Arrays of Objects

CSC 1051 – Data Structures and Algorithms I
Dr. Mary-Angela Papalaskari
Department of Computing Sciences
Villanova University
Course website:
www.csc.villanova.edu/~map/1051/

Some slides in this presentation are adapted from the slides accompanying Java Software Solutions by Lewis & Loftus
Outline

Declaring and Using Arrays

Arrays of Objects

Variable Length Parameter Lists

Two-Dimensional Arrays

Polygons and Polylines

Mouse Events and Key Events
Arrays - review

- Declaration:

  ```java
  int[] scores = new int[10];
  ```

This array holds 10 values that are indexed from 0 to 9

```
scores.length = 10
```
Alternate Array Syntax

• The brackets of the array type can be associated with the element type or with the name of the array.

• Therefore the following two declarations are equivalent:

```
double[] prices;
double prices[];
```

• The first format generally is more readable and should be used.
Initializer Lists

• An *initializer list* can be used to instantiate and fill an array in one step

• The values are delimited by braces and separated by commas

• Examples:

```c
int[] units = {147, 323, 89, 933, 540, 269, 97, 114, 298, 476};

char[] grades = {'A', 'B', 'C', 'D', 'F'};
```
Initializer Lists

• Note that when an initializer list is used:
  – the `new` operator is not used
  – no size value is specified

• The size of the array is determined by the number of items in the list

• An initializer list can be used only in the array declaration

• See `Primes.java`
public class Primes
{
    // Stores some prime numbers in an array and prints them.
    public static void main (String[] args)
    {
        int[] primeNums = {2, 3, 5, 7, 11, 13, 17, 19};

        System.out.println ("Array length: "+ primeNums.length);
        System.out.println ("The first few prime numbers are:");
        for (int prime : primeNums)
            System.out.print (prime + " ");
    }
}
public class Primes {
    // Store some prime numbers in an array and print them.
    public static void main (String[] args) {
        int[] primeNums = {2, 3, 5, 7, 11, 13, 17, 19};

        System.out.println ("Array length: " + primeNums.length);
        System.out.println ("The first few prime numbers are: " + Arrays.toString(primeNums));

        for (int prime : primeNums) {
            System.out.print (prime + "  ");
        }
    }
}
Arrays as Parameters

• An entire array can be passed as a parameter to a method (just like any other object).

• **Exercise:** Write a method that increments the value of each element in an array.
Arrays of Objects

- Example: An array of Strings
  
  ```java
  String[] words = new String[5];
  ```

- It does NOT create the `String` objects themselves
Arrays of Objects

• The `words` array when initially declared:

![Diagram of an empty array]

• At this point, the following line of code would throw a `NullPointerException`:

```java
System.out.println(words[0]);
```
Arrays of Objects

- After some `String` objects are created and stored in the array:

```java
String[] words = {
    "friendship",
    "loyalty",
    "honor"
};
```
Arrays of Objects

- The following declaration creates an array object called `verbs` and fills it with four `String` objects created using string literals

```java
String[] verbs = {"play", "work", "eat", "sleep", "run"};
```
Arrays of Objects

• The following example creates an array of Grade objects, each with a string representation and a numeric lower bound.

• The letter grades include plus and minus designations, so must be stored as strings instead of char.

• See GradeRange.java
• See Grade.java
//********************************************************************
//  GradeRange.java       Author: Lewis/Loftus
//
//  Demonstrates the use of an array of objects.
//********************************************************************

public class GradeRange
{
    //------------------------------------------------------------------
    //  Creates an array of Grade objects and prints them.
    //------------------------------------------------------------------
    public static void main (String[] args)
    {
        Grade[] grades =
        {
            new Grade("A", 95), new Grade("A-", 90),
            new Grade("B+", 87), new Grade("B", 85), new Grade("B-", 80),
            new Grade("C+", 77), new Grade("C", 75), new Grade("C-", 70),
            new Grade("D+", 67), new Grade("D", 65), new Grade("D-", 60),
            new Grade("F", 0)
        };

        for (Grade letterGrade : grades)
            System.out.println (letterGrade);
    }
}
public class GradeRange {
  // -----------------------------
  // Creates an array of Grade objects and prints them.
  // -----------------------------
  public static void main (String[] args) {
    Grade[] grades = {
      new Grade("A", 95), new Grade("A-", 90),
      new Grade("B+", 87), new Grade("B", 85), new Grade("B-", 80),
      new Grade("C+", 77), new Grade("C", 75), new Grade("C-", 70),
      new Grade("D+", 67), new Grade("D", 65), new Grade("D-", 60),
      new Grade("F", 0)
    };

    for (Grade letterGrade : grades)
      System.out.println (letterGrade);
  }
}
public class Grade
{
    private String name;
    private int lowerBound;

    // Constructor: Sets up this Grade object with the specified grade name and numeric lower bound.
    public Grade (String grade, int cutoff)
    {
        name = grade;
        lowerBound = cutoff;
    }

    // Returns a string representation of this grade.
    public String toString()
    {
        return name + "\t" + lowerBound;
    }
}
// Name mutator.
public void setName (String grade)
{
    name = grade;
}

// Lower bound mutator.
public void setLowerBound (int cutoff)
{
    lowerBound = cutoff;
}
continue

//------------------------------
//  Name accessor.
//------------------------------
public String getName()
{
    return name;
}

//------------------------------
//  Lower bound accessor.
//------------------------------
public int getLowerBound()
{
    return lowerBound;
}
Arrays of Objects

• Now let's look at an example that manages a collection of DVD objects

• An initial capacity of 100 is created for the collection

• If more room is needed, a private method is used to create a larger array and transfer the current DVDs

• See Movies.java
• See DVDCollection.java
• See DVD.java
public class Movies {
    public static void main (String[] args) {
        DVDCollection movies = new DVDCollection();
        movies.addDVD ("The Godfather", "Francis Ford Coppala", 1972, 24.95, true);
        movies.addDVD ("District 9", "Neill Blomkamp", 2009, 19.95, false);
        movies.addDVD ("Iron Man", "Jon Favreau", 2008, 15.95, false);
        movies.addDVD ("All About Eve", "Joseph Mankiewicz", 1950, 17.50, false);
        movies.addDVD ("The Matrix", "Andy & Lana Wachowski", 1999, 19.95, true);
        System.out.println (movies);
        movies.addDVD ("Iron Man 2", "Jon Favreau", 2010, 22.99, false);
        movies.addDVD ("Casablanca", "Michael Curtiz", 1942, 19.95, false);
        System.out.println (movies);
    }
}
public class Movies {
    public static void main (String[] args) {
        DVDCollection movies = new DVDCollection();
        movies.addDVD("The Godfather", "Francis Ford Coppola", 1972, 24.95, true);
        movies.addDVD("District 9", "Neill Blomkamp", 2009, 19.95, false);
        movies.addDVD("Iron Man", "Jon Favreau", 2008, 15.95, false);
        movies.addDVD("All About Eve", "Joseph Mankiewicz", 1950, 17.50, false);
        movies.addDVD("The Matrix", "Andy & Lana Wachowski", 1999, 19.95, true);
        System.out.println(movies);
        movies.addDVD("Iron Man 2", "Jon Favreau", 2010, 22.99, false);
        movies.addDVD("Casablanca", "Michael Curtiz", 1942, 19.95, false);
        System.out.println(movies);
    }
}
public class Movies {
    public static void main(String[] args) {
        DVDCollection movies = new DVDCollection();
        movies.addDVD("The Godfather", "Francis Ford Coppala", 1972, 24.95, true);
        movies.addDVD("District 9", "Neill Blomkamp", 2009, 19.95, false);
        movies.addDVD("Iron Man", "Jon Favreau", 2008, 15.95, false);
        movies.addDVD("All About Eve", "Joseph Mankiewicz", 1950, 17.50, false);
        movies.addDVD("The Matrix", "Andy & Lana Wachowski", 1999, 19.95, true);
        System.out.println(movies);
        movies.addDVD("Iron Man 2", "Jon Favreau", 2010, 22.99, false);
        movies.addDVD("Casablanca", "Michael Curtiz", 1942, 19.95, false);
        System.out.println(movies);
    }
}
import java.text.NumberFormat;

public class DVDCollection
{
    private DVD[] collection;
    private int count;
    private double totalCost;

    //----------------------------------------------------------------------------
    // Constructor: Creates an initially empty collection.
    //----------------------------------------------------------------------------
    public DVDCollection ()
    {
        collection = new DVD[100];
        count = 0;
        totalCost = 0.0;
    }

    continue
```
continue

// Adds a DVD to the collection, increasing the size of the
// collection array if necessary.
public void addDVD (String title, String director, int year,
                   double cost, boolean bluRay)
{
    if (count == collection.length)
        increaseSize();

    collection[count] = new DVD (title, director, year, cost, bluRay);
    totalCost += cost;
    count++;
}
continue
```
public String toString()
{
    NumberFormat fmt = NumberFormat.getCurrencyInstance();

    String report = "~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
                       My DVD Collection
                       ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~\n"
    report += "Number of DVDs: " + count + "\n"
    report += "Total cost: " + fmt.format(totalCost) + "\n"
    report += "Average cost: " + fmt.format(totalCost/count) + "\n"
    report += "DVD List:\n"
    for (int dvd = 0; dvd < count; dvd++)
        report += collection[dvd].toString() + "\n";

    return report;
}
private void increaseSize ()
{
    DVD[] temp = new DVD[collection.length * 2];

    for (int dvd = 0; dvd < collection.length; dvd++)
        temp[dvd] = collection[dvd];

    collection = temp;
}
import java.text.NumberFormat;

public class DVD {
    private String title, director;
    private int year;
    private double cost;
    private boolean bluRay;

    //-------------------------------------------------------------------------------
    // Creates a new DVD with the specified information.
    //-------------------------------------------------------------------------------
    public DVD (String title, String director, int year, double cost, boolean bluRay) {
        this.title = title;
        this.director = director;
        this.year = year;
        this.cost = cost;
        this.bluRay = bluRay;
    }

    continue
public String toString()
{
    NumberFormat fmt = NumberFormat.getCurrencyInstance();

    String description;

    description = fmt.format(cost) + "\t" + year + "\t";
    description += title + "\t" + director;

    if (bluRay)
    {
       description += "\t" + "Blu-Ray";
    }

    return description;
}
Arrays of Objects

- A UML diagram for the Movies program:
Homework

- Review Section 8.1- 8.3
- Read Section 8.4 to prepare for next class

Exercises Handout

Some slides in this presentation are adapted from the slides accompanying Java Software Solutions by Lewis & Loftus