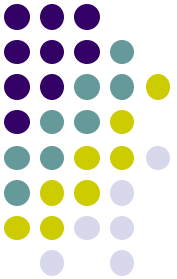
OOP

Ionut

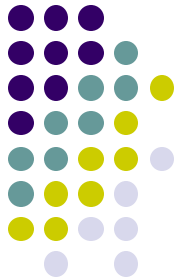
Spalatelu

Reference vs Object vs Instance vs Class



- Let's use the analogy of building a house to better understand Classes and objects
- A **class** is basically a blueprint for a house, using the blueprint (plans) we can build as many houses as we want based on those plans;
- Each house you build (in other words **instantiate** using the **new** operator) is a new object also known as an **instance**
- Each built house has an address (a physical location). In other words if you want to tell someone where you live, you give your address (perhaps written on a paper). This is known as the reference.
- You can copy that reference as many times as you like but there is still only one house. In other words, we are copying the paper that has the address on it not the house itself.
- We can pass **references** as **parameters** to **constructors** and **methods**.

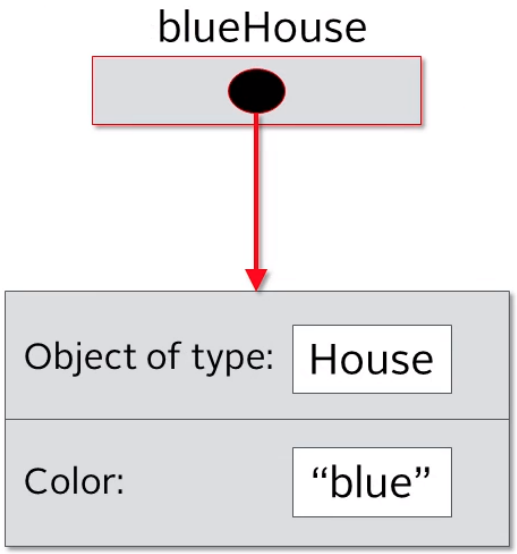
Reference vs Object vs Instance vs Class



```
class House {

    private String color;

    public House(String color) {
        this.color = color;
    }

    public String getColor() {
        return color;
    }

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

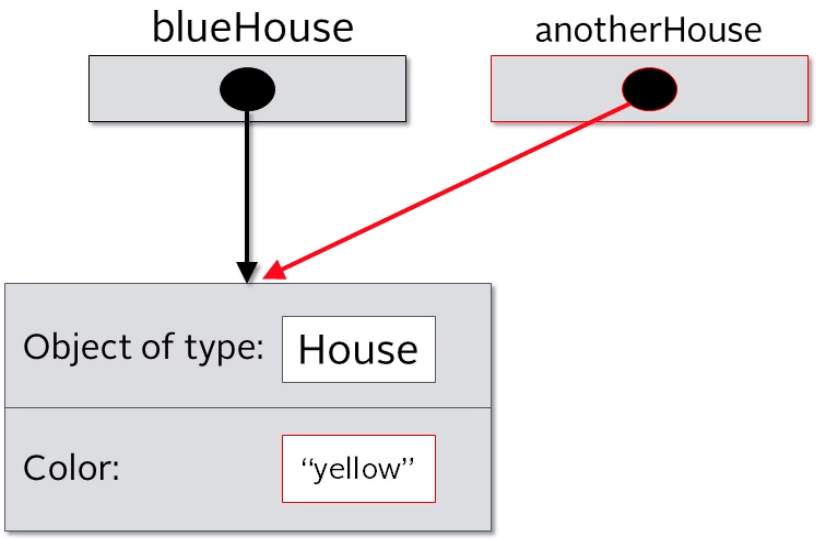
```
public class Main {

    public static void main(String[] args) {
        House blueHouse = new House("blue");
        House anotherHouse = blueHouse;

        System.out.println(blueHouse.getColor()); // prints blue
        System.out.println(anotherHouse.getColor()); // blue

        anotherHouse.setColor("red");
        System.out.println(blueHouse.getColor()); // red
        System.out.println(anotherHouse.getColor()); // red

        House greenHouse = new House("green");
        anotherHouse = greenHouse;

        System.out.println(blueHouse.getColor()); // red
        System.out.println(greenHouse.getColor()); // green
        System.out.println(anotherHouse.getColor()); // green
    }
}
```

Reference vs Object vs Instance vs Class



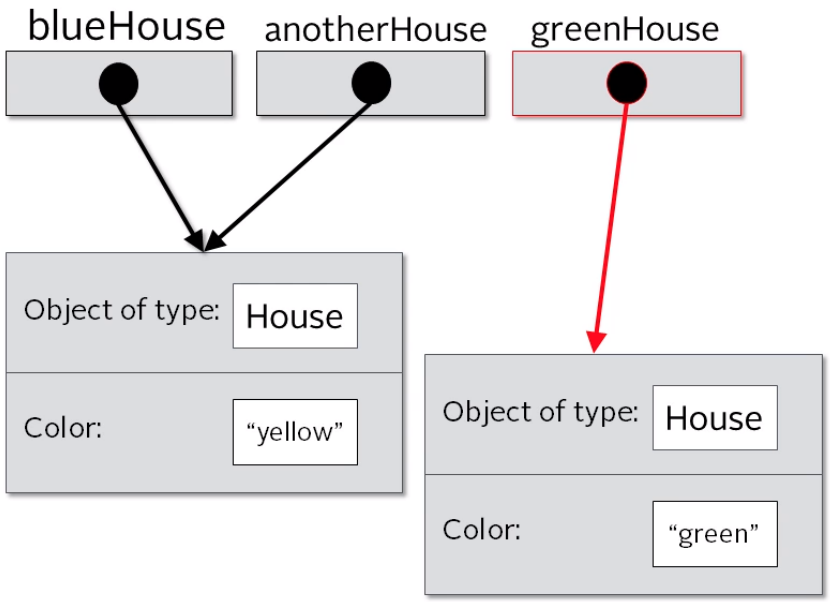

```
public class Main {

    public static void main(String[] args) {
        House blueHouse = new House("blue");
        House anotherHouse = blueHouse;

        System.out.println(blueHouse.getColor()); // prints blue
        System.out.println(anotherHouse.getColor()); // blue

        anotherHouse.setColor("red");
        System.out.println(blueHouse.getColor()); // red
        System.out.println(anotherHouse.getColor()); // red

        House greenHouse = new House("green");
        anotherHouse = greenHouse;

        System.out.println(blueHouse.getColor()); // red
        System.out.println(greenHouse.getColor()); // green
        System.out.println(anotherHouse.getColor()); // green
    }
}
```

Reference vs Object vs Instance vs Class



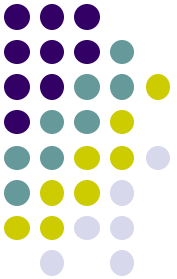

```
public class Main {

   public static void main(String[] args) {
      House blueHouse = new House("blue");
      House anotherHouse = blueHouse;

      System.out.println(blueHouse.getColor()); // blue
      System.out.println(anotherHouse.getColor()); // blue

      anotherHouse.setColor("yellow");
      System.out.println(blueHouse.getColor()); // yellow
      System.out.println(anotherHouse.getColor()); // yellow

      House greenHouse = new House("green");
      anotherHouse = greenHouse;

      System.out.println(blueHouse.getColor()); // yellow
      System.out.println(greenHouse.getColor()); // green
      System.out.println(anotherHouse.getColor()); // green
   }
}
```

Reference vs Object vs Instance vs Class




```
public class Main {

    public static void main(String[] args) {
        House blueHouse = new House("blue");
        House anotherHouse = blueHouse;

        System.out.println(blueHouse.getColor()); // blue
        System.out.println(anotherHouse.getColor()); // blue

        anotherHouse.setColor("yellow");
        System.out.println(blueHouse.getColor()); // yellow
        System.out.println(anotherHouse.getColor()); // yellow

        House greenHouse = new House("green");
        anotherHouse = greenHouse;

        System.out.println(blueHouse.getColor()); // yellow
        System.out.println(greenHouse.getColor()); // green
        System.out.println(anotherHouse.getColor()); // green
    }
}
```

Summary



- In Java, you always have a references to an object in memory.
- There's no way to access an object directly, everything is done using that reference.

Questions