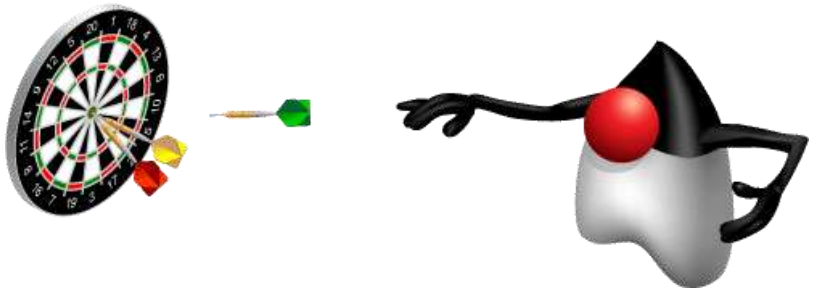
Describing Objects and Classes

Objectives

After completing this lesson, you should be able to:

- List the characteristics of an object
- Define an object as an instance of a class
- Instantiate an object and access its fields and methods
- Describe how objects are stored in memory
- Instantiate an array of objects
- Describe how an array of objects is stored in memory
- Declare and instantiate an object as a field
- Use the NetBeans IDE to create and test Java classes

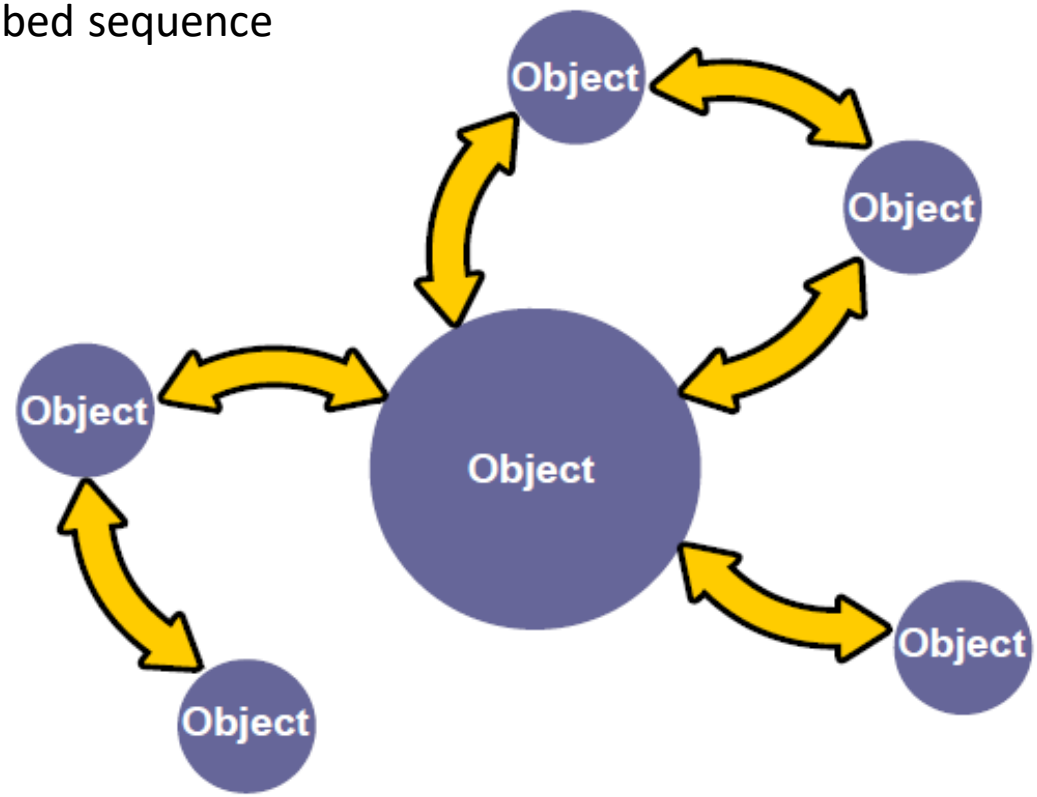


Topics

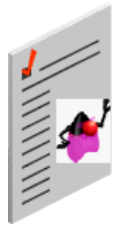
- Describing objects and classes
- Defining fields and methods
- Declaring, instantiating, and using objects
- Working with object references
- Doing more with arrays
- Introducing NetBeans IDE
- Introducing the soccer league use case

Object-Oriented Programming

- Interaction of objects
- No prescribed sequence



Characteristics of Objects



Objects are physical or conceptual.

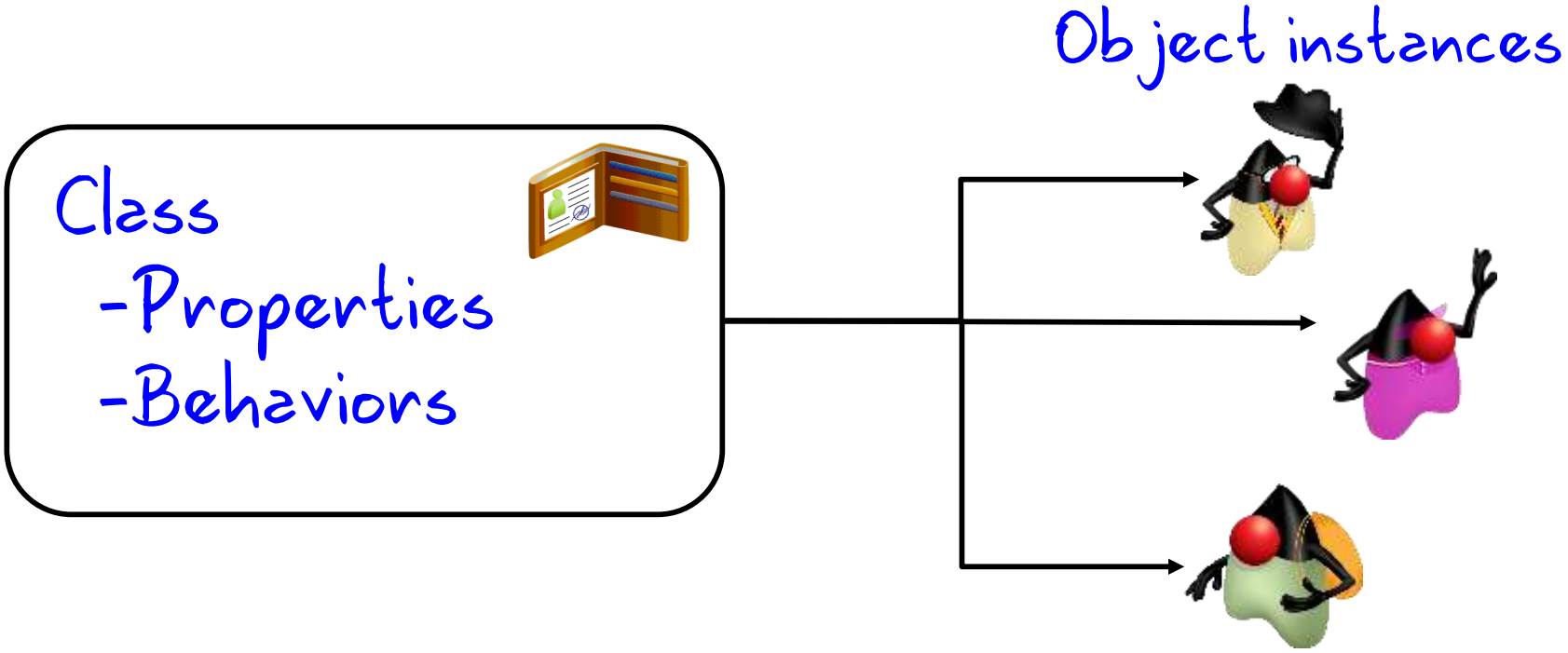
- Objects have **properties**:
  - Size
  - Shape
  - Name
  - Color
- Objects have **behaviors**:
  - Shop
  - Put item in cart
  - Pay

Mrs. Duke

Color property value is red

Classes and Instances

- A class:
  - Is a blueprint or recipe for an object
  - Describes an object's properties and behaviors
  - Is used to create object instances



Quiz

Which of the following statements is true?

a. An object is a blueprint for a class.
b. An object and a class are exactly the same.
c. An object is an instance of a class.
d. A class is an instance of an object.

Topics

- Describing objects and classes
- Defining fields and methods
- Declaring, instantiating, and using objects
- Working with object references
- Doing more with arrays
- Introducing NetBeans IDE
- Introducing the soccer league use case

The Customer Properties and Behaviors

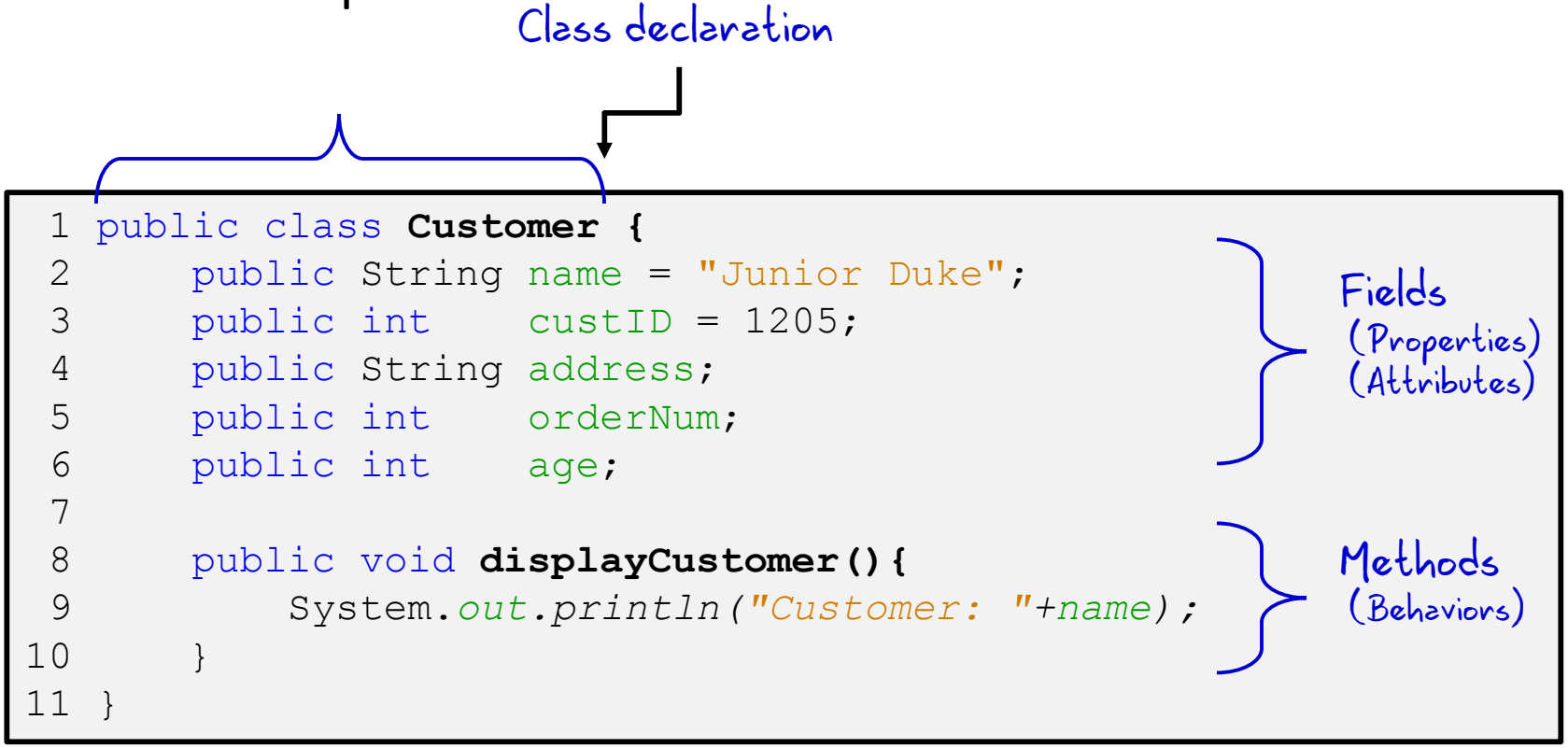
Properties:

- Name
- Address
- Age
- Order number
- Customer number

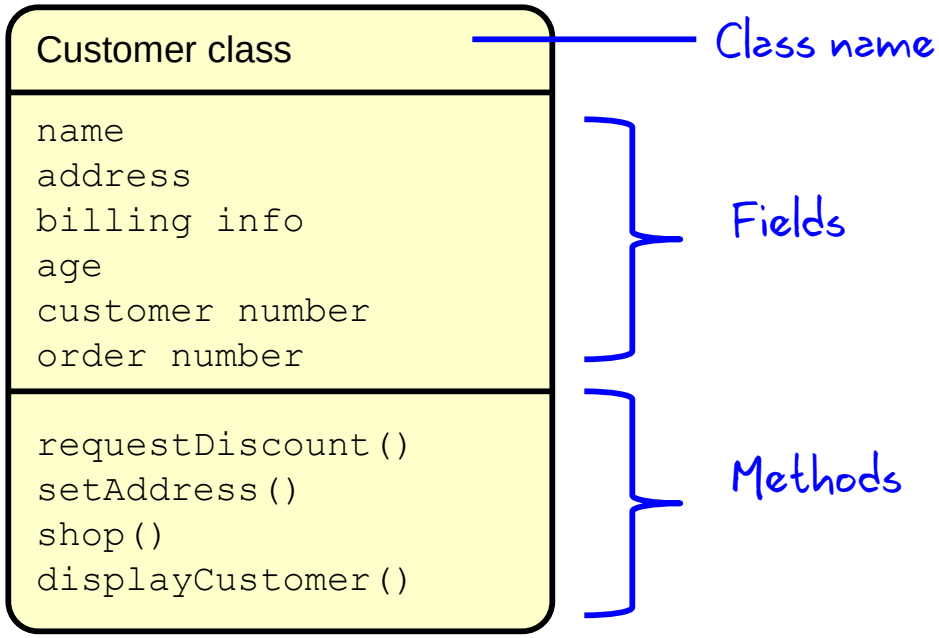
Behaviors:

- Shop
- Set Address
- Add item to cart
- Ask for a discount
- Display customer details

The Components of a Class



Modeling Properties and Behaviors



Exercise 6-1: Creating the Item Class

In this exercise, you create the Item class and declare public fields for ID (`int`), descr, quantity (`int`), and price (`double`).

Topics

- Describing objects and classes
- Defining fields and methods
- Declaring, instantiating, and using objects
- Working with object references
- Doing more with arrays
- Introducing NetBeans IDE
- Introducing the soccer league use case

Customer Instances

customer01

customer02

```
public static void main(String[] args){

    Customer customer01 = new Customer();
    Customer customer02 = new Customer();

    customer01.age = 40;
    customer02.name = "Duke";

    customer01.displayCustomer();
    customer02.displayCustomer();
    }
}
```

Create new instances (instantiate).

Fields are accessed.

Methods are called.

Object Instances and Instantiation Syntax

variable becomes a reference to that object.

The new keyword creates (instantiates) a new instance.

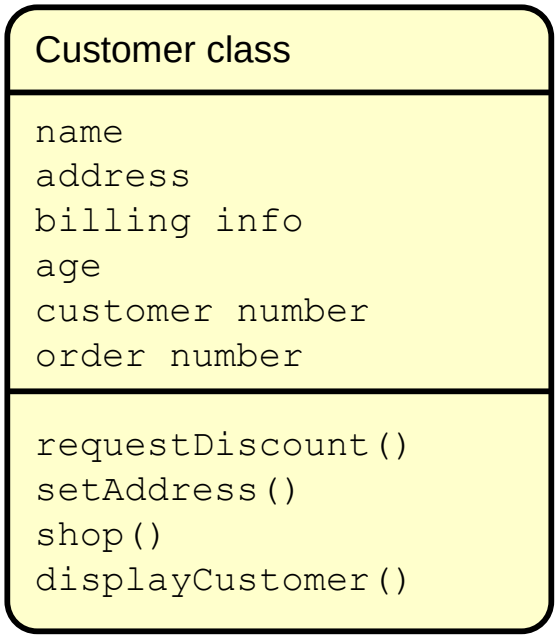
The syntax is:

```
<class name> variable = new <class name>()
```

```
public static void main(String[] args){

    Customer customer01 = new Customer();   //Declare and instantiate

    Customer customer02;                    //Declare the reference
    customer02 = new Customer();            //Then instantiate

    new Customer();                 //Instantiation without a reference
                                    //We can't use this object later
                                    //without knowing how to reference it.
    }
}
```

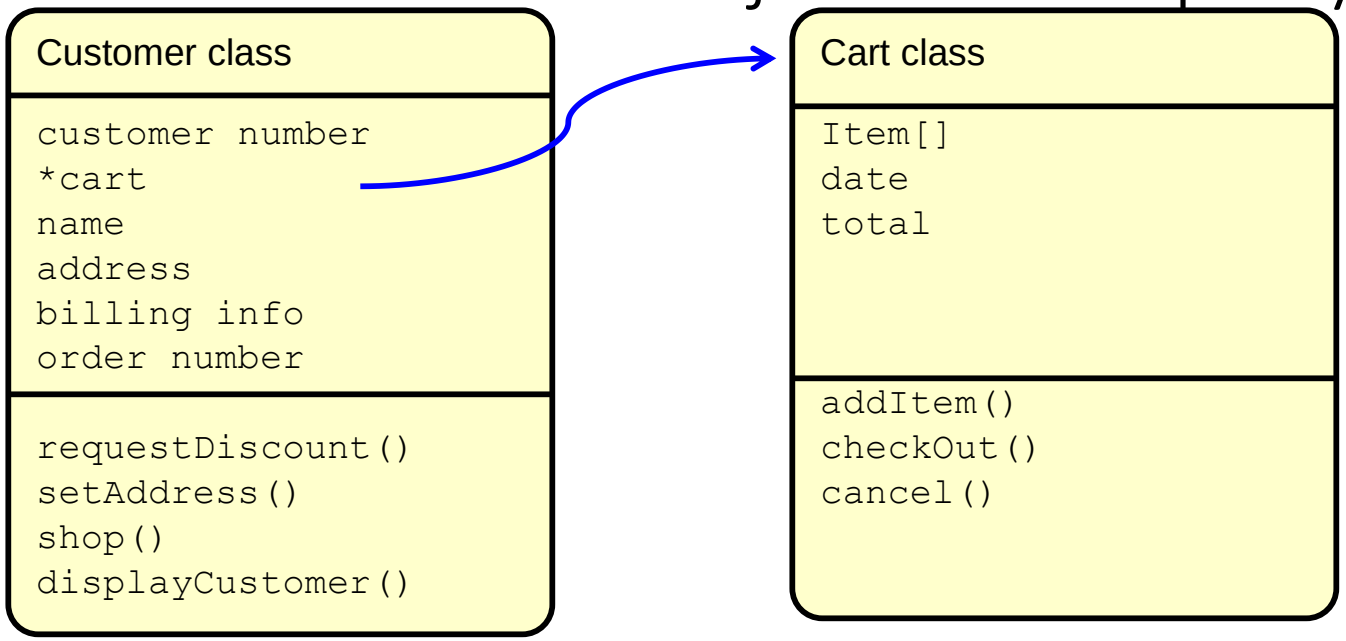
The Dot (.) Operator

Follow the reference variable with a dot operator (.) to access the fields and methods of an object.




```
public static void main(String[] args){

    Customer customer01 = new Customer();

    //Accessing fields
    System.out.println(customer01.name);
    customer01.age = 40;

    //Calling methods
    customer01.requestDiscount();
    customer01.displayCustomer();
    }
}
```

Objects with Another Object as a Property




```
public static void main(String[] args){

    Customer customer01 = new Customer();
    customer01.cart.cancel();                  //How to access methods of an
                                               //object within another object
}
```

Quiz

Which of the following lines of code instantiates a `Boat` object and assigns it to a `sailBoat` object reference?

a. `Boat sailBoat = new Boat();`
b. `Boat sailBoat;`
c. `Boat = new Boat()`
d. `Boat sailBoat = Boat();`

Topics

- Describing objects and classes
- Defining fields and methods
- Declaring, instantiating, and using objects
- Working with object references
- Doing more with arrays
- Introducing NetBeans IDE
- Introducing the soccer league use case

Accessing Objects by Using a Reference

The camera is like the **object** that is accessed using a reference.

The remote is like the **reference** used to access the camera.

Working with Object References

1 Pick up the remote to gain access to the camera.

2 Press the remote's controls to have camera do something.

1 Create a Camera object and get a reference to it.

```
11 Camera remote1;
12
13 remote1 =  new Camera();
14
15 remove1.play();
```

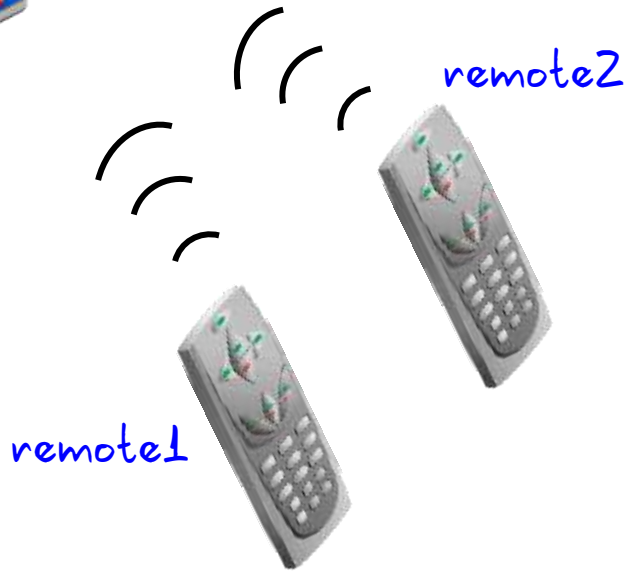
2 Call a method to have the Camera object do something.

Working with Object References


```
12  Camera remote1 = new Camera();
13
14  Camera remote2 = new Camera();
15
16  remote1.play();
17
18  remote2.play();
```

There are two Camera objects.

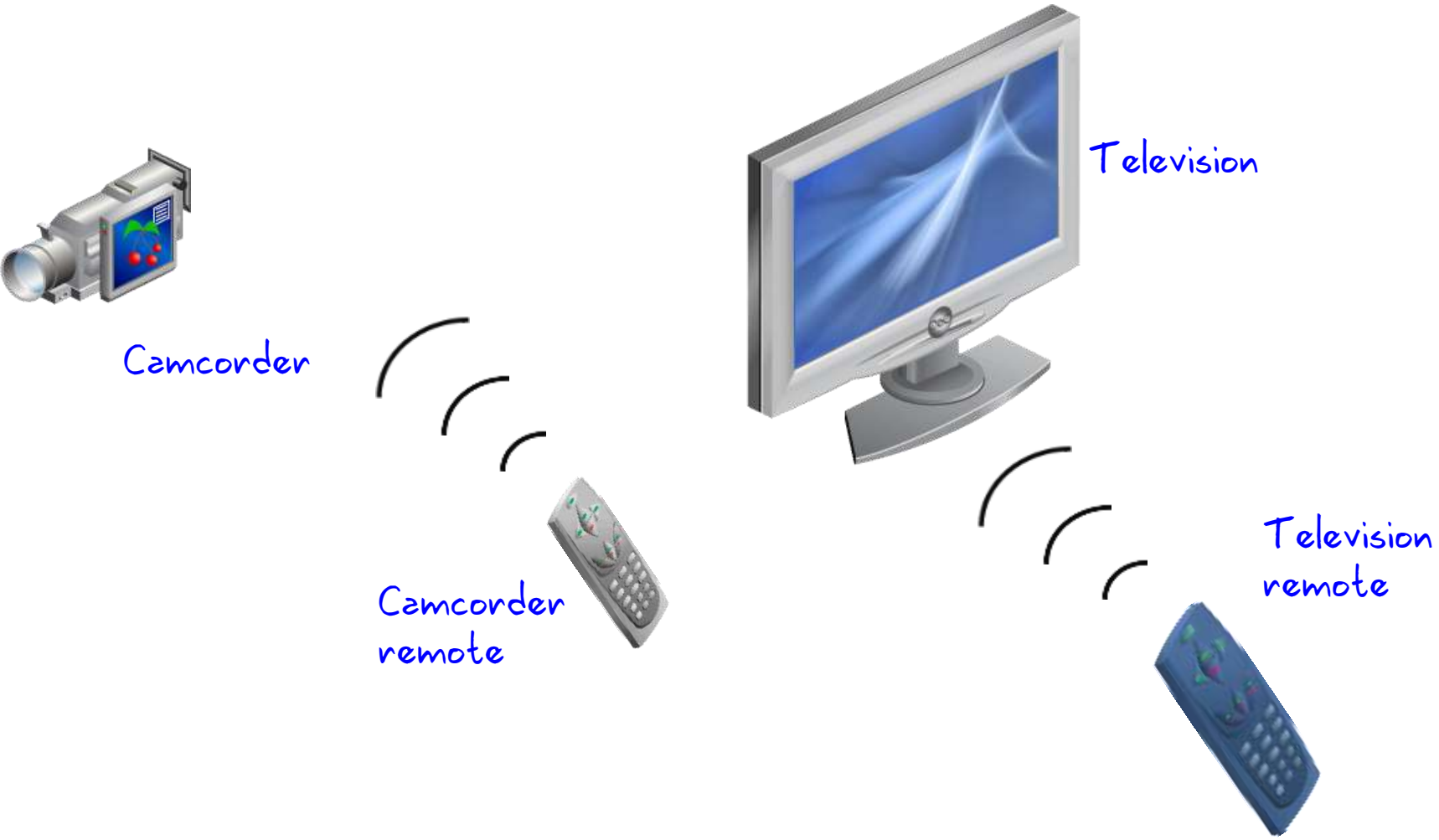
Working with Object References



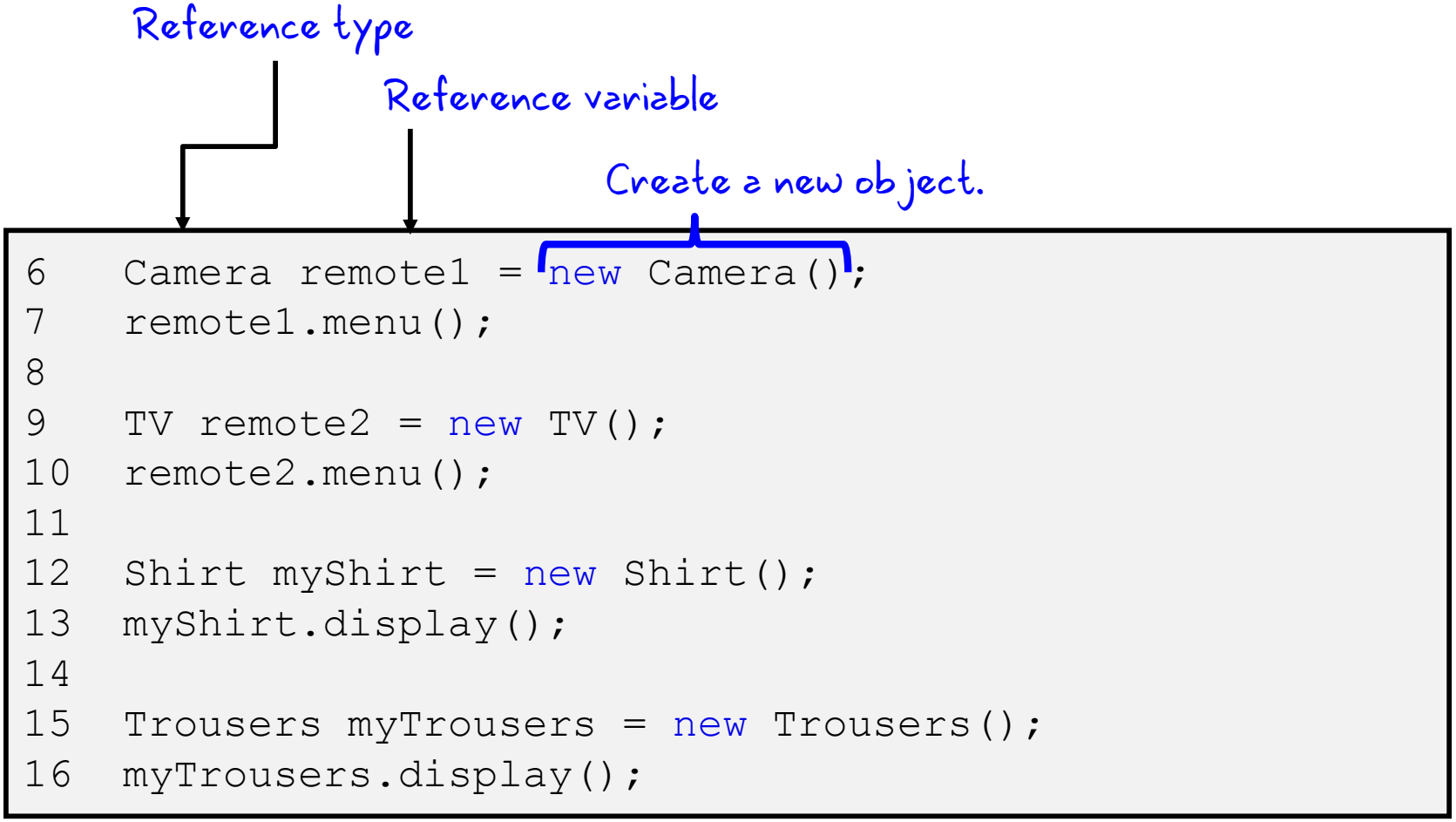
There is only one Camera object.

```
12  Camera remote1 = new Camera();
13
14  Camera remote2 = remote1;
15
16  remote1.play();
17
18  remote2.stop();
```

References to Different Objects

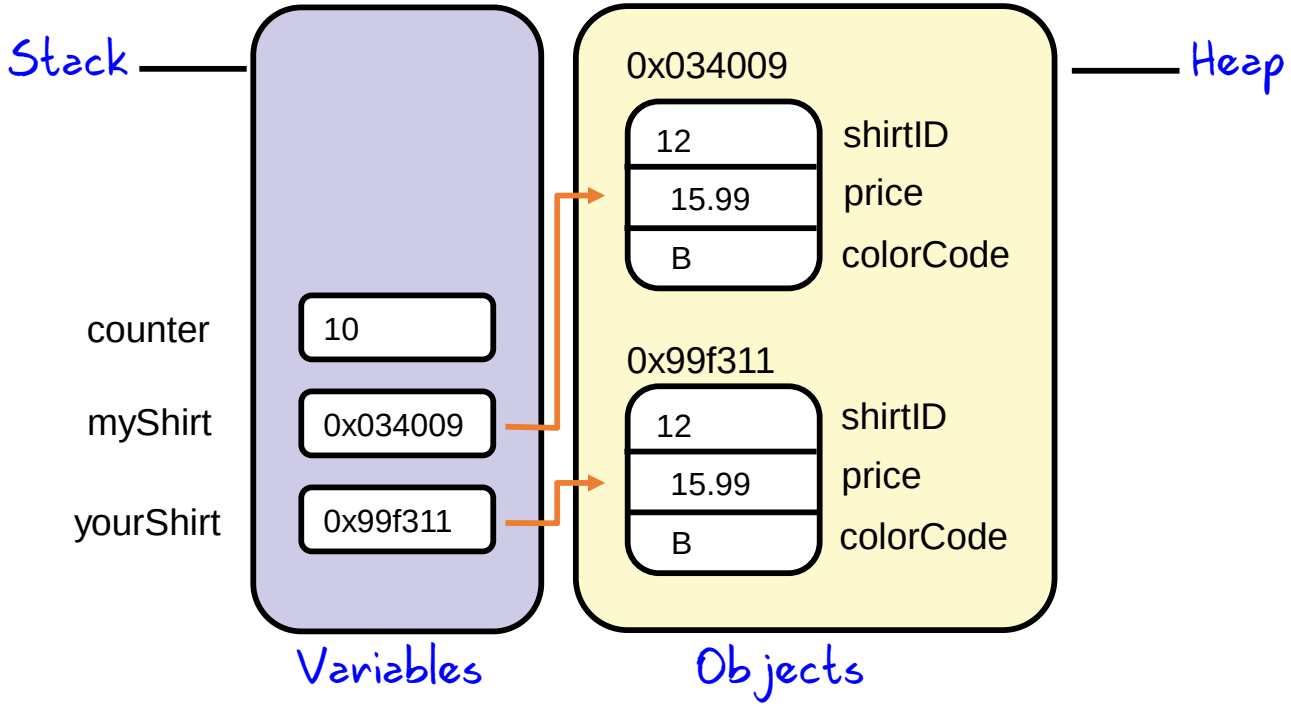


References to Different Objects



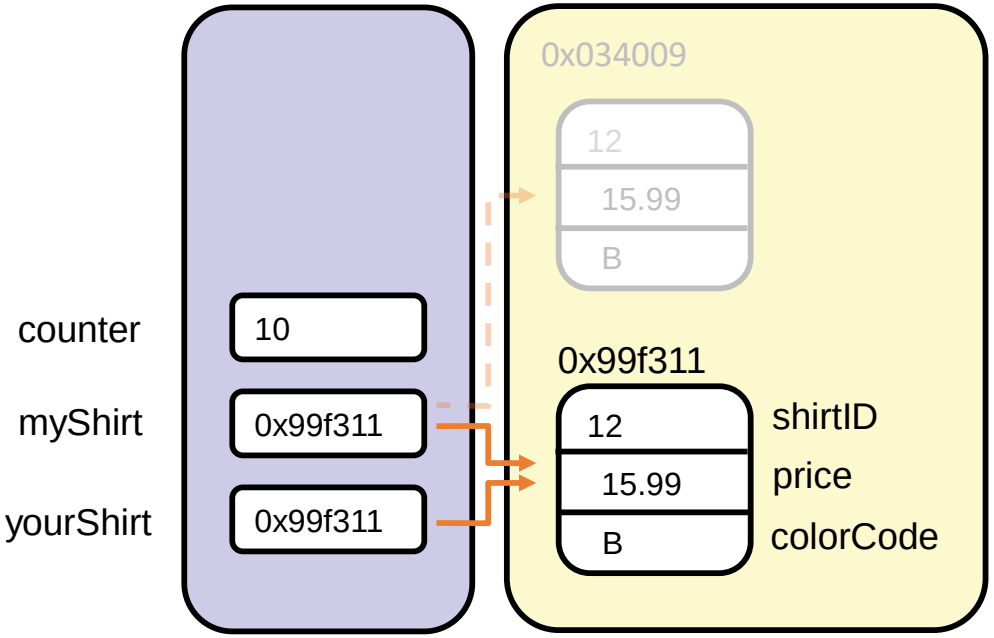
References and Objects in Memory

```
12 int counter = 10;
13 Shirt myShirt = new Shirt();
14 Shirt yourShirt = new Shirt();
```



Assigning a Reference to Another Reference

```
myShirt = yourShirt;
```



Two References, One Object

Code fragment:

```
12 Shirt myShirt = new Shirt();
13 Shirt yourShirt = new Shirt();
14
15 myShirt = yourShirt;          //The old myShirt object is
16                               //no longer referenced
17 myShirt.colorCode = 'R';
18 yourShirt.colorCode = 'G';
19
20 System.out.println("Shirt color: " + myShirt.colorCode);
```

Output from code fragment:

```
Shirt color: G
```

Topics

- Describing objects and classes
- Defining fields and methods
- Declaring, instantiating, and using objects
- Working with object references
- Doing more with arrays
- Introducing NetBeans IDE
- Introducing the soccer league use case

Arrays Are Objects

Arrays are handled by an implicit Array *object*.

- The Array variable is an *object reference*, not a primitive data type.
- It must be instantiated, just like other objects.
  - Example:

```
int[] ages = new int[4];
```

- Previously, you have been using a shortcut to instantiate your arrays.
  - Example:

```
int[] ages = {8,7,4,5};
```

This array can hold four elements.

Declaring, Instantiating, and Initializing Arrays

- Examples:

All in one line

```
1    String[] names = {"Mary","Bob","Carlos"};
2
3    int[] ages = new int[3];
4    ages[0] = 19;
5    ages[1] = 42;
6    ages[2] = 92;
```

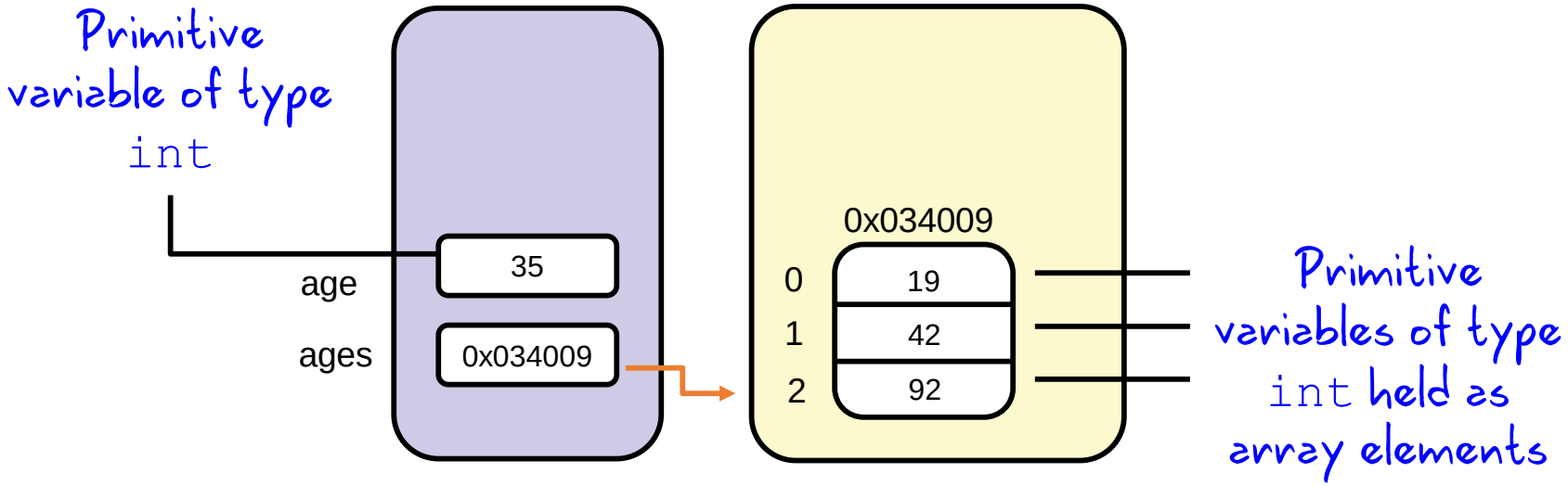
Multistep approach

- Not permitted (compiler will show an error):

```
int [] ages;
ages = {19, 42, 92};
```

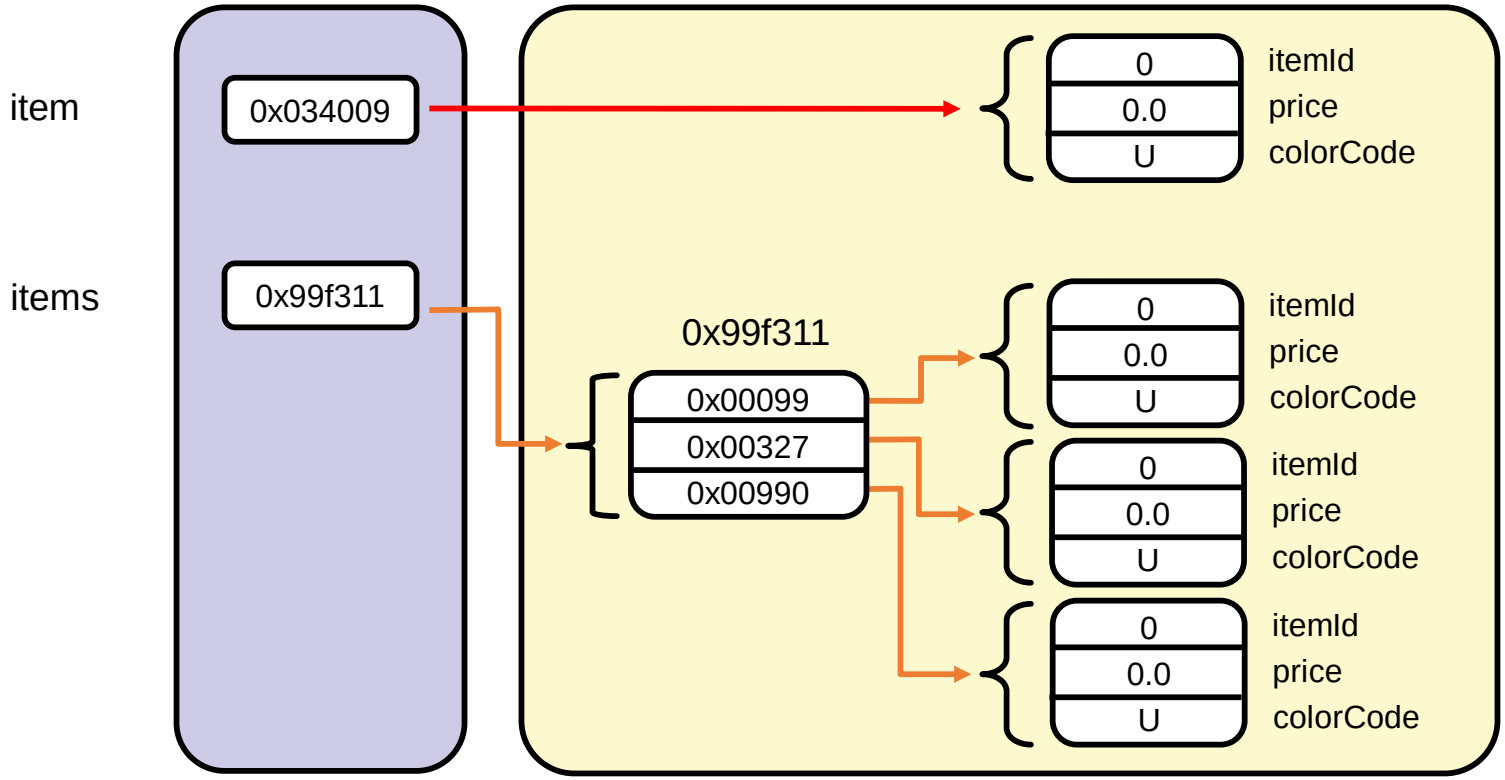
Storing Arrays in Memory

```
int age = 35;
int[] ages = {19, 42, 92};
```



Storing Arrays of Object References in Memory

```
Item item = new Item();
Item[] items = { new Item(), new Item(), new Item() };
```



Quiz

The following code is the correct syntax for _____ an array:

```
array_identifier = new type[length];
```

a. Declaring
b. Setting array values for
c. Instantiating
d. Declaring, instantiating, and setting array values for

Quiz

Given the following array declaration, which of the following statements are true?

- `int [ ] ages = new int [13];`

a. `ages[0]` is the reference to the first element in the array.

b. `ages[13]` is the reference to the last element in the array.

c. There are 13 integers in the `ages` array.

d. `ages[5]` has a value of 0.

Summary

In this lesson, you should have learned how to:

- Describe the characteristics of a class
- Define an object as an instance of a class
- Instantiate an object and access its fields and methods
- Describe how objects are stored in memory
- Instantiate an array of objects
- Describe how an array of objects is stored in memory
- Declare an object as a field
- Use the NetBeans IDE