Describing Objects and Classes

Interactive Quizzes

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
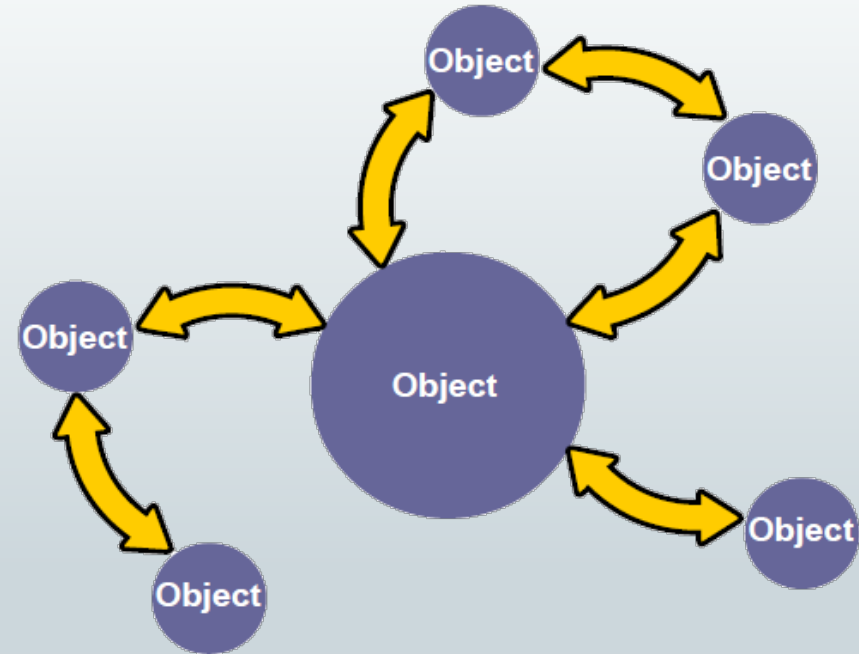- Declare and instantiate an object as a field

Topics

- **Describing objects and classes**

- Defining fields and methods

- Declaring, instantiating, and using objects

- Working with object references

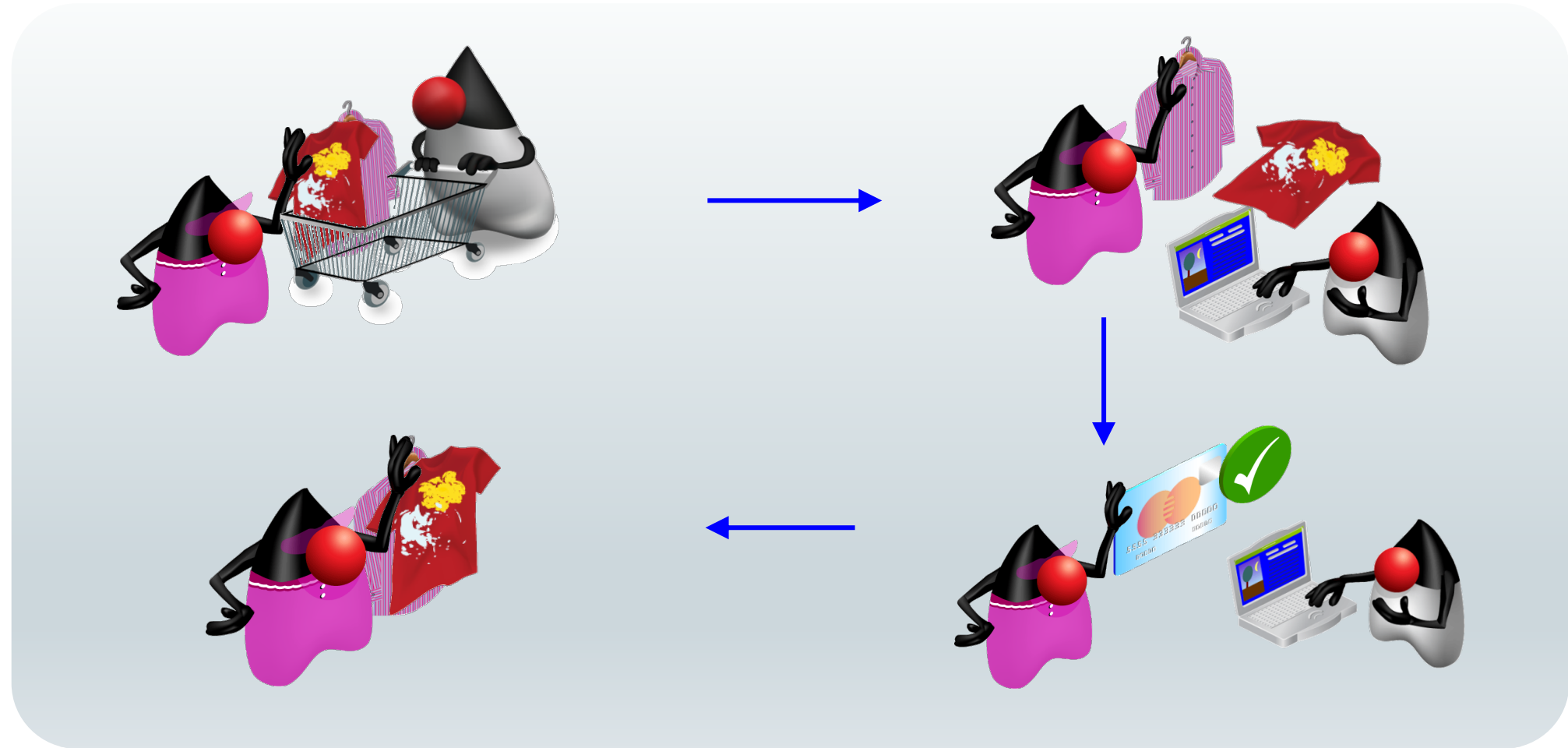- Doing more with arrays

- Introducing the soccer league use case

Object-Oriented Programming

- Interaction of objects
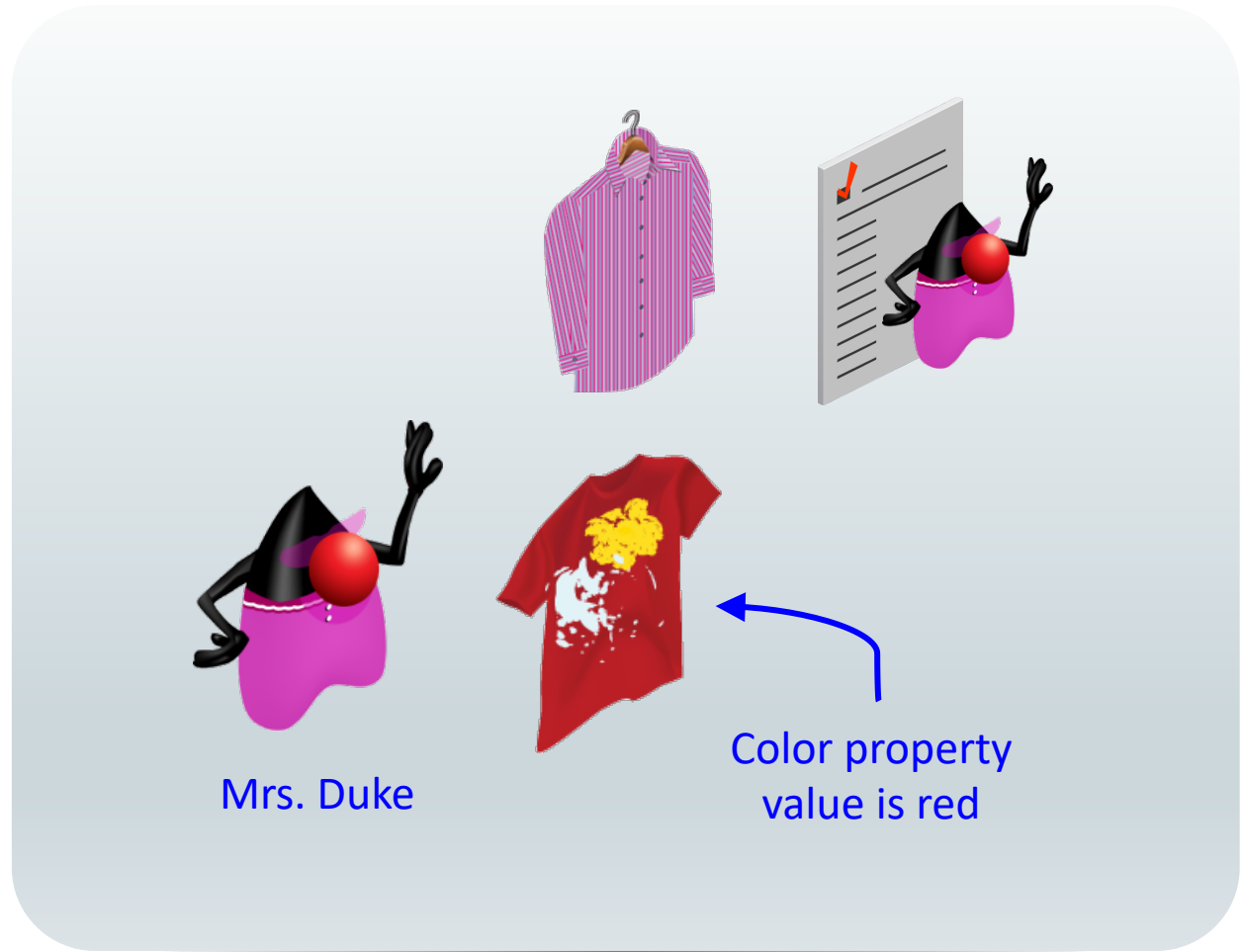- No prescribed sequence

Duke's Choice Order Process

Characteristics of Objects
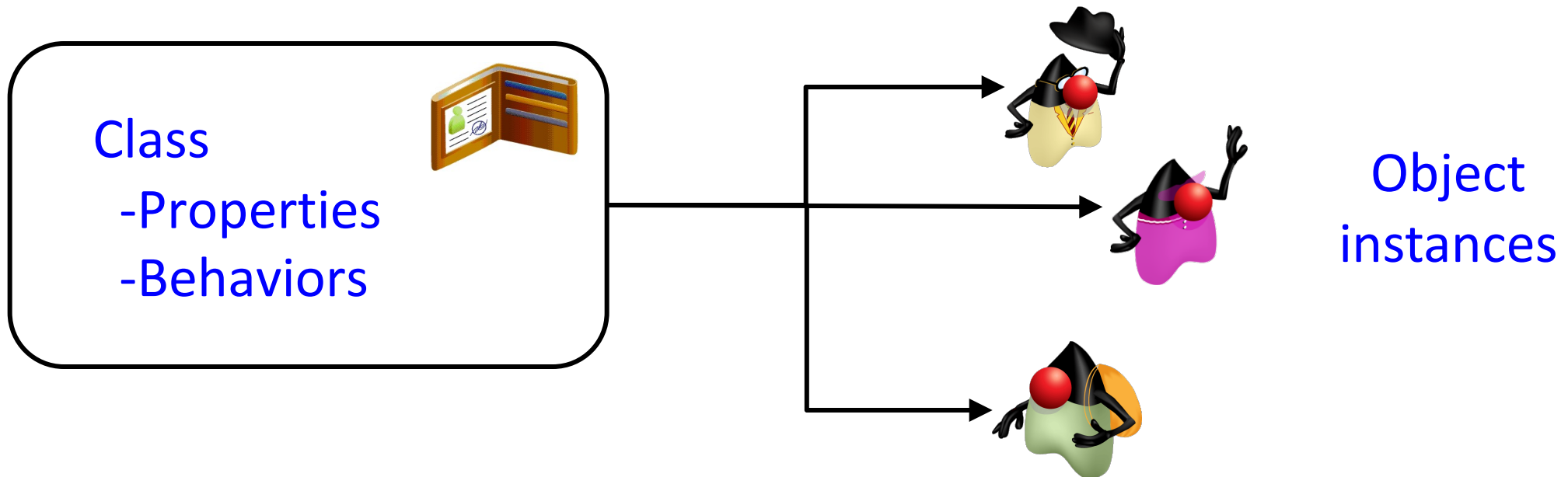
Objects are physical or conceptual.

- 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

Class
-Properties
-Behaviors

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 the soccer league use case

The Customer Properties and Behaviors
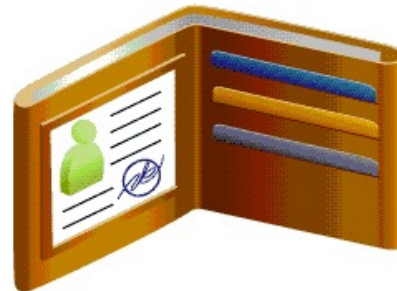
Properties:

- 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
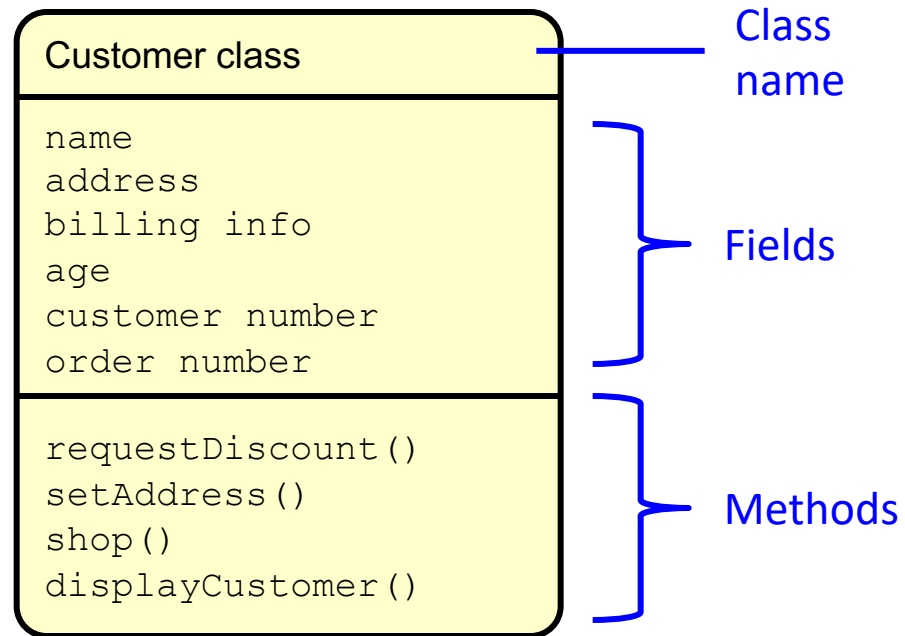
Class declaration

```
 1 public class Customer {
 2     public String name = "Junior Duke";
 3     public int     custID = 1205;
 4     public String address;
 5     public int     orderNum;
 6     public int     age;
 7
 8     public void displayCustomer(){
 9         System.out.println("Customer: "+name);
10     }
11 }
```

Fields
(Properties)
(Attributes)

Methods
(Behaviors)
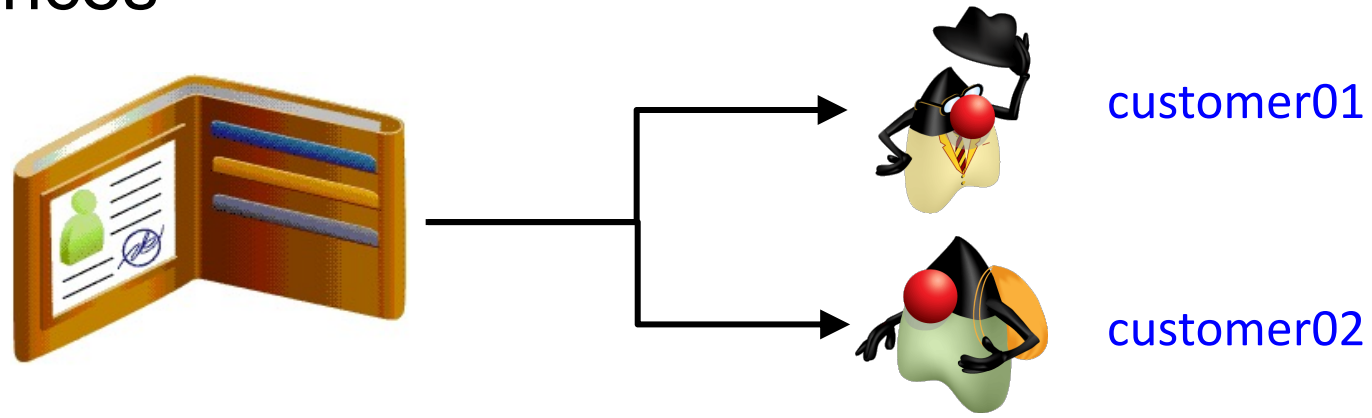
Modeling Properties and Behaviors

Topics

- Describing objects and classes

- Defining fields and methods

- **Declaring, instantiating, and using objects**

- Working with object references

- Doing more with arrays

- 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.

Customer class
name address billing info age customer number order number
requestDiscount() setAddress() shop() displayCustomer()

```java
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

Customer class

```
customer number
*cart
name
address
billing info
order number
```

```
requestDiscount()
setAddress()
shop()
displayCustomer()
```

Cart class

```
Item[]
date
total
```

```
addItem()
checkOut()
cancel()
```
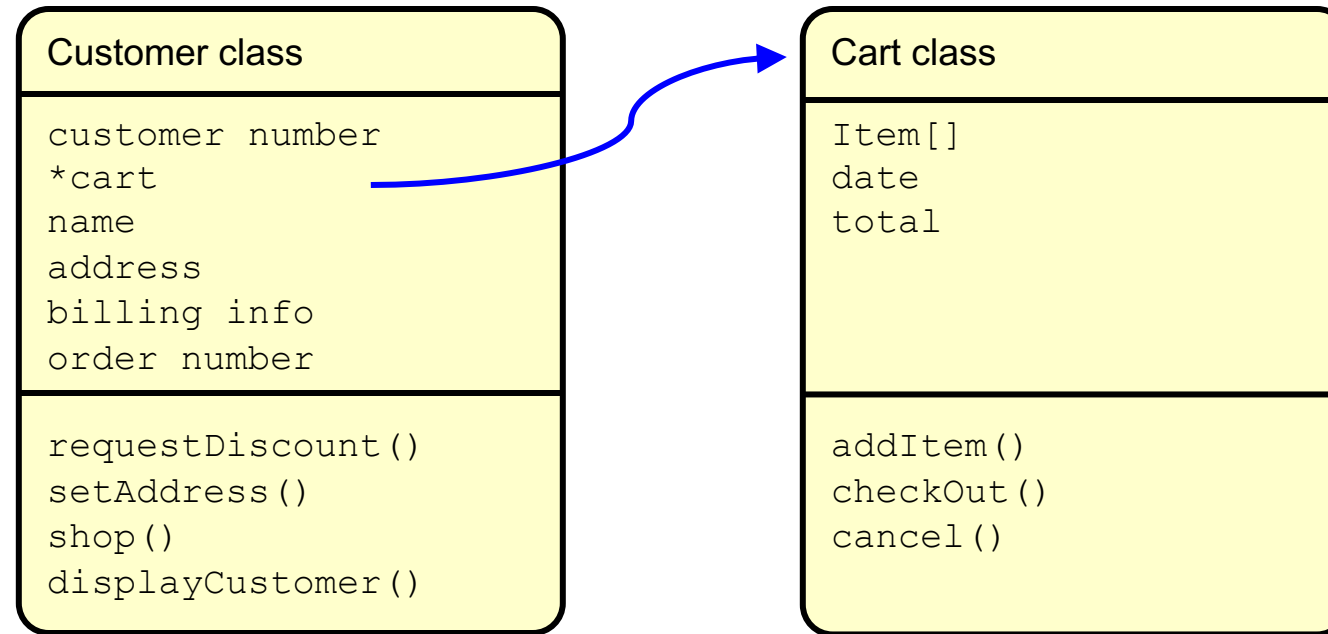
```java
public static void main(String[] args){


    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()`
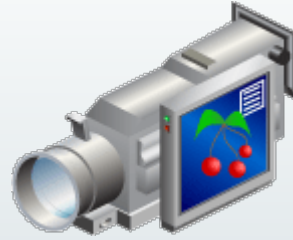
d. `Boat sailBoat = Boat();`

Topics

- Describing objects and classes

- Defining fields and methods

- Declaring, instantiating, and using objects

- **Working with object references**

- Doing more with arrays
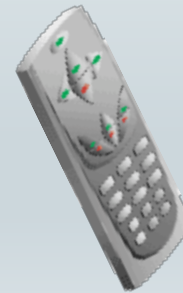
- Introducing the soccer league use case

Accessing Objects by Using a Reference

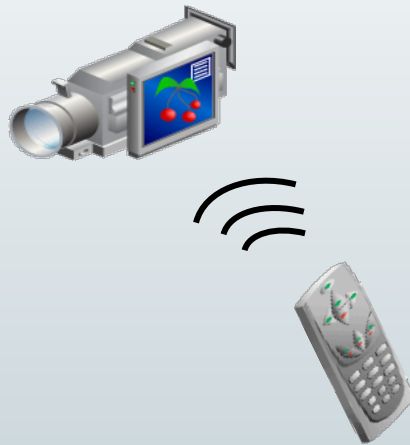The camera is like the **object** that is accessed using a reference.

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

Working with Object References
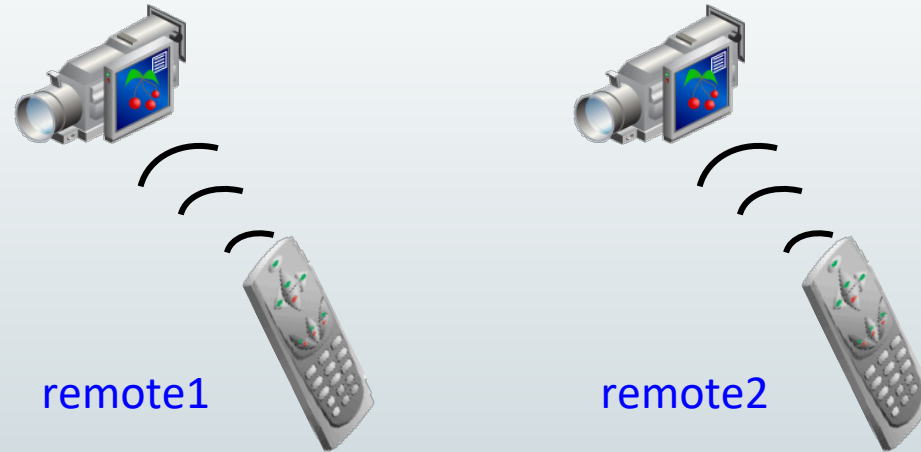
1 Pick up the remote to gain access to the camera.

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

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

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

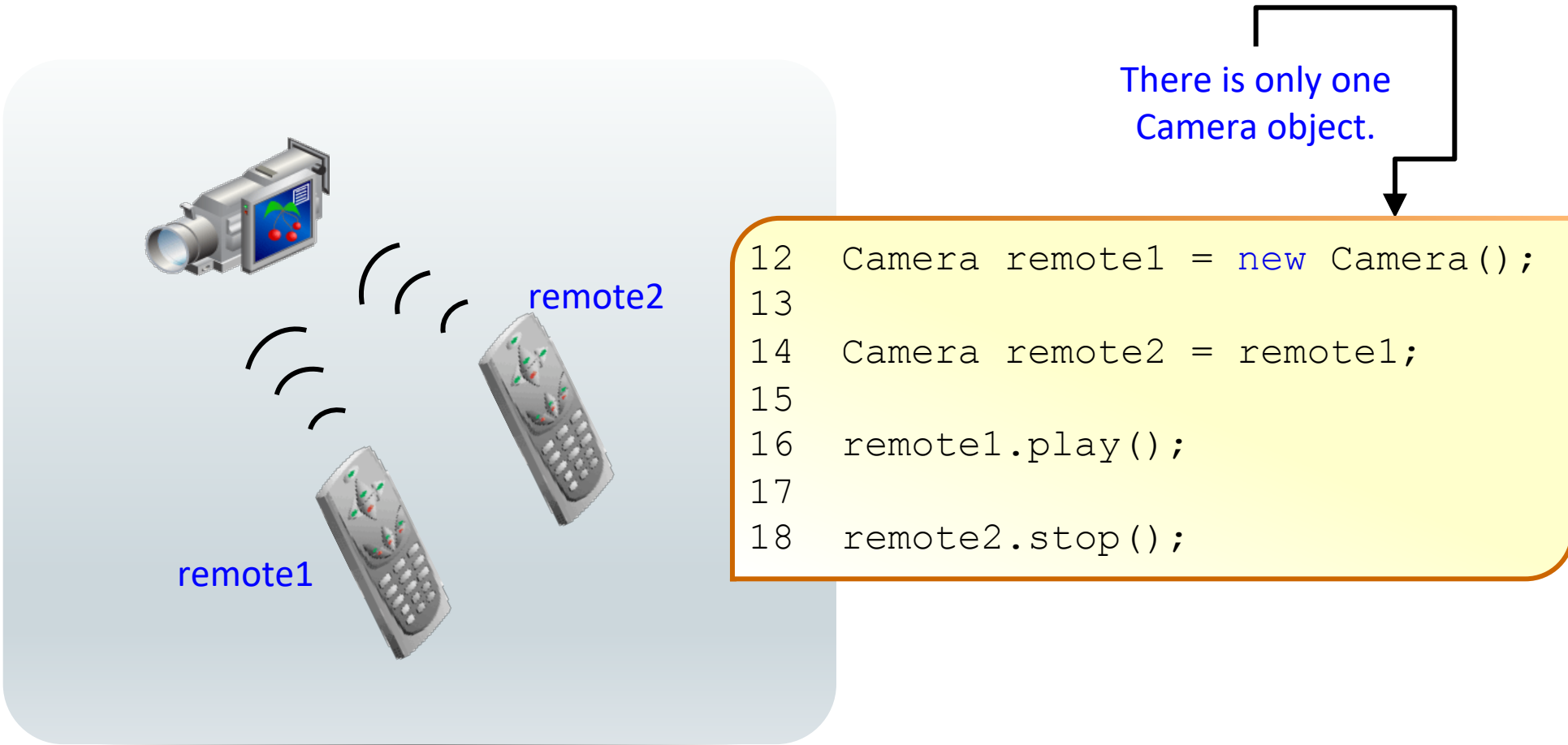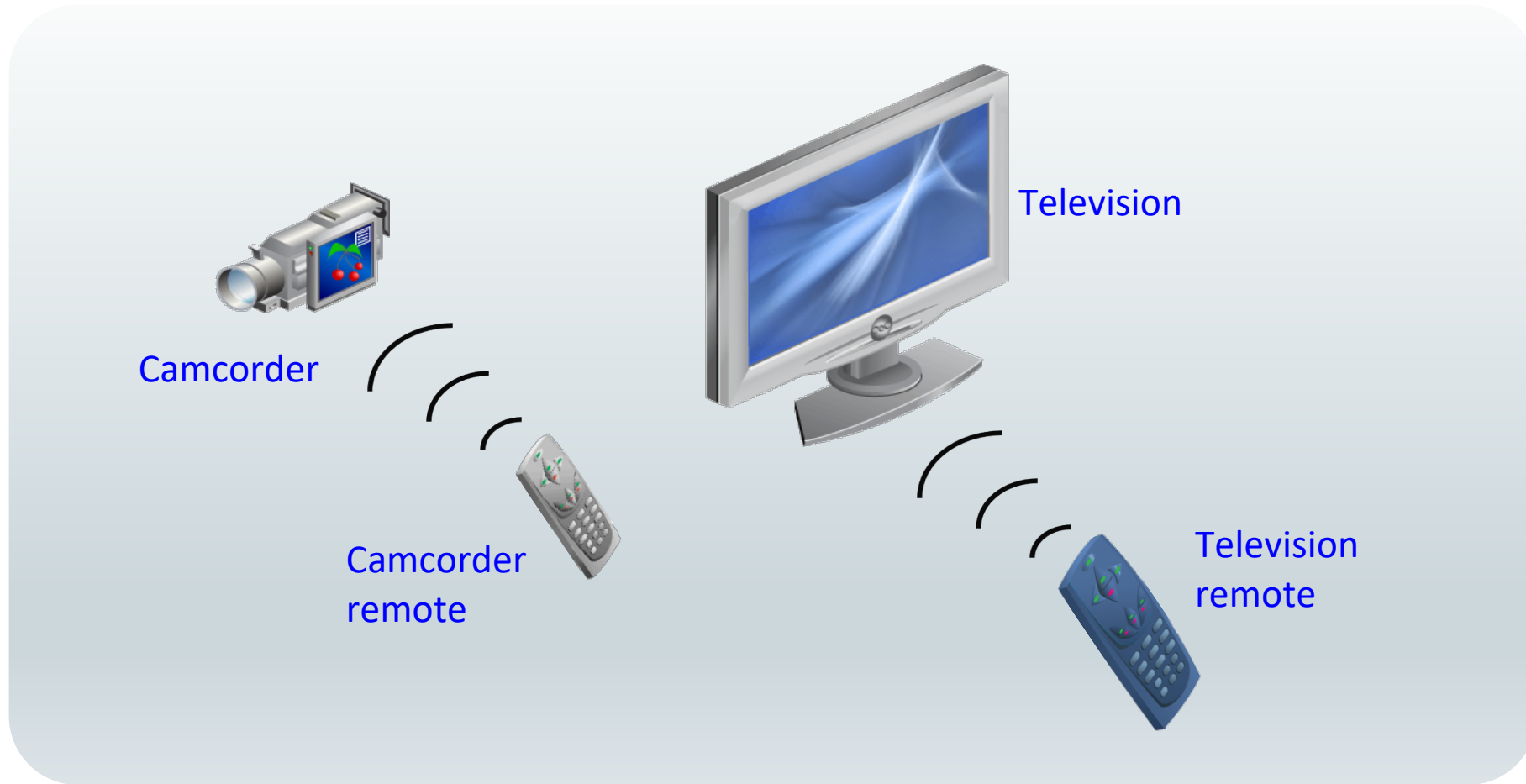**2** Call a method to have the Camera object do something.

Working with Object References


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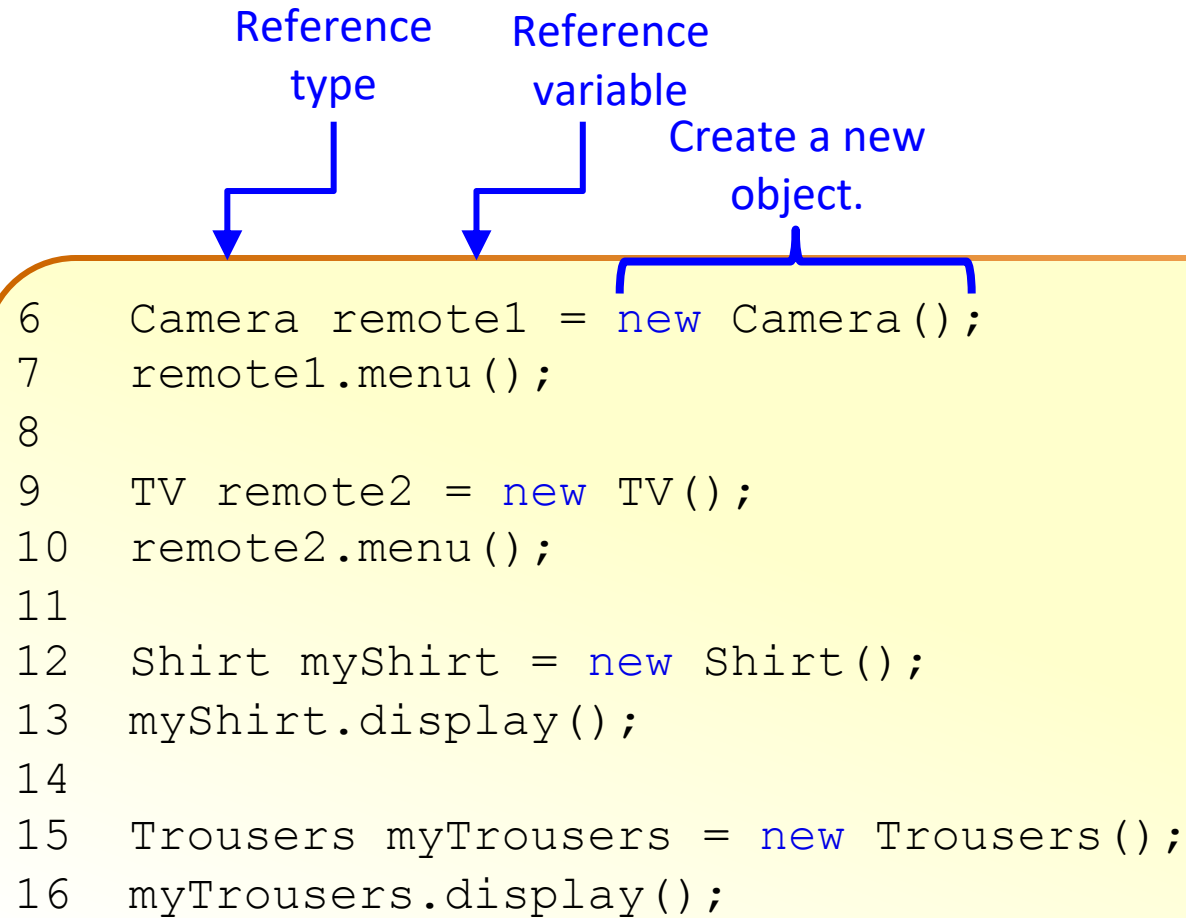
remote1   remote2

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();
```

remote2

remote1

References to Different Objects

Camcorder

Camcorder remote

Television

Television remote

References to Different Objects

Reference type

Reference variable

Create a new object.

```
6    Camera remote1 = new Camera();
7    remote1.menu();
8
9    TV remote2 = new TV();
10   remote2.menu();
11
12   Shirt myShirt = new Shirt();
13   myShirt.display();
14
15   Trousers myTrousers = new Trousers();
16   myTrousers.display();
```

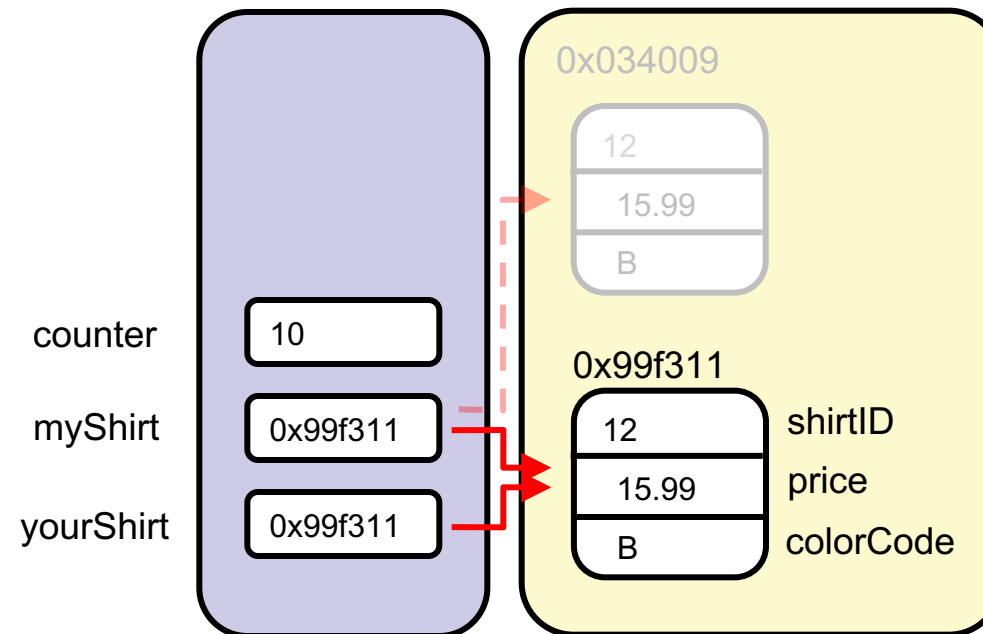# References and Objects in Memory

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


Stack ——

Heap

0x034009

12	shirtID
15.99	price
B	colorCode

0x99f311

12	shirtID
15.99	price
B	colorCode

counter   10

myShirt   0x034009

yourShirt   0x99f311

Variables
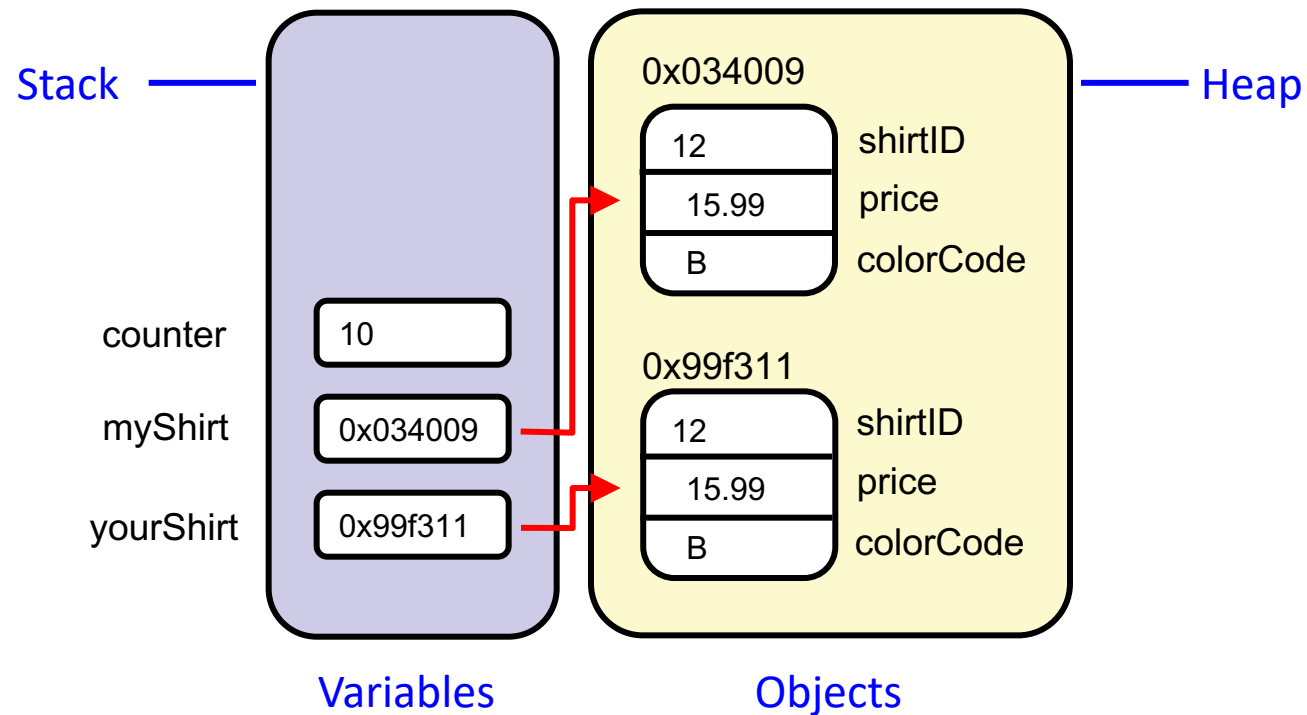
Objects

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 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];
      ```
      This array can hold four elements.

- Previously, you have been using a shortcut to instantiate your arrays.
    - Example:
      ```
      int[] ages = {8,7,4,5};
      ```

Declaring, Instantiating, and Initializing Arrays

- Examples:

```
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;
```

This statement declares an integer array `ages` and initializes the elements with default value of 0
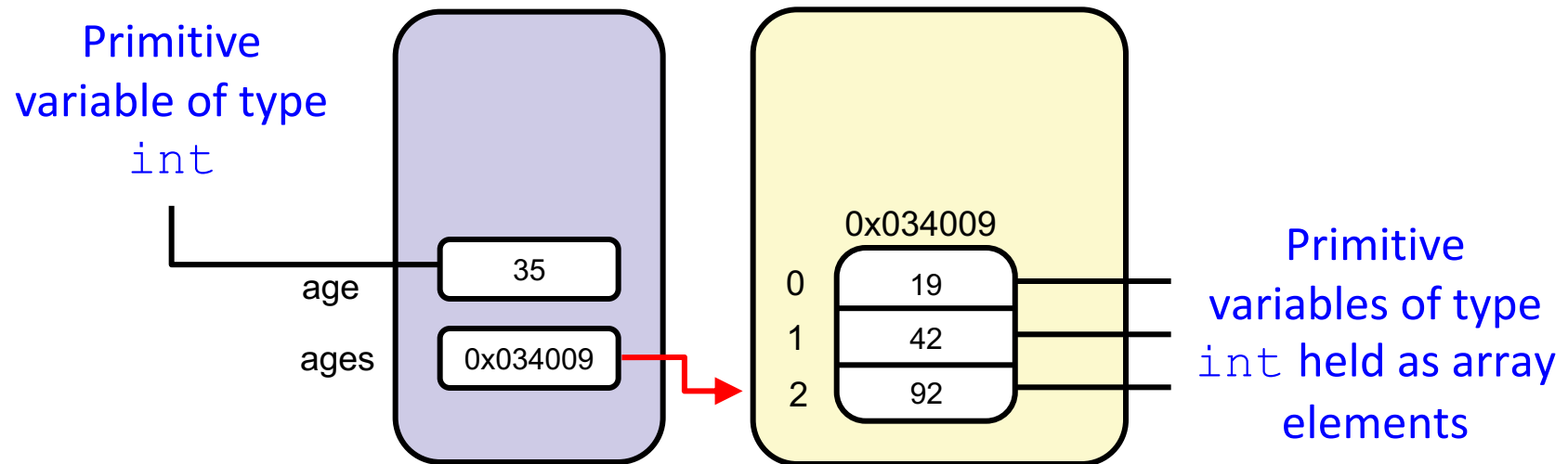
- Not permitted (compiler will show an error):
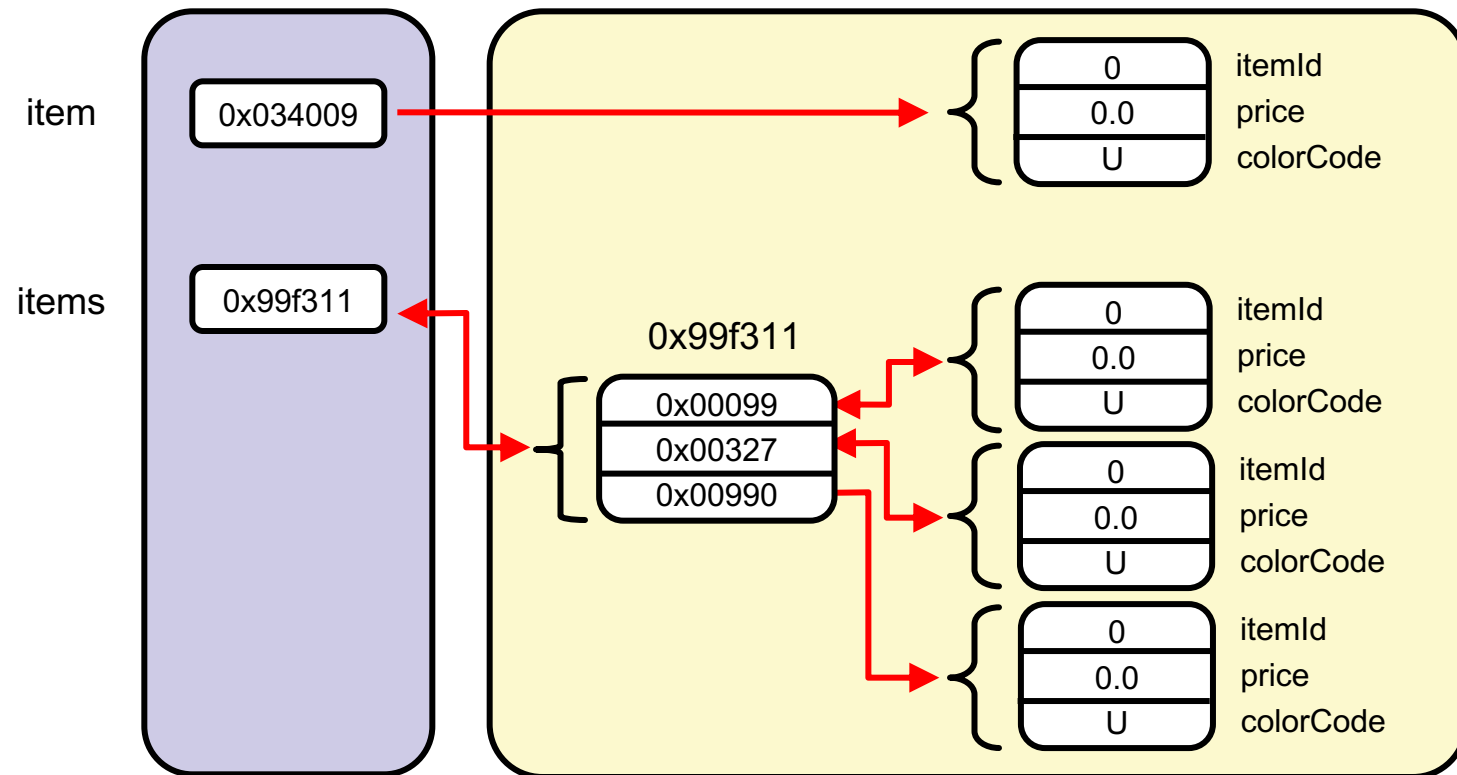
```
int[] ages;
ages = {19, 42, 92};
```

Storing Arrays in Memory

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

Primitive variable of type `int`

35
age

ages
0x034009

0x034009

0	19
1	42
2	92

Primitive variables of type `int` held as array elements

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.

Topics

- Describing objects and classes

- Defining fields and methods

- Declaring, instantiating, and using objects

- Working with object references

- Doing more with arrays

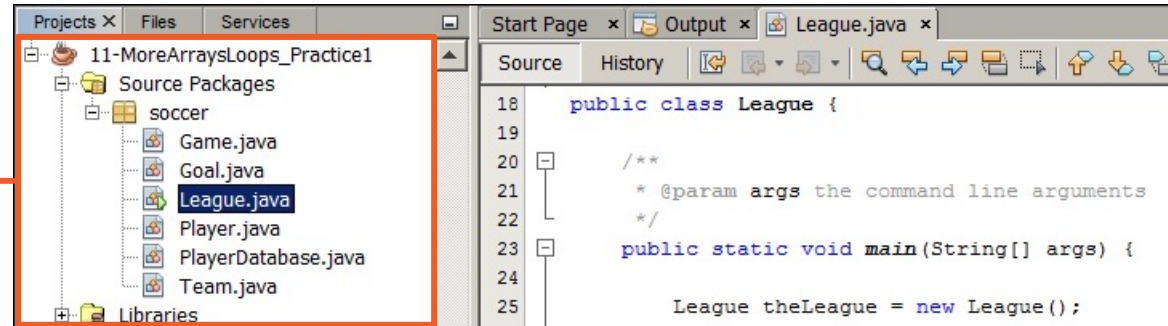- **Introducing the soccer league use case**

Soccer Application

Practices 6 through 14 build a soccer league application with the following features:

- Any number of soccer teams, each with up to 11 players

- Set up an all-play-all league.

- Use a random play game generator to create test games.

- Determine the rank order of teams at the end of the season.

Creating the Soccer Application

A separate project for each practice

Sample output showing events in a game

Sample output showing rank order of teams

Soccer Web Application

Soccer League Games

[Replay games]

		Away Teams							
		The Magpies	**The Crows**	**The Reds**	**The Blues**	**The Rovers**	**The Harriers**	**Goals**	**Points**
	The Magpies	X	(0 - 1)	(4 - 2)	(1 - 0)	(3 - 0)	(1 - 0)	15	18
	The Crows	(2 - 1)	X	(1 - 0)	(0 - 1)	(0 - 0)	(0 - 0)	10	18
Home Teams	**The Reds**	(0 - 1)	(0 - 1)	X	(1 - 1)	(1 - 0)	(1 - 0)	13	14
	The Blues	(4 - 1)	(0 - 2)	(0 - 1)	X	(3 - 4)	(1 - 0)	12	14
	The Rovers	(3 - 0)	(5 - 2)	(2 - 4)				18	13
	The Harriers	(1 - 3)	(1 - 1)	(3 - 3)				8	7

The Rovers vs. The Reds (2 - 4)

Team	Player	Time
The Reds	Jane Austin	7
The Rovers	J. M. Synge	21
The Reds	Jane Austin	41
The Reds	Mark Twain	46
The Reds	Brian Moore	76
The Rovers	Charlotte Bronte	83

Return to main page

Teams listed in rank order

Click the score of a game to show game details.

Points and goals scored used for ordering

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