ORACLE

Academy

Java Fundamentals

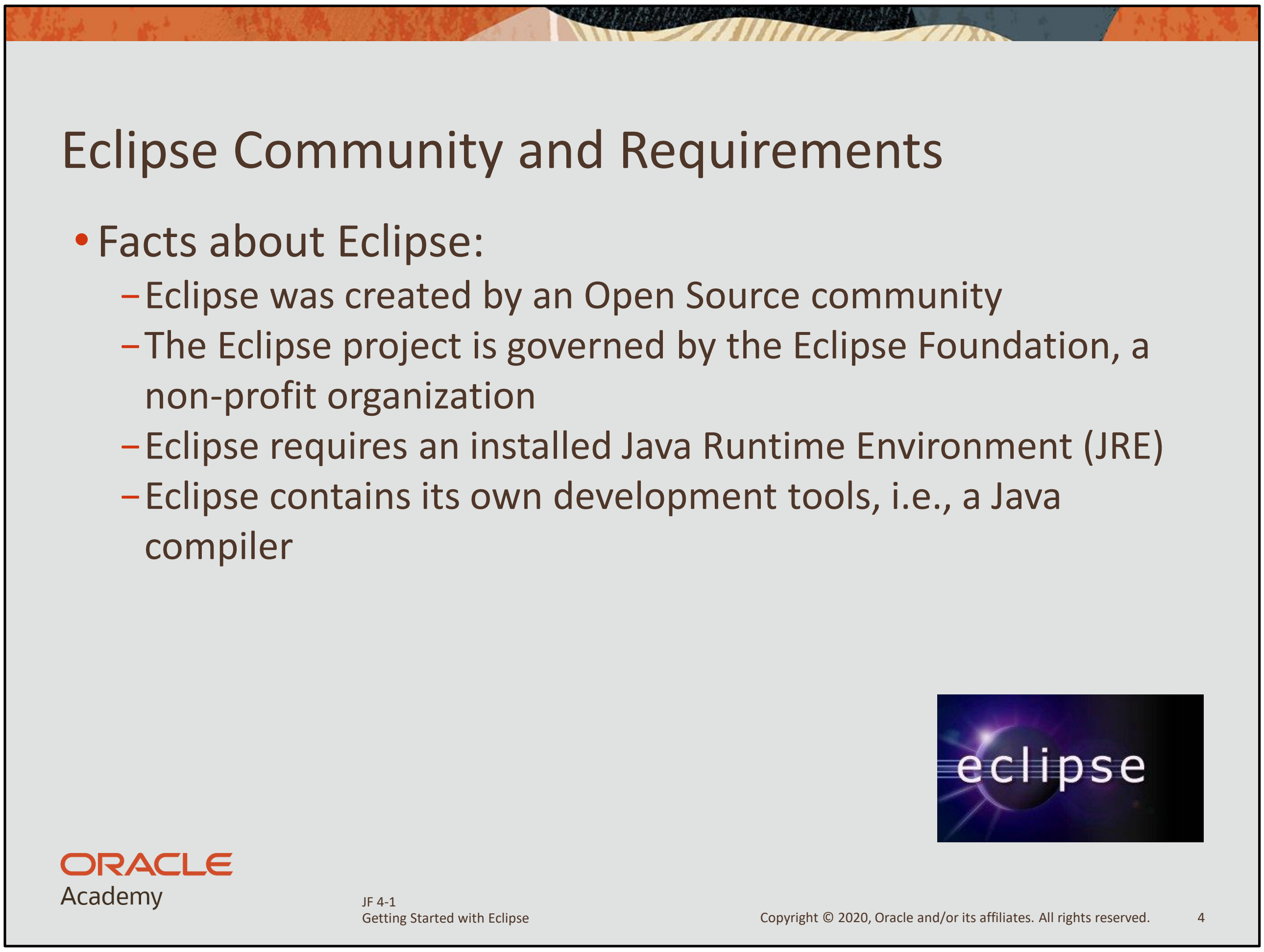
4-1
Getting Started with Eclipse

ORACLE
Academy

Objectives

- This lesson covers the following objectives:
  - Identify components of Eclipse
  - Identify components of a Java application
  - Compile an application
  - Test to ensure application is complete
  - Write the code for GalToLit.java
  - Modify a program to execute error free
  - Modify a program to use a formula to convert units of measure

ORACLE
Academy



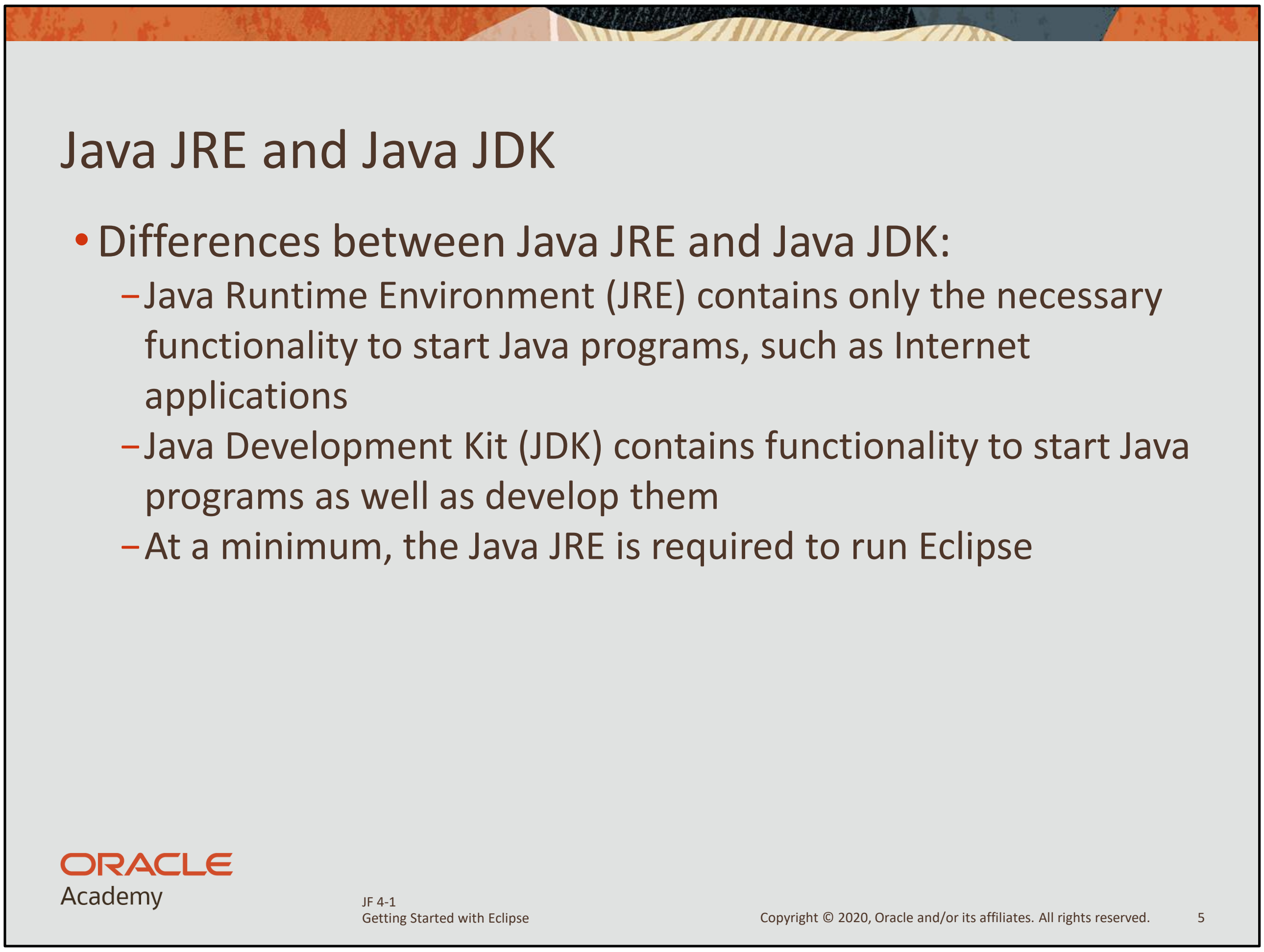
Eclipse Community and Requirements

- Facts about Eclipse:
  - Eclipse was created by an Open Source community
  - The Eclipse project is governed by the Eclipse Foundation, a non-profit organization
  - Eclipse requires an installed Java Runtime Environment (JRE)
  - Eclipse contains its own development tools, i.e., a Java compiler

ORACLE
Academy

While this course officially uses Eclipse the course can be taught with any IDE that allows a program to be edited, compiled, and run. S04 L01 is the only lesson in the course that directly discusses Eclipse.

Oracle Academy is a member (sponsor) of the Eclipse Foundation.

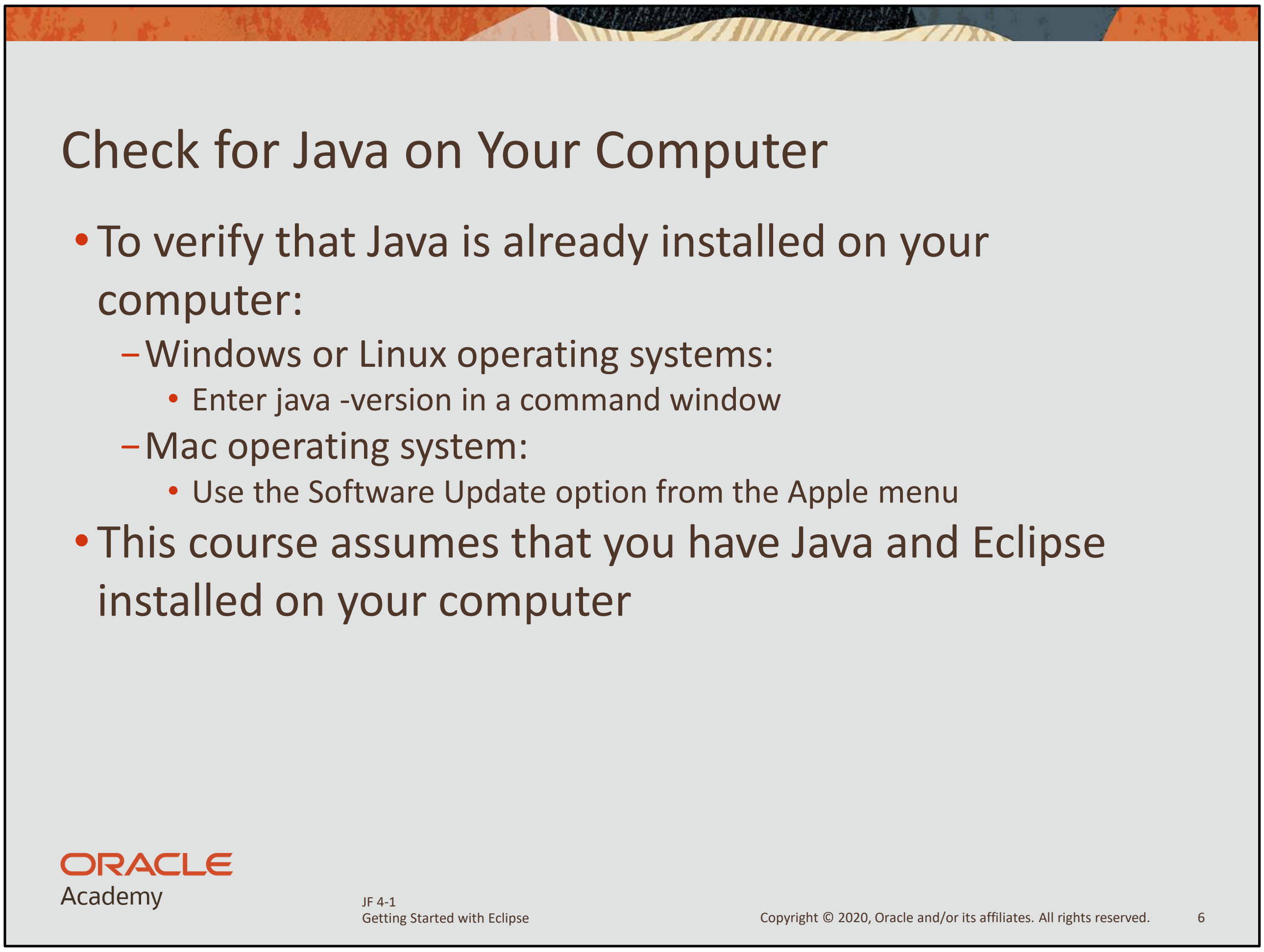


Java JRE and Java JDK

- Differences between Java JRE and Java JDK:
  - Java Runtime Environment (JRE) contains only the necessary functionality to start Java programs, such as Internet applications
  - Java Development Kit (JDK) contains functionality to start Java programs as well as develop them
  - At a minimum, the Java JRE is required to run Eclipse

ORACLE Academy

The JRE is sometimes referred to as the VM (Virtual Machine). For this course, the JDK will be required.



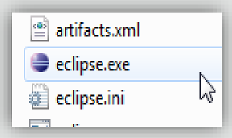
Check for Java on Your Computer

- To verify that Java is already installed on your computer:
  - Windows or Linux operating systems:
    - Enter java -version in a command window
  - Mac operating system:
    - Use the Software Update option from the Apple menu
- This course assumes that you have Java and Eclipse installed on your computer

With appropriate arguments, java is the command that runs Java programs. javac is the command that compiles .java source code files into .class files that can then be executed with the java command. The IDE is the front end interface that uses these commands.

Steps to Launch Eclipse

- On a computer with Windows double-click on the file eclipse.exe
- On a Linux or Mac computer double click on the file eclipse
- When prompted, enter the pathname for the workspace into which you will store your Java projects and click the OK button
- This can be your c:\ drive, or possibly a network drive
- Eclipse will start and display the Welcome page
- Close the Welcome page by clicking the X next to the Welcome tab name

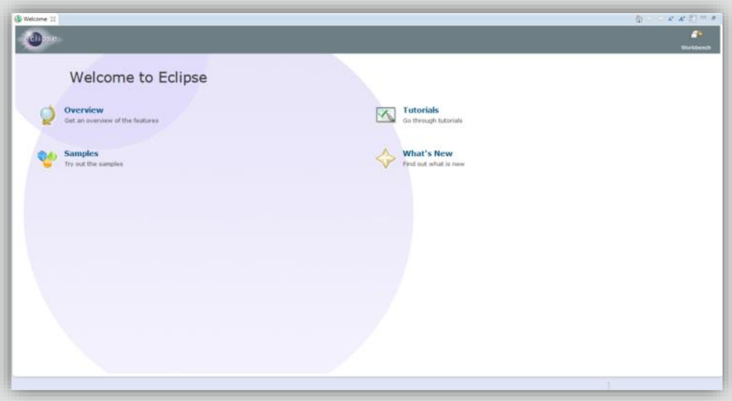


ORACLE Academy

As of May 2017, the screen captures are for Eclipse version Kepler, this is only apparent with the Welcome screen. The most recent version of Eclipse should be used with this course.

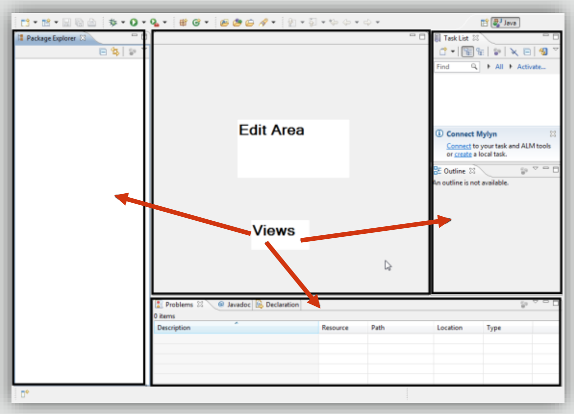
Eclipse Welcome Page

- There are valuable resources available from the Welcome page
- You can return to the Welcome page by choosing Welcome from the Help menu



Eclipse Edit Area and Views

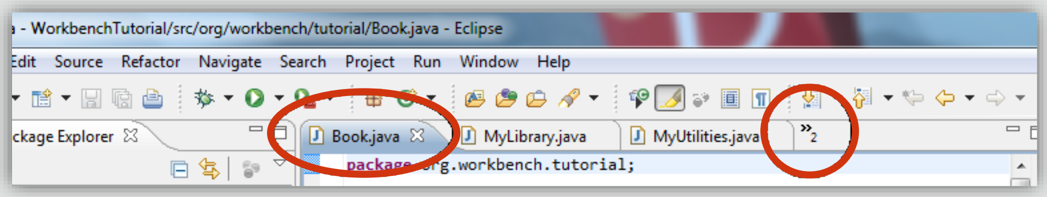
- Eclipse provides an edit area and several views
- An editor is where you type in your Java source code
- Views are sub-windows that provide information about your project



ORACLE
Academy

Eclipse Edit Area Tabs

- The edit area uses tabs when more than one file is open

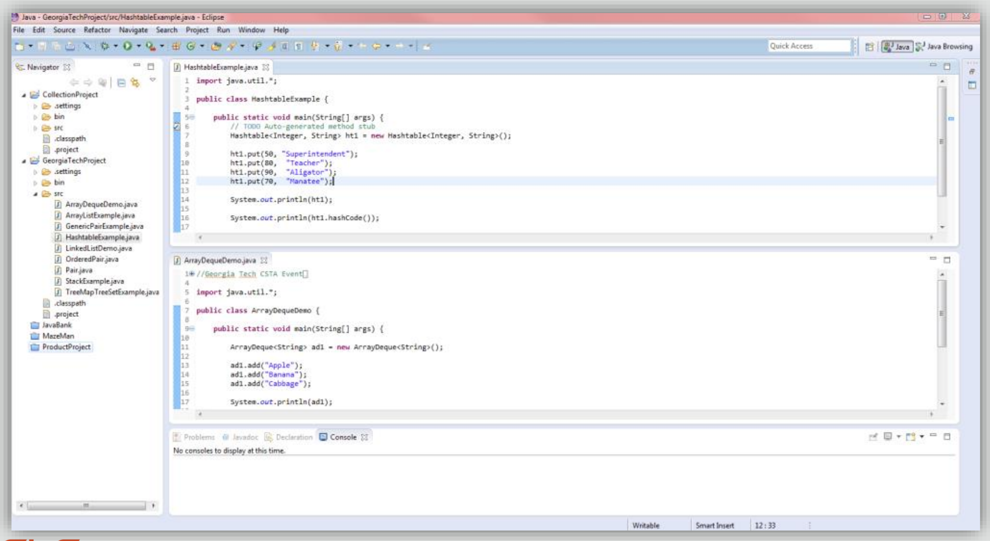

a - WorkbenchTutorial/src/org/workbench/tutorial/Book.java - Eclipse
Edit Source Refactor Navigate Search Project Run Window Help
ckage Explorer
Book.java
MyLibrary.java
MyUtilities.java
»2

```
package org.workbench.tutorial;
```

ORACLE Academy

Eclipse Edit Area Windows

- The edit area can have multiple windows occupy the space



HashtableExample.java

```
import java.util.*;

public class HashtableExample {

    public static void main(String[] args) {
        // TODO Auto-generated method stub
        Hashtable<Integer, String> ht1 = new Hashtable<Integer, String>();

        ht1.put(50, "Superintendent");
        ht1.put(80,  "Teacher");
        ht1.put(90,  "Aligator");
        ht1.put(70,  "Manatee");

        System.out.println(ht1);

        System.out.println(ht1.hashCode());

```

ArrayDequeDemo.java

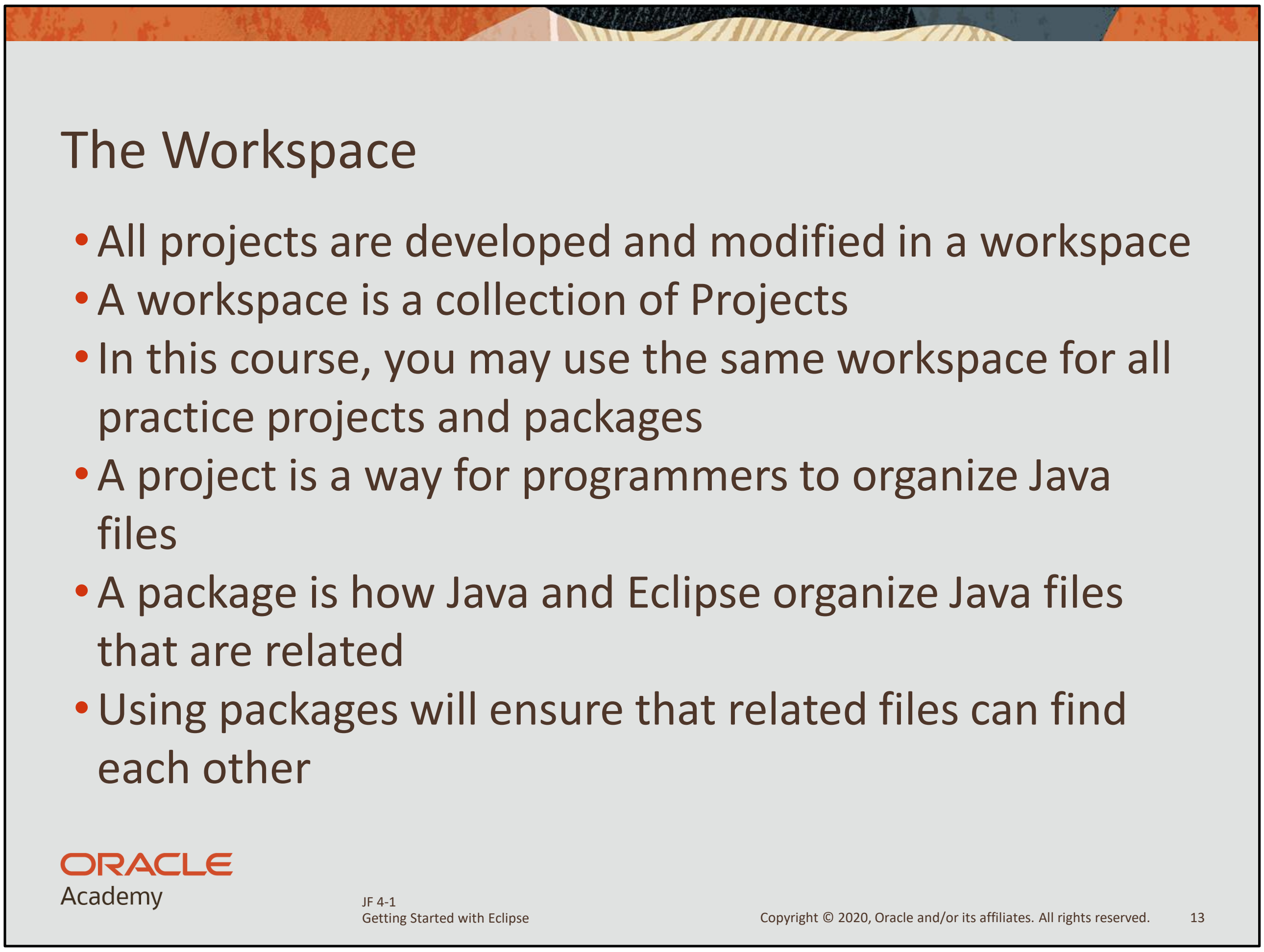
```
//Georgia Tech CSTA Event

import java.util.*;

public class ArrayDequeDemo {

    public static void main(String[] args) {

        ArrayDeque<String> ad1 = new ArrayDeque<String>();

        ad1.add("Apple");
        ad1.add("Banana");
        ad1.add("Cabbage");

        System.out.println(ad1);
```

```
No consoles to display at this time.
```

ORACLE Academy

Additional Details on Edit Areas and Views

- A combination of views and editors are referred to as a perspective
- You can choose Open Perspective from the Window menu



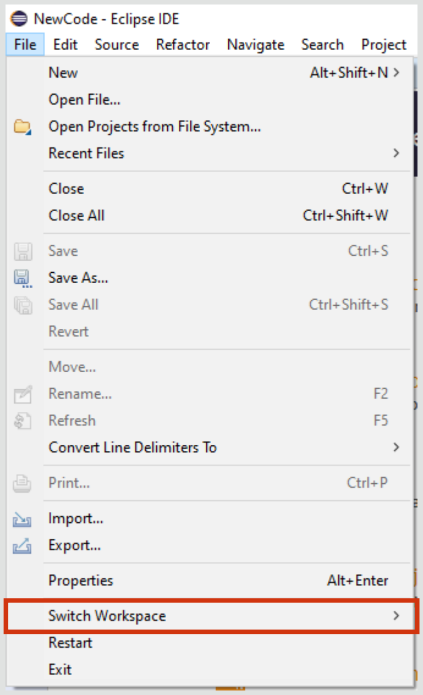
The Workspace

- All projects are developed and modified in a workspace
- A workspace is a collection of Projects
- In this course, you may use the same workspace for all practice projects and packages
- A project is a way for programmers to organize Java files
- A package is how Java and Eclipse organize Java files that are related
- Using packages will ensure that related files can find each other

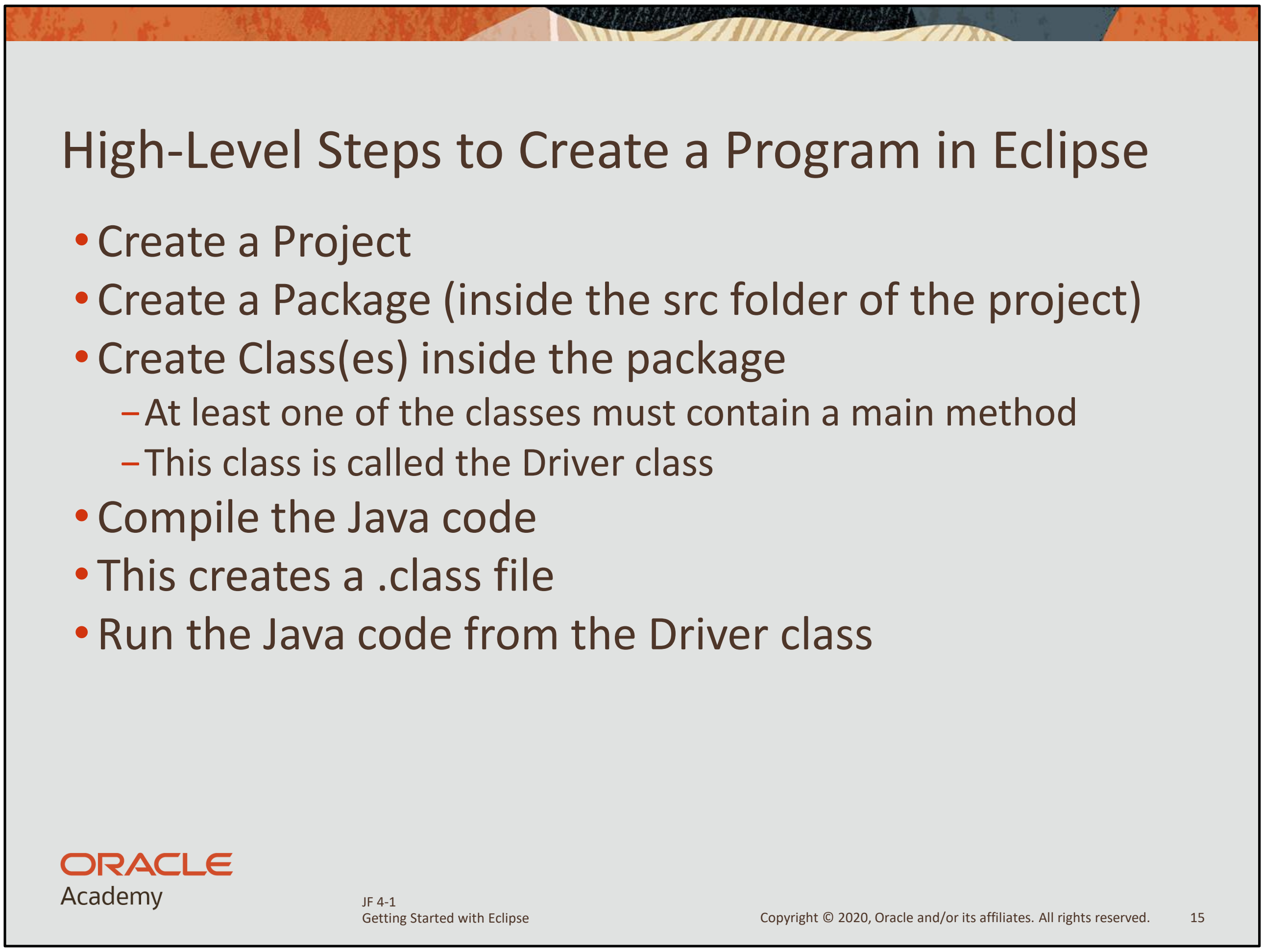
Do not keep the workspace in the same folder as Eclipse itself. That way if Eclipse is corrupted or upgraded the workspace will not be lost. Keeping both on a portable device such as a flash drive is an option. BACK UP THE WORKSPACE!

Switching Workspaces

- You can Switch Workspaces
  - From the File menu change to a different physical location for your files



ORACLE Academy



High-Level Steps to Create a Program in Eclipse

- Create a Project
- Create a Package (inside the src folder of the project)
- Create Class(es) inside the package
  - At least one of the classes must contain a main method
  - This class is called the Driver class
- Compile the Java code
- This creates a .class file
- Run the Java code from the Driver class

ORACLE Academy

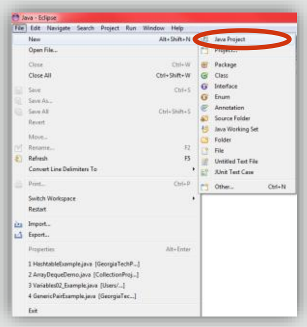
Packages are covered in detail in Java Programming and are used to organize Java classes. A project may be created without using packages. Eclipse refers to this as the default package and warns that the practice is discouraged.

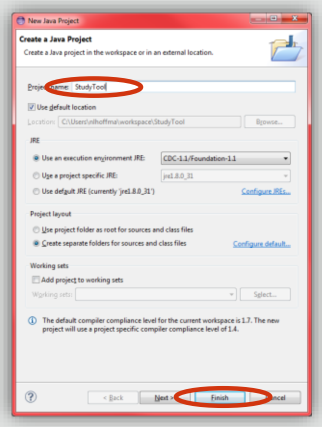
Projects in Eclipse

- In Eclipse:
  - All programs must reside inside a project for proper compilation
  - You may have one or multiple class files in one project
  - One of the classes must contain a main method

Steps to Create a Project in Eclipse

- Choose File → New → Java Project*
- Enter a project name and click Finish
- All project values are set to default by clicking the Finish button




New Java Project

Create a Java Project

Create a Java project in the workspace or in an external location.

Project name: StudyTool

[✓] Use default location

Location: C:\Users\nlhoffma\workspace\StudyTool

JRE

[● Use an execution environment JRE:] CDC-1.1/Foundation-1.1

Use a project specific JRE: jre1.8.0_31

Use default JRE (currently 'jre1.8.0_31')

Project layout

Use project folder as root for sources and class files

[● Create separate folders for sources and class files]

Working sets

[ ] Add project to working sets

Working sets:

The default compiler compliance level for the current workspace is 1.7. The new project will use a project specific compiler compliance level of 1.4.

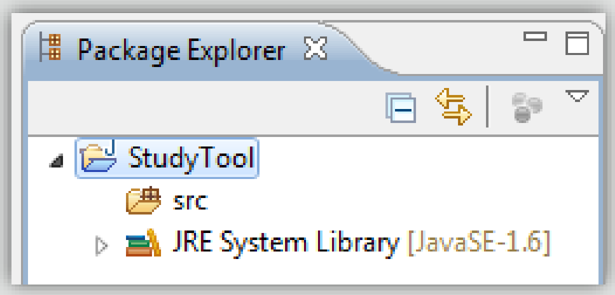
ORACLE
Academy

If the project pane is closed, it may be reopened from the Window -> Show View menus and then selecting Project Explorer. (Note that for the purposes of this course, there is no significant difference between the Project Explorer and the Package Explorer.)

*Update: If New -> Java Project is not available, choose Window -> Open Perspective -> Java. Then New -> Java project will be available.

Project Display

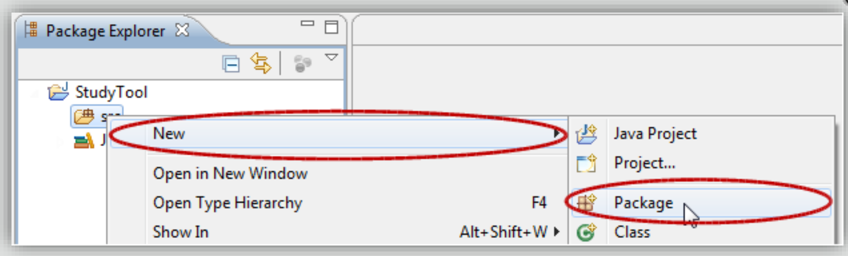
- The project is created and displayed as a folder
- It displays in the Package view to the left of the edit area



Steps to Create a Package

- Select the src folder in the Package view
- Right click the src folder and choose New → Package

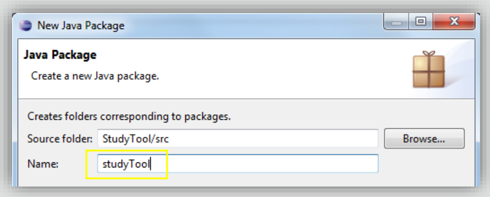
A package, in Java, is a mechanism for organizing Java classes into namespaces or containers. In Eclipse, packages are created inside projects.



Naming Packages

- Name this package the same name as the Project using lower camel case

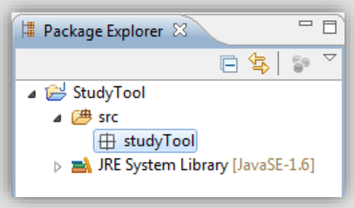
Camel case is the practice of stringingCapitalizedWords together with no spaces. Lower camel case does not capitalize the lead word.



New Java Package

Java Package
Create a new Java package.

Creates folders corresponding to packages.
Source folder: StudyTool/src
Name: studyTool



Package Explorer

```
StudyTool
  src
    studyTool
  JRE System Library [JavaSE-1.6]
```

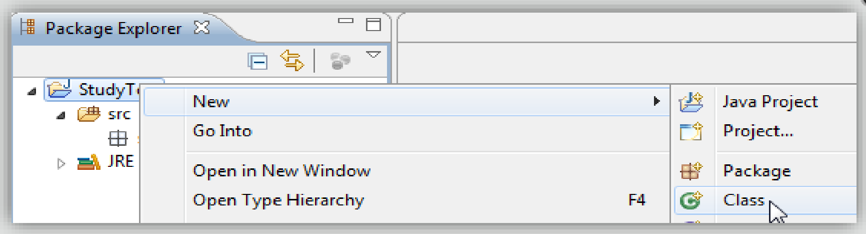
ORACLE Academy

Conventional naming practices for packages will be covered in Java Programming. For now, we use the same name for the package and the project for convenience.

Steps to Create a Class

- Right click on the project name to create a new class named StudyPage in the studyTool package
- Select the option to create the main method

A class in Java is a construct that is used as a blueprint to create objects. It can also be a construct in which objects are created.

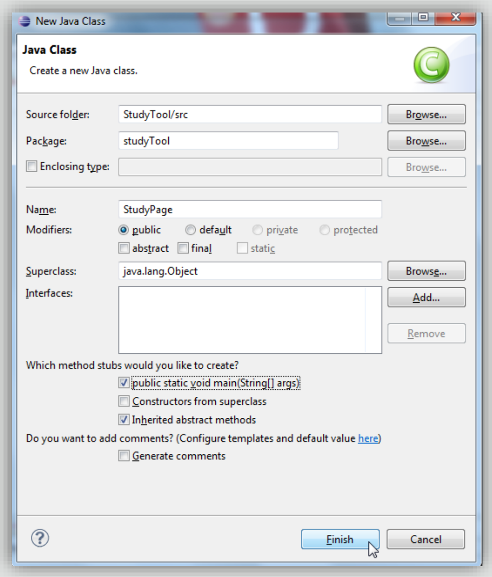


ORACLE
Academy

Create a Class Display

- The option is selected to create the main method

A main method, in Java, is the method inside a class that runs when the class is compiled and run. This class is referred to as the Driver Class.



Java Class
Create a new Java class.

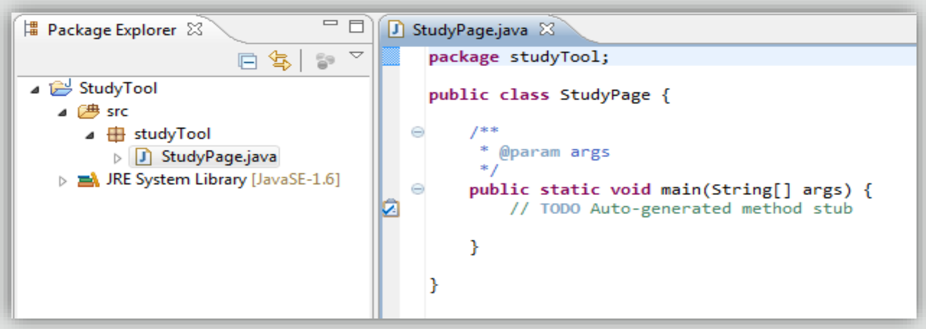
Source folder: StudyTool/src
Package: studyTool
Enclosing type: [ ]
Name: StudyPage
Modifiers: [● public] ○ default ○ private ○ protected
[ ] abstract [ ] final [ ] static
Superclass: java.lang.Object
Interfaces:

Which method stubs would you like to create?
[✓] public static void main(String[] args)
[ ] Constructors from superclass
[✓] Inherited abstract methods
Do you want to add comments? (Configure templates and default value here)
[ ] Generate comments

ORACLE
Academy

New Class

- Notice the following:
  - The StudyPage.java class appears in the studyTool package in the Package Explorer View
  - The StudyPage.java class is created with the main method



```
package studyTool;

public class StudyPage {

    /**
     * @param args
     */
    public static void main(String[] args) {
        // TODO Auto-generated method stub

    }

}
```

The class name in the .java file must match the name of the .java file (without the .java extension). If the class name is changed in the source code, the file name must also be changed. It is recommended that this NOT be done manually. Right click on the class in the Project pane and select Refactor -> Rename. This has the added benefit of updating the class name change in all files where it is used in the project.

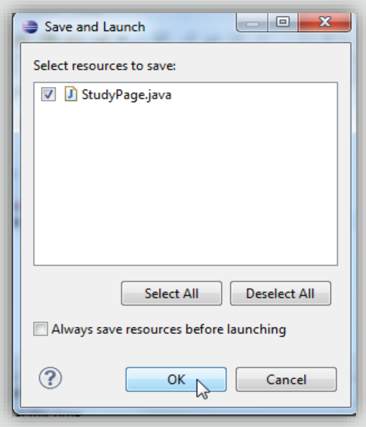
Steps to Create and Run Java Code

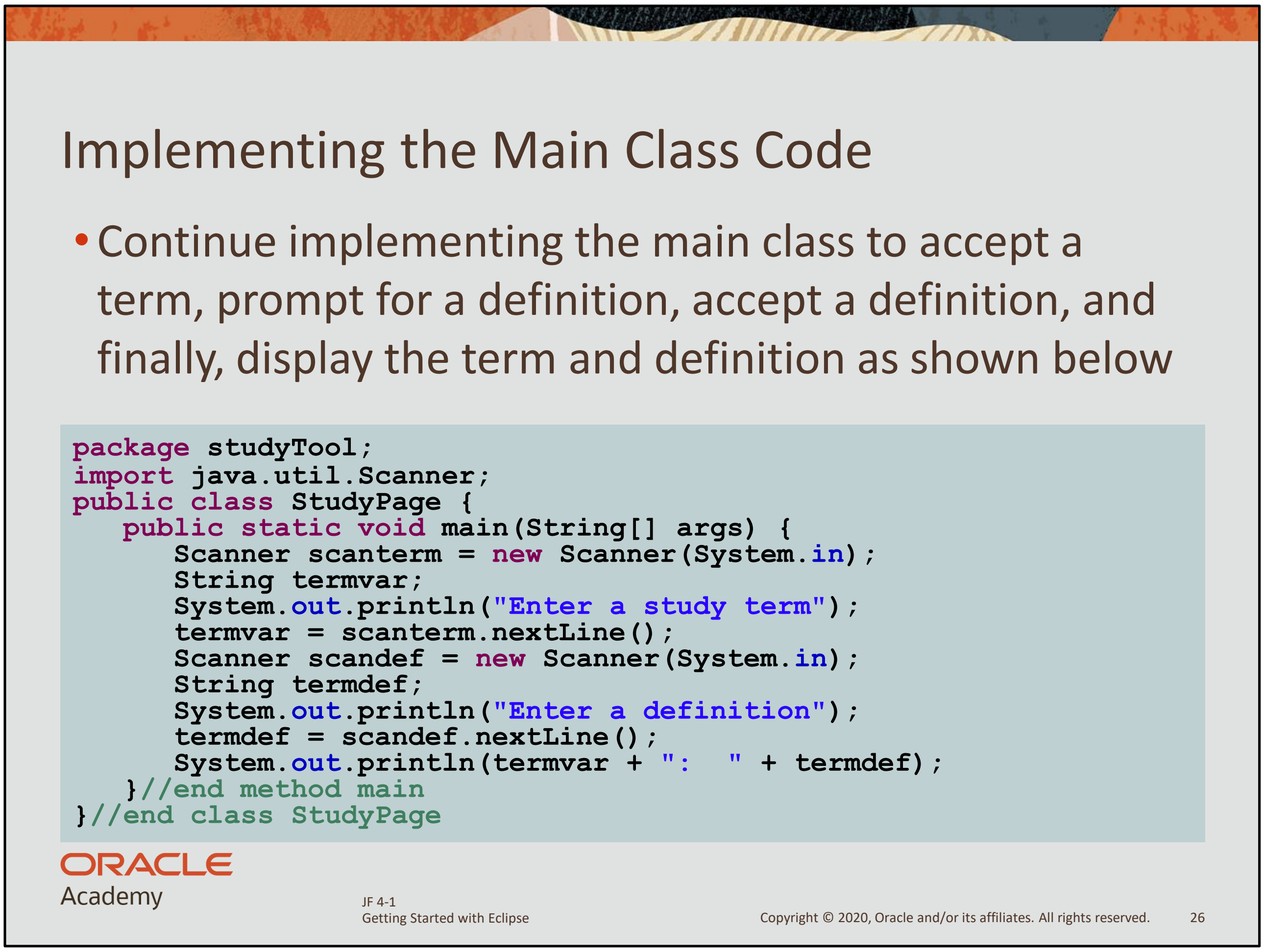
- To implement the main method and run the program, enter the following information into the main class
- The method inside a class that runs when the class is compiled and ran is the main method

```
public class StudyPage {
   public static void main(String[] args) {
      System.out.println("Enter a study term");
   }//end method main

}//end class StudyPage
```

Steps to Create and Run Java Code

- Right click on StudyPage.java in the package view
- Choose Run As → Java Application
- Save the class when prompted
- Note results display in the Console View


Save and Launch
Select resources to save:
☑ StudyPage.java
Select All
Deselect All
☐ Always save resources before launching
OK
Cancel



Implementing the Main Class Code

- Continue implementing the main class to accept a term, prompt for a definition, accept a definition, and finally, display the term and definition as shown below

```
package studyTool;
import java.util.Scanner;
public class StudyPage {
   public static void main(String[] args) {
      Scanner scanterm = new Scanner(System.in);
      String termvar;
      System.out.println("Enter a study term");
      termvar = scanterm.nextLine();
      Scanner scandef = new Scanner(System.in);
      String termdef;
      System.out.println("Enter a definition");
      termdef = scandef.nextLine();
      System.out.println(termvar + ":  " + termdef);
   }//end method main
}//end class StudyPage
```

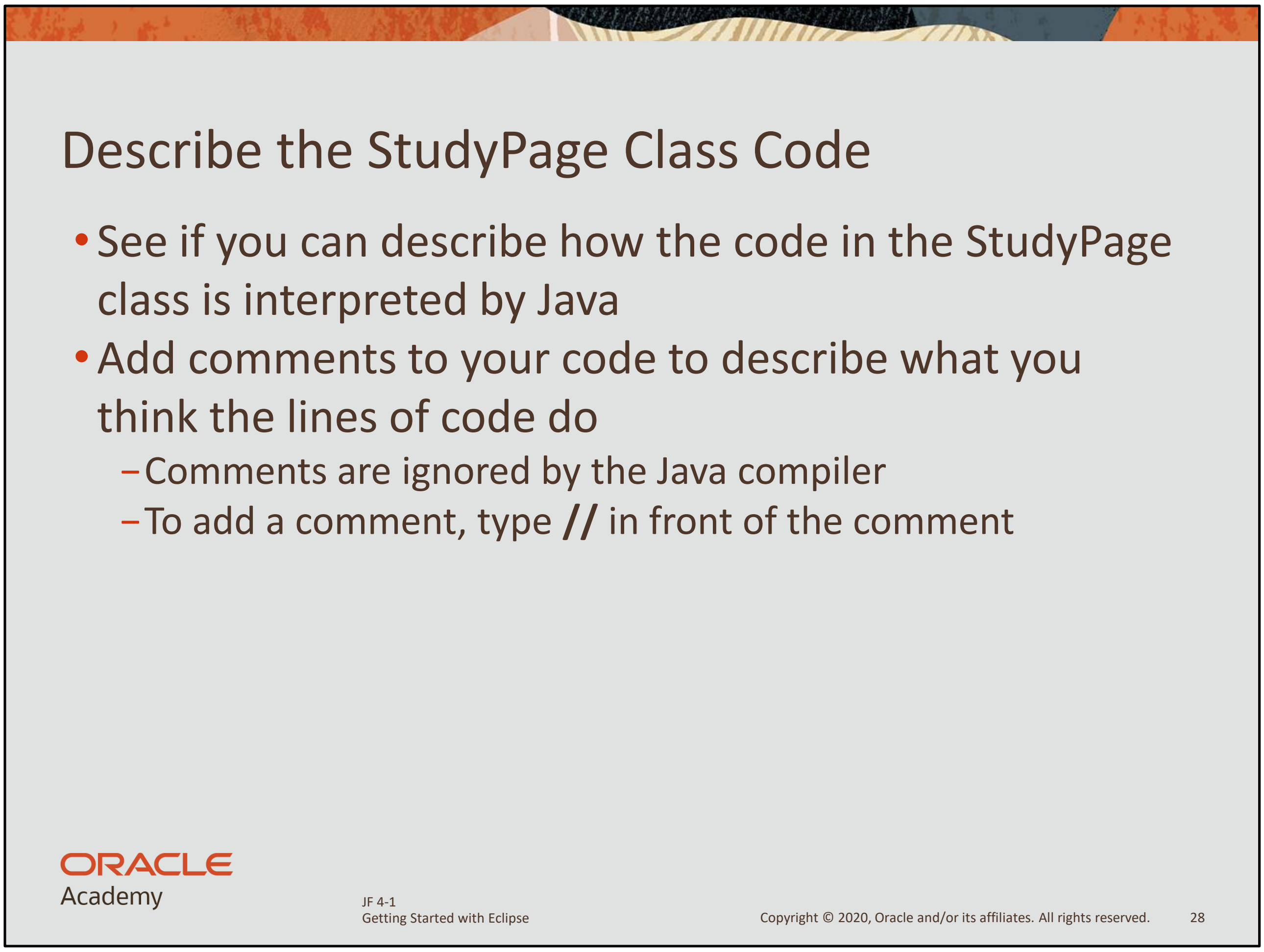
ORACLE Academy

The Scanner class is covered in detail in Section 5, Lesson 1.

Implementing the Main Class Code

- You may have to correct some syntax errors that are a result of typing errors
- See if you can correct them without asking for assistance
- Pay particular attention to the "**;**" at the end of each line, and that your "**{**" (left curly brace) has a matching "**}**" (right curly brace

ORACLE
Academy



Describe the StudyPage Class Code

- See if you can describe how the code in the StudyPage class is interpreted by Java
- Add comments to your code to describe what you think the lines of code do
  - Comments are ignored by the Java compiler
  - To add a comment, type **//** in front of the comment

ORACLE
Academy

/* Of course, the multi-line comment may be used, too. */

Program to Convert Gallons to Liters

- What is the formula to convert gallons to liters?
  - 1 US gallon is equivalent to 3.78541178 Liters

Pseudocode for Gallons to Liters Conversion

- What is the math calculation to complete the conversion?
- Pseudocode would look like:
  - Use Scanner to read in the number of gallons
  - Store the number of gallons in a variable
  - Multiply that variable by 3.785 (variable*3.785) and store this new value into a second variable (Hint: use double for the variables, not int)
  - Display the output
  - Use the StudyPage program as a guide to create the program on your own

ORACLE
Academy

Hints for Gallons to Liters Conversion

- Hints:
  - an int variable type is used to store whole numbers
  - a double variable type is used to store decimal numbers
  - int numgallons = 0; //declare a variable for number of gallons
  - double converttoliters = numgallons * 3.785

Terminology

- Key terms used in this lesson included:
  - Camel case
  - Eclipse:
    - edit and view areas, perspective, workspace
  - Java JRE vs. Java JDK
  - Java classes
  - Java packages
  - Java main methods

Summary

- In this lesson, you should have learned how to:
  - Identify components of Eclipse
  - Identify components of a Java application
  - Compile an application
  - Test to ensure application is complete
  - Write the code for GalToLit.java
  - Modify a program to execute error free
  - Modify a program to use a formula to convert units of measure

ORACLE Academy

ORACLE

Academy