Nicholas Imamshah
1251-9659
nimamshah@ufl.edu

CNT 4007: Programming Assignment #2

Compilation

```
javac *.java
```

Run

```
[Console 1] java network [port]
[Console 2] java receiver [serverURL] [port]
[Console 3] java sender [serverURL] [port] [messageFileName]
```

*****NOTE*** receiver must be run before sender to ensure the sender receives DROP packets**

Code Structure

The network.java holds two additional classes: NetworkManager and NetworkClient. The sender and receiver are managed as two NetworkClients to the NetworkManager. The emulation of a lossy channel is handled by the relay() function of the NetworkManager which is called whenever a NetworkClient has passed a packet to the network. Once an action has been chosen by the NetworkManager, the appropriate NetworkClient's sendRelay() function is called, which will pass the proper packet to the client.

At a higher abstraction, the network is the server and the sender and receiver are both clients.

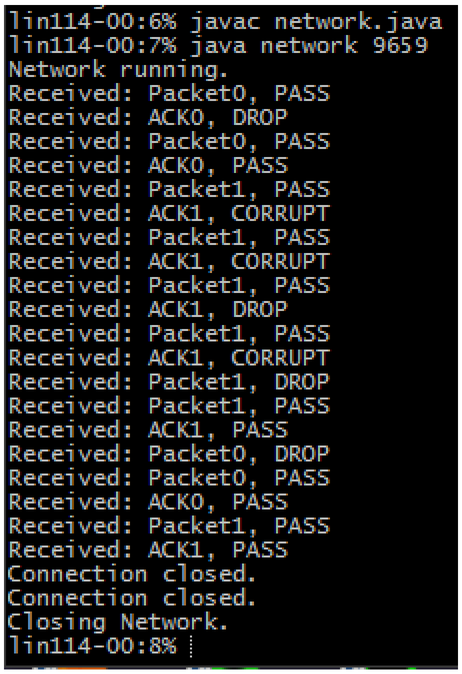
The sender completes three operations: (1) it connects to the network, which is handled in the sender class' constructor, (2) it creates packets from a given message file, (3) it begins sending packets according to the RDT 3.0 Sender protocol, which is wrapped in the function sendPackets().

The receiver completes two operations: (1) it connects to the network, which is handled in the receiver class' constructor, (2) it begins receiving packets according to the RDT 2.2/3.0 Receiver protocol, which is wrapped in the function receivePackets().

The sender and receiver accomplish their tasks with a number of helper methods to encapsulate various bits of logic required to implement the protocols.

All three programs (network, sender, and receiver) use a custom Message class that models the required packet structure. This class supports a static toString() method and a static extract() method which can be used to convert from and object to a string and from a string to an object.

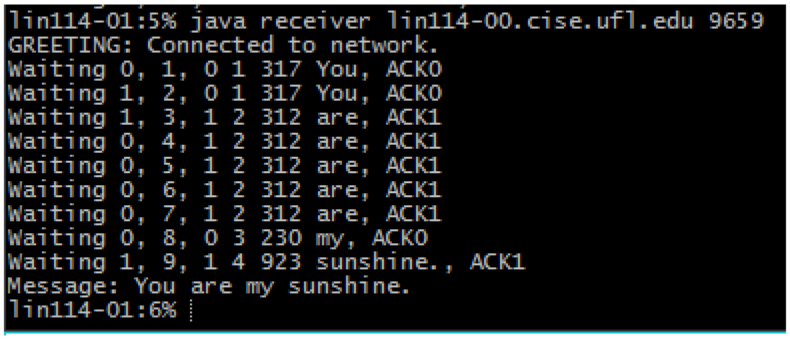
Execution Results



```
lin114-00:6% javac network.java
lin114-00:7% java network 9659
Network running.
Received: Packet0, PASS
Received: ACK0, DROP
Received: Packet0, PASS
Received: ACK0, PASS
Received: Packet1, PASS
Received: ACK1, CORRUPT
Received: Packet1, PASS
Received: ACK1, CORRUPT
Received: Packet1, PASS
Received: ACK1, DROP
Received: Packet1, PASS
Received: ACK1, CORRUPT
Received: Packet1, DROP
Received: Packet1, PASS
Received: ACK1, PASS
Received: Packet0, DROP
Received: Packet0, PASS
Received: ACK0, PASS
Received: Packet1, PASS
Received: ACK1, PASS
Connection closed.
Connection closed.
Closing Network.
lin114-00:8%
```

Network log on lin114-00.cise.ufl.edu

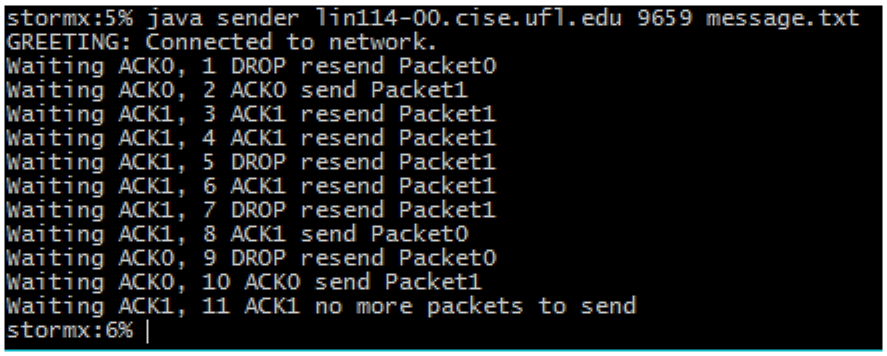
We can see the decisions of PASS, DROP, and CORRUPT of the network to each incoming packet, as well as additional logs of the networks initiation and when the sender and receiver disconnect. Each execution of the programs is different according to the probabilities specified for passing, dropping, and corrupting packets.



```
lin114-01:5% java receiver lin114-00.cise.ufl.edu 9659
GREETING: Connected to network.
Waiting 0, 1, 0 1 317 You, ACK0
Waiting 1, 2, 0 1 317 You, ACK0
Waiting 1, 3, 1 2 312 are, ACK1
Waiting 0, 4, 1 2 312 are, ACK1
Waiting 0, 5, 1 2 312 are, ACK1
Waiting 0, 6, 1 2 312 are, ACK1
Waiting 0, 7, 1 2 312 are, ACK1
Waiting 0, 8, 0 3 230 my, ACK0
Waiting 1, 9, 1 4 923 sunshine., ACK1
Message: You are my sunshine.
lin114-01:6%
```

Receiver log on lin114-01.cise.ufl.edu

We can see the RDT state indicated by “Waiting #”, and the appropriate ACK to send provided the packet received.



```
stormx:5% java sender lin114-00.cise.ufl.edu 9659 message.txt
GREETING: Connected to network.
Waiting ACK0, 1 DROP resend Packet0
Waiting ACK0, 2 ACK0 send Packet1
Waiting ACK1, 3 ACK1 resend Packet1
Waiting ACK1, 4 ACK1 resend Packet1
Waiting ACK1, 5 DROP resend Packet1
Waiting ACK1, 6 ACK1 resend Packet1
Waiting ACK1, 7 DROP resend Packet1
Waiting ACK1, 8 ACK1 send Packet0
Waiting ACK0, 9 DROP resend Packet0
Waiting ACK0, 10 ACK0 send Packet1
Waiting ACK1, 11 ACK1 no more packets to send
stormx:6% |
```

Sender log on storm.cise.ufl.edu

We can see the proper Packet# chosen according to the various conditions of the RDT 3.0 Sender protocol.

Bugs

None, to my knowledge

Comments

None