Lab Exercise: Creating Classes

INTERTECH

Instructors Who Consult. | **Consultants** Who Teach.

About Intertech

Thank you for choosing Intertech for your training. The next page of this document will get you started with your lab if you'd like to skip ahead. Below is a brief overview of Intertech as well as a promo code for you for future live Intertech trainings.

Intertech (www.intertech.com) is the largest combined software developer **training** and **consulting** firm in Minnesota. Our unique blend of training, consulting, and mentoring has empowered technology teams in small businesses, Fortune 500 corporations and government agencies since 1991.

Our training organization offers live in-classroom and online deliveries, private on-site deliveries, and on-demand options. We cover a broad spectrum of .NET, Java, Agile/Scrum, Web Development, and Mobile technologies. See more information on our training and search for courses by **clicking here**.

We appreciate you choosing Intertech!

Lab Exercise

Creating Classes

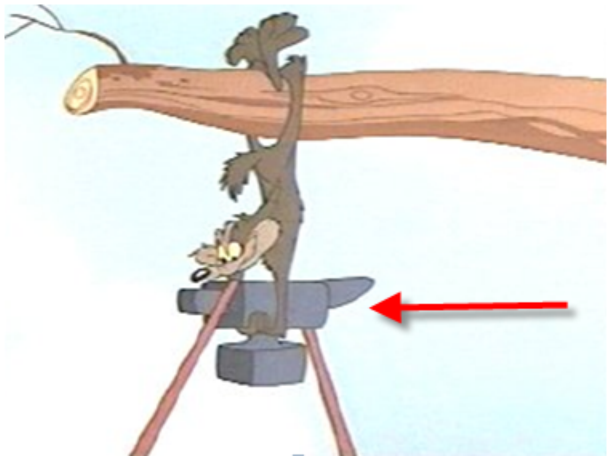
Java classes are the fundamental building blocks in Java applications. In this lab, you create a Java class that is used to create many Java objects. The objects are used to hold data and to invoke actions in the form of methods.

Specifically, in this lab you will:

- Create a reusable object type (class) called MyDate
- Add attributes to the MyDate class
- Add multiple constructors to the MyDate class
- Add methods to the MyDate class
- Test your definition of the MyDate class with a previously coded test class
- Explore an initialization block in MyDate

Scenario

Congratulations! You have just been hired by the Acme Company, a worldwide conglomerate with a diverse product portfolio, from adding machines to X-ray machines (see http://en.wikipedia.org/wiki/Acme_Corporation). One of its best-known products is the ever-popular Acme anvil.



Over the remaining labs, as a member of Acme, you are going to help Acme build a product order system.

Particularly, in this lab you make your first Java class for this project. A class defines a type that is a template for objects. It is your job to create a utility class called MyDate. An application that will use and test your version of the MyDate type has already been created. The test code appears below.

```
public class TestMyDate{

  public static void main(String[] args){
        MyDate date1 = new MyDate(11,11,1918);
        MyDate date2 = new MyDate();
        date2.day = 11;
        date2.month = 11;
        date2.year = 1918;
        MyDate date3 = new MyDate();
        date3.setDate(4,21,1968);
        String str1 = date1.toString();
        String str2 = date2.toString();
```

```
            String str3 = date3.toString();
            System.out.println(str1);
            System.out.println(str2);
            System.out.println(str3);
    }
}
```

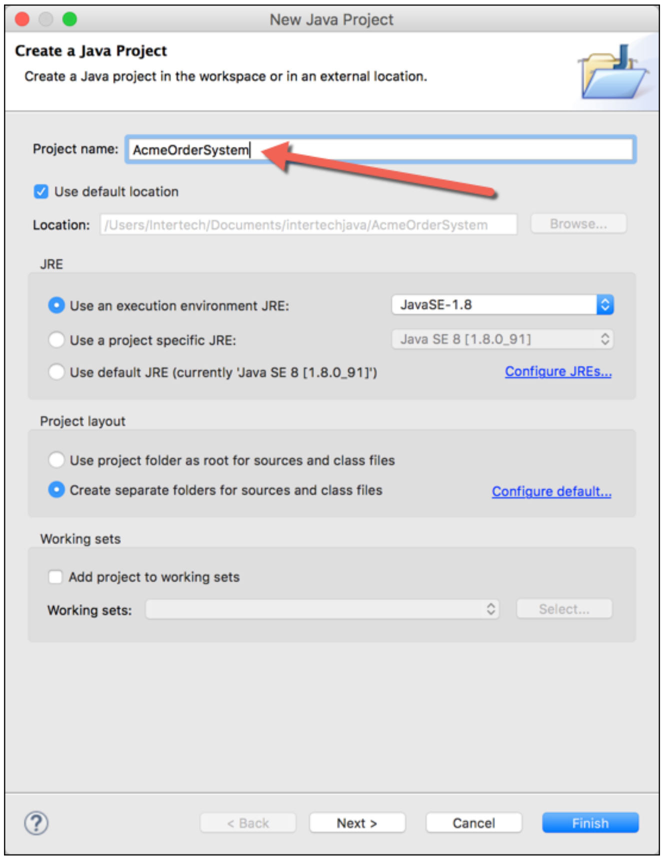
Step 1: Create a MyDate class

In this step, you create a MyDate type. The Date type already exists in Java (see java.util.Date and java.util.Calendar). However, you create another much simpler date type called MyDate to demonstrate object-orientated programming (OOP) concepts. You will modify MyDate later to demonstrate more advanced OOP concepts.

1.1 Make a "Java Project" to store your new classes and code.

1.1.1 Select *File > New > Java Project …*

1.1.2 In the window that appears, enter AcmeOrderSystem as the project name and click the *Finish* button.



New Java Project

Create a Java Project

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

Project name: AcmeOrderSystem

[✓] Use default location

Location: /Users/intertech/Documents/intertechjava/AcmeOrderSystem

JRE

[●] Use an execution environment JRE: JavaSE-1.8

[ ] Use a project specific JRE: Java SE 8 [1.8.0_91]

[ ] Use default JRE (currently 'Java SE 8 [1.8.0_91]')

Project layout

[ ] Use project folder as root for sources and class files

[●] Create separate folders for sources and class files

Working sets

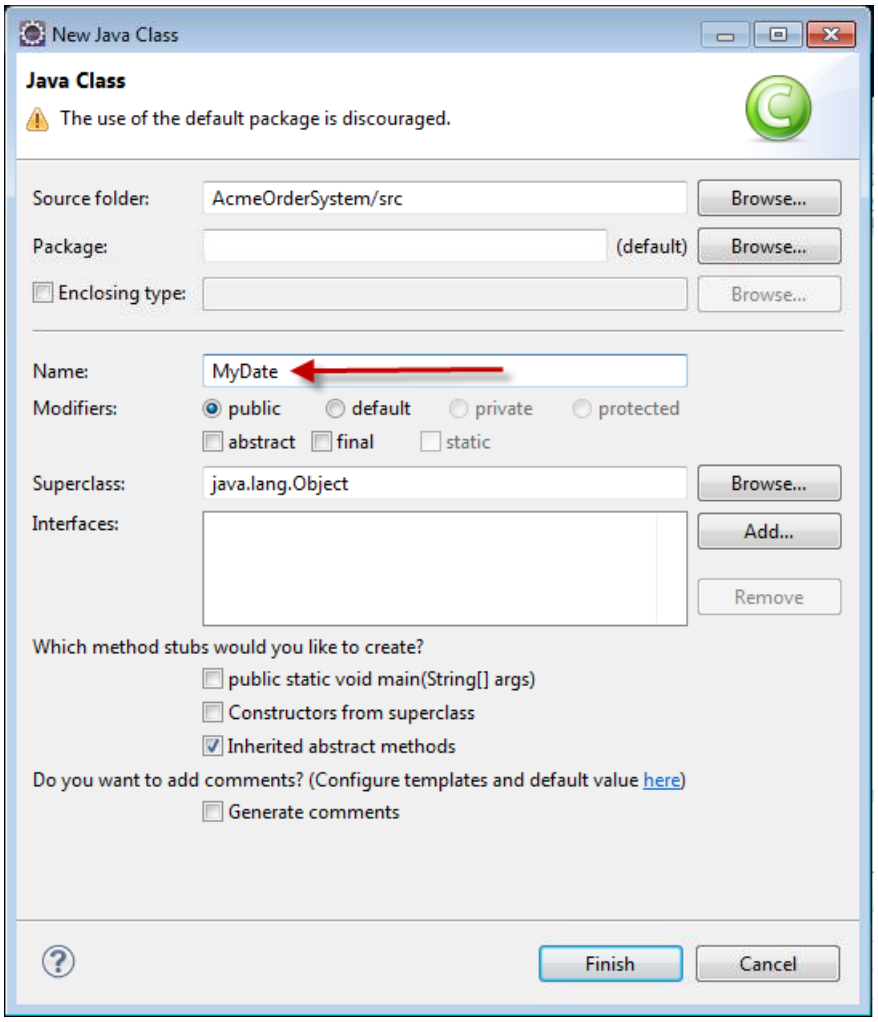
[ ] Add project to working sets

Working sets:

1.2 Create a class called MyDate.

1.2.1 Right-click on the AcmeOrderSystem project, and select *New > Class*.

1.2.2 In the dialog window that displays, enter MyDate as the name of the class, and press the *Finish* button.


New Java Class

Java Class

The use of the default package is discouraged.

Source folder: AcmeOrderSystem/src

Package: (default)

Enclosing type: [ ]

Name: MyDate

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

Note*:* Don't add a main method to MyDate, as it is not the application's starting point.

This should create a MyDate.java file in the src folder of the project (under the default package) and open a MyDate.java editor for you to add code.

1.3 Add day, month, and year attributes to MyDate. These three attributes (member variables) should be of type int.

```
int day;
int year;
int month;
```

Step 2: Create two constructors

Constructors allow you to initialize an object when it is created (instantiated). You are required to add two constructors to the MyDate class.

2.1 Add a no-argument (no-args) constructor. The no-args constructor enables you to make a MyDate using default values.

```
public MyDate(){}
```

2.2 Add a constructor that has three integer arguments (parameters). This will enable the user to create a MyDate using a constructor like this: new MyDate(2, 6, 2004)

```
public MyDate(int m, int d, int y){
 //...use the parameters of m, d and y to set the three
attributes
}
```

Note: As a reminder, constructors look like a method with no return type.

2.3 Save your file, and fix any compiler errors before moving to the next step.

Step 3: Add some methods to MyDate

The MyDate class you have created needs two methods. Recall that a method represents an action or something an object can do. Users of MyDate should be able to see the dates represented by the object. Therefore, you must provide a method, called toString( ), that turns the MyDate object into a String. Users of MyDate should also be able to reset the date represented by MyDate by feeding in three parameters, so you must create a setDate( ) method.

3.1 In the MyDate.java editor, add a toString( ) method. The code below shows a partially completed method. The method should return a String that contains the values of day, month, and year.

```
public String toString(){
   //TODO return a string with month/day/year like "01/20/1964"
   return "";
}
```

Note: Java Strings can be concatenated using the + symbol like "Cat" + "Dog", so the month and day can be concatenated as month + "/" + day.

3.2 Create a setDate(m, d, y) method. The setDate( ) method enables the user to call one method to set day, month, and year of a MyDate object. An empty version of this method appears below.

```
public void setDate(int m, int d, int y){
  //TODO set the MyDate attributes with m, d, and y values here!
}
```

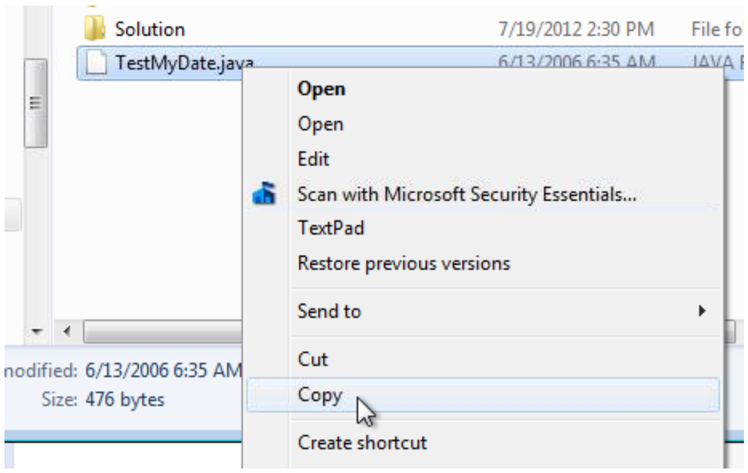
Step 4: Run TestMyDate

A program is provided to test your new class. It appeared at the beginning of this lab. You don't need to write this program, as it is available in the lab folder.

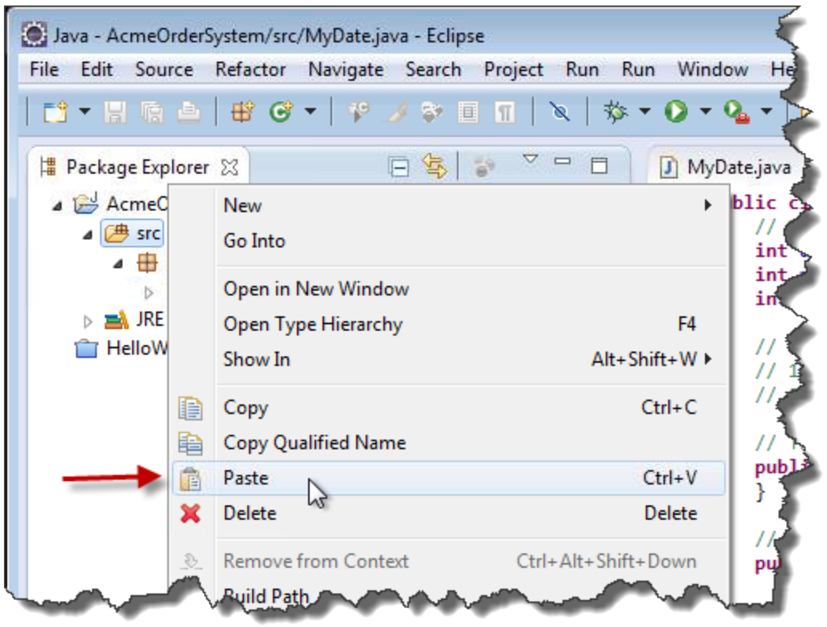
4.1 Import TestMyDate.java from the IntertechLearnJava zip.

4.1.1 Find the file where you downloaded it,

4.1.2 Right-click on the file, and request a copy of the file.



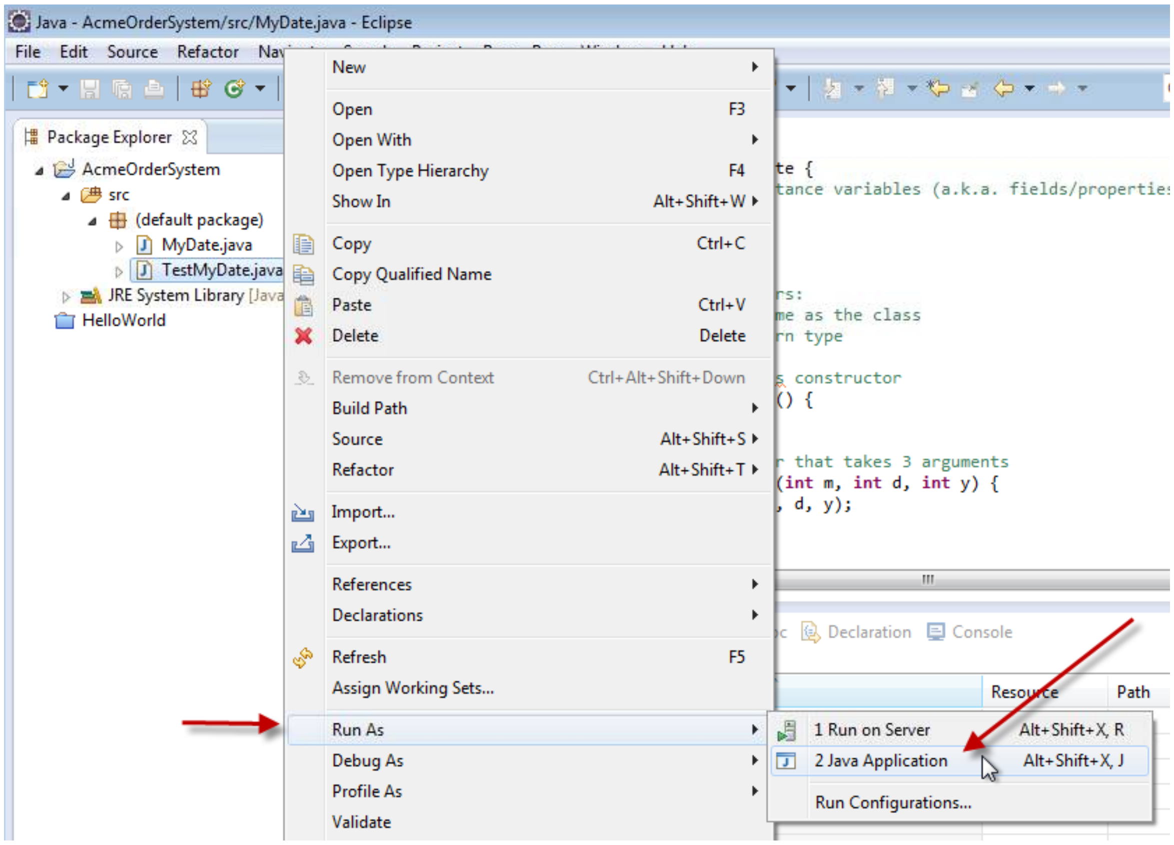
4.1.3 Back in the Eclipse IDE, right-click on the src folder in the AcmeOrderSystem project, and request to paste the copied file into the project.



4.2 Fix any errors in MyDate or the TestMyDate file. The most likely place for an error is in the MyDate type. If you didn't follow the directions exactly, the test program can fail to compile.

4.3 Run the TestMyDate class.

4.3.1 Right-click on TestMyDate and select *Run As > Java Application*.



4.3.2 When you run the test, the output in the Console view should look like the following. If it looks different, go back and try to fix it. Ask for help if you get stuck.

```
11/11/1918
11/11/1918
4/21/1968
```

Note: Throughout class, the lab book will show you the output of running your Java application as shown in the block above. In most cases, your output should be identical to that shown in the lab book. However, there may be cases, especially later in class, where your output will vary slightly. This may be due to a number or circumstances such as parameters/data used or how many of the bonus labs you choose to complete.

Lab Solution

MyDate.java

```
public class MyDate{
   // Member/instance variables (a.k.a.
fields/properties/attributes)
    int day;
    int month;
    int year;

   // Constructors:
   //     1. Same name as the class
   //     2. No return type

   //The no-args constructor
   public MyDate(){
   }

   //Constructor that takes 3 arguments
   public MyDate(int m, int d, int y){
         setDate(m, d, y);
   }

   //Methods
   public String toString(){
         return month + "/" + day + "/" + year;
   }
   public void setDate(int m, int d, int y){
         day            = d;
         year           = y;
         month          = m;
   }
}
```

TestMyDate.java

```
public class TestMyDate{

   public static void main(String[] args){
         MyDate date1 = new MyDate(11,11,1918);

         MyDate date2 = new MyDate();
         date2.day = 11;
         date2.month = 11;
         date2.year = 1918;

         MyDate date3 = new MyDate();
         date3.setDate(4,21,1968);

         String str1 = date1.toString();
         String str2 = date2.toString();
         String str3 = date3.toString();

         System.out.println(str1);
         System.out.println(str2);
         System.out.println(str3);
   }
}
```

Bonus Lab

Step 5: Add an initialization block

Add an initialization block to MyDate that defaults the day, month, and year to January 1, 2000. After completing the initialization block, add code similar to that shown below to test your initialization block in the main method of TestMyDate.java.

```
MyDate date4 = new MyDate();
String str4 = date4.toString();
System.out.println(str4);
```

Give Intertech a call at **1.800.866.9884** or visit Intertech's website.