Sockets

Apr. 17th

Fool in the rain

```
bool Socket::waitOnEvents(int timeout){
  struct pollfd pfd;
  pfd.fd = fd;
  pfd.events = POLLIN | POLLRDHUP | POLLHUP;
  return poll(&pfd,1,timeout) > 0;
}
```

Threaded Server

```
void threadedServer(string p){
   Socket listener;
   vector<Client*> clients;
   listener.listen(p);
   while(1){
      listener.waitOnEvents(-1);
      //generate a Client thread
      if(listener.hasData()){
         Client* c = new Client(listener.accept())
         clients.push_back(c);
         c->start();
      }
      //remove clients who are done
      for( int i =clients.size()-1;  i >= 0; i -= 1){
         if(clients[i]->done){
            clients[i]->end();
            delete clients[i];
            clients.erase(clients.begin() + i);
         }
      }
   }
}
```

Client Struct

```
struct Client{
    thread* exe = NULL;
    Socket sock;
    bool done = false;

    Client(Socket s){
        sock = s;
    }

    ~Client(){
        delete exe;
    }

    void start( ){
        exe = new thread( &Client::handle, this);
    }

    void end( ){
        if( exe  != NULL){
            exe->join();
        }
    }

    void handle( ){
        //handle the connection
    }
};
```

Client Struct

```
void handle( ){
   while(!done){
      sock.waitOnEvents(-1);
      if(sock.isBroken()){
         done = true;
      }else if(sock.hasData()){
         //read — and write to sock
         //set done to true if you are done
      }
   }
   sock.close();
}
```