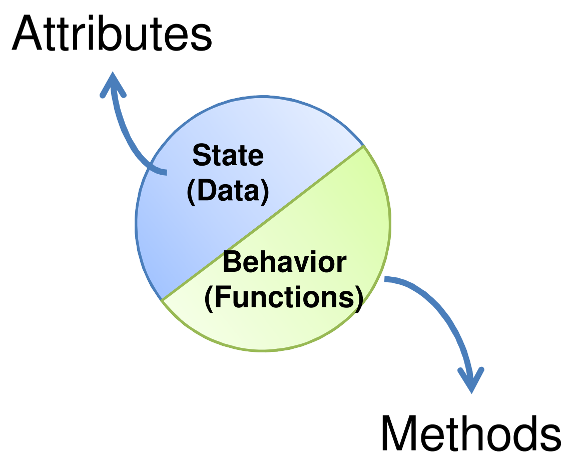
# Some Java Basics

Object Oriented Programming
2016375 - 5

Camilo López

# Objects and Classes

- We know what an Object is…



- We also know what we need in order to build/create new objects

Total surface area = 144 m² (122 m²)
STUDIO
TOILET
BOILER
LUMBER ROOM
HALLWAY 2
KITCHEN
LIBRARY
HALLWAY 1
DINING ROOM
LIVING ROOM

# Classes

Objects – Instances of a class

Width, Height and Color are Common features of the rectangles

They are relevant in our abstraction of a rectangle

```
public class Rectangle {
    String color;
    double width;
    double height;

    // ... method declarations
}
```

## adding some methods/behaviors

We may want to know the perimeter or the area of a given rectangle. Who can tell us?

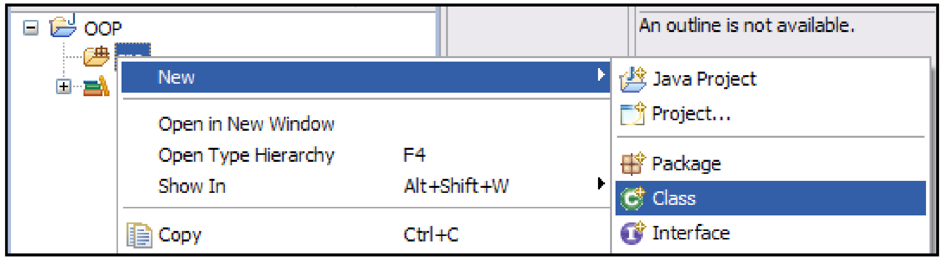
It’s the rectangle’s responsibility to inform us about these data → Telling us its area or perimeter is **a behavior**

```
public class Rectangle {
    String color;
    double width;
    double height;

    double area(){
        return (width * height)
    }
}
```

# In Eclipse

*Rectangle.java*

Now, let’s create the Rectangle class. It will be in its own .java file



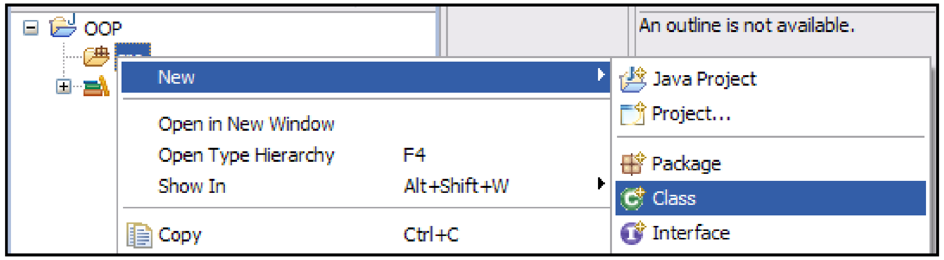
This class doesn’t need a Main method, we’re not thinking on execute it!

```
public class Rectangle {
    String color;
    double width;
    double height;

    double area(){
        return (width * height)
    }
}
```

# In Eclipse

*Rectangle.java*

Now, let's create the Rectangle class. It will be in its own .java file



This class doesn't need a Main method, we're not thinking on execute it!

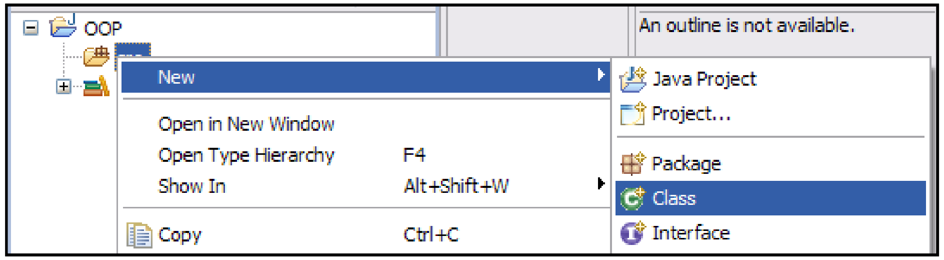
```
public class Rectangle {
    private String color;
    double width;
    double height;

    double area(){
        return (width * height)
    }
}
```

**change the [Access level modifiers](#) to private**

# In Eclipse

*Rectangle.java*

Now, let’s create the Rectangle class. It will be in its own .java file



This class doesn’t need a Main method, we’re not thinking on execute it!

```
public class Rectangle {
    private String color;
    double width;
    double height;

    double area(){
        return (width * height)
    }
}
```

change the Access level modifiers to private

**You need accessor methods (get/set)**
**Add a method to compute the perimeter.**
**Methods should be public**

## Creating Objects

As you know, a class provides the blueprint for objects; you create an object from a class.

To create an object and assign it to a variable we need:

1. **Declaration**: Associate a variable name with an object type.

```
Rectangle r
```

## Creating Objects

As you know, a class provides the blueprint for objects; you create an object from a class.

To create an object and assign it to a variable we need:

1. **Declaration**: Associate a variable name with an object type.
2. **Instantiation**: The new keyword is a Java operator that creates the object.

```
Rectangle r = new Rectangle();
```

## Creating Objects

As you know, a class provides the blueprint for objects; you create an object from a class.

To create an object and assign it to a variable we need:

1. **Declaration**: Associate a variable name with an object type.
2. **Instantiation**: The new keyword is a Java operator that creates the object.
3. **Initialization**: The new operator is followed by a call to a constructor, which initializes the new object.

```
Rectangle r = new Rectangle();
```

**This is done in Client Code**

## Create a RectangleTest Class

*RectangleTest.java*

```
public class RectangleTest {
    public static void main(String[] args) {
        double width = 4;
        Rectangle re = new Rectangle();

        re.setWidth(width);
        re.setLength(2);
        re.setColor(“green”);

        System.out.println(“The rectangle is “ + re.getColor() +
                    “\n and the area is “ + re.getArea());
    }
}
```

**First, we create two variables**

**A double and a Rectangle**

# Create a RectangleTest Class

*RectangleTest.java*

```
public class RectangleTest {
    public static void main(String[] args) {
        double width = 4;
        Rectangle re = new Rectangle();

        re.setWidth(width);
        re.setLength(2);
        re.setColor(“green”);

        System.out.println(“The rectangle is “ + re.getColor() +
                    “\n and the area is “ + re.getArea());
    }
}
```

**Set the attribute’s values**
**width – length – color**

# Create a RectangleTest Class

*RectangleTest.java*

```
public class RectangleTest {
    public static void main(String[] args) {
        double width = 4;
        Rectangle re = new Rectangle();

        re.setWidth(width);
        re.setLength(2);
        re.setColor("green");

        System.out.println("The rectangle is " + re.getColor() +
                    "\n and the area is " + re.getArea());
    }
}
```

**Now we can use this values and "communicate" with the object**

# Create a RectangleTest Class

*RectangleTest.java*

```
public class RectangleTest {
    public static void main(String[] args) {
        double width = 4;
        Rectangle re = new Rectangle(width, 2, “green”);

        System.out.println(“The rectangle is “ + re.getColor() +
                    “\n and the area is “ + re.getArea());
    }
}
```

**Same example.**
**Now the Rectangle Class has an explicit constructor**

**This is: The constructor has a list of formal parameters**

## References

- J. Barker, *Beginning Java Objects: From Concepts To Code, Second Edition*, Apress, 2005.
- H.M. Deitel and P.J. Deitel, Java How to Program: Early Objects Version, Prentice Hall, 2009.
- Java SE Tutorials (Last Updated 5/27/2009), which can be found at: http://java.sun.com/docs/books/tutorial