JAVA PROGRAMMING BASICS

Module 2: Java Object-oriented Programming

Training program

1. Classes and Instances
2. The Methods
3. **The Constructors**
4. Static Elements
5. Initialization sections
6. Package
7. Inheritance and Polymorphism
8. Abstract classes and Interfaces
9. String processing
10. Wrapper classes for primitive types
11. Exceptions and Assertions
12. Nested classes
13. Enums
14. Generics
15. Collections
16. Method overload resolution
17. Multithreads
18. Core Java classes
19. Object Oriented Design
20. Functional Programming

Module contents

- The Constructors
  - Constructors declaration
  - Default Constructors
  - Constructors overloading
  - Calling one constructor from another

Module contents

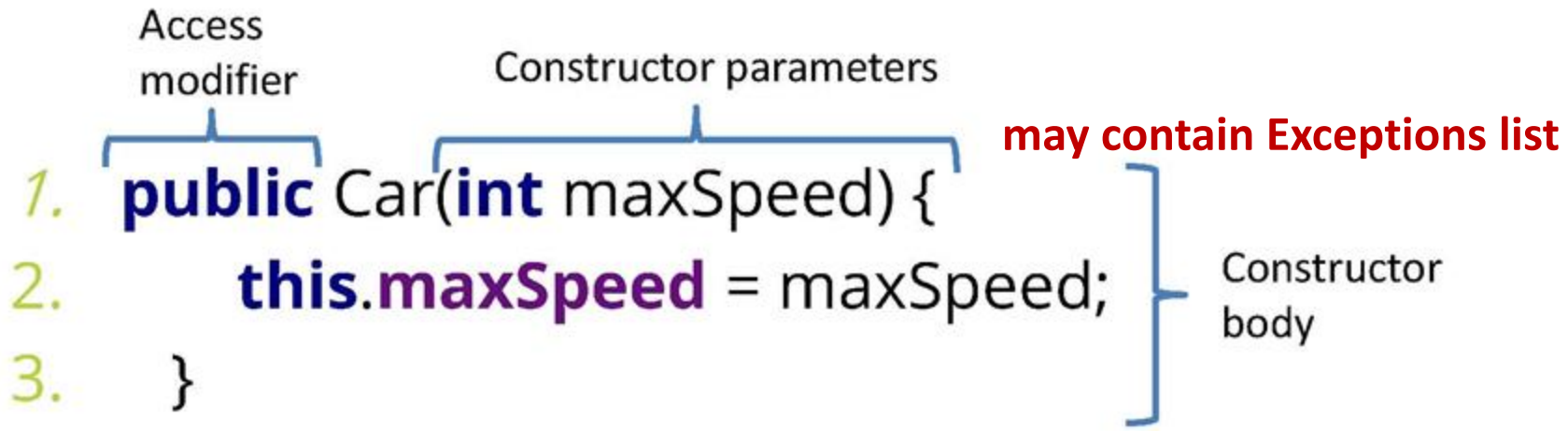
- The Constructors
  - Constructors declaration
  - Default Constructors
  - Constructors overloading
  - Calling one constructor from another

Constructor declaration 1/4

```
1.  public class Car {
2.      private int maxSpeed;
3.      //...
4.      public Car(int maxSpeed) {
5.          this.maxSpeed = maxSpeed;
6.      }
7.      //...
8.      public int getMaxSpeed() {
9.          return maxSpeed;
10.     }
11. }
```

Constructor declaration 2/4

- Constructors are special methods defined in a class that used while class instances are created for the instances initialization.



Constructor declaration 3/4

Get a block of unused memory in the heap, large enough to hold an object of the specified type

Call the default constructor of the superclass if no constructor is defined

Initialize member variables to the specified values (or default initial value is used)

Executes the body of the constructor

Module contents

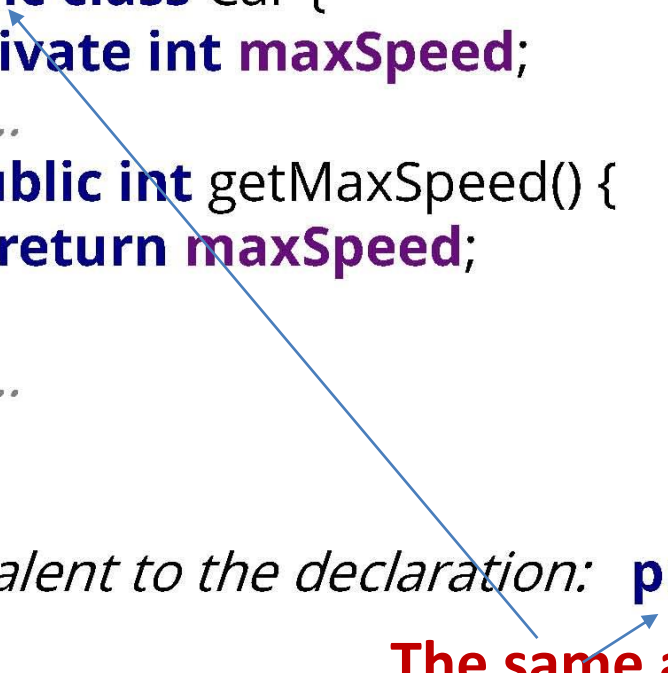
- The Methods
  - Constructors declaration
  - Default Constructors
  - Constructors overloading
  - Calling one constructor from another

Default Constructors 1/4

- When we don't define any constructor, the compiler creates the default constructor

```
1.  public class Car {
2.      private int maxSpeed;
3.      //...
4.      public int getMaxSpeed() {
5.          return maxSpeed;
6.      }
7.      //...
8.  }
```

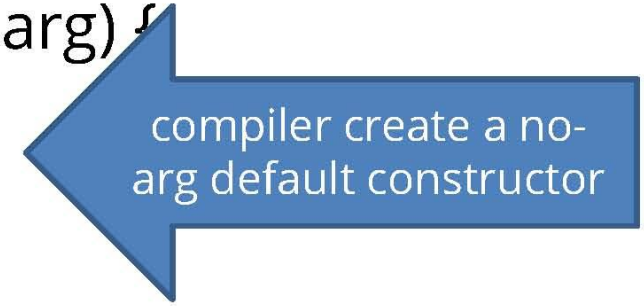
- *equivalent to the declaration:* **public** Car() { }



The same access modifier as the class

Default Constructors 2/4

```
1.  public class Main {
2.      public static void main(String[] arg) {
3.          Car myCar1 = new Car();
4.          Car myCar2 = new Car();
5.          Car myCar3 = new Car();
6.          System.out.println(myCar1.getMaxSpeed());
7.          System.out.println(myCar2.getMaxSpeed());
8.          System.out.println(myCar3.getMaxSpeed());
9.      }
10. }
```



Default Constructors 3/4

- When we define any constructor, the compiler don't creates the default constructor

```
1.  public class Car {
2.      private int maxSpeed;
3.      //...
4.      public Car(int maxSpeed) {
5.          this.maxSpeed = maxSpeed;
6.      }
7.      //...
8.      public int getMaxSpeed() {
9.          return maxSpeed;
10.     }
11. }
```

Default Constructors 4/4

```
1.  public class Main {
2.      public static void main(String[] arg)
3.          Car myCar1 = new Car();
4.          Car myCar2 = new Car(100);
5.          Car myCar3 = new Car(120);
6.          System.out.println(myCar1.getMaxSpeed());
7.          System.out.println(myCar2.getMaxSpeed());
8.          System.out.println(myCar3.getMaxSpeed());
9.      }
10. }
```

compiler doesn't create a no-arg constructor

See objcreation

Module contents

- The Methods
  - Constructors declaration
  - Default Constructors
  - Constructors overloading
  - Calling one constructor from another

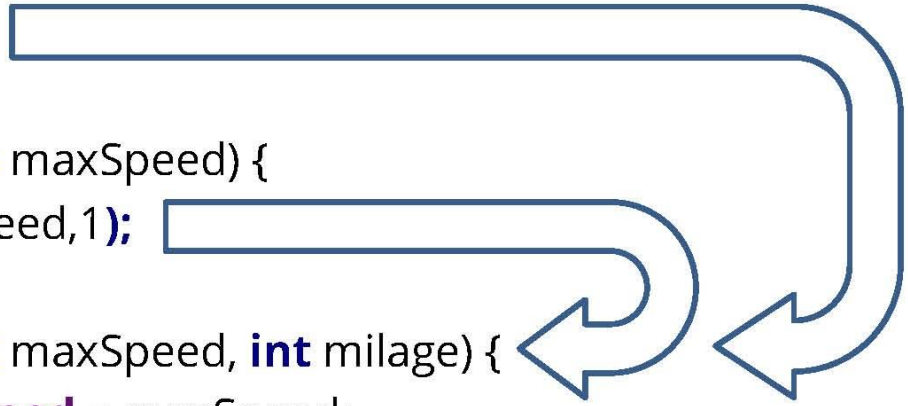
See constructoroverloadpkg

Module contents

- The Methods
  - Constructors declaration
  - Default Constructors
  - Constructors overloading
  - Calling one constructor from another

Calling one constructor from another 1/3

```
1.  public class Car {
2.      private int maxSpeed, milage;
3.      public Car() {
4.          this(100,1) ;
5.      }
6.      public Car(int maxSpeed) {
7.          this(maxSpeed,1);
8.      }
9.      public Car(int maxSpeed, int milage) {
10.         this.maxSpeed = maxSpeed;
11.         this.milage = milage;
12.     }
13. }
```



explicit constructor invocation

See constructoroverloadpkg

This() call must be first!!!