

Describing Objects and Classes

6



Copyright © 2018, Oracle and/or its affiliates. All rights reserved.



ORACLE®

Moisés Ocampo Sámano (aos_moyer@hotmail.com) has a
non-transferable license to use this Student Guide.

Interactive Quizzes



Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

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



Copyright © 2018, Oracle and/or its affiliates. All rights reserved.



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

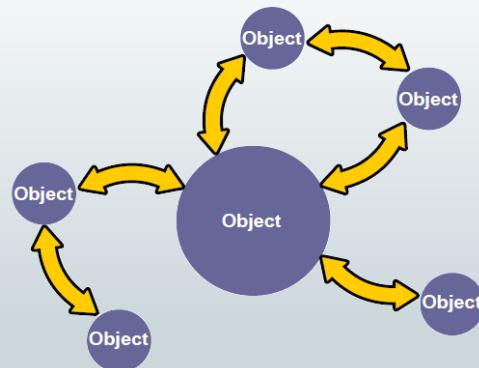


Copyright © 2018, Oracle and/or its affiliates. All rights reserved.



Object-Oriented Programming

- Interaction of objects
- No prescribed sequence

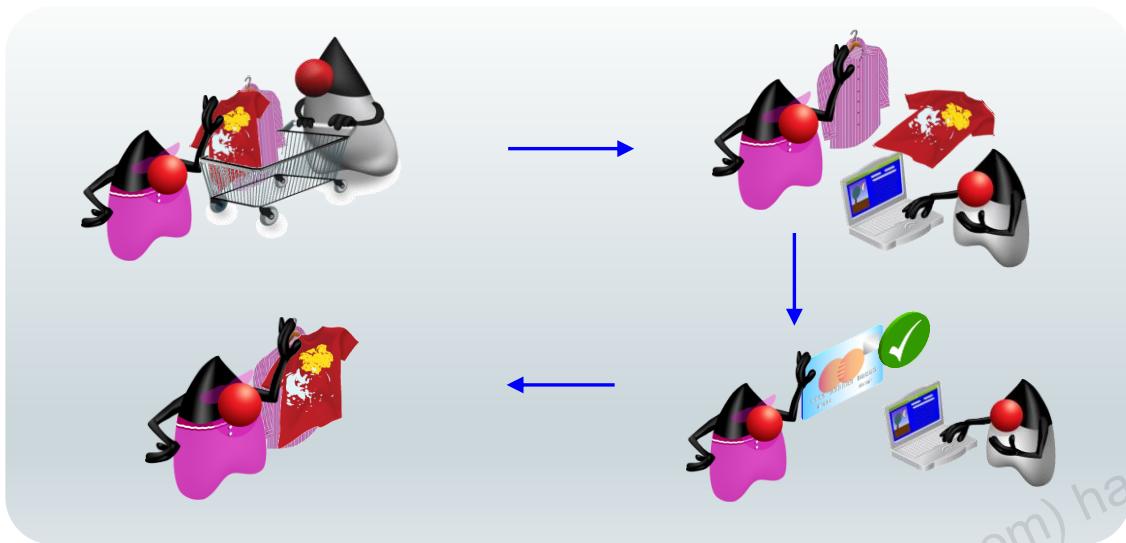


Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

You have seen this diagram before in the “What Is a Java Program?” lesson. The diagram illustrates how object-oriented programming stresses the interaction of objects. The current lesson teaches you how to identify the objects that are required for the application that you would like to build. You first identify what the objects are, you determine the object’s characteristics or properties, and then you determine the object’s behaviors or operations. You then translate that analysis into Java code to create your application. It is time to learn more about objects.



Duke's Choice Order Process



Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

In the first five lessons, the exercises mention a shopping cart class that contains items. Take another look at the shopping cart scenario.

Imagine an online store called Duke's Choice. His number one shopper is his mother, Mrs. Duke. As Mrs. Duke shops, she places items in a shopping cart. Mrs. Duke likes shirts, so she places shirts in her cart. After she fills the cart, she checks out. The checkout process applies the purchase to a credit card, which is verified, and then Mrs. Duke receives an order number so that she can track her order or return it.

As a software developer, when you are presented with a scenario such as Duke's Choice for an application that you need to develop, you can analyze the scenario by breaking it into steps and defining the objects of the scenario.



Characteristics of Objects

Objects are physical or conceptual.

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

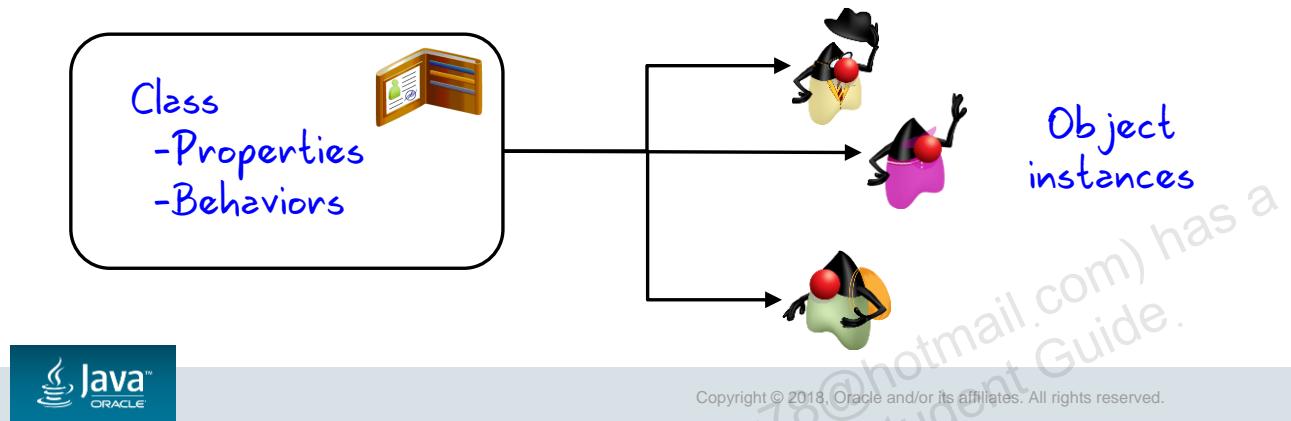


Copyright © 2018, Oracle and/or its affiliates. All rights reserved.



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



Copyright © 2018, Oracle and/or its affiliates. All rights reserved.



You just learned about some of the objects, characteristics, and behaviors in the Duke's Choice scenario. Here is an example of one of Duke's Choice objects, the `Customer`, and its function in the store. `Customer` is the class, and a class is a blueprint or recipe for an object. The class describes an object's properties and behaviors.

Classes are used to create object instances, such as the three `Customer` object instances, as illustrated by the three images.



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.



Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

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



Copyright © 2018, Oracle and/or its affiliates. All rights reserved.



You have just learned about objects, classes, and their characteristics (properties and behaviors). Now it is time to look at fields and methods.

The Customer Properties and Behaviors

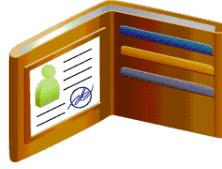


Properties:

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

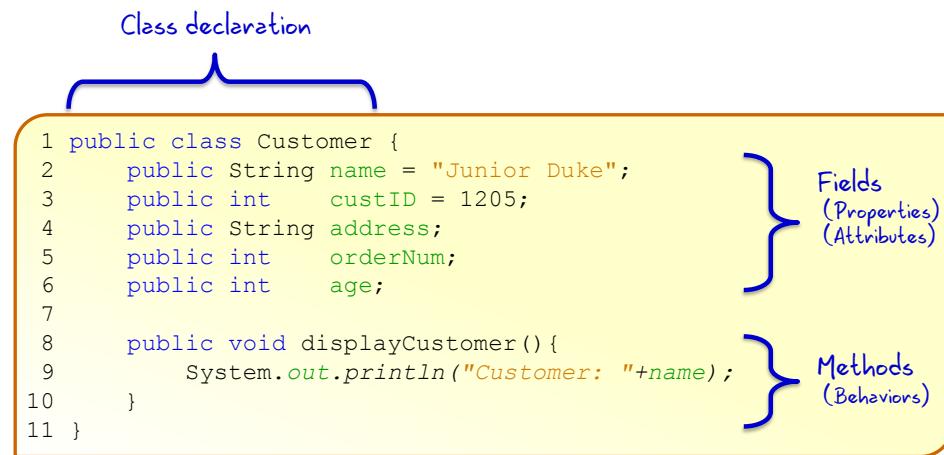
Behaviors:

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



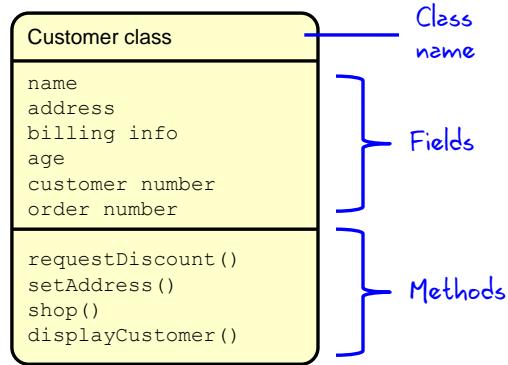
Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

The Components of a Class



Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

Modeling Properties and Behaviors



Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

Exercise 6-1: Creating the Item Class

1. Open the project **Exercise_06-1** in NetBeans
2. Create the Item class as a plain **Java class**.
3. Declare public fields for `ID` (`int`), `descr` (`String`), `price` (`double`), and `quantity` (`int`).
 - You will not be able to test the `Item` class until Exercise 6-2.



Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

Topics

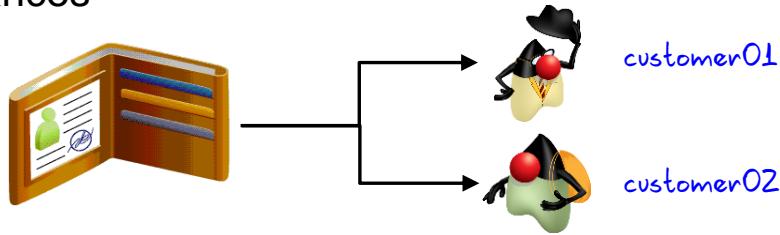
- Describing objects and classes
- Defining fields and methods
- Declaring, instantiating, and using objects
- Working with object references
- Doing more with arrays
- Introducing the soccer league use case



Copyright © 2018, Oracle and/or its affiliates. All rights reserved.



Customer Instances



```
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.

Copyright © 2018, Oracle and/or its affiliates. All rights reserved.



Object Instances and Instantiation Syntax

The syntax is:
`<class name> variable = new <class name>()`

variable becomes a reference to that object.
The new keyword creates (instantiates) a new instance.

```
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.  
}
```



Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

By using the `new` keyword, a new instance of the class is now available to be accessed through the variable, which stores a reference to that object. It can be referred to as a reference variable or an object reference.

Notice that, following the `new` keyword, you see the class name followed by parentheses. This looks similar to calling a method, doesn't it? You are calling a method—the `constructor` method of the `Customer` class. Every class has a `constructor` method that has the same name as the class. Constructors are covered in more detail in the lesson titled “Creating and Using Methods.”

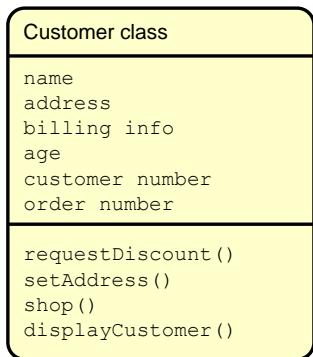
To summarize, there are three steps to getting an object reference:

1. Declare the reference.
2. Instantiate the object using the `new` keyword and the class `constructor` method.
3. Assign the object to the reference.

Note that the way that the assignment operator (an `=` symbol) works requires that the reference and the newly created object must be in the same statement. (Statements are ended with the semicolon symbol and are not the same as lines. The end of a line means nothing to the Java compiler; it only helps make the code more readable.)

The Dot (.) Operator

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

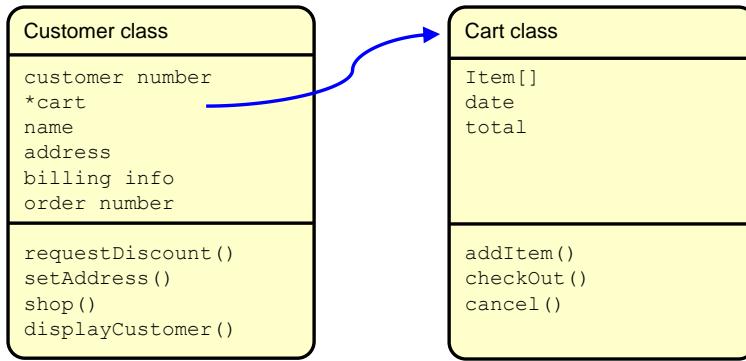


```
public static void main(String[] args) {  
  
    Customer customer01 = new Customer();  
  
    //Accessing fields  
    System.out.println(customer01.name);  
    customer01.age = 40;  
  
    //Calling methods  
    customer01.requestDiscount();  
    customer01.displayCustomer();  
}  
}
```



Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

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
```



Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

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();`



Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

Answer: a

Topics

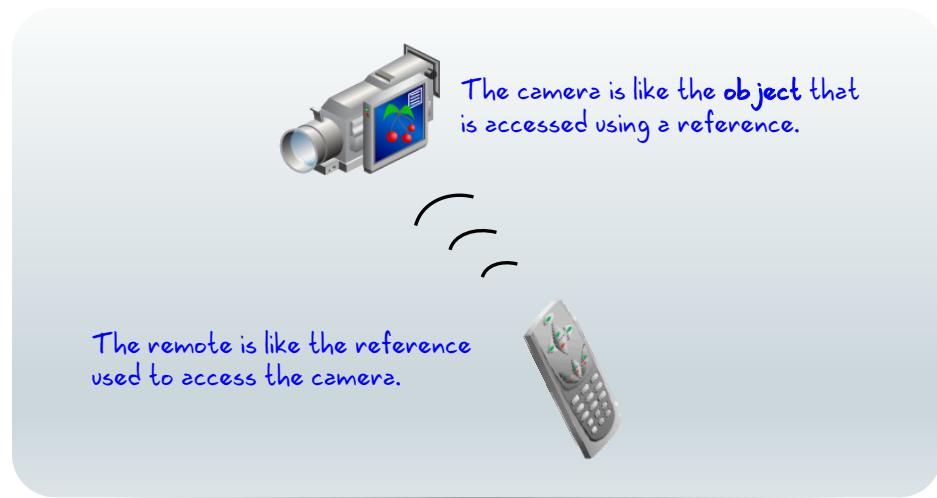
- Describing objects and classes
- Defining fields and methods
- Declaring, instantiating, and using objects
- **Working with object references**
- Doing more with arrays
- Introducing the soccer league use case



Copyright © 2018, Oracle and/or its affiliates. All rights reserved.



Accessing Objects by Using a Reference



What you have learned up to this point is:

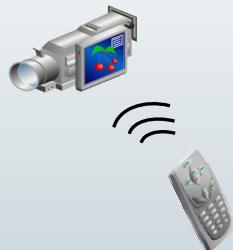
- Objects are accessed using references.
- Objects are instantiated objects of their class type.
- Objects consist of properties and operations, which in Java are fields and methods.

To work with an object, you need to access it using a reference. A good analogy is using a remote control to operate an electronic device. The buttons on the remote control can be used to modify the behavior of the device (in this case, a camera). For example, you can make the camera stop, play, or record by interacting with the remote.

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 remote1.play();
```

2

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

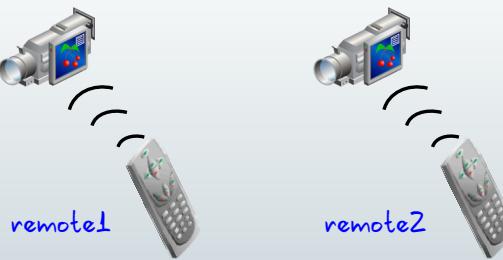
2

Call a method to have the Camera object do something.



Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

Working with Object References



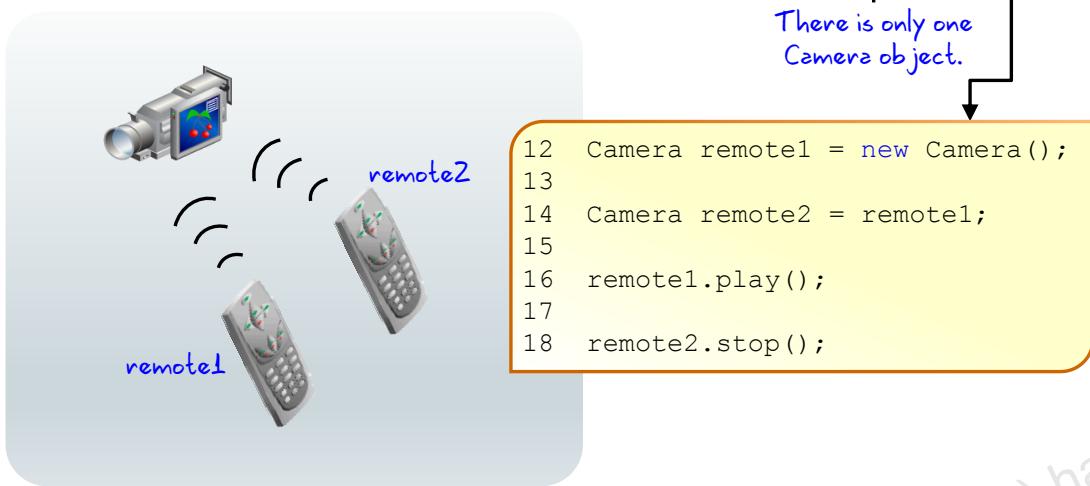
```
12 Camera remote1 = new Camera();  
13  
14 Camera remote2 = new Camera();  
15  
16 remote1.play();  
17  
18 remote2.play();
```

There are two Camera objects.

Copyright © 2018, Oracle and/or its affiliates. All rights reserved.



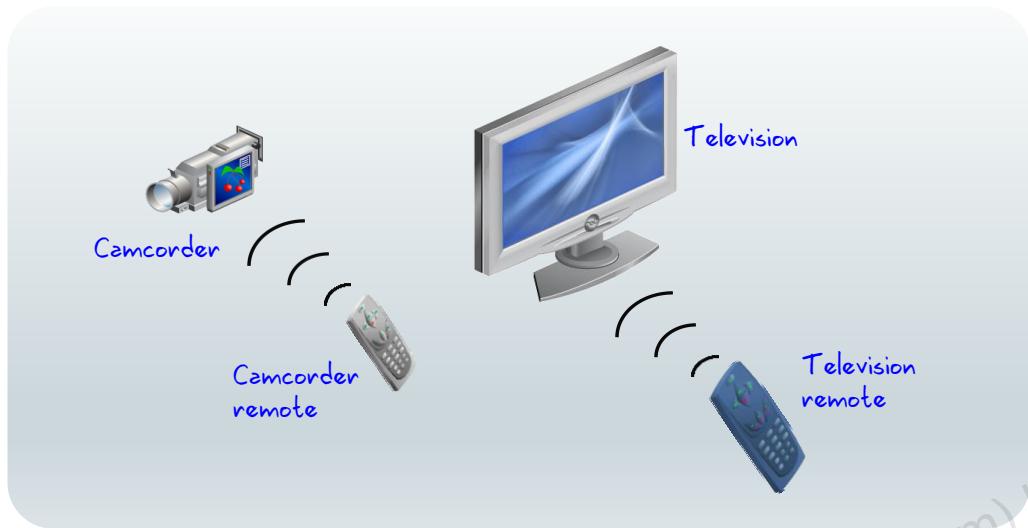
Working with Object References



Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

The diagram shows another important aspect of how references work. In this example, a `Camera` object is created and its reference assigned to a `Camera` reference, `remote1`. This reference is then assigned to another `Camera` reference, `remote2`. Now both references are associated with the same `Camera` object, and methods called on either reference will affect the same `Camera` object. Calling `remote1.play` is no different than calling `remote2.play`. Both remotes operate the same camera.

References to Different Objects



To extend the analogy just a little further, to work with a different type of object (for example, a television), you need a remote for that object. In the Java world, you need a reference of the correct type for the object that you are referencing.

You can ignore the fact that there is such a thing as a universal remote controller, although later in the course you will discover that Java also has the concept of references that are not limited to a single object type! For the moment, let's just say that a reference of the same type as the object is one of the reference types that can be used, and is a good place to start exploring the world of Java objects.

References to Different Objects

```
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();
```



Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

remote1 references a Camera object.

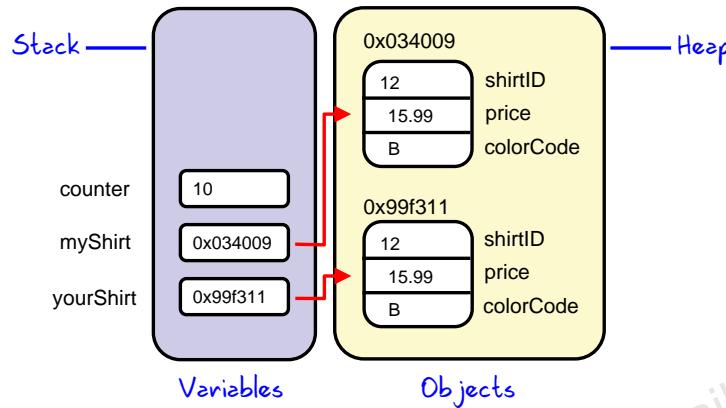
remote2 references a TV object.

myShirt references a Shirt object.

myTrousers references a Trousers object.

References and Objects in Memory

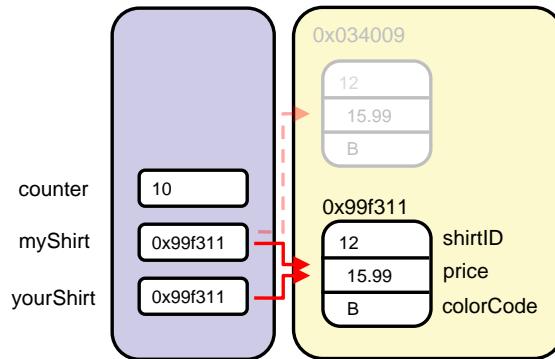
```
12 int counter = 10;  
13 Shirt myShirt = new Shirt();  
14 Shirt yourShirt = new Shirt();
```



Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

Assigning a Reference to Another Reference

```
myShirt = yourShirt;
```



Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

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
```



Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

This example now shows what happens if you use either reference to make a change or get a value from the object. References `yourShirt` and `myShirt` refer to the same object, so making a change or getting a field value by using one reference is exactly the same as doing it with the other reference. The old object that was previously referenced by `myShirt` goes away.

Exercise 6-2: Modifying the ShoppingCart to Use Item Fields

1. Continue editing **Exercise_06-1** or open **Exercise_06-2** in NetBeans.
2. Create a new Java Main Class called `ShoppingCart`. This class contains a single main method. The rest of this exercise is spent modifying `ShoppingCart.java`.
3. Declare and instantiate two objects of type `Item`. Initialize only the `descry` field in each, using different values for each.
4. Print the description for each item and run the code.
5. (Optional) Above the code that prints the descriptions, assign `item2` to `item1`. Run it again.



Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

In this exercise, you declare and instantiate two variables of type `Item` in the `ShoppingCart` class and experiment with accessing properties and calling methods on the 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



Copyright © 2018, Oracle and/or its affiliates. All rights reserved.



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};
```



Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

An array is actually an object type and is handled implicitly through a class called `Array` (not available in the Java API documentation). Therefore, like other object types (`String` is an exception) it must be instantiated using the `new` keyword.

- In the top example, an `int` array called `ages` is declared and instantiated with a capacity to hold four elements.

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;
```

- Not permitted (compiler will show an error):

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

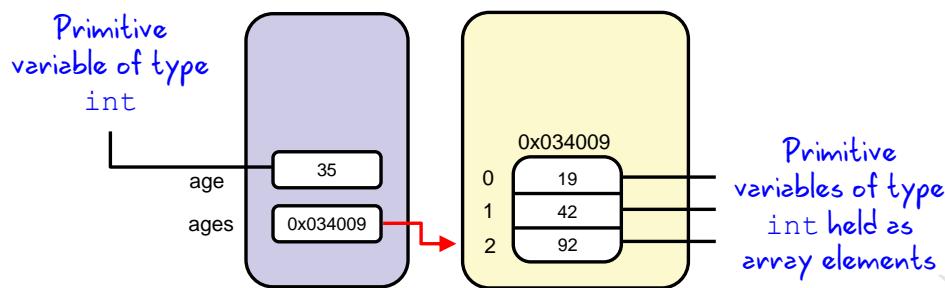


Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

As introduced in the lesson titled “Managing Multiple Items,” there are two approaches for creating and initializing arrays. Using the **new** keyword allows you to declare and instantiate an array of a particular size and initialize it at a later time.

Storing Arrays in Memory

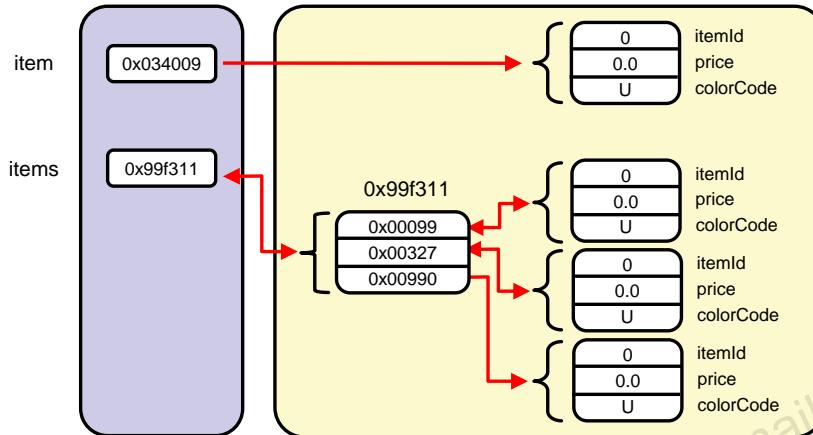
```
int age = 35;  
int[] ages = {19, 42, 92};
```



Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

Storing Arrays of Object References in Memory

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



Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

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



Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

Answer: c

a is incorrect. Declaring the array would look like this, assuming an array of object types: Type []
array_identifier;

b is incorrect. Setting array values would look like this, assuming an array of object types:

```
array_identifier[0] = new Type();
```

c is correct. The code example shows the array being initialized to a specific size.

d is incorrect. Declaring, instantiating, and setting array values would look like this, assuming an array of object types:

```
Type[] array_identifier = {new Type(), new Type(), new Type()};
```

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.



Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

**Answer: a, c, d**

## 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



Copyright © 2018, Oracle and/or its affiliates. All rights reserved.



## 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.



Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

In the remaining practices in this course, you will build an application that manages a Soccer League. The application will keep details on teams and players, as well as the results of games.

You will also write code that will randomly generate game results so that you can then develop code to list the Teams in rank order.

## Creating the Soccer Application

A separate project for each practice

Sample output showing events in a game

Sample output showing rank order of teams

The Greys vs. The Pinks (2014-03-08)  
Kickoff by Agatha Christie of The Greys. (0.0 mins.)  
Arthur Conan Doyle of The Pinks currently has possession. (6.0 mins.)  
GOAL! Scored by W. B. Yeats of The Greys. (7.0 mins.)  
Kickoff by Alan Patton of The Pinks. (8.0 mins.)  
Alexander Solzhenitsyn of The Pinks currently has possession. (11.0 mins.)  
GOAL! Scored by Arthur Conan Doyle of The Pinks. (14.0 mins.)  
Kickoff by Agatha Christie of The Greys. (18.0 mins.)  
Alan Patton of The Pinks currently has possession. (23.0 mins.)  
Agatha Christie of The Greys currently has possession. (24.0 mins.)  
GOAL! Scored by Agatha Christie of The Greys. (40.0 mins.)  
Kickoff by Arthur Conan Doyle of The Pinks. (44.0 mins.)  
Arthur Conan Doyle of The Pinks currently has possession. (49.0 mins.)  
GOAL! Scored by Arthur Conan Doyle of The Pinks. (55.0 mins.)  
Kickoff by Agatha Christie of The Greys. (59.0 mins.)  
Alan Patton of The Pinks currently has possession. (73.0 mins.)  
GOAL! Scored by W. B. Yeats of The Greys. (89.0 mins.)  
The Pinks win! (3 - 2)

Team Points  
The Reds:17:20  
The Blues:17:17  
The Pinks:12:17  
The Greens:8:12  
The Greys:6:13  
BUILD SUCCESSFUL (total time: 0 seconds)

Java™ ORACLE

Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

# Soccer Web Application

| Soccer League Games                 |                  |         |         |         |         |         |         |       |
|-------------------------------------|------------------|---------|---------|---------|---------|---------|---------|-------|
| <a href="#">Replay games</a>        |                  |         |         |         |         |         |         |       |
| Home Teams                          | Away Teams       |         |         |         |         |         |         |       |
|                                     | 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 |
|                                     | 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 15 |
|                                     | 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      |         |         |         |         |         |       |
| <a href="#">Return to main page</a> |                  |         |         |         |         |         |         |       |

Teams listed in rank order

Click the score  
of a game to  
show game  
details.

Points and goals scored  
used for ordering



Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

The code that you write in the practices can be used by a simple web application to view the results of games in the league. You will see a demonstration of this.

## 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



Copyright © 2018, Oracle and/or its affiliates. All rights reserved.



## Practices Overview

- 6-1: Creating Classes for the Soccer League
- 6-2: Creating a Soccer Game



Copyright © 2018, Oracle and/or its affiliates. All rights reserved.

