**ICOM 4015-Advanced Programming**

Spring 2014

Instructor: Dr. Amir H. Chinaei

TAs: Hector Franqui, Jose Garcia, and Antonio Tapia

Reference: **Big Java**

By Hortsmann, Ed 4

**Lab 5**

**Introduction to Jar Files, RESTFUL Methods and Web Sockets.**

Department

of

Electrical and Computer Engineering

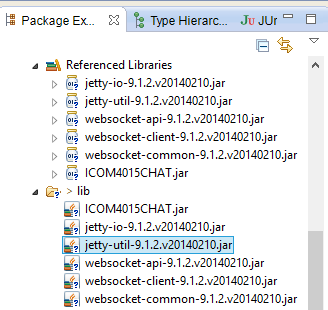
University of Puerto Rico at Mayagüez

**Before laboratory:**

1. Review Java Jar Files
2. Revisit JPanels
3. Revisit Action Listeners
4. Review JComboBox,JTextField, JTextArea
5. Print (at least) the Evaluation sheet in the last page.
6. **Login to computer (1 minute)**
7. **Using JAR Files(20 minutes)**
   1. Today we will be building a chat client and we will start by download all the jar files from :

<https://ece.uprm.edu/~ahchinaei/courses/2014jan/icom4015/lab5>

* 1. Create a new Java project by: *Ctrl+n > Java Project* . (Feel free to use any name for your project.)
  2. Now, you will create a folder ( named *lib* ) inside the *src* folder of your just-created project*.* Select the project first, then: *Ctrl+n >General> Folder* (make sure to use *lib* as the name of your folder)
  3. Drag/Drop the Jar files that you downloaded into the *lib* folder.
  4. You need now to refresh your project so that Eclipse can acknowledge the jar files: select the project > right mouse click > refresh . Or, just press F5.
  5. Right click on each jar file in the *lib* folder and: Build Path > Add To build path
  6. So, by now, you have added all the jar files to your build path and the package Explorer of your project should look as Figure 1.
  7. **Show your Package Explorer to the instructor (20 points)**



**Figure 1**

*You should have a paragraph or two here explaining what the downloaded jar files are for?*

A JAR(Java Archive) file may be used to store libraries. In this case we are adding these jar files because they contain web libraries and pre-made classes which we can utilize.

1. **Building a GUI (25 minutes)** 
   1. *Ambiguous: What do you mean by compile in the JFrame?*
   2. Let’s create a new class, inside the project we recently created, and call it *WelcomePanel*. On the class creation wizard have it extend the *SimplePanel* class.
   3. This will create an empty class with a method with a signature of “void init(){}” above it you will see the an annotation that reads “@Override” why is this ? … We will now begin building an interface for our soon to be chat client, add the following code to this class: *Ambiguous: where this class has to be created? Inside the above project? Or in a new project? What the annotation @override is for? Is this implementing an interface or what? This is going to be very confusing for students*

@Override // Ignore this for now

void init() {

JButton eButton = new JButton("Enter");

eButton.setActionCommand("enterRoom");

eButton.setName("enter");

String[] roomNames = getChatNames();

JComboBox<String> combo = new JComboBox<String>(roomNames);

combo.setName("roomDropDown");

combo.setActionCommand("checkRoom");

JTextField field = new JTextField("Username", 2);

field.setName("nameField");

JPanel buttonPane = new JPanel();

buttonPane.setLayout(new GridLayout(0,5));

buttonPane.add(new JLabel("Name:"));

buttonPane.add(field);

buttonPane.add(new JLabel("Room Number:"));

buttonPane.add(combo);

buttonPane.add(eButton);

buttonPane.setPreferredSize(new Dimension(630, 30));

JPanel mainPanel = new JPanel();

mainPanel.add(buttonPane,BorderLayout.CENTER);

mainPanel.setPreferredSize(new Dimension(630,400));

this.add(mainPanel,BorderLayout.CENTER);

JLabel status = new JLabel("Welcome");

status.setName("status");

status.setPreferredSize(new Dimension(630, 30));

this.add(status,BorderLayout.PAGE\_END);

this.setBorder(new EmptyBorder(20,20,20,20));

}

private String[] getChatNames() {

String knownNames[] = {

"b/",

"irc",

"Mofongo"

};

String roomNames[] = new String[20];

for(int i=1;i<=20;i++){

roomNames[i-1]=knownNames[(i-1)%knownNames.length]+((i-1)/knownNames.length);

}

return roomNames;

}

* 1. Now add the following main method to the *WelcomePanel class*: *Ambiguous: Where this main method has to be added to? To the same WelcomePanel Class or in a new class? What the comments are about?*

/\*\*

\* Javadoc stuff !

\* @param args

\*/

public static void main(String... args){

JFrame window = new JFrame();

window.setContentPane(new EnterPanel());

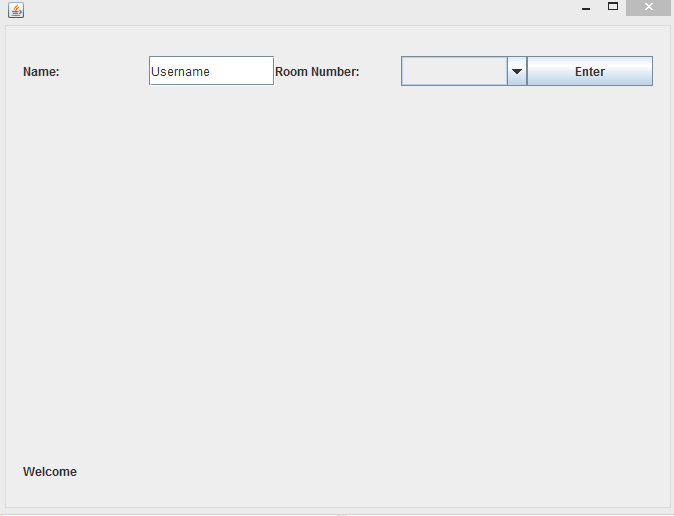
window.setVisible(true);

window.setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE);

window.setSize(680, 520);

}

* Run this class and you should see the window below:

****

**Figure 2**

* 1. Let’s create a new class, inside the project we recently created, and call it *ChatPanel*. On the class creation wizard have it extend the *SimplePanel* class.
  2. This will once again create an empty class with a method with a signature of “void init(){}”. Let’s add the following implementation to the chat client:

@Override

void init() {

/\*\*

\* Text Area

\*/

JTextArea chatArea = new JTextArea("Welcome");

chatArea.setName("chatArea");

chatArea.setPreferredSize(new Dimension(630,300));

chatArea.setEditable(false);

/\*\*

\* Button Bar

\*/

JButton sButton = new JButton("Send!");

sButton.setActionCommand("sendMessage");

sButton.setName("send");

sButton.setPreferredSize(new Dimension(50, 30));

JTextField messageField = new JTextField("Hey, sup?");

messageField.setName("messageField");

messageField.setPreferredSize(new Dimension(550, 30));

JPanel buttonPane = new JPanel();

buttonPane.setLayout(new GridLayout(0,2));

buttonPane.setPreferredSize(new Dimension(630, 30));

buttonPane.add(messageField);

buttonPane.add(sButton);

/\*\*

\* Main Pane

\*/

JPanel chatPane = new JPanel();

chatPane.add(chatArea,BorderLayout.CENTER);

chatPane.add(buttonPane,BorderLayout.PAGE\_END);

chatPane.setPreferredSize(new Dimension(630,350));

/\*\*

\* Status Label

\*/

JLabel status = new JLabel("Welcome");

status.setName("status");

status.setPreferredSize(new Dimension(630, 30));

/\*\*

\* Exit Button

\*/

JButton exitRoom = new JButton("Exit Room!");

exitRoom.setName("exitRoom");

exitRoom.setActionCommand("exitRoom");

exitRoom.setPreferredSize(new Dimension(630,30));

/\*\*

\* Main Panel

\*/

this.add(exitRoom,BorderLayout.PAGE\_START);

this.add(chatPane,BorderLayout.CENTER);

this.add(status,BorderLayout.PAGE\_END);

this.setBorder(new EmptyBorder(20,20,20,20));

}

* 1. Now add the following main method to the *ChatPanel* class:

/\*\*

\* Test UI

\* @param args

\*/

public static void main(String... args){

JFrame window = new JFrame();

window.setContentPane(new ChatPanel());

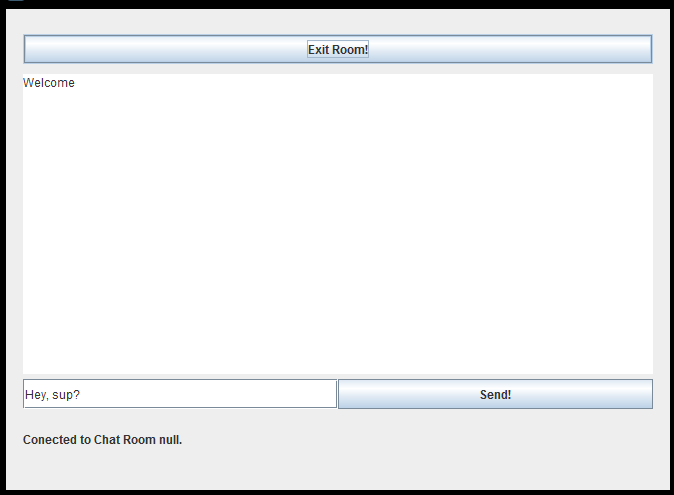
window.setVisible(true);

window.setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE);

window.setSize(680, 520);

}

* 1. Run this class and you should see the window below:

****

**Figure 4**

* 1. Let us now customize the UI Code below so that it looks more personal (change button colors, background color... )

.

* 1. **Show your code to instructor (20 points)**

1. **Chat Engine** (10 minutes)
   1. This an abstract class that already implements most of what we need to make our chat client. But we still need to complete some of its implementation.
   2. Create a new class that extends the “ChatEngine” class, namely “MyClassEngine”.
   3. This will create a new class with various methods with a “@Override” annotation.
   4. We will now Implement one of the method, “setName()” :

@Override // ignore this

public void setName() {

JTextField name = (JTextField)fetchComponent(null, "nameField");

this.name = name.getText();

}

* 1. Notice that the class chat engine has some objects and methods already implemented. The fetch Component methods searches for whatever swing component you need to find in the current window!
  2. Let’s test our classes by creating a runner; this is a class with a single method which is a main method. Create a new class and name it as you wish and make sure it has a main method.
  3. Add the following lines to the above class’s main method:

public static void main(String[] args) {

SimplePanel chatPanel = new ChatPanel();

SimplePanel enterPanel = new EnterPanel();

ChatEngine myEngine = new MyChatEngine(chatPanel, enterPanel);

ChatListener listener = new ChatListener(myEngine);

ChatApp app = new ChatApp(new JFrame(),myEngine, chatPanel,enterPanel,listener);

app.start();

}

1. **HTTP . (35 minutes)**
   1. The Hypertext Transfer Protocol is the most used protocol on the web for information sharing.
   2. TAs explain information on slides (15 minutes). For TAs please note that for this lab the “HTTPConnector.sendGet()” and “HTTPConnector.sendPost()” method already takes care of all the details of creating the respective HTTP request.
   3. Now implement the “checkRoomImp” method.

@Override

public boolean checkRoomImp(String roomName, String url) {

String[] parameters = {

"room="+roomName

};

String result = HTTPConnetor.sendGet(url,parameters , System.out);

return result.contains("\"success\":true");

}

1. **Do a POST that adds User to the Room parsing the server response.**
   1. Implement the “exitRoomImp”

@Override

public boolean exitRoomImp(String roomName, String name, String url) {

String[] parameters = {

"room="+roomName,

"name="+name

};

String result = HTTPConnetor.sendPost(url,parameters , System.out);

return result.contains("\"success\":true");

}

* 1. **Show your code to instructor (20 points)**
  2. Now it’s up to you to implement the POST for “registerUserInRoomImp”.
  3. **Show your code to instructor (20 points)**

1. **Sockets (19 minutes)**
   1. Is another method of communication on the web and it is commonly used in chat services, games and all services needing duplex(two way) communication.
   2. TAs explain information on slides (10 minutes). Notice that the socket is passed as an object and that all messages send to the server must be send in the JSON format.

JSON format in a nutshell:

{

“name”:name,

"message":message,

"room":roomName,

"isMessage":true,

"acknowledge":true

}

* 1. Ask to implement a WEB SOCKET for “sendMessageImp” user clicks send message to chat(15 minutes).
  2. **Show your code to instructor (20 points)**

**After Lab**

* **Read on HTTP Methods**
* **Evaluation Sheet**
* **(optional)Check out the chat server on Github “https://github.com/Apo45ty/NodeSimpleChatServer”**

### Evaluating Lab 4

### Write Your Section# here: ……..

### Please evaluate the quality of the lab and performance of the instructors by filling up the following table and give it to your lab representative. (Choose 5 as the highest and 1 as the lowest grade).

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Items | 5 | 4 | 3 | 2 | 1 |
| The lab started on time SHARP. |  |  |  |  |  |
| The instructor covered adequately the HTTP Methods, WebSockets and answered the group’s questions thoroughly. |  |  |  |  |  |
| The instructor covered adequately Jar files and answered the group’s questions thoroughly. |  |  |  |  |  |
| The instructor overall followed the specified timeline for each step |  |  |  |  |  |
| You found the lab today overall Great (helpful, fruitful, interesting, etc.). |  |  |  |  |  |