A2 Installation and Setup

Overview

A2 consists of database applications that track customer orders, shipping, and inventory. The system is missing some functionality such as the ability to modify and delete orders and inventory entries in the database as well as other similar features. This has been done to reduce the code base and complexity of the system, reduce system scope, leaving only the essential functionality, enabling you to focus on the architectural design issues. The key tradeoffs are between realism and minimizing complexity of installation and scope of modifications. Despite the fact that all has been done to reduce complexity, setup your system up for A2 can be tricky. Please read the steps below before attempting installation and setup. Note that this installation guide assumes you are using the Windows 7 or better operating system.

This assignment makes use of the following technologies:

- ○ MySQL Community Server (5.6.15) – Open source database that stores all of the order and inventory data.
- ○ NetBeans IDE 7.4 – Allows you to use Java to create sophisticated applications including graphical user interfaces (GUI), Beans, applets, and so forth.
- ○ J Connector – This is the Java Database Connectivity standard that allows Java programmers to access data in databases from within a java application.

You will have to download each of these products from the following URLs:

- ○ NetBeans IDE – The NetBeans IDE 6.8 installation archive can be found at: http://netbeans.org/downloads/index.html For A2, all you need is the Java bundle.
- ○ MySQL Community Server – http://dev.mysql.com/downloads/installer/5.6.html You can download the MySQL community server installer here. Note that the assignment was designed to work with the win32 version – it has not been tested with the 64 bit option.
- ○ MySQL J Connector – http://dev.mysql.com/downloads/connector/j/ Again the assignment was developed with the 32 bit connector. This is a .msi file. You should place this file were you plan to do your development. In Netbeans you can specify the location of the connector under the libraries folder for each project. You can see this for yourself in the example code for OrderApp, ShippingApp, and AddInventory projects.

The A2 Archive, Database Installation

The A2 Archive

Once you unzip the A2-2014 archive, you will see a directory named *A2 Applications* that contains the following directories:

- AddInventory – This directory contains the files for the application that is used to add items to the inventory database. Data is stored in one of three tables: seeds, shrubs, trees, depending upon the type of product.

- OrderApp – This directory contains the files for the application that is used to submit orders that is stored in the orderinfo database, in the orders table. Each order also has a table associated with it that contains an itemized list of the products ordered.

- ShippingApp – This directory contains the files for the application that is used to mark orders as shipped. This application lists orders and changes the shipped status to true in the orderinfo database, orders table.

- MySQLData – This directory contains the database table configuration and data that you will import once you install MySQL.

MySQL

To install MySQL, double click on the mysql-installer-community-5.6.15.0.msi file. This will start the installation process for MySQL database. You should use a typical installation and all defaults. This include using the default port number 3306 (this is required if you intend to use the database remotely) and leaving the root password blank (you can change it later if you like).

Once the installation is complete, you should add the path to the MySQL executables to your system's PATH file. The executables are in the bin directory of the MySQL directory (mysql-5.6.15-win32). So generally speaking, you will add the following to your PATH:

    drive:\<path to mysql-5.6.15-win32>\bin

For example assuming that you have installed MySQL in the root directory of your C drive you would add the following to your PATH:

    c:\mysql-5.6.15-win32\bin

Once you have added this to your PATH variable, you can check out the installation by starting the MySQL data base as follows:

    c:\start mysqld –u root

This will start the MySQL server in the background as a root user. Once you have started the server, you can run the command line MySQL interface as follows:

    c:\mysql --user=root

Note that in the above command, there are two dashes before user. If everything installed properly, you should see the following prompt:

    mysql>

Now you can access the server as an administrator. To run the provided applications, you will need to create the databases and import the data. First we will create the EEPs and LeafTech databases as follows…

    mysql> *create database inventory;*

    mysql> *create database orderinfo;*

    mysql> *create database leaftech;*

You can list the databases you just created by typing *show databases;* at the MySQL command line. You should see the following:

These databases are now created so when you access your instance of MySQL you can access them. The blue lines at the left show the three databases that you just created. Ignore the other databases as they are used by MySQL. Note that thes databases we just created are totally empty have no table structures defined and obviously are not populated with data. Next we will set up the table structures and import data into these databases.

At this point exit from mysql by typing "exit" at the mysql prompt. At the command window prompt (not the mysql command line window), type the following:

mysql -u root –p[root pwd] -D inventory < inventory.sql

mysql -u root –p[root pwd] -D orderinfo < orderinfo.sql

mysql -u root –p[root pwd] -D leaftech < leaftech.sql

Note that the files "inventory.sql" and "orderinfo.sql" are located in the *A2-2014* archive in the directory *A2-2014 Source/EEPs/MySQLData.* The "leaftech.sql" file is located in the directory *A2-2014 Source/LeafTech/MySQLData* You can use a full path name to the .sql files or you can go to the MySQLData directory and import the data from there using the commands above. The two .sql files listed above are not databases, but contain the meta data to create the database tables and populate the databases with data. This is all done in the two steps listed above.

Restart the command line MySQL interface. At this point you can list the databases you just created by typing *show databases;* at the MySQL command line.

You can select a database by typing *use <DB>;* where <DB> is the name of one of the databases you created: *orderinfo, inventory, or leaftech.* Once you have selected one of these databases, you can see the tables by typing *show tables;* at the MySQL command line.

You can show the structure of a table by typing *describe <TB>;* where <TB> is the name of one of the tables in the database you selected. In the example below you can see where the inventory database was selected (line 1), the tables in the inventory database were listed (line 2), and the seeds table structure is shown (line3).

```
                    Administrator: Command Prompt - mysql --user=root
mysql> use inventory;
Database changed
mysql> show tables;
+---------------------+
| Tables_in_inventory |
+---------------------+
| seeds               |
| shrubs              |
| trees               |
+---------------------+
3 rows in set (0.00 sec)

mysql> describe seeds
    -> ;
+--------------+-------------+------+-----+---------+-------+
| Field        | Type        | Null | Key | Default | Extra |
+--------------+-------------+------+-----+---------+-------+
| product_code | varchar(10) | YES  |     | NULL    |       |
| description  | varchar(80) | YES  |     | NULL    |       |
| quantity     | int(11)     | YES  |     | NULL    |       |
| price        | float(10,2) | YES  |     | NULL    |       |
+--------------+-------------+------+-----+---------+-------+
4 rows in set (0.00 sec)

mysql>
```

1
2
3

In order to run the applications, you will need to create an account for the entire database called *remote*, with its password set to *remote_pass*. You can change this if you like, but this is currently hard-coded into the demo applications provided. This is used by the applications (local or remote) to access the data in the orders, inventory, and leaftech databases. The syntax for creating this account is the following from the MySQL command line:

mysql> create user "remote"@"%" identified by "remote_pass";

Now that you have created the account you must set the permissions as follows:

*mysql> grant all on *.* to "remote"@"%" identified by "remote_pass";*

The "%" above allows any IP to access the database. You can replace this with a specific IP address if you like. If you plan to only use the database locally you can replace it with "localhost." Note that if you choose to access the database remotely, you will have to open up port 3306 on the computer hosting the database.

Installing the J Connector

Next you will need to install the J Connector. This allows you to access MySQL databases directly from the Java language. To install the J Connector double-click on the mysql-connector-java-gpl-5.1.28.msi file – this will extract the connector. Note where you install the: mysql-connector-java-5.1.28-bin.jar file. This is the connector and you will have to tell Netbeans were to find this file. Once installed, copy the J Connector to the java runtime

environment's lib and lib\ext folders. For example, assume you are running Java 7, then you would copy the J Connector jar file to *c:\program files\java\jre7\lib* and also to *c:\program files\java\jre7\lib/ext.* This will allow you to export and run the applications outside of Netbeans should you need to.

Installing NetBeans IDE

To install NetBeans IDE you will need to double click on the *netbeans-7.4-javaee-windows.exe* file and it will start the installation process. Simply follow the instructions and the application framework will install automatically. You will have to play with NetBeansIDE a bit to learn how it works. The good news is that all of the applications in A2 are simple Java applications. Once NetBeans IDE is installed you can start it up and you should see the main screen:

Projects Window

Work Space

Messages/Output

Running the Applications

To run any of the applications, you will have to open the application project. Assume you want to run the *OrderApp* application. Click on: File -> Open Project, and then navigate to the *A2 Applications* folder. Select the *OrderApp* project as shown below:

These are projects… select the OrderApp project.
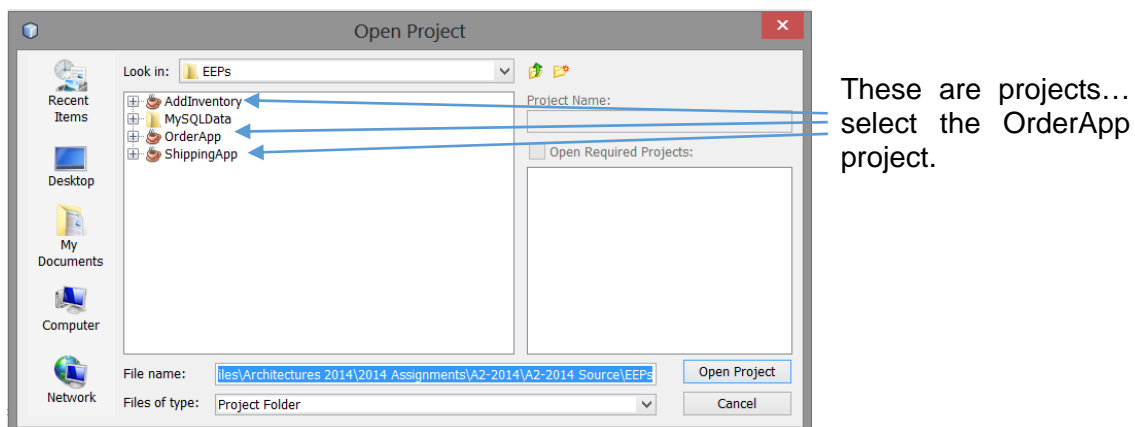
Once selected, the project will appear in the projects window of Netbeans. You can expand the project and project folders by clicking on them as shown below:
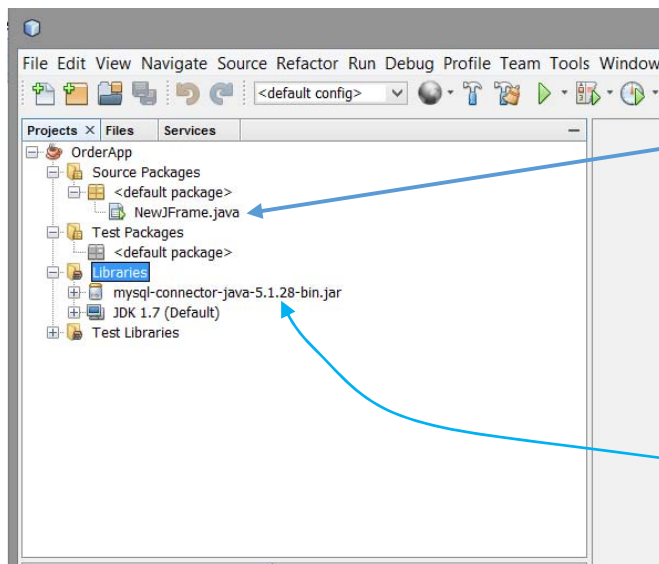
Note that in all of the applications, the source code will be located in one source file: *NewJFrame.Java*

There are many more sophisticated ways to set this up, this was done for simplicity.

The libraries directory contains all the libraries required to build the application, note that the J Connector jar file is shown here.

To add the J Connector jar file to the libraries, left click on the libraries and you can choose to add a Project, Library, or JAR/Folder… select JAR/Folder. A window will pop-up allowing you to select the mysql-connector-java-5-1-28-bin.jar file.

You can run the *OrderApp* inside NetBeans which is useful for coding and debugging. Select the source code under the *Source Packages* directory. In the case above you would select *NewJFrame.java*. This will open the source code as shown below. At this point you can select the "Run" option from the main tool bar and the application will execute.
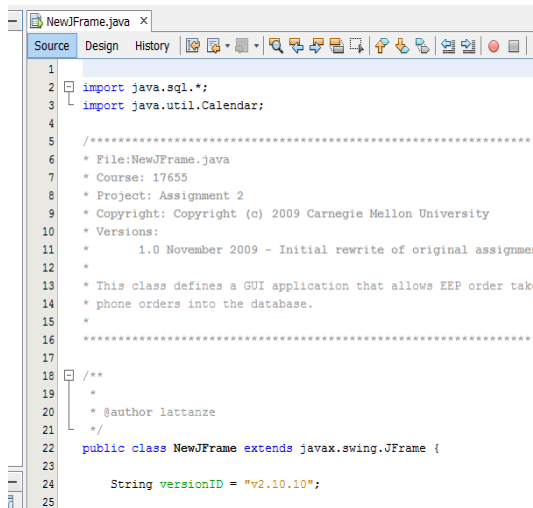
In addition to running the code, you can switch between the "Source" or "Design" views. The Source view displays the code and lets you edit and develop code. If you edit code you can optionally choose to "Build" the code under the run menu, or by running the program, NetBeans will automatically rebuild the application. Errors are displayed in the output display at the center bottom of the GUI.

The Design view shows you the GUI layout and lets you edit and design the layout. The "Palette" panel to the right lets you drag and drop widgets onto the GUI and edit their properties. This view is shown below.

Here you can see the GUI design view here…   The palette pane is here…
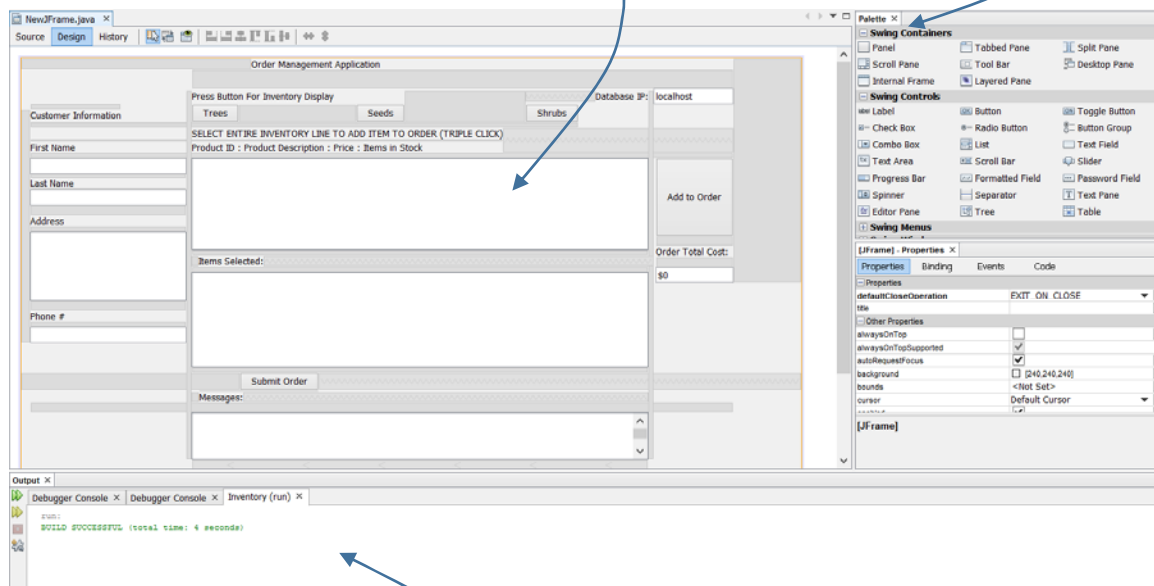
Any errors will be written here….

To write code for GUI widget actions, select the widget and right click. This will bring up a window of options. Select Events. This will bring up a list of events that the widget can be programmed to respond to. Selecting one of these will automatically insert an abstract method for that handler in your code. All you have to do is insert your code. Cool eh?! The two pop-up menus are shown below.
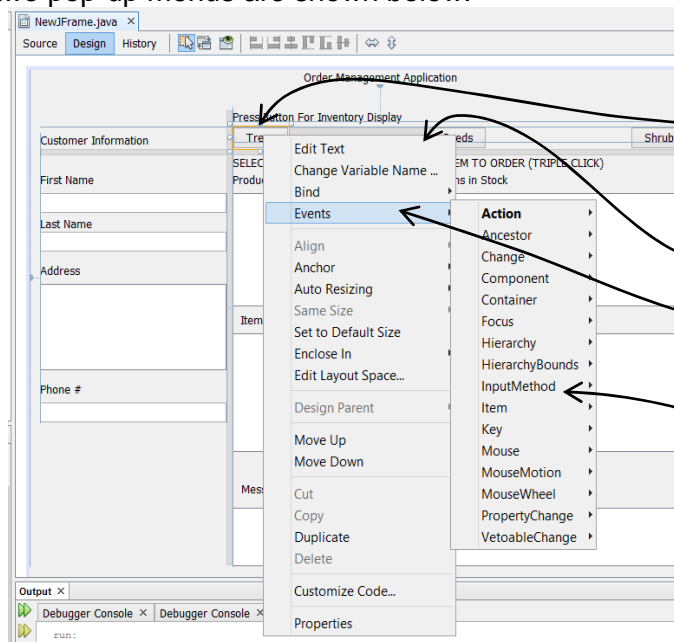
Here I have selected a button whose text property is "trees." This correlates to jButton1.

When I right click I get a pop-up presenting me with a list of options.

Now I select "Events".

This gives me a bunch of events I can select from.

Again, when you select one of these events, Netbeans automatically inserts an abstract class for the handler into the source code.

If you switch to the source code view you will see the abstract class that NetBeans has inserted for you. I will look like this…

```
58
59            pack();
60        }// </editor-fold>
61
62  ⊟     private void jButton1ActionPerformed(java.awt.event.ActionEvent evt) {
63            // TODO add your handling code here:
64        }
65
66  ⊟     /**
67         * @param args the command line arguments
68         */
69  ⊟     public static void main(String args[]) {
70            /* Set the Nimbus look and feel */
71  ⊞        Look and feel setting code (optional)
```

Cool eh? Whenever the button (jButton1 in this case) is pushed it will execut whatever code you put here. All the other events and widgets work exactly the same way. NetBeans will autogenerate all the supporting code required to create widgets and handle events from them. If this is still confusing or you have no experience with event oriented programming and GUI builders, then you should see the section below: **More NetBeans Hints**. This will walk you through creating a simple application.

It is possible to run applications built using NetBeans without the NetBeans environment. In order to run the *OrderApp* without NetBeans, open a command window and navigate to the distribution directory is. There you will find the OrderApp jar file. It is located inA2-2014\A2-2014 Source\EEPs\OrderApp\dist. Type the following at a command line or setup a shortcut as follows: java –jar OrderApp.jar. At this point you should see the OrderApp GUI as shown below:

This GUI allows order takers (see A2 for details) to submit orders to the system for shipping. To use this form, first you must specify the location of the database. If the database is on your local system, then leave the default *localhost* in the database IP. Note that while this system mimics an enterprise wide distributed client-server system, however to make the assignment easier, it can operate on one system. If you like, you can host the

database on another system – like one of your teammate's laptops. If you choose to do this, you should put the IP address of the system that is hosting the databases.

More NetBeans Hints

The NetBeans IDE framework facilitates the construction of a variety of java applications including a variety of beans, GUIs, client-server, tiered applications, and much more. The applications created for A2 are standalone GUIs that access a local or remote MySQL database. A2 is obviously a simple client-server system; however you are not bound by this style, nor are you required to deviate from it as you work through your modifications. You have a great deal of freedom in selecting how you will design and implement your changes. Whatever you decide, NetBeans IDE should support any design choices you make for A2 from an implementation stand point.

Creating a Simple Application

In order to create a simple GUI application, you will have to create a new project. Under file, select create new project. This will pop-up the following menu:
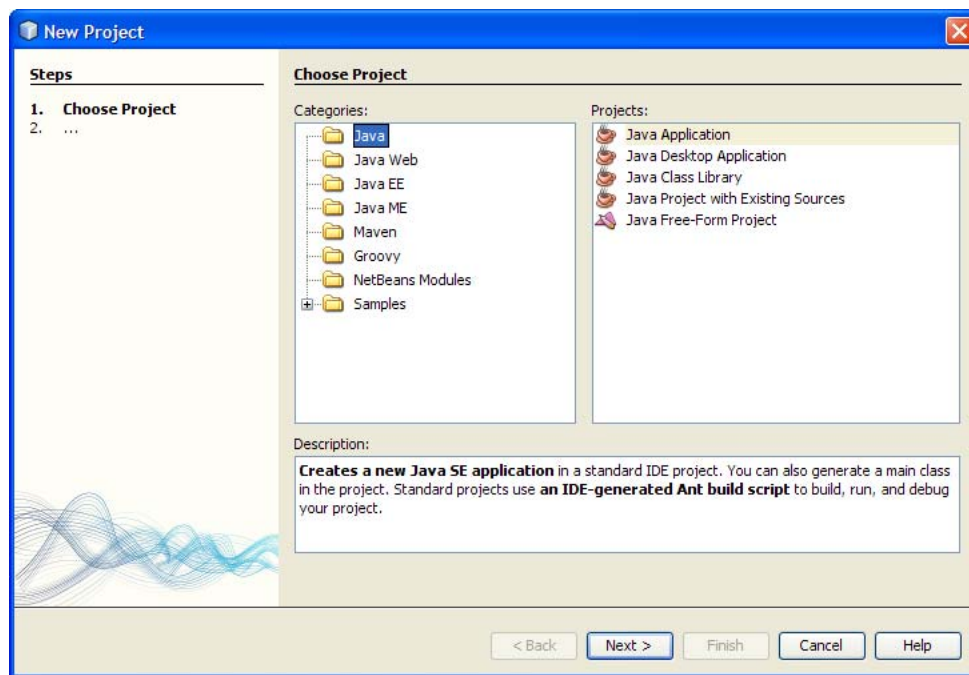

For the purpose of our demo, select Java, then Java Application, then click on *Next*. Once you click on next, you will get the following pop-up:
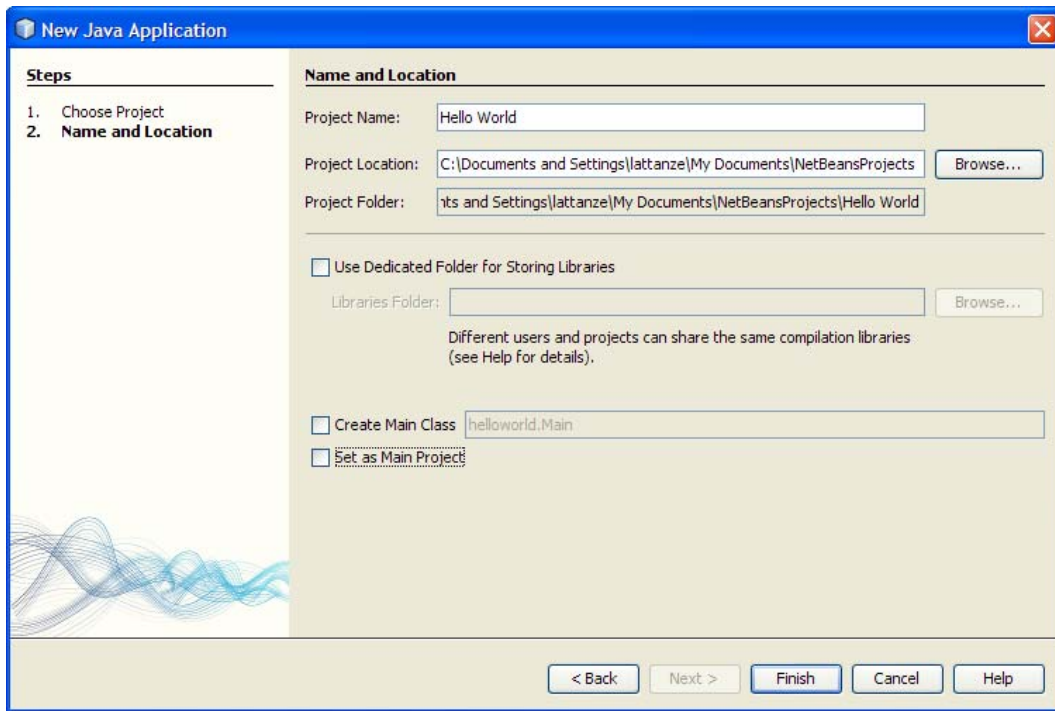
Select a project name. I have selected *Hello World* here. The default project location is in the NetBeansProjects directory, as is the project folder. You should deselect the *Create Main Class* and *Set as Main Project* options. When you have done this, click *Finish*. At this point, your project is created and you will see the Hello World project with the folders: *Source Packages*, *Test Packages*, *Librarie*s, and *Test Libraries*. Next we will add a jFrame with a couple of buttons and a text field. To create the frame, double click *Source Packages*, then select *<default packages>*; then right click; select *new*; select *jFrame Form* as shown below:
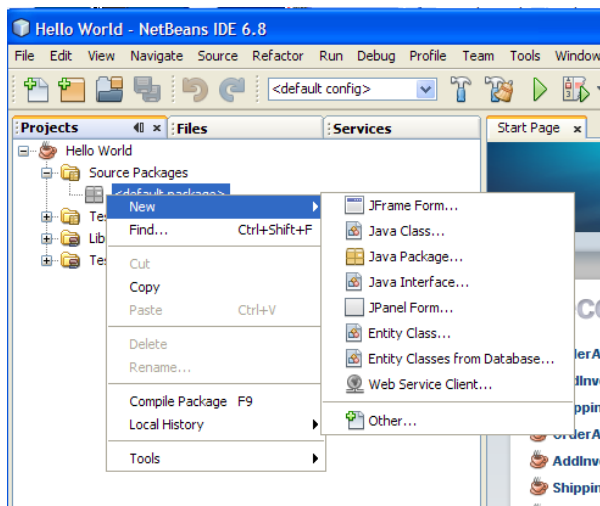

In the next pop-up window, you can provide a new class name or use the default. The remainder can be left with their default values. When complete, click on finish. Now it gets fun. At this point you can drag widgets from the palette located on the right side panel and drag them onto the frame. In the example below, I have placed two buttons and a text field. You can change the properties of the widgets using the properties panel after selecting the widget whose properties you intend to change. This is shown below:
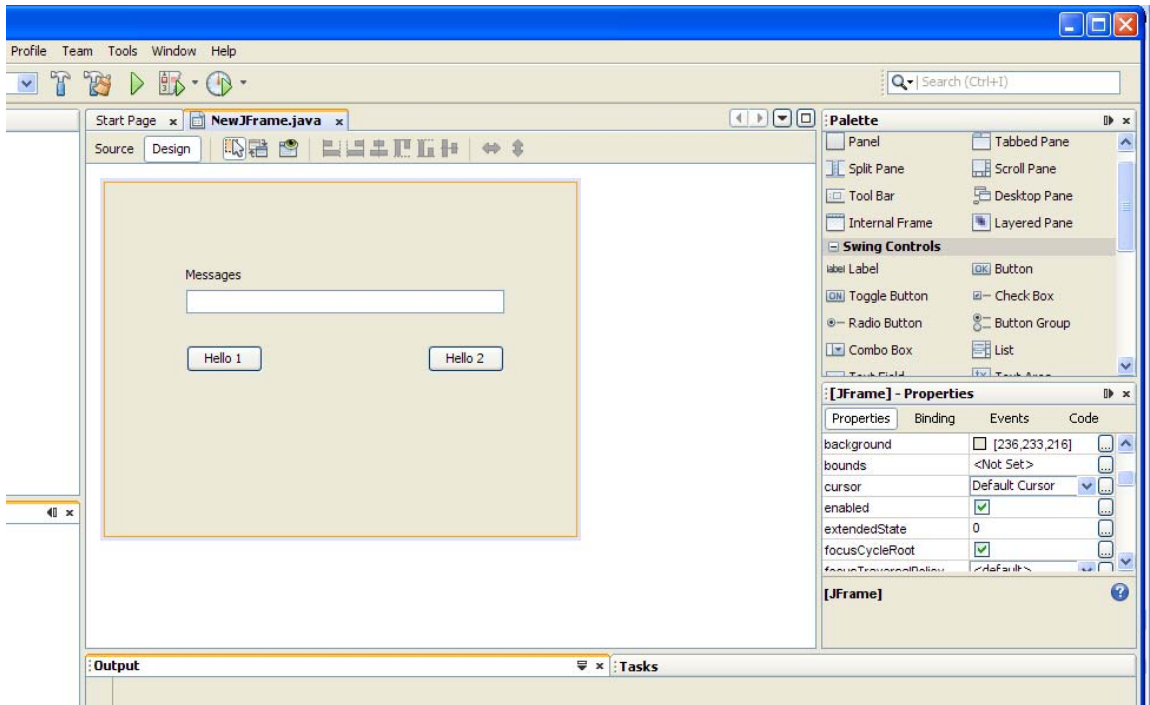
In this example there is a jLabel (jLabel1), a jTextField (jTextField1), and two buttons (jButton1 and jButton2). The NetBeansIDE will compile your application and let you run it without having to use a command line interface or create a shortcut. At this time, our application will only display the panel, but it's a good time to play with the compile and run options. In order to compile the program, you click on the hammer icon on the menu bar. The green right pointing arrow will execute your program. These are show below:

This is the compile tool…

This is the run tool…
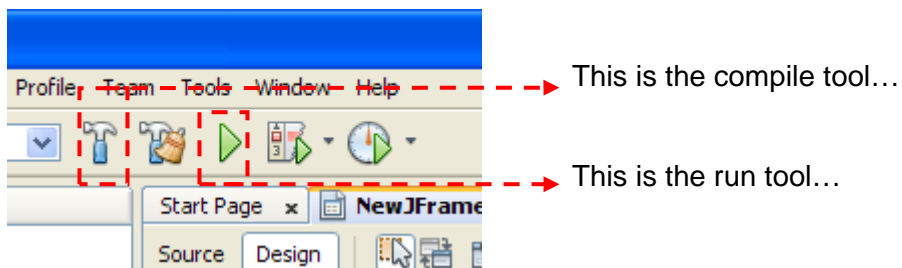
At this point if you compile and run the application, you will get a simple GUI that really doesn't do anything as shown below. You will also get a message the first time you compile indicating that you do not have a main defined, select the default. You won't have to do this again.
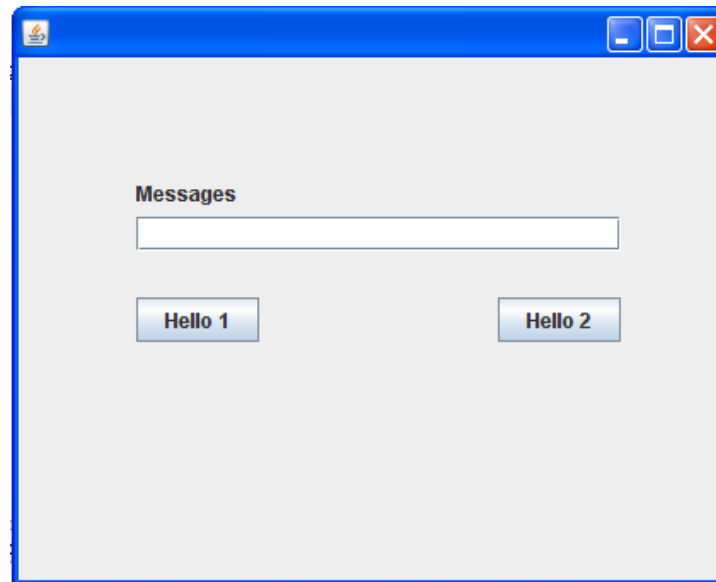
Now we can add some code to make the application do something. GUIs activate code when events happen within the frame. Just about anything can cast an event, but for our purposes lets write some code that will be activated when the buttons are pressed. On the work space, double click the Hello 1 button, this will switch from design view to code view. Note there is a tab that will also allow you to switch between these views as well. At this point you should see the following:
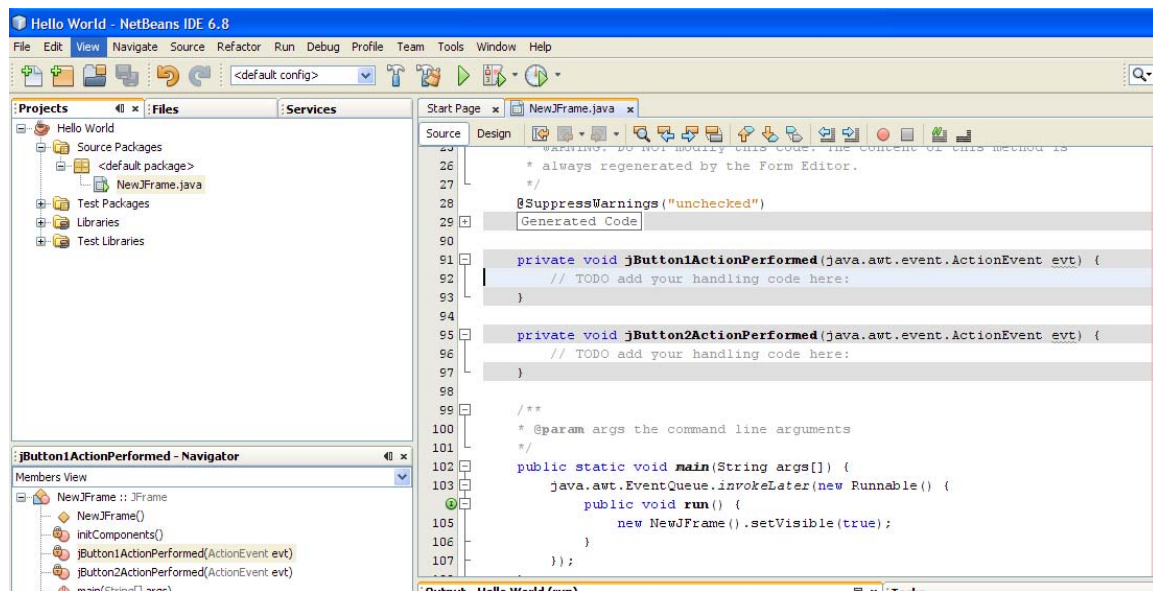
The jButton1ActionPerformed() method is added by the NetBeansIDE when you placed the button on the frame. Now let's add the following code to the method:

```
91      private void jButton1ActionPerformed(java.awt.event.ActionEvent evt) {
92          // Button one handler
93
94          jTextField1.setText("Hello world from button 1");
95
96      }
```

This is a very simple method that will print "Hello world from button 1" when button 1 is pushed. While this example is really simple, this could be as complex as any other computer program. Let's add some code for button 2. Switch back to the design view, then double click on button 2. This will take you to the handler for button 2. Add the following code:

```
98      private void jButton2ActionPerformed(java.awt.event.ActionEvent evt) {
99          // Button one handler
100
101          jTextField1.setText("Hello world from button 2");
102
103      }
```

Now compile and run the program. This time when you hit button one or button two, you should get messages displayed in the textfield as show here to the right. You may declare any variables, link various libraries, or do anything in NetBeans IDE that you would in any other Java program. NetBeans makes it really easy to create some complex applications. However, you just learned all you will need to know to complete A2.
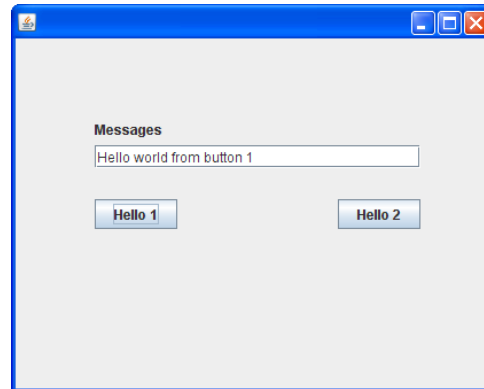
It's important to note that you can move this application around to any computer with JRE by moving the project folder. The project folder will be "Hello World" and will have the directories: *build, dist, nbproject, src*, and *test*. The executable jar file for the project is located in the dist directory. In the dist directory there will be a jar file named *Hello World.jar*. To run the program, type java –jar Hello World.jar at a command line or in a shortcut.