Fields and starting with Constructors

1. Declaring fields
2. Constructors
3. Default constructor
4. Accessing object fields

Declaring fields

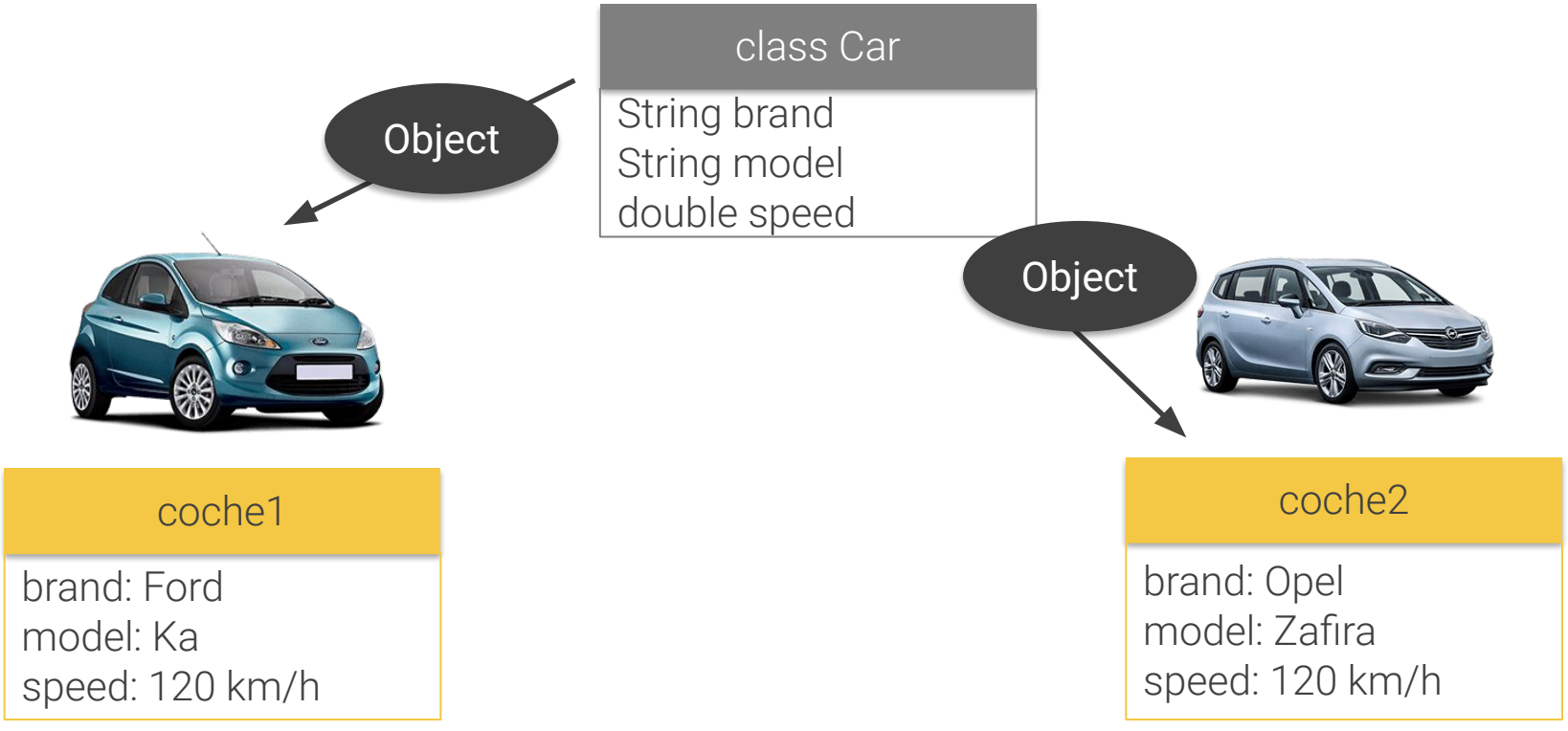
1. Declaring fields

```
public class MyClass {

 // field declarations
 // constructors
 // method implementations

}
```

1. Declaring fields

Atributos (fields): características de un objeto

1. Declaring fields



1. Declaring fields

Atributos (fields): características de un objeto

Dependen del escenario

1. Declaring fields

APPLICATION: CAR RACING GAME

class Car

String brand
String model
double speed

Object

coche1

brand: Ford
model: Ka
speed: 120 km/h

APPLICATION: RENT A CAR

class Car

String type
String brand
String model
int seats
double price
boolean isRented

Object

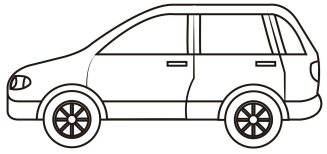
coche1

type: Berlina
brand: Ford
model: Ka
seats:4
price: 25.99€/día
isRented:true

1. Declaring fields

```
public class Car{

 //field declarations
   String brand;
   String model;
   double speed;

 //constructors
 //method implementations
}
```

Field declarations = Variable declarations

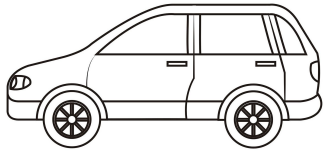


1. Declaring fields

```
public class Car{

 //field declarations
   String brand;
   String model;
   double speed;

}
```

Data type

Field declarations = Variable declarations

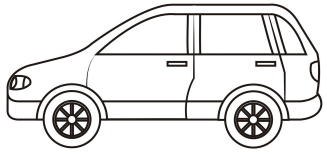


1. Declaring fields

```
public class Car{

 //field declarations
   String brand;
   String model;
   double speed;

}
```

Identifier

Field declarations = Variable declarations

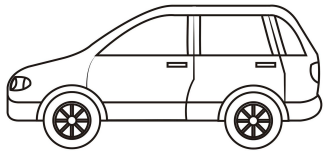


1. Declaring fields

```
public class Car{

 //field declarations
   String brand;
   String model;
   double speed;

}
```

lower camel Case

Field declarations = Variable declarations

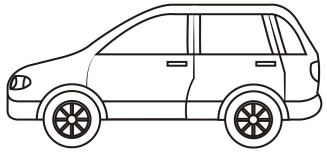


1. Declaring fields

```
public class Car{

 //field declarations
   String brand;
   String model;
   double speed;

 //constructors
 //method implementations
}
```

Primitive variables

Object variables

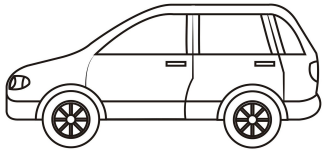


1. Declaring fields

```
public class Car{

 //field declarations
   String brand;
   String model;
   double speed;

 //constructors
 //method implementations
}
```

Primitive variables

Object variables

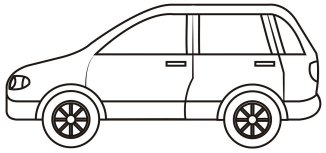


1. Declaring fields

```
public class Car{

 //field declarations
   String brand;
   String model;
   double speed;

 //constructors
 //method implementations
}
```

Primitive variables

Object variables



1. Declaring fields

```
public class Producto{

 //field declarations
    String nombre;
    double precio;
    int stock;

}
```

1. Declaring fields

```
public class Producto{

 //field declarations
    String nombre;
    double precio;
    int stock;

}
```

Primitive variables

Object variables

1. Declaring fields

```
public class Producto{

 //field declarations
    String nombre;
    double precio;
    int stock;

}
```

Primitive variables

Object variables

1. Declaring fields

```
public class Supermercado{

 //field declarations
    ArrayList<Producto> productos;
    String nombre;
    String direccion;
    String telefono;
    String email;

}
```

1. Declaring fields

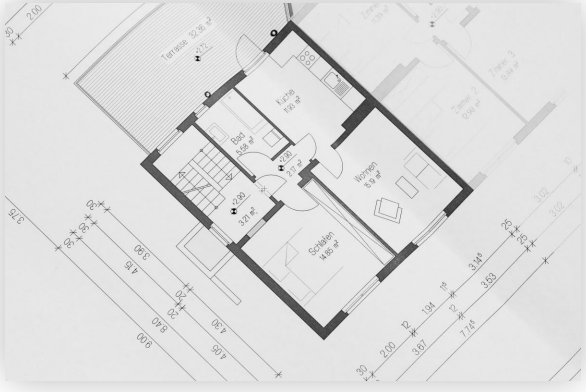
```
public class Supermercado{

 //field declarations
    ArrayList<Producto> productos;
    String nombre;
    String direccion;
    String telefono;
    String email;

}
```

1. Declaring fields

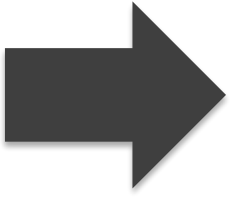
```
public class Supermercado{

 //field declarations
    ArrayList<Producto> productos;
    String nombre;
    String direccion;
    String telefono;
    String email;

}
```

Constructors

Constructors are invoked to create objects from the class blueprint



Class blueprint

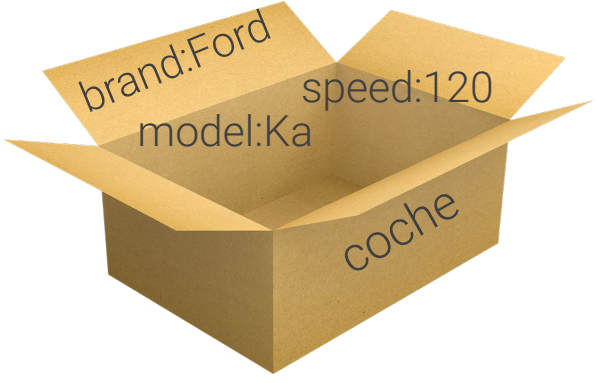


Object created from the class blueprint

Object

class Car
String brand String model double speed

coche
brand: Ford model: Ka speed: 120 km/h



2. Constructors

Default constructor

Is the no-argument constructor automatically generated unless you define another constructor

Defined constructors

Default constructor

3. Default constructor

Default constructor

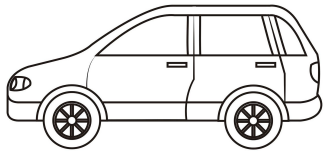
Defined constructors

Is the no-argument constructor automatically generated unless you define another constructor

3. Default constructor

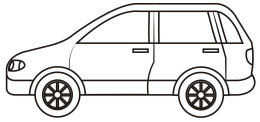
```
public class Car{

 //field declarations
   String brand;
   String model;
   double speed;

 //constructors

 //method implementations
   public double applyBrake(double decrement) {
      speed -= decrement;
      return speed;
   }

}
```

No constructor implementations:
Default constructor is available



Creating objects with Default Constructor

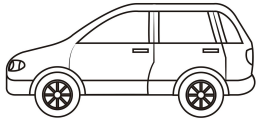
```
public class Car{

 //field declarations
   String brand;
   String model;
   double speed;

 //constructors

 //method implementations
   public double applyBrake(double decrement) {
      speed -= decrement;
      return speed;
   }
}
```



```
Car coche = new ¿?;
```

Creating objects with Default Constructor

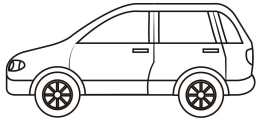
```
public class Car{

 //field declarations
   String brand;
   String model;
   double speed;

 //constructors

 //method implementations
   public double applyBrake(double decrement) {
      speed -= decrement;
      return speed;
   }
}
```



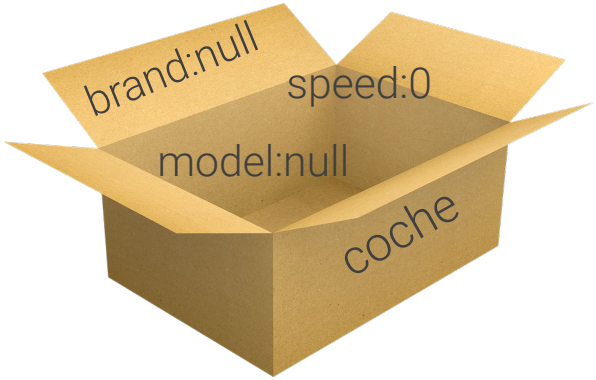
```
Car coche = new Car();
```

Creating objects with Default Constructor

```
public class Car{

 //field declarations
   String brand;
   String model;
   double speed;

 //constructors

 //method implementations
   public double applyBrake(double decrement) {
      speed -= decrement;
      return speed;
   }
}
```



```
Car coche = new Car();
```

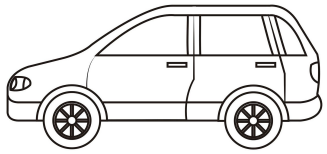


3. Default constructor

```
public class Car{

 //field declarations
   String brand;
   String model;
   double speed;

 //constructors

 //method implementations
   public double applyBrake(double decrement) {
      speed -= decrement;
      return speed;
   }
}
```

Variable declarations

Declaration is not to declare "value" to a variable; it's to declare the type of the variable



3. Default constructor

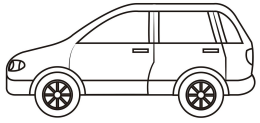
Data Type	Descripción	Valor por defecto
long	Número entero	0
int		0
short		0
byte		0
double	Número decimal	0
float		0

3. Default constructor

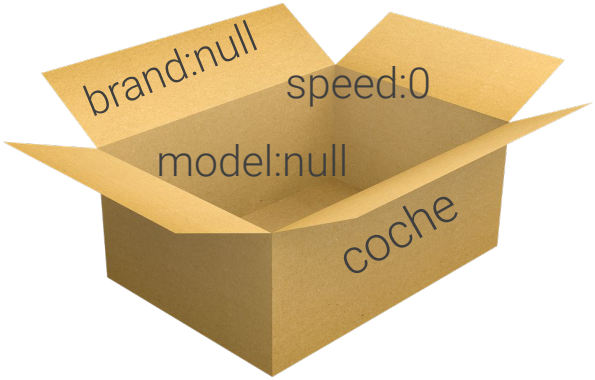
Data Types	Descripción	Valor por defecto
boolean	valor lógico	false
char	carácter unicode	nul
String	cadena de caracteres	null
XXX	Cualquier objeto	null

Creating objects with Default Constructor

```
public class Car{

 //field declarations
   String brand;
   String model;
   double speed;

 //constructors

 //method implementations
   public double applyBrake(double decrement) {
      speed -= decrement;
      return speed;
   }
}
```

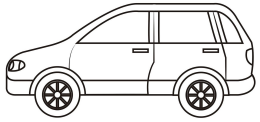


```
Car coche = new Car();
```



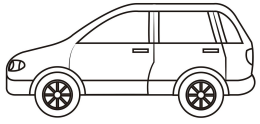
Creating objects with Default Constructor

```
public class Car{

 //field declarations
   String brand;
   String model;
   double speed=100;

 //constructors

 //method implementations
   public double applyBrake(double decrement) {
      speed -= decrement;
      return speed;
   }
}
```

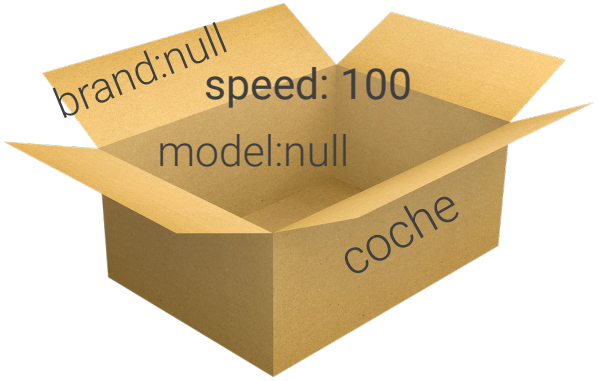


Creating objects with Default Constructor

```
public class Car{

 //field declarations
   String brand;
   String model;
   double speed=100;

 //constructors

 //method implementations
   public double applyBrake(double decrement) {
      speed -= decrement;
      return speed;
   }
}
```



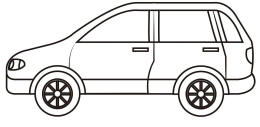
```
Car coche = new Car();
```



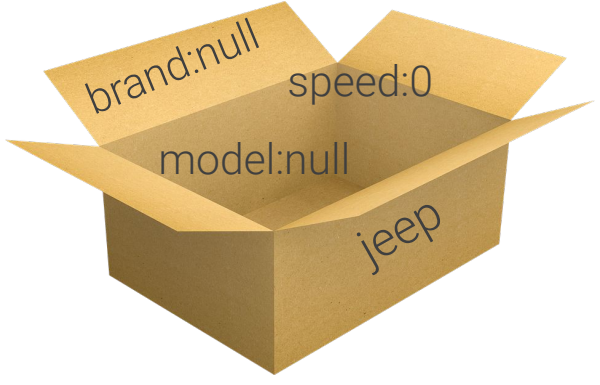
Accessing object fields

4. Accessing object fields

```
public class Car{

 //field declarations
   String brand;
   String model;
   double speed;

 //constructors

 //method implementations
   public double applyBrake(double decrement) {
      speed -= decrement;
      return speed;
   }
}
```

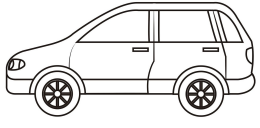


```
Car jeep = new Car();
```

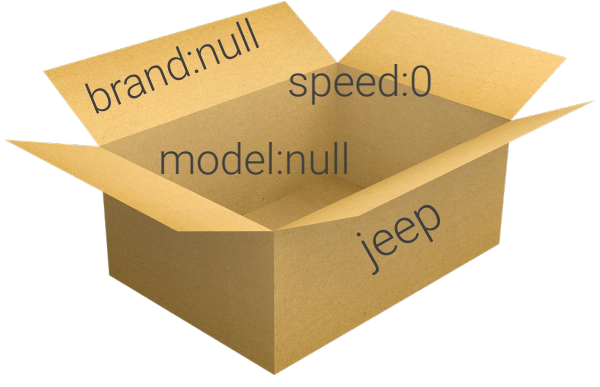


4. Accessing object fields

```
public class Car{

 //field declarations
   String brand;
   String model;
   double speed;

 //constructors

 //method implementations
   public double applyBrake(double decrement) {
      speed -= decrement;
      return speed;
   }

}
```

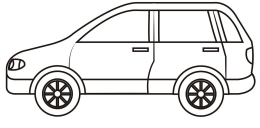


```
Car jeep = new Car();
jeep.speed
```

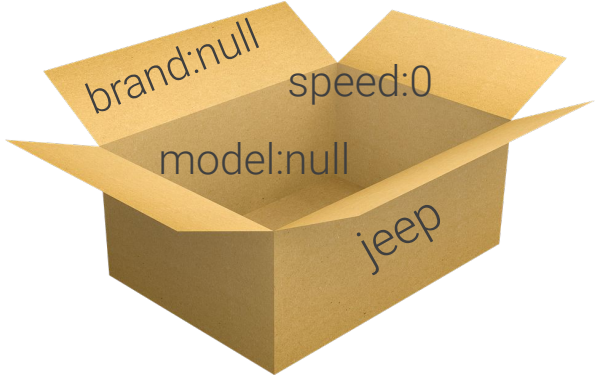


4. Accessing object fields

```
public class Car{

 //field declarations
   String brand;
   String model;
   double speed;

 //constructors

 //method implementations
   public double applyBrake(double decrement) {
      speed -= decrement;
      return speed;
   }

}
```

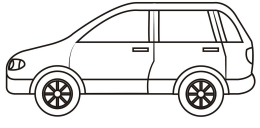


```
Car jeep = new Car();
double n = jeep.speed
```

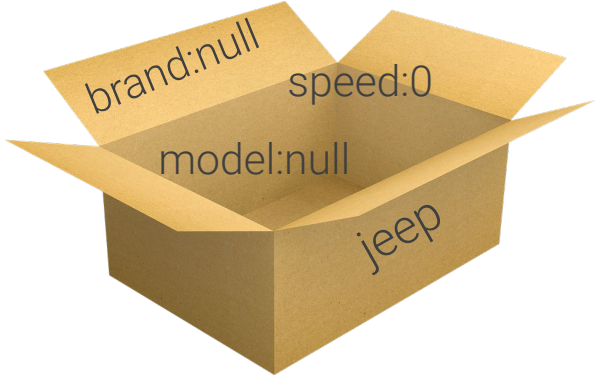


4. Accessing object fields

```
public class Car{

 //field declarations
   String brand;
   String model;
   double speed;

 //constructors

 //method implementations
   public double applyBrake(double decrement) {
      speed -= decrement;
      return speed;
   }

}
```

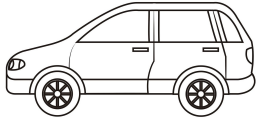


```
Car jeep = new Car();
System.out.println(jeep.speed);
```

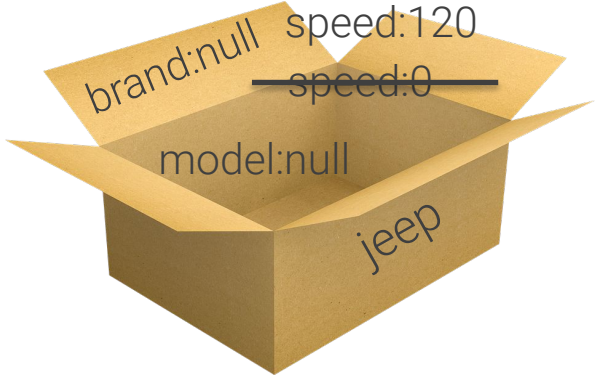


4. Accessing object fields

```
public class Car{

 //field declarations
   String brand;
   String model;
   double speed;

 //constructors

 //method implementations
   public double applyBrake(double decrement) {
      speed -= decrement;
      return speed;
   }

}
```

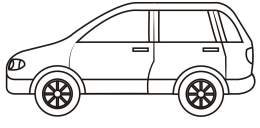


```
Car jeep = new Car();
jeep.speed = 120;
```

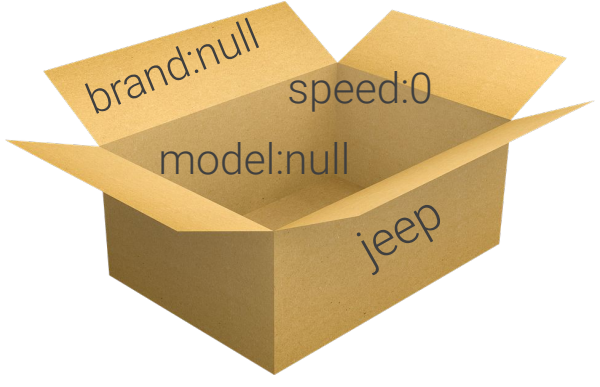


4. Accessing object fields

```
public class Car{

 //field declarations
   String brand;
   String model;
   double speed;

 //constructors

 //method implementations
   public double applyBrake(double decrement) {
      speed -= decrement;
      return speed;
   }

}
```

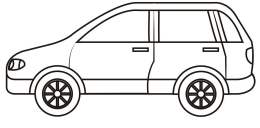


```
Car jeep = new Car();
jeep.speed = "Ford";
```



4. Accessing object fields

```
public class Car{

 //field declarations
   String brand;
   String model;
   double speed;

 //constructors

 //method implementations
   public double applyBrake(double decrement) {
      speed -= decrement;
      return speed;
   }

}
```



```
Car jeep;
jeep.speed = 120;
System.out.println(jeep.speed);
```

Compilation error:
Variable 'jeep' might not have been initializated

“La paciencia es un elemento clave del éxito.“

Bill Gates, empresario e informático estadounidense cofundador de la empresa Microsoft