Practices for Lesson 1

Practices Overview

These practices cover configuring a development environment for Java SE 7. These practices should not be performed unless directed to do so by your instructor.

In these practices, you will use the NetBeans IDE and create a project, create packages, and a Java main class, and then add classes and subclasses. You will also run your project from within the IDE and learn how to pass command-line arguments to your main class.

Note: There are two levels of practice for most of the practices in this course. Practices that are marked "Detailed Level" provide more instructions and, as the name implies, at a more detailed level. Practices that are marked "Summary Level" provide less detail, and likely will require additional review of the student guide materials to complete. The end state of the "Detailed" and "Summary" level practices is the same, so you can also use the solution end state as a tool to guide your experience.

Practice 1-1: Verifying Software Installation

Overview

In this practice, you will verify the installation of the necessary software to perform Java 7 software development. If verification fails, you will proceed with the remaining practices.

Assumptions

Your instructor has instructed you to perform these steps.

Summary

You have been given a computer system that will be used for Java SE 7 software development. You must validate that the Java 7 SE Development Kit is installed, the NetBeans 7.0.1 IDE is installed, and the NetBeans IDE is correctly configured to use JDK 7.

Tasks

1. Open a command or terminal window.

   Note: If you are using Windows, you can open a command window by performing the following steps:

   a. Click the Start button.

   b. Click Run.

   c. Type `cmd` in the Run dialog box and click the OK button.

2. Execute the `java -version` command. This verifies that a *JRE* is installed; this does not verify that the *JDK* is installed.

   a. Verify that the output of the `java -version` command matches the following example output. For 64-bit machines, the output should be:

   ```
   java version "1.7.0"
   Java(TM) SE Runtime Environment (build 1.7.0-b147)
   Java HotSpot(TM) 64-Bit Server VM (build 21.0-b17, mixed mode)
   ```

   For 32-bit machines, the output should be:

   ```
   java version "1.7.0"
   Java(TM) SE Runtime Environment (build 1.7.0-b147)
   Java HotSpot(TM) Client VM (build 21.0-b17, mixed mode, sharing)
   ```

   b. If a different version number or an error message is displayed, you may have one of the following possible problems:

   - The JRE/JDK is not installed.
   - The `java` command is not in your path.
   - The wrong JRE/JDK version is installed.
   - Multiple JREs/JDKs are installed.

Note: To exclude an incorrect `PATH` environment variable as the reason for an incorrect or unrecognized Java version, you may attempt to locate the path to your JDK and execute `java -version` by using a fully qualified path. For example:

```
"C:\Program Files\Java\jdk1.7.0\bin\java.exe" -version
```

3. Execute the `javac -version` command. This verifies that a JDK is installed.

   a. Verify that the output of the `javac -version` command matches the following example output:

   ```
   javac 1.7.0
   ```

   b. If a different version number or an error message is displayed, you may have one of the following possible problems:

   - The JDK is not installed.
   - The `javac` command is not in your path.
   - The wrong JDK version is installed.
   - Multiple JDKs are installed.

   Note: It is very common that the directory containing `javac` is not listed in your `PATH` environment variable. You do not need to modify the `PATH` for most IDEs to function; instead, locate the path to your JDK and execute `javac -version` by using a fully qualified path to verify the presence and version of the JDK. For example:

   ```
   "C:\Program Files\Java\jdk1.7.0\bin\javac.exe" -version
   ```

4. Start the NetBeans IDE and verify the version number of the JDK used by the IDE.

   a. The installation of NetBeans places a menu in your Start menu. Locate the NetBeans IDE 7.0.1 shortcut within the Start menu and click it.

   b. Within NetBeans, click the Help menu, and then click About.

   c. The About dialog box displays both the NetBeans and JDK version numbers that are being used. For 64-bit machines, you should see:

   ```
   Product Version: NetBeans IDE 7.0.1 (Build 201107282000)
   Java: 1.7.0; Java HotSpot(TM) 64-Bit Server VM 21.0-b17
   ```

   For 32-bit machines, you should see:

   ```
   Product Version: NetBeans IDE 7.0.1 (Build 201107282000)
   Java: 1.7.0; Java HotSpot(TM) Client VM 21.0-b17
   ```

   Note: Even if JDK7 was discovered in a previous step, you must verify that NetBeans is using Java 1.7.0.

   d. Click the Close button to close the About dialog box.

   Note: NetBeans 7.0.1 was the first version of NetBeans to fully support the final release of JDK 7.

Practice 1-2: Software Installation

Overview

In this practice, you will install the software needed to perform Java 7 SE development.

Assumptions

Your instructor has instructed you to perform these steps.

Summary

You have been given a computer system that will be used for Java SE 7 software development. You will use the information obtained in the previous practice to determine the software to be installed, and then you will install the software.

A JDK and NetBeans cobundle is available that will reduce the number of files to download. Downloading the JDK and NetBeans separately provides greater flexibility for selecting the JDK to be used (32-bit or 64-bit) and reduces the required amount of data to download if either the JDK or NetBeans is already installed.

For more information about Oracle JDK 7 supported operating systems, go to http://www.oracle.com/technetwork/java/javase/config-417990.html. Java 7 support for additional operating systems may be available from other channels.

Tasks

1. Obtain the required software.
    a. If you require both the JDK and NetBeans, the easiest method is to download the “JDK 7 with NetBeans 7.0.1” cobundle.
        1) Using a web browser, go to http://www.oracle.com/technetwork/java/javase/downloads/index.html.
        2) Locate the Java SE Development Kit (JDK) Cobundles table.
        3) Click the Download button for “JDK 7 with NetBeans 7.0.1.”
        4) You must accept the JDK 7 and NetBeans 7.0.1 Cobundle License Agreement to download the software.
        5) Download the file for your operating system. As of this writing, Linux, Solaris, and Windows downloads are available.
    b. If you require just the JDK, download the Java SE 7 JDK.
        1) Using a web browser, go to http://www.oracle.com/technetwork/java/javase/downloads/index.html.
        2) Locate the Java Platform, Standard Edition table.
        3) Click the Download button for Java SE 7 JDK.

            Note: Be sure to download the *JDK* and not the *JRE*.
        4) You must accept the JDK 7 and NetBeans 7.0.1 Cobundle License Agreement to download the software.
        5) Download the file for your operating system. As of this writing, Linux, Solaris, and Windows downloads are available.

    c. If you require just the NetBeans IDE, download “NetBeans IDE 7.0.1 for Java SE.”
        1) Using a web browser, go to http://download.netbeans.org/netbeans/7.0.1/final/.
        2) Click the Download button for Java SE.
        3) If the download does not start automatically, click the “download it here” link.

2. Install the required software.
    - Install the software downloaded in the previous step.
        - The JDK includes optional demos and a database known as the JavaDB. Install all optional JDK components if you are installing the JDK.
        - If you downloaded the JDK and NetBeans separately (not the cobundle), complete the installation of the JDK before installing NetBeans.
        - If you have multiple JDK versions installed, be sure to select the Java SE 7 JDK if prompted while installing NetBeans.
        - If NetBeans 7.0.1 was installed before JDK 7 or an older version of the JDK was selected during NetBeans installation, perform practice 1-3 to reconfigure NetBeans to use JDK 7.
3. Verify the software installation.
    - Repeat practice 1-1 to verify software installation.

Practice 1-3: Configuring NetBeans 7.0.1 to Utilize JDK 7

Overview

In this practice, you will configure NetBeans 7.0.1 to use a locally installed instance of JDK 7.

Assumptions

Your instructor has instructed you to perform these steps.

Summary

These steps are required only if both NetBeans 7.0.1 and JDK 7 are installed but NetBeans is running with the wrong JDK. You can verify the JDK being used by NetBeans with the steps outlined in practice 1-1, step 4. Failure to correct the NetBeans configuration to use JDK 7 will result in an inability to create Java projects that use Java 7 language features.

Tasks

1. Configure NetBeans to be aware of the Java 7 Platform.
    a. In the NetBeans IDE, select Tools, and then Java Platforms from the Main menu.
    b. Click the Add Platform button and specify the directory that contains your JDK 7 installation.
    c. In the Platform Name step, verify that the default locations of the Platform Sources zip file and API documentation are valid.
    d. Click the Finish button to close the Add Java Platform dialog box.
    e. Ensure that JDK 1.7 is selected in the Platforms list and click Close.
2. Configure NetBeans to start with Java SE 7 JDK.
    a. Open the directory containing the NetBeans configuration files, typically: C:\Program Files\NetBeans 7.0.1\etc\.
    b. Use a text editor to edit the `netbeans.conf` file.
    c. Modify the `netbeans_jdkhome` property to have a value of the JDK 7 installation location, for example: `netbeans_jdkhome="C:\Program Files\Java\jdk1.7.0"`.
3. Restart NetBeans and verify the JDK being used by NetBeans with the steps outlined in practice 1-1, step 4.

Practice 1-4: Summary Level: Creating Java Classes

Overview

In this practice, using the NetBeans IDE, you will create an `Employee` class, create a class with a `main` method to test the `Employee` class, compile and run your application, and print the results to the command line output.

Tasks

1. Start the NetBeans IDE by using the icon from Desktop.
2. Create a new project `EmployeePractice` in the `D:\labs\01\practices` directory (or your another directory) with an `EmployeeTest` main class in the `com.example` package.
3. Set the Source/Binary format to JDK 7.
    a. Right-click the project and select Properties.
    b. Select JDK 7 from the drop-down list for Source/Binary Format.
    c. Click OK.
4. Create another package called `com.example.domain`.
5. Add a Java Class called `Employee` in the `com.example.domain` package.
6. Code the `Employee` class.
    a. Add the following data fields to the `Employee` class—use your judgment as to what you want to call these fields in the class. Refer to the lesson materials for ideas on the field names and the syntax if you are not sure. Use `public` as the access modifier.

Field use	Recommended field type
Employee id	`int`
Employee name	`String`
Employee Social Security Number	`String`
Employee salary	`double`

7. Create a `no-arg` constructor for the `Employee` class.

   NetBeans can format your code at any time for you. Right-click anywhere in the class and select Format, or press the Alt-Shift-F key combination.

8. Add accessor/mutator methods for each of the fields.

   Note that NetBeans has a feature to create the getter methods for you. Click in your class where you want the methods to go, then right-click and choose Insert Code (or press the Alt-Insert keys). Choose getters from the Generate menu, and click the boxes next to the fields for which you want getter and setter methods generated.

9. Write code in the `EmployeeTest` class to test your `Employee` class.
    a. Construct an instance of `Employee`.
    b. Use the setter methods to assign the following values to the instance:

Field	Value
Employee id	101
Employee name	Jane Smith
Employee Social Security Number	012-34-4567
Employee salary	120_345.27

    c. In the body of the main method, use the `System.out.println` method to write the value of the employee fields to the console output.
    d. Resolve any missing import statements.
    e. Save the `EmployeeTest` class.
10. Run the `EmployeePractice` project.
11. (Optional) Add some additional employee instances to your test class.

Practice 1-5: Detailed Level: Creating Java Classes

Overview

In this practice, using the NetBeans IDE, you will create an `Employee` class, create a class with a `main` method to test the `Employee` class, compile and run your application, and print the results to the command-line output.

Tasks

1. Start the NetBeans IDE by using the icon from Desktop.
2. Create a new Project called `EmployeePractice` in NetBeans with an `EmployeeTest` class and a `main` method.
    a. Click File > New Project.
    b. Select Java from Categories, and Java Application from Projects.
    c. Click Next.
    d. On the New Application screen, enter the following values:

Window/Page Description	Choices or Values
Project Name:	`EmployeePractice`
Project Location	`D:\labs\01\practices (or your other directory)`
Use Dedicated Folder for Storing Libraries	Deselected
Create Main Class	Selected Change the name to `com.example.EmployeeTest` – `com.example` is the package name.
Set as Main Project	Selected

    e. Click Finish.

    You will notice that NetBeans has saved you a great deal of typing by creating a class called `EmployeeTest`, including the package name of `com.example`, and writing the skeleton of the `main` method for you.

3. Set the Source/Binary format to JDK 7.
    a. Right-click the project and select Properties.
    b. Select JDK 7 from the drop-down list for Source/Binary Format.
    c. Click OK.

4. Create another package called `com.example.domain`.
   a. Right-click the current package `com.example` under Source Packages.
   b. Select New > Java Package. The New Java Package dialog box highlights the new subpackage name.
   c. Enter `com.example.domain` in the Package Name field, and click Finish.

   You will notice that the icon beside the package name is gray in the Project—this is because the package has no classes in it yet.

5. Create a new Java Class called `Employee` in the `com.example.domain` package.
   a. Right-click the `com.example.domain` package and select New > Java Class.
   b. In the Class Name field, enter `Employee`.
   c. Click Finish to create the class.

   Notice that NetBeans has generated a class with the name `Employee` in the package `com.example.domain`.

6. Code the `Employee` class.
   a. Add the following data fields to the `Employee` class.

Field use	Access	Recommended field type	Field name
Employee id	`public`	`int`	`empId`
Employee name	`public`	`String`	`name`
Employee Social Security Number	`public`	`String`	`ssn`
Employee salary	`public`	`double`	`salary`

   b. Add a constructor to the `Employee` class:

```
public Employee() { }
```

   NetBeans can format your code at any time for you. Right-click anywhere in the class and select Format, or press the Alt-Shift-F key combination.

   c. Create accesor/mutator (getter/setter) methods for each of the fields.

   Note that NetBeans has a feature to create the getter methods for you. Click in your class where you want the methods to go, then right-click and choose Insert Code (or press the Alt-Insert keys). Select “Getter and Setter” from the Generate menu, and click the boxes next to the fields for which you want getter and setter methods generated. You can also click the class name (Employee) to select all fields. Click Generate to insert the code.

   The built-in automated syntax checker for NetBeans should have provided you with hints when you have syntax errors or code errors. Save your class.

7. Write code in the `EmployeeTest` main class to test your `Employee` class.
    a. Add an import statement to your class for the `Employee` object:

    ```
    import com.example.domain.Employee;
    ```

    b. In the `main` method of `EmployeeTest`, create an instance of your `Employee` class, like this:

    ```
    Employee emp = new Employee();
    ```

    c. Using the employee object instance, add data to the object using the setter methods. For example:

    ```
    emp.setEmpId(101);
    emp.setName("Jane Smith");
    emp.setSsn ("012-34-5678");
    emp.setSalary(120_345.27);
    ```

    Note that after you type the "`emp.`", Netbeans provides you with suggested field names (in green) and method names (in black) that can be accessed via the `emp` reference you typed. You can use this feature to cut down on typing. After typing the dot following `emp`, use the arrow keys or the mouse to select the appropriate method from the list. To narrow the list down, continue typing some of the first letters of the method name. For example, typing `set` will limit the list to the method names that begin with `set`. Double-click the method to choose it.

    d. In the body of the main method, use the `System.out.println` method to write messages to the console output.

    ```
    System.out.println ("Employee id:        " + emp.getEmpId());
    System.out.println ("Employee name:      " + emp.getName());
    System.out.println ("Employee Soc Sec #: " + emp.getSsn());
    System.out.println ("Employee salary:    " + emp.getSalary());
    ```

    The `System` class is in the `java.lang` package, which is why you do not have to import it (by default, you always get `java.lang`). You will learn more about how this multiple dot notation works, but for now understand that this method takes a string argument and writes that string to the console output.

    e. Save the `EmployeeTest` class.
8. Examine the Project Properties.
    a. Right-click the project and select Properties.
    b. Expand Build, if necessary, and select Compiling. The option at the top, “Compile on Save,” is selected by default. This means that as soon as you saved the Employee and `EmployeeTest` classes, they were compiled.
    c. Select Run. You will see that the Main Class is `com.example.EmployeeTest`. This is the class the Java interpreter will execute. The next field is Arguments, which is used for passing arguments to the main method. You will use arguments in a future lesson.
    d. Click Cancel to close the Project Properties.
9. Run the `EmployeePractice` project.

a. To run your `EmployeePractice` project, right-click the project and select Run, or click the Run Main Project icon (the green triangle), or press F6.

b. If your classes have no errors, your should see the following output in the Output window:

```
run:
Employee id:         101
Employee name:       Jane Smith
Employee Soc Sec #: 012-34-5678
Employee salary:    120345.27
BUILD SUCCESSFUL (total time: 1 second)
```

10. (Optional) Add some additional employee instances to your test class.