Objective

The purpose of this project is to give you experience writing a UDP server-type application using the Winsock library. You will create a "Quote of the Day" (qotd) server that will respond to incoming requests with a quote obtained from a text file on the server. The qotd service is commonly provided on port 17.

Implementation

The qotd server expects one of two commands to be received from a client:
1. `sendQOTD` – Server should respond with the quote of the day.
2. `shutdown` – Server should shutdown (***if*** the command is sent from a client running on the **same** machine).

Any other text received by the server should be logged but then ignored.

The text for the quote of the day should be stored in a text file called **`qotd.txt`** that resides in the same directory as qotd.exe. When the server receives a "`sendQOTD`" command (*case insensitive*), it will (1) open the file `qotd.txt`, (2) send the entire contents of the file to the client and then (3) close the file. By not keeping the file open for an extended period of time allows the daily quote to be changed (by simply changing the contents of the file) without having to shutdown and restart the server application.

Your qotd server should run on any windows-based networked computer. Your server should log all activity to the standard output device (eg. cout). The example below shows the output the qotd server would produce if running on a computer with IP address 10.1.6.31:

```
C:\> qotd
Wed Mar 9 14:23:38 2016 - Started qotd service on 10.1.6.31:17
Wed Mar 9 14:23:49 2016 - Datagram received from: 10.1.154.14:2349
Wed Mar 9 14:23:50 2016 - Ignored command: hey
Wed Mar 9 14:23:59 2016 - Datagram received from: 10.1.5.20:12849
Wed Mar 9 14:24:00 2016 - Received command: sendQOTD
Wed Mar 9 14:24:00 2016 - Sent file: qotd.txt to 10.1.5.20:12849
Wed Mar 9 14:24:07 2016 - Datagram received from: 10.1.6.31:5876
Wed Mar 9 14:24:07 2016 - Received command: shutdown
Wed Mar 9 14:24:07 2016 - Shutting down qotd service
```

The output log shows a client from 10.1.154.14 sent the server the unknown command "`hey`". Since it was not "`sendQOTD`" or "`shutdown`", it was ignored. A client from 10.1.5.20 then sent a "`sendQOTD`" command (the client used port 12849), and the server responded by sending the quote in the `qotd.txt` file back to IP address 10.1.5.20 and port 12849. Finally a "`shutdown`" command was sent from a client running on 10.1.6.31 (the same computer the server was running on), and the server shut down. Had the "`shutdown`" been received from any other IP address, it would have been ignored.

The qotd server should run according to the following algorithm:

1. Initialize the Winsock library (WSAStartup).
2. Create a UDP socket using `passivesock()` and output the startup message to the log.
3. Perform the following actions until receiving a valid shutdown request:
    a. Wait (forever if necessary) for a client to send a datagram.
    b. Using the client's IP address and port number report the connection to the log.
    c. If the received message is "`sendQOTD`" (case insensitive), open the `qotd.txt` file and send it to the client, using its IP address and port number (and report to log).
    d. If the received message is "`shutdown`" ***and*** it was sent from the same machine as that running the server, send the client the message: "`Service: qotd - will be shutdown`" and set a flag to exit the loop. Otherwise send the client a deny message: "`SHUTDOWN request denied`". You should also log the message.
4. Close the socket (closesocket) and log the shutdown message.

Several supporting routines have already been implemented for you. Below is a summary of those routines.

1. `char* timestamp()` – Returns a timestamp string formatted like this: "Wed Mar 9 14:23:38 2016"
2. `SOCKET passivesock(char *service, char *protocol)` – Allocates a server socket for a given service (or port) and protocol (TCP or UDP).
3. `char* getMyIPAddress()` – Returns a pointer to the host's IP address as a null-terminated, dotted-decimal string.

These functions can be found on EASEL. Click on the "Source Code Examples" link under Class Resources.

Below is an example of what may be stored in the qotd.txt file:

```
A soft answer turneth away wrath:
but grievous words stir up anger.
       - Proverbs 15:1
```

All characters (including EOLN chars) in the file should be transmitted to the client. Each line has a maximum of 2000 characters. There is no limit to the number of lines in the file.

Testing

You can test your qotd server using your qotd client program.

Turn In

Submit your qotd.cpp file to EASEL by midnight on the due date.

Not everything that can be counted counts, and not everything that counts can be counted.
- Albert Einstein