**DC3580 PP1 Abdullah Alajaj**

**Connectionless Datagram Socket**

In this program we are testing and modifying the Java codes from example 2 in figure 4.10, 4.11 and 4.12 in Liu’s textbook.

For code modularity, a class called MyDatagramSocket (Figure 4.10) is created as a subclass DatagramSocket, with two instance method for sending and receiving a message respectively. The Example2SenderReceiver code (4.11) instantiates a MyDatagramSocket object, then calls its sendMessage method, followed by a call to receiveMessage method. The Example2ReceiverSender code (4.12) instantiates a MyDatagramSocket object, then calls its receiveMessage method, follow by a call to its sendMessage method.

Flow diagram:

Example2SenderReceiver

* Send msg1 by satisfying MyDatagramSocket
* Receive msg2
* Auto repeat for task2 and 3 only

Example2ReceiverSender

* Have to run first to unlock datagram socket to guarantee message arrival
* Wait to receive msg1
* Send msg2
* Auto repeat for task2 and 3 only

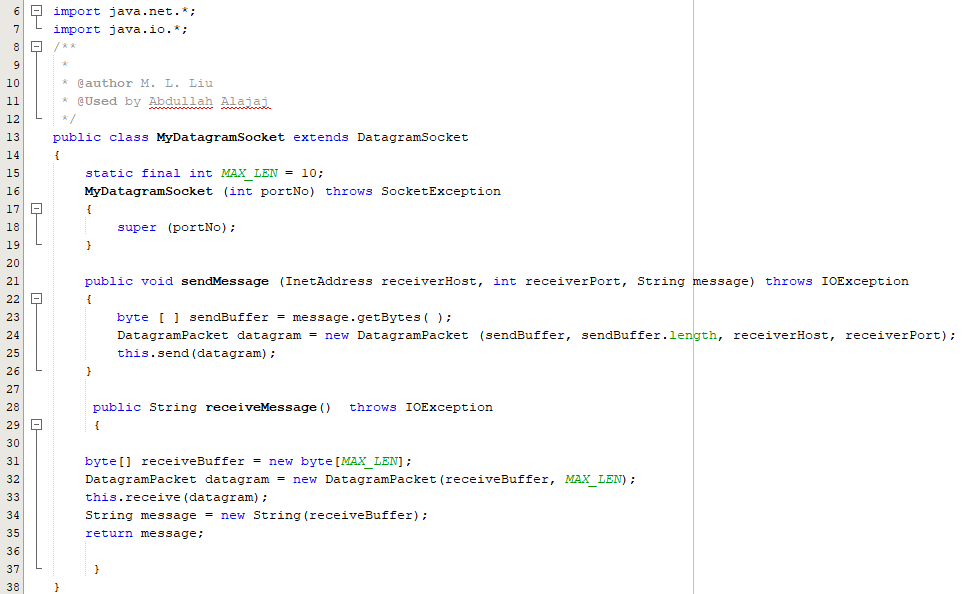
MyDatagramSocket

* 4 protocol have to satisfy to communicate

1. Host address or location
2. Receiver portal
3. Sender portal
4. Message (max 10 characters)

Functions description:

MyDatagramSocket.Java

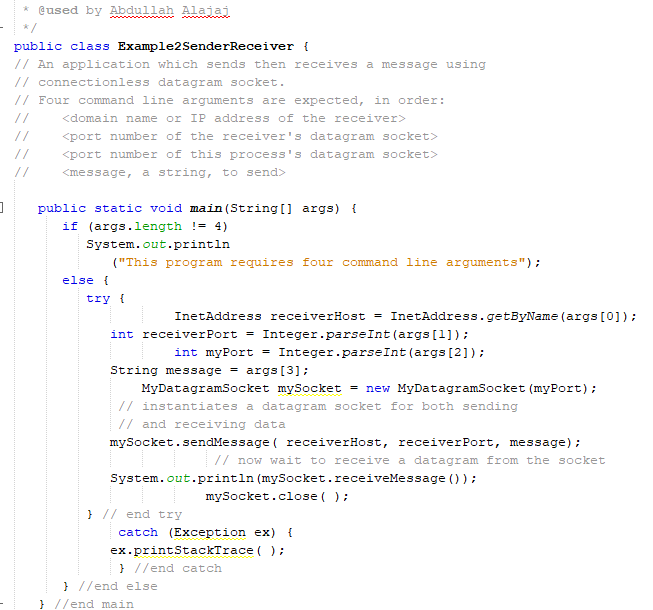


This class used to complete communication between client and server or two clients. Line 15 will set the max characters allow to send. If the message more than 10 characters, only the first 10 will be delivered. Line 16 to 19, is the constructor for the class with on one parameter.

This class has two public functions. First one SendMessage (line 21 to 26) where it converts the message to bytes and loads it with receiver address and portal in a dynamic DatagramPacket which get send to the receiver. Second function is the receiverMessage which will convert back the message to string and enforce the max length. And return the message.

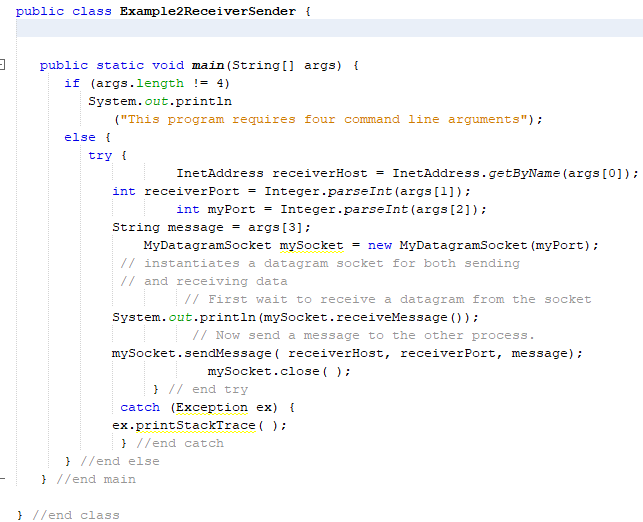
Task1:

Example2SenderReceiver.Java



When sending a message we need to satisfy the protocols for the datagramSocket. After having the protocols the program will create new constructor for MyDatagramSocket with one parameter which allow us to call sendMessage function. Finally, wait to receive message back and print it out.

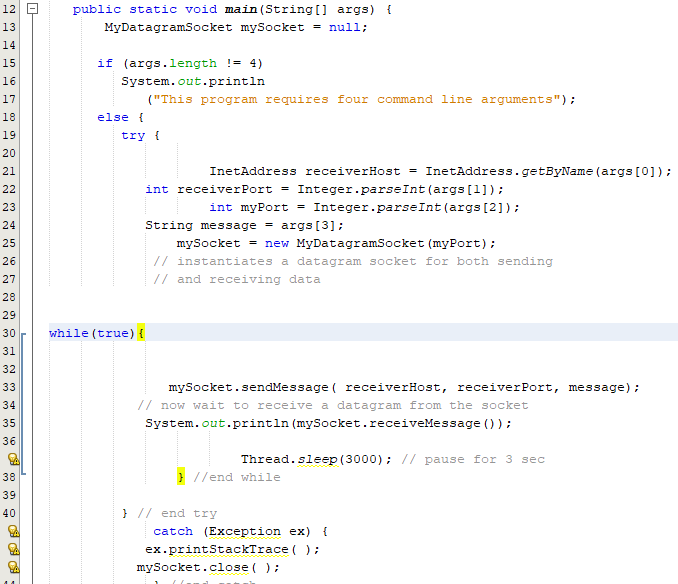
Example2ReceiverSender.Java



When receiving a message then first you need to make sure your API not block by send a message we need to satisfy the protocols for the datagramSocket. After having the protocols the program will create new constructor for MyDatagramSocket with one parameter which allow us to call sendMessage function. But first the receiver waits to receive message before sending and print it out.

Task2:

Example2SenderReceiver.Java



Same as Example2SenderReceiver task1 but the only deferent that initialization of mySocket is outside of the if/else statement since sending and receiving messages in a loop. Also, the while loop have a timeout for 3 sec.

Example2ReceiverSender.Java



The only different than task1 process is a while loop to allow repeat sending and receiving

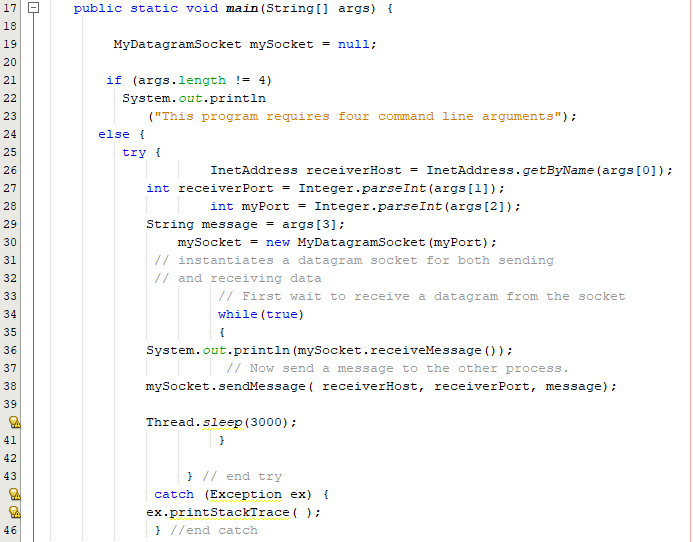
Task3:

Example2SenderReceiver.Java



The only different than task1 process is a while loop to allow repeat sending and receiving

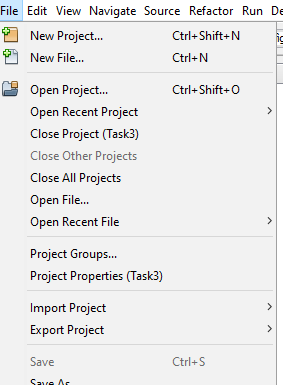
Example2ReceiverSender.Java

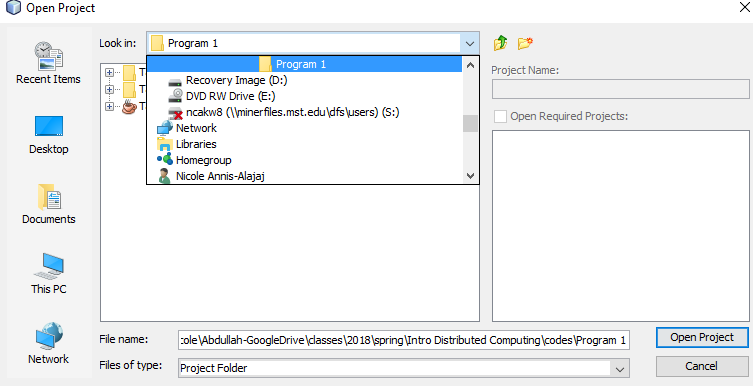


Same as Task2 with a system sleep for 3 sec

Compiled and executed:

* Unzip “program 1 – Alajaj” folder then open NetBean IDE (prefer 8.2 version)
  + From top Manu bar : file >> (file location)>>(choose a task)





To compiled the code:

* I test it through local network and on the same PC using windows PowerShell and command prompt in windows 10 Home Version 1709
  + The easy way to open with the correct path i shift + right click in src folder for the task you want to run << 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.

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 receiver2Sender before Sender2Receiver. For example
  + java Example2ReceiverSender localhost 20000 10000 msg1
  + java Example2SenderReceiver localhost 10000 20000 msg2
* make the sender run on PowerShell and receiver run on command prompt or the other way around. Do not run both code on the same promot windows.

Code test result:

Let

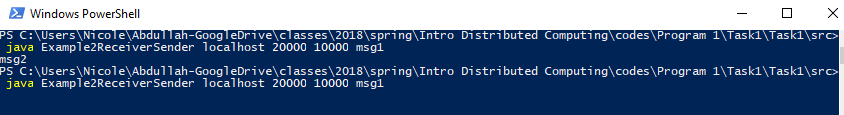
Example2ReceiverSender to be A

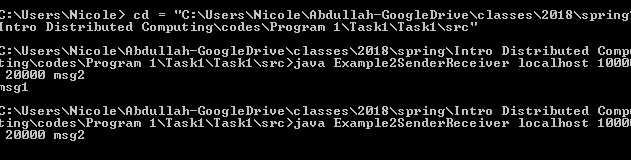
Example2SenderReceiver to be B

B First received msg2, A Received again msg1, the order in which they run is important, Because in this routine, Using a connectionless transfer, sending data. One side cannot guarantee that the data must be received by the receiver, only the first run Receiver makes it into the blocked state, and then shipped row sender, In order to allow the smooth transfer of information. If the receiver has not yet opened the case to send data, then the number it will be lost the recipient will therefore remain blocked. Therefore, the receiver's service should be the first to open, In obstruction state, waiting for the sender to send data.

Task1:

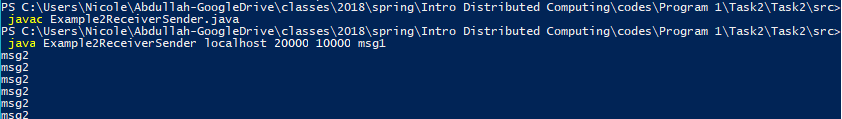
* Running A first and B second :- msg1 arrived to B and msg2 arrived to A.
* Running B first and A second: - none of the messages arrived and both API waiting for each other. When I stop B control c then run it again one message arrived at each console where one msg from B got lost and did not deliver. When I stop A then it does not matter how many time I try to connect to B it does not send or receive anything.





Task2:

* Running A first and B second:- msg2 arrived once to A and msg1 arrived to B then B wait for 3 sec and send another message (msg2). For the code to do that I have to add wile loop to both sides. Otherwise, B will not be able to repeat sending and receiving
* Running B first and A second: - none of the messages arrived and both API waiting for each other.





Task3:

First Executive A, its meeting Enter Into Resistance Plug shape state, It Rear Shipped Row B After sending the message to run receive Into the blocked state, A received After the message, continue to send msg2give B, After the implementation sleep. B Also received the information, then run send Message and execute it sleep. In this way, the two sides continue to exchange data, has been repeating this process

* Running A first and B second:- msg2 arrived once to A and msg1 arrived to B then B wait for 3 sec and send another message (msg2). For the code to do that I have to add wile loop to both sides. Otherwise, B will not be able to repeat sending and receiving
* Running B first and A second: - none of the messages arrived and both API waiting for each other.

