

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. Exceptions and Assertions
11. Nested classes
12. Enums
13. Wrapper classes for primitive types
14. Generics
15. Collections
16. Method overload resolution
17. Multithreads
18. Core Java Classes
19. Object Oriented Design

Module contents

- Static elements
 - The static keyword
 - The static fields
 - The static final fields
 - The static methods
 - The static methods restrictions
 - Using static fields and methods

Module contents

- Static elements
 - The static keyword
 - The static fields
 - The static final fields
 - The static methods
 - The static methods restrictions
 - Using static fields and methods

The static keyword 1/2

There are different uses of static keyword in java:

- static fields
- static methods
- static classes
- static blocks
- static import

Module contents

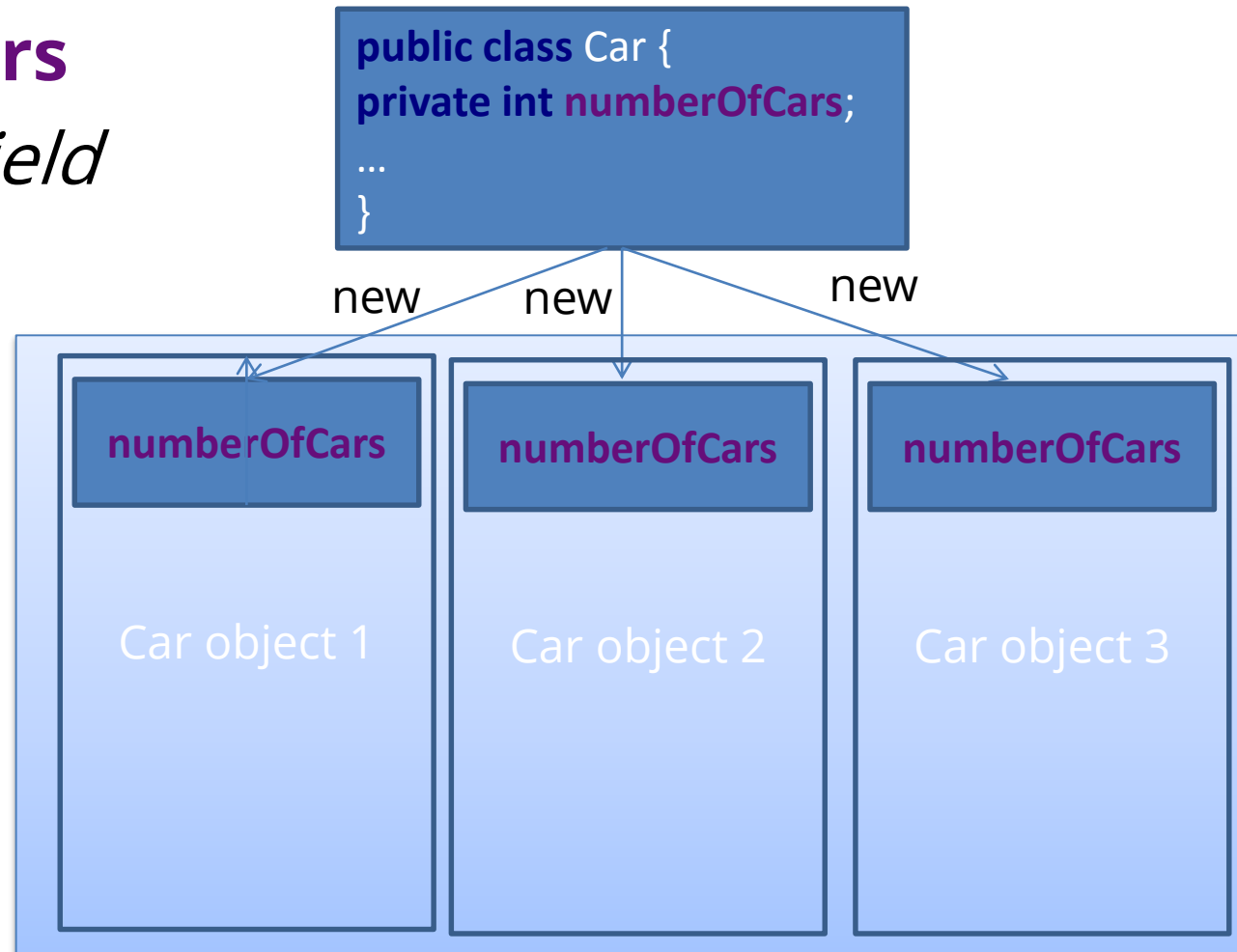
- Static elements
 - The static keyword
 - The static fields
 - The static final fields
 - The static methods
 - The static methods restrictions
 - Using static fields and methods

The static keyword 2/2

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

The static fields 1/5

numberOfCars
as Instance Field



The static fields 2/5

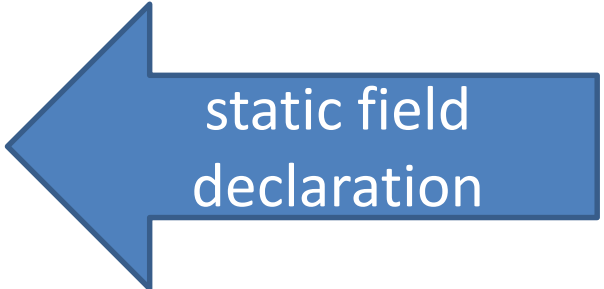
- **numberOfCars as Instance Field**

```
1. public static void main(String[] arg) {  
2.     Car myCar1 = new Car();  
3.     Car myCar2 = new Car();  
4.     Car myCar3 = new Car();  
5.     System.out.println(myCar1.getNumOfCars());  
6. }
```

Result : 1

The static fields 3/5

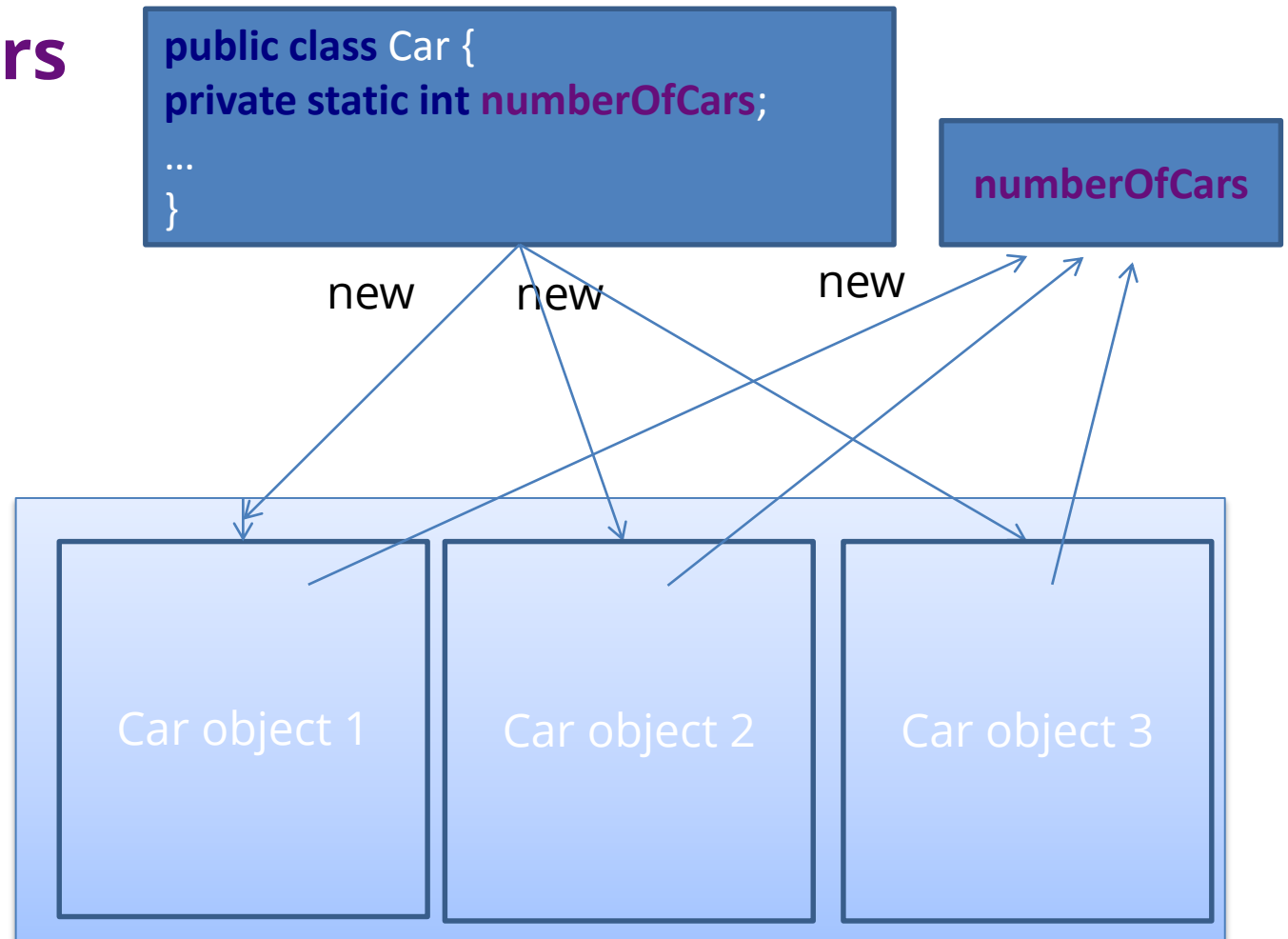
```
1. public class Car {  
2.     private static int numOfCars;  
3.     //...  
4.     public Car() {  
5.         numOfCars++;  
6.     }  
7.     //...  
8.     public int getNumOfCars() {  
9.         return numOfCars;  
10.    }  
11. }
```



static field
declaration

The static fields 4/5

numberOfCars
as static Field



The static fields 5/5

- **numberOfCars as static Field**

```
1. public static void main(String[] arg) {  
2.     Car myCar1 = new Car();  
3.     Car myCar2 = new Car();  
4.     Car myCar3 = new Car();  
5.     System.out.println(myCar1.getNumOfCars());  
6. }
```

Result : 3

Module contents

- Static elements
 - The static keyword
 - The static fields
 - The static final fields
 - The static methods
 - The static methods restrictions
 - Using static fields and methods

The static final fields 1/3

- The static modifier, in combination with the final modifier, is also used to define constants.

```
1. public class Car {  
2.     //...  
3.     public static final int NUM_OF_WHEELS = 4;  
4.     //...  
5. }
```

The static final fields 2/3

- It is a compile-time error if a static final variable is not definitely assigned by a static initializer of the class in which it is declared.

```
1. public class Car {  
2.     //...  
3.     public static final int NUM_OF_WHEELS;  
4.     //...  
5. }
```

The static final fields 3/3

- The final modifier indicates that the value of this field cannot change.

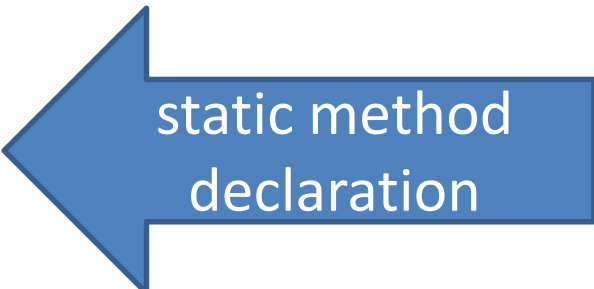
```
1. public class Car {  
2.     public static final int NUM_OF_WHEELS = 4;  
3.     //...  
4.     public Car() {  
5.         NUM_OF_WHEELS = 3;  
6.     }  
7. }
```

Module contents

- Static elements
 - The static keyword
 - The static fields
 - The static final fields
 - The static methods
 - The static methods restrictions
 - Using static fields and methods

The static methods 1/3

```
1. public class Car {  
2.     private static int numOfCars;  
3.     //...  
4.     public Car() {  
5.         numOfCars++;  
6.     }  
7.     //...  
8.     public static int getNumOfCars() {  
9.         return numOfCars;  
10.    }  
11. }
```

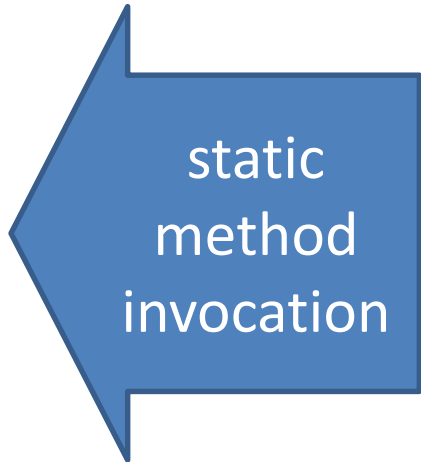


static method
declaration

The static methods 2/3

- Static methods should be invoked with the class name:

```
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.         int numOfCars = Car.getNumOfCars();  
7.         System.out.println(numOfCars);  
8.     }  
9. }
```

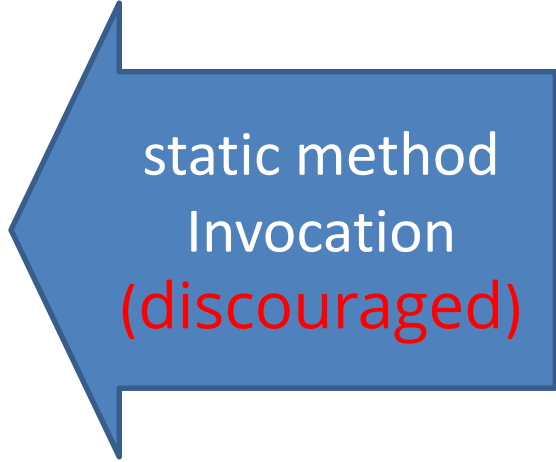


static
method
invocation

The static methods 3/3

- You can also refer to static methods with an object reference but this is discouraged:

```
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.         int numOfCars = myCar1.getNumOfCars();  
7.         System.out.println(numOfCars);  
8.     }  
9. }
```



static method
Invocation
(discouraged)

Module contents

- Static elements
 - The static keyword
 - The static fields
 - The static final fields
 - The static methods
 - The static methods restrictions
 - Using static fields and methods

The static methods restrictions 1/3

- Instance methods can access instance variables and instance methods directly.
- Instance methods can access class variables and class methods directly.
- Class methods can access class variables and class methods directly.

The static methods restrictions 2/3

- Class methods **cannot** access instance variables or instance methods directly—they must use an object reference.

```
1. public class Car {  
2.     public static final int NUM_OF_WHEELS = 4;  
3.     private int milage;  
4.     //...  
5.     public static void testCar(){  
6.         int numOfBolt = NUM_OF_WHEELS*5; // OK!  
7.         int numOfDiag = milage/4000;  
8.     }  
9. }
```



The static methods restrictions 3/3

- Also, static methods cannot use the ***this*** keyword as there is no instance for this to refer to.

```
1. public class Car {  
2.     public static final int NUM_OF_WHEELS = 4;  
3.     private int milage;  
4.     //...  
5.     public static void testCar(){  
6.         int numOfBolt = NUM_OF_WHEELS * 5; // OK!  
7.         int numOfDiag = this.milage / 4000;  
8.     }  
9. }
```



Using static fields and methods

- Static elements
 - The static keyword
 - The static fields
 - The static final fields
 - The static methods
 - The static methods restrictions
 - Using static fields and methods

Using static fields and methods

- Statics should be used when data is not instance dependent and for all existing instances of static member you want to apply same state.
- Static should be used for the methods where you have mostly utility methods in the class and we hardly need to maintain any object state to achieve the required action.