School of Computing

Bachelor of Science in Computing.

Programme Code: DT211/3 DT228/3

2012 – 2016

Network Programming

Lecturer: Mark Deegan

Student's Name	James Wilson
Class Group	DT211/3
Assignment Number	2
Assignment Title	Simple Server
Date Issued	17/1/2015
Date Due For Return	6/3/2015
Date Returned	28/3/2015

Brief

- Complete the **SimpleCServer** example to a suitable standard. This includes *fully commenting* the code and *making it robust* by having it handle all foreseeable error conditions.
- Build a client **SimpleCClient.c** to work with **SimpleCServer**.
- Build a client **SimpleJavaClient** to work with **SimpleCServer**. This should behave in exactly the same way as **SimpleCClient**.
- Build a server **SimpleJavaServer** to work with both **SimpleCClient** and **SimpleJavaClient** above. This should work in exactly the same way as **SimpleCServer**.
- You must request access to a GitHub repoDIT-School-of-Computing/DT211-3-NP-CA-2.
- Your request should take the form of an email, including your github username.
- You must fork this repo and use it as your starting point for this assignment.
- You must then invite mark deegan as a collaborator to the forked repo.
- Your ongoing use of GitHub throughout the project will be used for tracking your progress and for offering appropriate feedback.
- The ongoing engagement with GitHub throughout the project is a key determinant in the overall mark you will receive.
- You must use only a private repo for this project. No other collaborators should be invited to, or should have access to that repo.
- To your forked repo you must add
- All design documentation
- All build and run instructions for both clients and both servers
- All test documentation, test scripts and test results
- All code developed as part of the project

What is my approach to this Assignment?

Section(i)

I have broken the assignment into three sections that I hope to complete successfully. I firstly will build the connection in C programming language, that will work with the Simple C Server which will allow for a TCP connection to be established using sockets.

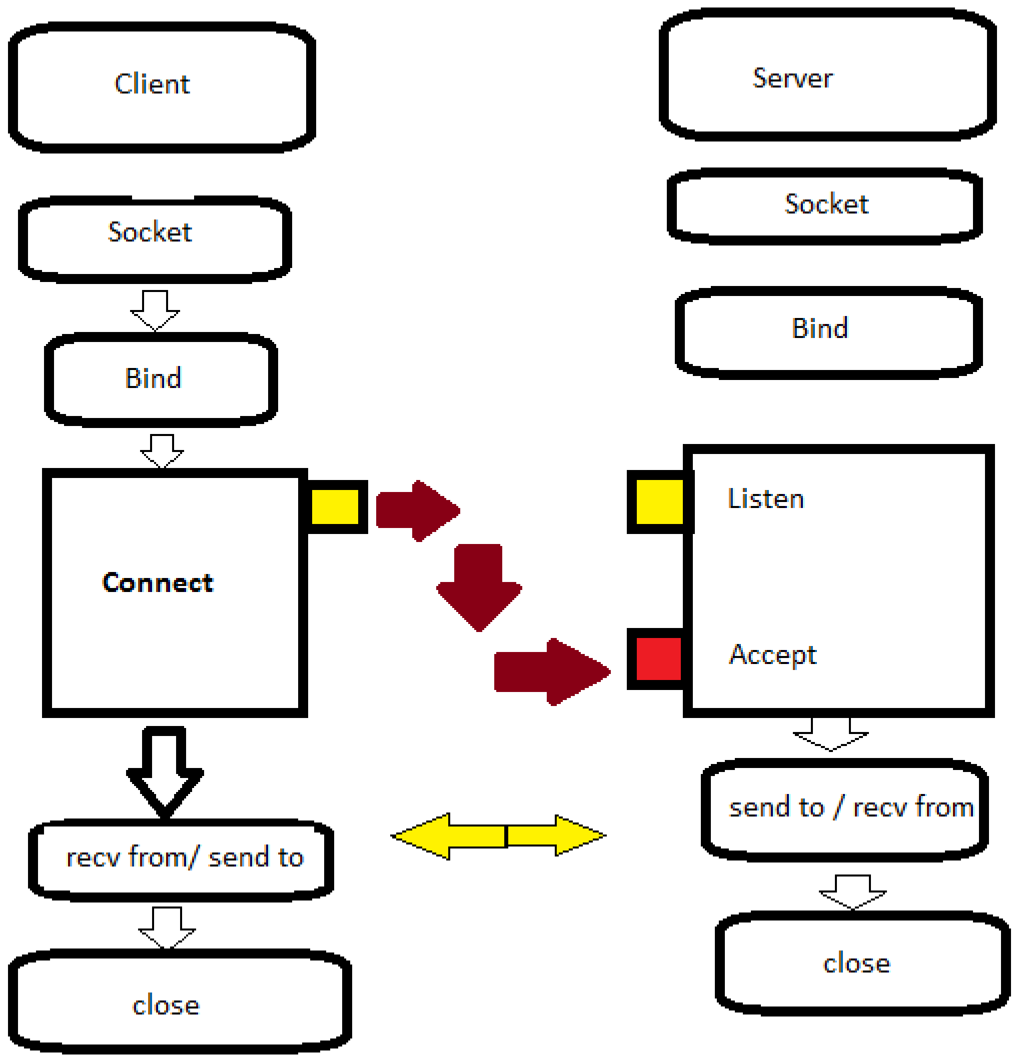
Section(ii)

I will first complete the SimpleCServer before I start coding the SimpleCClient. Once the Simple C Server is complete I can then get the Simple C Client to work with it. Once it is fully developed, I will be able to develop my next simple client in Java, which I will implement with the Simple Java Server. The Java Client should act exactly like the C Client.

Section(iii)

Once I have built the two Simple client (both in C and in Java) working with the Simple C Server and the Simple Java Server, I will try and implement the clients to connect to connect to the other servers on a different programming language i.e. the C Client should be able to communicate with the Java Server.

Flowchart Diagram



The flowchart is my overall design and outlook on the program. It shows the interactions between the client and the server.

Step(i): We create the socket

Step(ii): We then bind the socket to the port

Step(iii): Then, we send the connect request to the server i.e. LocalHost 4444

Step(iv): The client traffic is lining up and waiting to be accepted

Step(v): Once accepted the client can send and receive data

Step(vi): Then close the program