

# Programming Assignment 3: Gnutella P2P Network with Consistency

Ismael J Lopez - [ilopez5@hawk.iit.edu](mailto:ilopez5@hawk.iit.edu)

---

## Peer Files

Peer files are provided for 16 peers with 10 files each. There are **no** duplicates across peers. Files are generated using [Gensort](#) and are **not actual mp4 files**. I know the instructions said to generate text files, these have text in them and are not binary files. These files are located in the `files/` directory and are numbered by port number 6001-6016.

---

## Install

The code was developed and tested on an Ubuntu 20.04 machine using OpenJDK 17. Ubuntu can be easily installed on a Virtual Machine, or use WSL if on Windows. OpenJDK-17 can be installed on Ubuntu using the following command:

```
$ sudo apt install openjdk-17-jdk
```

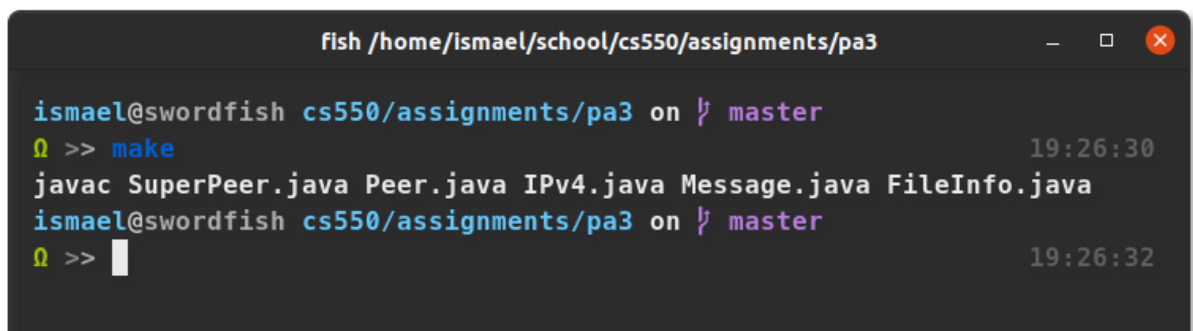
---

## Build

A Makefile has been provided. Running `make` will compile all the code and running `make clean` will remove any `.class` files.

Examples:

```
$ make # compiles
$ make clean # cleans the auxiliary files
```

A screenshot of a terminal window titled 'fish /home/ismael/school/cs550/assignments/pa3'. The terminal shows a user prompt 'ismael@swordfish cs550/assignments/pa3 on master' followed by the command 'make' being executed. The output shows 'javac SuperPeer.java Peer.java IPv4.java Message.java FileInfo.java'. The prompt then returns to 'ismael@swordfish cs550/assignments/pa3 on master' with a cursor. The timestamps '19:26:30' and '19:26:32' are visible on the right side of the terminal lines.

```
fish /home/ismael/school/cs550/assignments/pa3

ismael@swordfish cs550/assignments/pa3 on  master
❯ >> make                                     19:26:30
javac SuperPeer.java Peer.java IPv4.java Message.java FileInfo.java
ismael@swordfish cs550/assignments/pa3 on  master
❯ >>                                           19:26:32
```

Screenshot:

---

## Running

### Topology Config file

Please define a single topology config file which defines the network. The syntax for this topology is as follows:

```
c <push-pull> <ttr (pull only)> # e.g., 'c push' or 'c pull 1' for 1 minute TTR

s <address1:port1> <address2:port2> # defines the SuperPeer neighbor
s 127.0.0.1:5000 127.0.0.1:5001 # e.g., SuperPeer 5001 is neighbor to SuperPeer 5000
s 127.0.0.1:5001 127.0.0.1:5000 # e.g., SuperPeer 5000 is neighbor to SuperPeer 5001
```

```
p <address1:port1> <address2:port2> # defines the leaf peer of a SuperPeer
p 127.0.0.1:5000 127.0.0.1:6001 # e.g., Leaf Peer 6001 is under SuperPeer 5000
p 127.0.0.1:5001 127.0.0.1:6003 # e.g., Leaf Peer 6003 is under SuperPeer 5001
```

There are topology files already included. The \*.simple\*.config files contain a smaller network of 3 SuperPeers, each with 1 Leaf Peer. The \*.full\*.config files contain the network requested in the instructions: 8 SuperPeers with 1-3 Leaf Peers each.

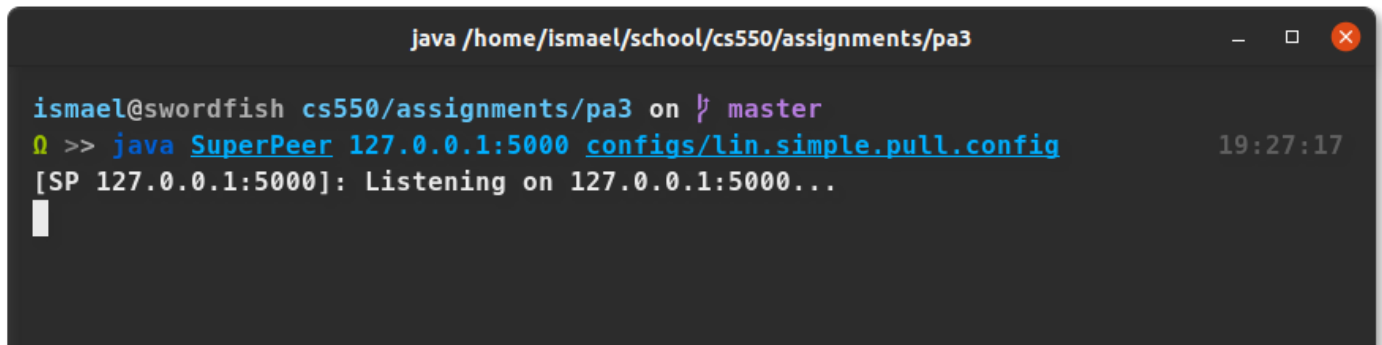
## Launching SuperPeers

To launch a SuperPeer, run the program while passing in the IPv4 address and port, and the config file containing the static topology of the network.

Example(s):

```
$ java SuperPeer 127.0.0.1:5000 configs/all.simple.pull.config
$ java SuperPeer 127.0.0.1:5000 configs/lin.full.push.config
```

Screenshot:

A terminal window titled "java /home/ismael/school/cs550/assignments/pa3" shows the command "java SuperPeer 127.0.0.1:5000 configs/lin.simple.pull.config" being executed. The output is "[SP 127.0.0.1:5000]: Listening on 127.0.0.1:5000...". The terminal prompt is "ismael@swordfish cs550/assignments/pa3 on master". The time "19:27:17" is shown in the top right corner.

```
java /home/ismael/school/cs550/assignments/pa3

ismael@swordfish cs550/assignments/pa3 on master
$ java SuperPeer 127.0.0.1:5000 configs/lin.simple.pull.config
[SP 127.0.0.1:5000]: Listening on 127.0.0.1:5000...
```

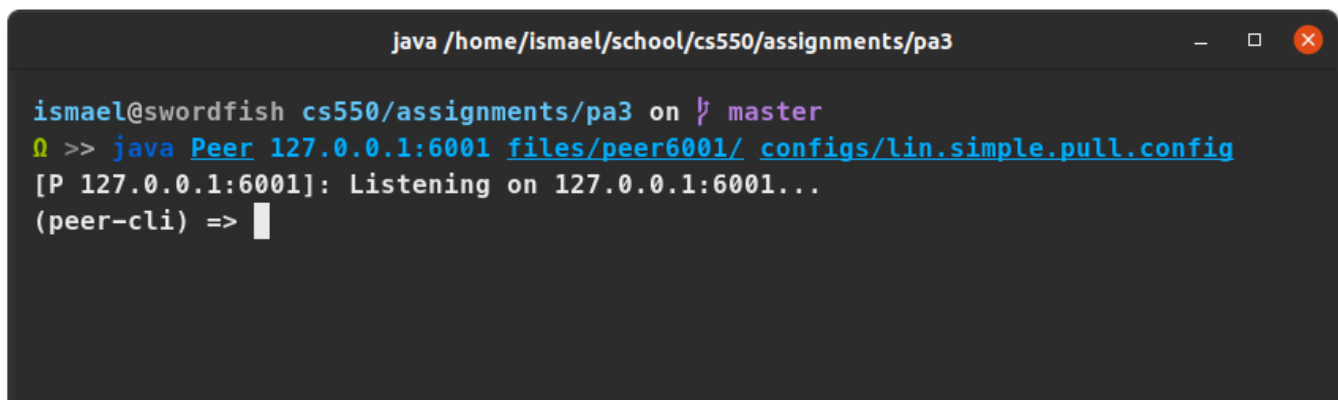
## Launching Peers

To launch a Peer, or leaf peer, run the program while passing in similar arguments to the SuperPeer, with the addition of a file directory containing the Peer's files. See below:

Example(s):

```
$ java Peer 127.0.0.1:6001 files/peer6001 configs/all.simple.pull.config
$ java Peer 127.0.0.1:6001 files/peer6001 configs/lin.full.push.config
```

Screenshot:

A terminal window titled "java /home/ismael/school/cs550/assignments/pa3" shows the command "java Peer 127.0.0.1:6001 files/peer6001/ configs/lin.simple.pull.config" being executed. The output is "[P 127.0.0.1:6001]: Listening on 127.0.0.1:6001..." followed by "(peer-cli) =>". The terminal prompt is "ismael@swordfish cs550/assignments/pa3 on master". The time "19:27:17" is shown in the top right corner.

```
java /home/ismael/school/cs550/assignments/pa3

ismael@swordfish cs550/assignments/pa3 on master
$ java Peer 127.0.0.1:6001 files/peer6001/ configs/lin.simple.pull.config
[P 127.0.0.1:6001]: Listening on 127.0.0.1:6001...
(peer-cli) =>
```

## Things you should know

- If the files/peer60XX directory does not have nested owned/ and downloads/ subdirectories, they will be created.
- The owned/ directory contains master files, that the given peer uniquely owns.

- The `downloads/` directory contains any downloaded files from other peers. This directory is not watched by the `EventListener` class for file modifications, given the assumption that only origin servers can modify their master copies.
- The communication messages in this network can be described as follows:

```
# fileinfo syntax is: filename,owner,version

register:  "register 0;0;fileinfo;sender"

deregister: "deregister 0;0;fileinfo;sender"

query:      "query msgid;ttl;fileinfo;sender"

queryhit:   "queryhit msgid;ttl;fileinfo;sender peerWhoHasFile"

invalidate: "invalidate msgid;ttl;fileinfo;sender"

# (pull model)
# request for file status
status: "status fileinfo"

# responses
deleted: "deleted"
uptodate: "uptodate"
outdated: "outdated"
```