**DS3580 PP4 Abdullah Alajaj**

**4/28/2018**

**Courses Registration System**

In this program we are creating courses registration system by developing distributed application based on RMI. There are two sides of such system, clients’ side where people can give a command to see class info, register for classes, drop out of class and see the last name of all students that register for a class. Server side, collect and modify all data been process by clients.

When the server run, it create a registry service to make the connection available to all clients and keep students registry list up to updated. This way multiple clients can register, drop, see information or see registry list for a class without any effect to the server.

On the other hand, each client can open their API on the same time give a command as they wish without interruption. There are five different command client can choose on this system. Each command represented by one letter. G to Get class Information, R to Register a student for a class, D to Drop a student out of a class, S to view a class roster and E to exit the system. Here are the discretion and requirement for each command.

Getting class information will provide you with the course name, number of credits and instructor name. You have to know the class code you want to see in order for the system to give you the information. Otherwise, the system will inform you that the code you entered wrong.

In order to sign up or drop a class, you need you provide the system with class code and student last name. The system then will search if the course available or not. If the course not available, the system rejects the command. Since, it is possible to have multiple last names on the same class, this program design to allow you to accomplish this task.

Viewing a course roster has the same information and requirement as getting course information but the only different is that it displays all students last name who registered for the class.

Exist command is the only way to disconnect with the server. By exist the server, all changes will be saved and can be viewed if you connect again.

**Courses available:**

**Code name credits Instructor**

1111 Coding Club 3 Na Yu

1514 Computer Science\_1 4 Ghassan Azar

4613 Computer Networks 3 David Brown

2534 Data Structures 4 David Fawcett

2514 Computer Science\_2 4 Oriehi Anyaiwe

3563 Game Design 3 Bryce Evans

4663 Operating Systems 3 Cameron Beyer

44 Basic Algebra 3 Ghassan Husseini

2423 Differential Equations 3 Michael Dabkowski

**Flow diagram:**

Client Server

Create or find Registry

Lookup Registry

Accept connection

Send total count

Connected

Object Registry

Receive or reject vote

Send command

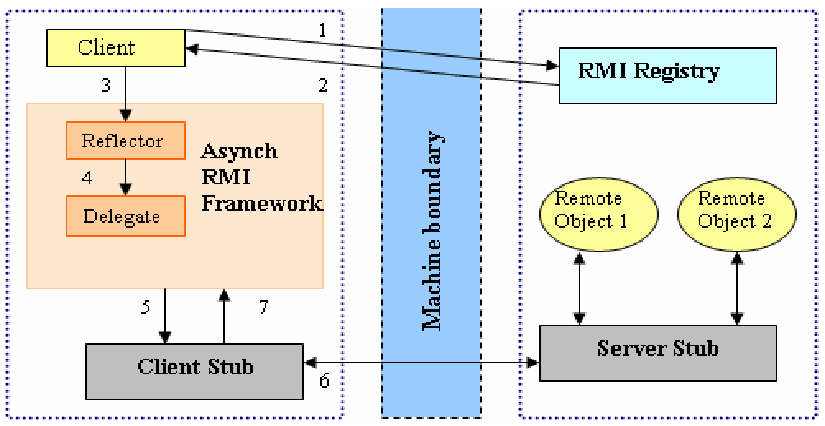
**again**

Receive result

Client Interface

Server Interface

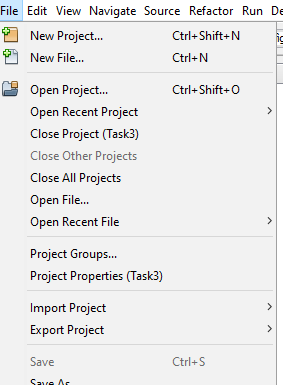
Multiple clients on the same time

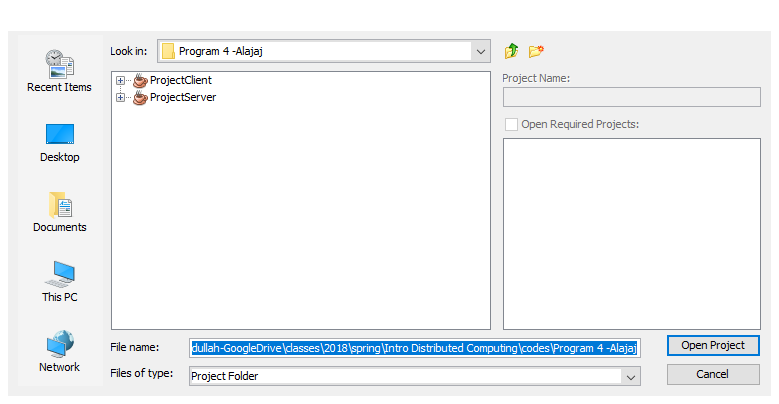
****

**Functions Description:**

**Compiled and executed:**

* Unzip “program 4 – Alajaj” folder then open NetBean IDE (prefer 8.2 version)
  + From top Manu bar : file >> (file location)>>(choose )





**To compiled the code:**

* I test it through local network and on the same PC using windows PowerShell , NetBeans and command prompt in windows 10 Home Version 1709
  + The easy way to open with the correct path on PowerShell is shift + right click in src folder for ProjectServer << open PowerShell window here
  + For command prompt, Go to the desktop search bar and type “cmd”. To set correct path use cd = “ “ where inside “” the location of the src file for ProjectClient.
  + Also, you can copy the PorjectClient folder to you desktop and run it on NetBeans as second client

Note:

* if the (classname).class does not exist then you can create one using this code Javac (the class name you want to compile).java
* You must run the server before the clients.
* Make the Server run on PowerShell and Client-1 run on command prompt and Client-2 on NetBeans or any way you like. Do not run both codes on the same program windows.

**Code test result:**

Let

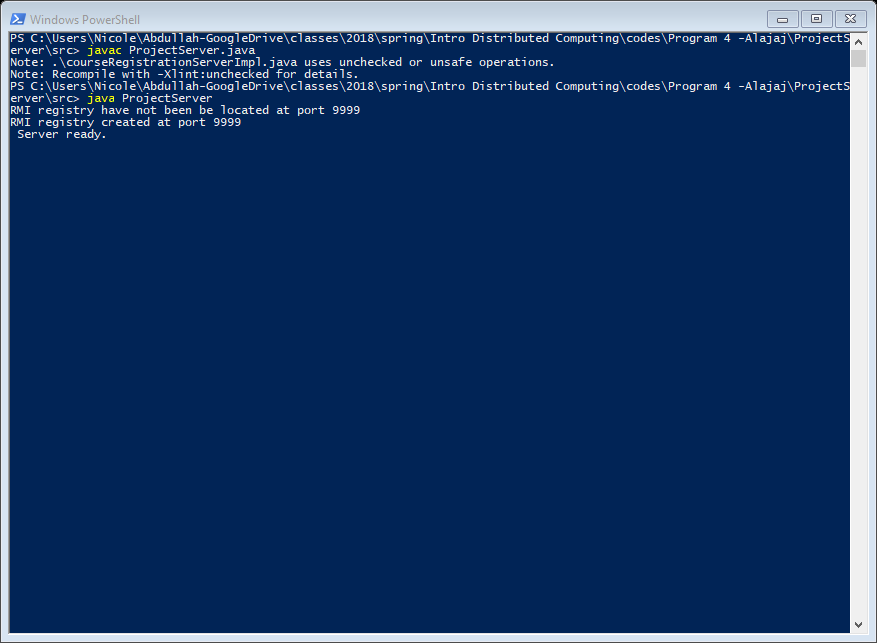
Client to be A

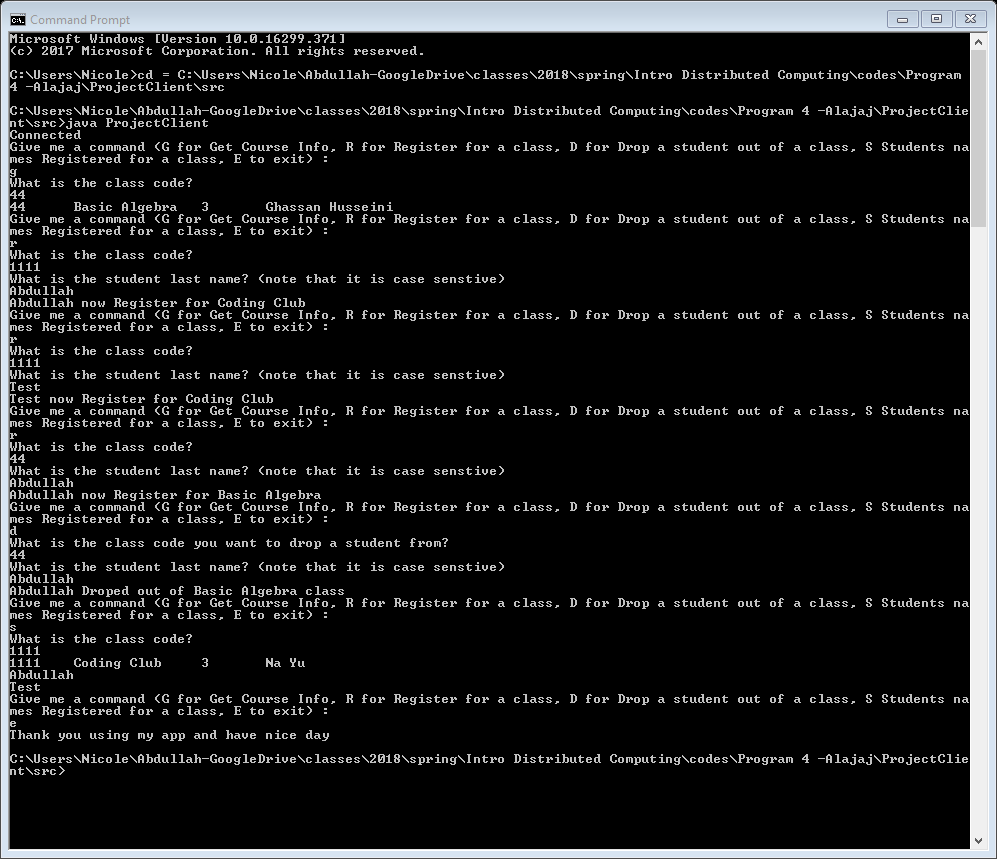
Server to be B

B First invoke method object registry, A try to remote connect using RMI registry lookup. The order in which they run is important. When connection established, A can start register, drop or find course info that you want wanted. There are only five choices. If A submits invalid command B will reject it. On the end, A has the choice to disconnect and exist the application. B will continue run even if A disconnect to keep data and service available. A can connect again and continue where she/he left.

**One-To-One:**

* Running B first and A second: - A can start using the system and B will keep track of any changes and send back the result or reject incorrect entry.





**Many-To-One:**

* Same as one-to-one but with two clients. Commands continue from first test.
* From the result, we see that A1 viewed basic algebra course info, registered Abdullah coding club and basic algebra classes, register Test on basic algebra class, dropped Abdullah out of basic algebra and viewed basic algebra class roster. A2 viewed coding club class roster, sign up Mohammad for coding class. When A2 try to drop Test out of coding club class, she/he entered the name instead of the class code which made the system to disconnect. A2 re-connect again and was able to delete Test with no issue. Finally she/he viewed coding club roster which show that only Abdullah and Mohammad are registered for it.



Re-connecting again



Error when trying to connect before running the polling server

