a Class
    /** The radius of this circle */
    double radius = 1.0;
}



public class test {

    public static void main(String[] args) {
    Circle c1= new Circle();                   // Creation of an object
    c1.radius=12;                              // Accessing a variable. Dot-notation
    System.out.println(c1.radius);
    }


}
```

Creation of two variables/objects

```
class Circle {                                    // This is a Class
    /** The radius of this circle */
    double radius = 1.0;
}



public class test {

    public static void main(String[] args) {
    Circle c1= new Circle();                              // Creation of an object
    c1.radius=12;                                         // Accessing a variable. Dot-notation
    Circle c2= new Circle();                              // Creation of an object
    c1.radius=24;                                         // Accessing a variable. Dot-notation
    System.out.println(c1.Radius+ c2.Radius);
    }

}
```

Example

- Write a Date class which stores the day, month and year values.

```
class Date {
        int day, month, year;
}
```

Example

- Read two different dates from the keyboard.

```
public class soru1 {
    public static void main(String[] args) {
    Date d1=new Date();
    Date d2=new Date();
    Scanner sc = new Scanner(System.in);
    System.out.print("Enter the first day\t:");        d1.day=sc.nextInt();
    System.out.print("Enter the first month\t:");      d1.month=sc.nextInt();
    System.out.print("Enter the first year\t:");       d1.year=sc.nextInt();

    System.out.print("Enter the first day\t:");        d2.day=sc.nextInt();
    System.out.print("Enter the first month\t:");      d2.month=sc.nextInt();
    System.out.print("Enter the first year\t:");       d2.year=sc.nextInt();

    System.out.println("Date 1 is  :"+d1.year+"-"+d1.month+"-"+d1.day);
    System.out.println("Date 2 is  :"+d2.year+"-"+d2.month+"-"+d2.day);
    }
```

- }

- Write a DisplayDate function which displays the Date depending on your display format.

```
public static void DisplayDate(Date d) {
    System.out.println("Date is  :"+d.year+"-"+d.month+"-"+d.day);

}
```

Example

- Generate an array which stores 100 date values in it.
- Date[] dates=new Date[100];

- Wriete a Generate function which generates 100(as parameter) random values and return this array as the return value.

```
public static Date[] Generate(int size) {
    Date[] array=new Date[size];
    for (int i=0; i<size; i++ ) {
    Date d = new Date();
    d.day=(int) (Math.random()*30);
    d.month=(int) (Math.random()*30);
    d.year=(int) (2000+Math.random()*50);
    array[i]=d;
    }
    return array;
}
```

- Display all these Dates on the screen

```
Date[] dates= new Date[100];
dates=Generate(100);
for (int i=0; i<100; i++ )
        DisplayDate(dates[i]);
```

Example

- Write a new Student class which have the following attributes
  - Name
  - Surname
  - StudentID
  - BirthDate
  - OSYMPoint