# Object-Oriented Programming in Java



## Learning objectives

- Object-Oriented Programming
- Classes and objects, static and non-static
- Wrapper types
- Pass by reference or pass by value



## **OOP - Object-Oriented Programming**

- Java is an Object-Oriented Language
- In OOP we often try to create a model of reality with objects
- Classes are used as templates to create objects from
- An object is created with the new keyword



# Objects

- Data is stored in variables
- Behavior is handled by methods
- The variables and methods are not static
- ► There is no main method in an object

Object

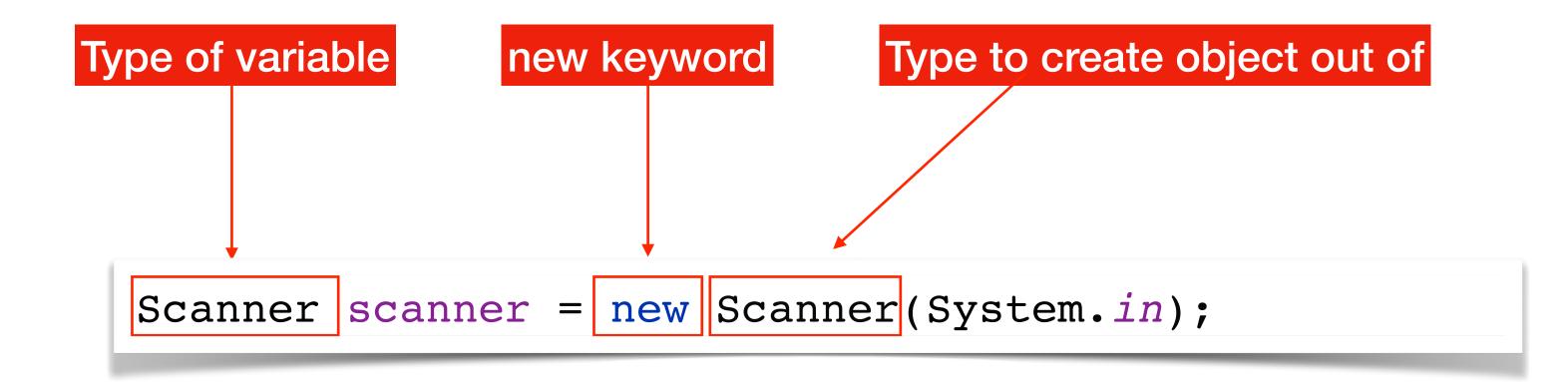
State (Instance Variables)

Behavior (Instance Methods)



## Object references

- Variables are used to access objects
- ▶ But the object isn't really stored inside the variable like a primitive type
- ► The variable stores an object reference that refers to the object in memory
- A variable with no reference to an object has the value null





## Demo 1 - Intro to objects

- Creating a class as an template for an object
- Create the object in runtime with the new keyword
- Instance variables and methods in the object



## Classes and objects

- ► There is only one unique class of every kind in the system
- ► There could be any number of unique objects from the same class
- Static variables belong to the class, not the object
- Instance variables (non-static variables) belong to the object, not the class
- Each object has its own unique data in its instance variables



#### Demo 2 - Static and Non-static

- Calling static methods from the static main method
- Create an object to be able to call a non-static method
- Static Class variables and non-static instance variables



## Exercise 1 - Objects

- Create a class called Rectangle as a template for rectangle objects
- Create the instance variables length and width in the Rectangle class
- Create a method called getArea that should return the area
- Create a method called getPerimiter that should return the perimiter
- In the main method, create two different rectangle objects with different length and width, and print out the area and perimeter of both objects



## Exercise 2 - Objects

- Create a class called Book that represents a book
- Create the instance variables title, author and price inside the Book class
- Create a Book array with a few elements
- ► Then create some book objects with values for title, author and price, and put them in the Book array
- Then loop over the Book array and print the book information (the title, author and price) to the console



### Demo 4 - 00P

- An object-oriented design for a situation where you will heat water, and for your disposal you have a stove, a pot and water.
- Classes are created for representing the stove, the pot and the water
- Objects are created at runtime from the classes and the objects interact for the stove to be able to heat the pot so that the water boils



## Exercise 3 - Objects

- Reuse the solution from the previous exercise (Exercise 2 with the books)
- Create a static method called printBook in the same class as the main method
- ► The method should take a Book object as an input argument, and it should print the information of the book to the console
- Refactor the code so that the book information is not printed in the loop, but instead a call is made in the loop to the new printBook method, and then that method prints the information



#### Wrapper types for primitive data types

- All primitive types have a corresponding wrapper type
- The wrapper type is used to create wrapper objects
- The wrapper class has static and non-static helper methods

Primitive type	Wrapper type
byte	Byte
short	Short
int	Integer
long	Long
float	Float
double	Double
boolean	Boolean
char	Character



# Demo 5 - Wrapper types

- Using wrapper types instead of primitive types
- Using the helper methods in wrapper types

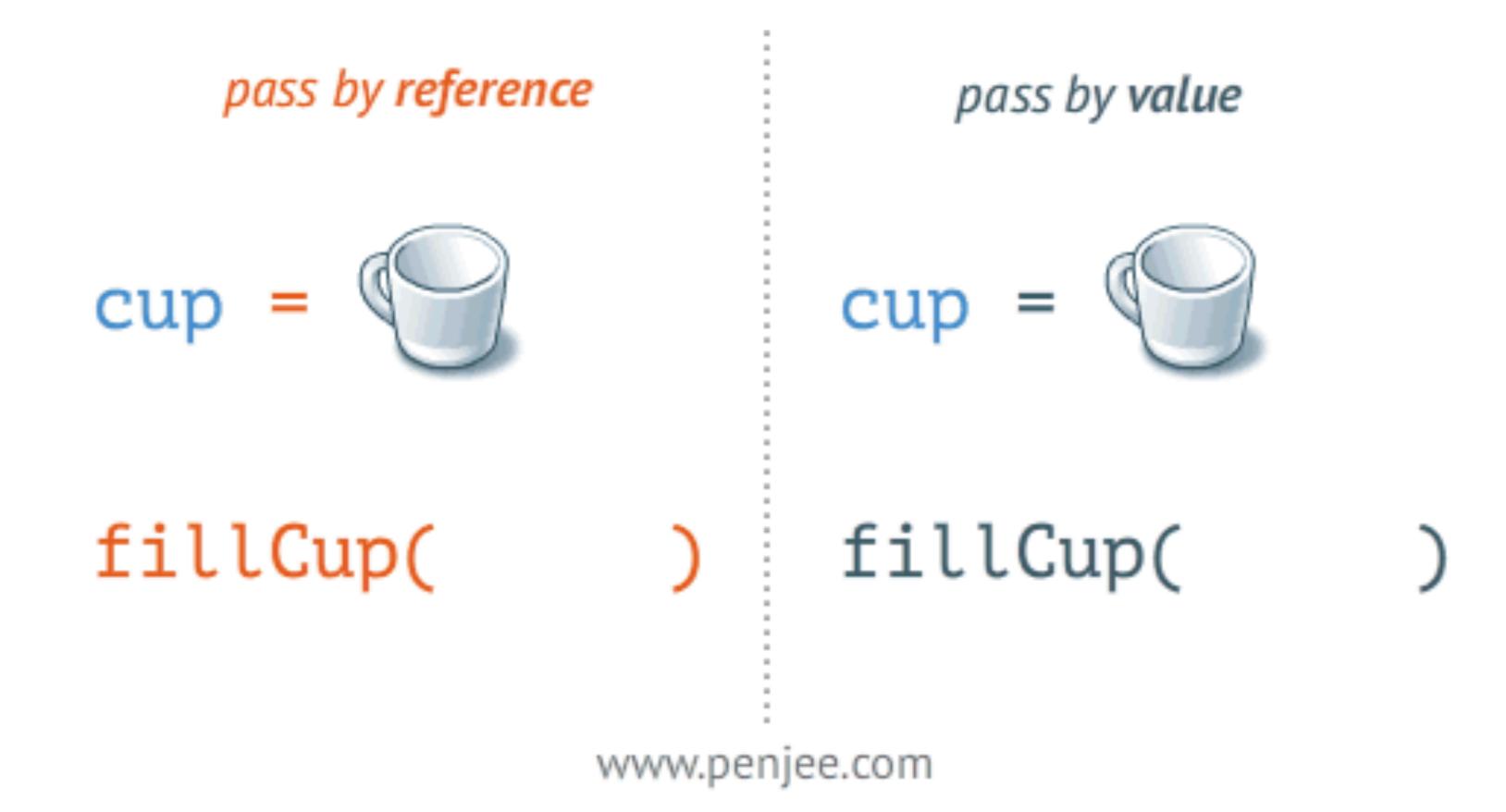


# Pass by reference or by value

- Java is always using pass by value
- For primitive types like an int, the value is the number in the int
- For object types like an Integer, the value is the reference to the object
- So for object types, the reference is passed and not the value of the object, since the value of the variable is only the reference



## Pass by reference or by value





## Demo 5 - Pass by value

- Comparing passing values of primitive types and object types
- Examining the difference of passing the reference or the value



#### Exercise 4 - 00P

- Reuse the solution from Challenges2 Finding the cheapest fruit
- Instead of one int array for the prices and one String array for the names, use one Fruit array with Fruit objects
- Create the Fruit class to be used as the template for the Fruit objects
- Create instance variables inside the Fruit class for name and price



## Learning objectives

- Object-Oriented Programming
- Classes and objects, static and non-static
- Wrapper types
- Pass by reference or pass by value

