Windows version:

This assignment was built using go version 1.6.2 for windows

C:\Users\Keith\work\src\github.com\user\blogpostapi>go version

go version go1.6.2 windows/amd64

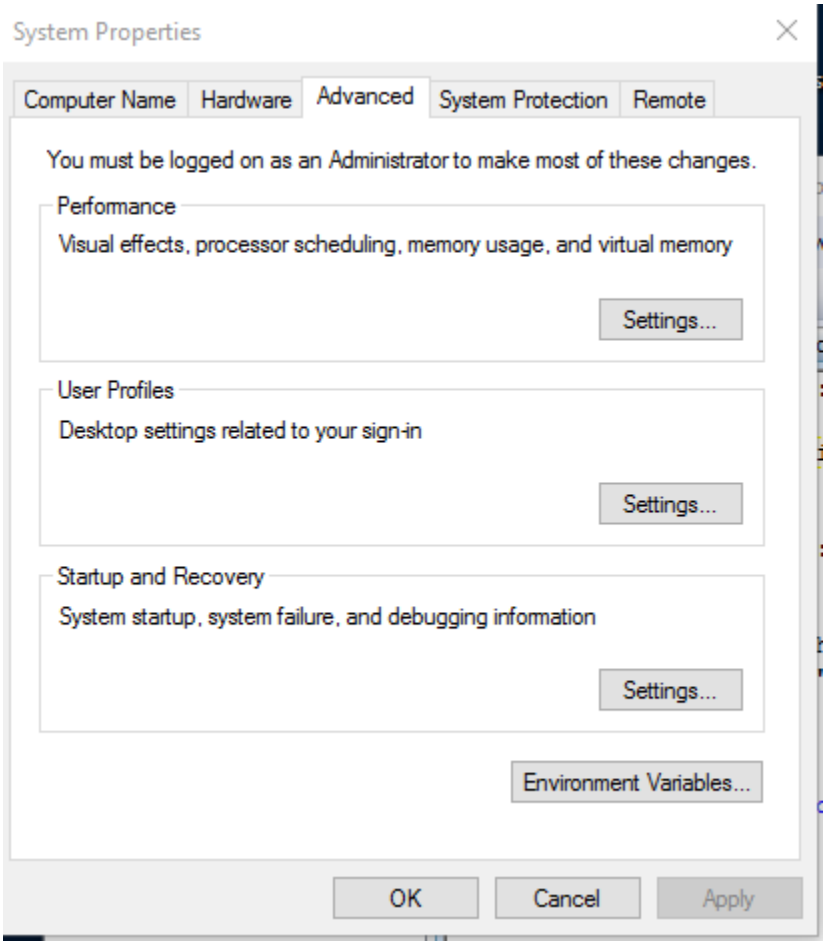
Directions:

1) Install GoLang for Windows (https://golang.org/doc/install)
2) Install curl for Windows (https://curl.haxx.se/download.html)
3) Install git for Windows (https://git-scm.com/download/win)
4) Install gcc for Windows (MinGW version is fine: http://mingw.org/)

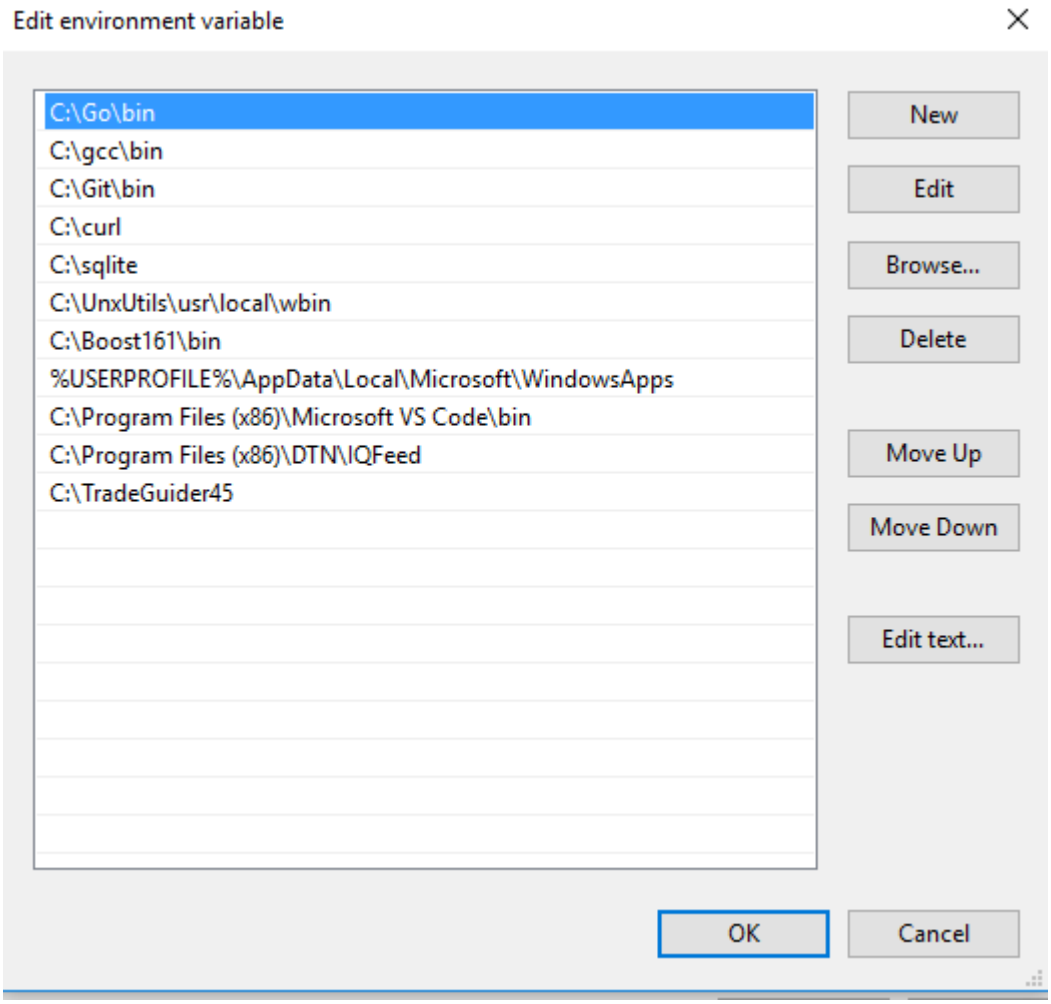
Create directory structure: Example: C:\Users\Keith\work\src\github.com

Update Path in Environment variables with bin directories for all programs:

Bring up System Properties in control panel:



Click on Environment Variables (should look something like this after you have added all the programs to your Path):



cd to C:\Users\Keith\work\src\github.com directory (or wherever your go directory is for source code). Get the following packages (if they are not present in your GoLang installation):

- go get github.com/gorilla/mux
- go get github.com/mattn/go-sqlite3
- go get github.com/jinzhu/gorm

<u>Instructions for installing blogpostapi:</u>

Create a directory in windows (under previous directory structure) called blogpostapi.

Example:

C:\Users\Keith\work\src\github.com\user> mkdir blogpostapi

Download the files to the directory.

cd to the directory and run the command “go build” (this includes all the *.go files in the build):

C:\Users\Keith\work\src\github.com\user\blogpostapi> go build

To run the server, just type blogpostapi (runs the binary blogpostapi.exe)

C:\Users\Keith\work\src\github.com\user\blogpostapi> blogpostapi

This spins up a web service using http://localhost:8080

Test cases using curl:

POST:

curl -X POST -d "{\"title\": \"My test post\", \"body\": \"This is my test blog post\"}" http://localhost:8080/post

output:

{"post_id":1,"title":"My test post","body":"This is my test blog post"}

curl -X POST -d "{\"title\": \"My second test post\", \"body\": \"This is my second test blog post\"}" http://localhost:8080/post

output:

{"post_id":2,"title":"My second test post","body":"This is my second test blog post"}

GET:

curl -i http://localhost:8080/posts

output:

HTTP/1.1 200 OK

Content-Type: application/json; charset=UTF-8

Date: Tue, 14 Feb 2017 04:13:00 GMT

Content-Length: 160

[{"post_id":1,"title":"My test post","body":"This is my test blog post"},{"post_id":2,"title":"My second test post","body":"This is my second test blog post"}]

Ubuntu Linux version:

The assignment was built using go version 1.6 for Ubuntu Linux

ubuntu@ip-172-31-16-115:~$ go version

go version go1.6 linux/amd64

Directions:

Installing gcc for Ubuntu (run the following commands):

sudo apt-get update

sudo apt-get upgrade

sudo apt-get install build-essential

gcc -v

make -v

Install git:

sudo apt install git

Install GoLang:

apt-get install golang

Create a directory for source code (Example /home/Ubuntu/Go)

Set GOPATH environment variable (export GOPATH=$HOME/Go)

Install curl

sudo apt-get install curl libcurl3 libcurl3-dev php5-curl

sudo service apache2 restart

Install additional go libraries if needed:

ubuntu@ip-172-31-16-115:~$ go get github.com/gorilla/mux

ubuntu@ip-172-31-16-115:~$ go get github.com/mattn/go-sqlite3

ubuntu@ip-172-31-16-115:~$ go get github.com/jinzhu/gorm

Create a directory in ubuntu (under previous directory structure) called blogpostapi.

Example:

ubuntu@ip-172-31-16-115:~/Go$ mkdir blogpostapi

Download the files to the directory.

cd to the directory and run the command “go build” (this includes all the *.go files in the build):

ubuntu@ip-172-31-16-115:~/Go/blogpostapi$ go build

To run the server, just type ./blogpostapi (runs the binary blogpostapi)

ubuntu@ip-172-31-16-115:~/Go/blogpostapi$ ./blogpostapi

This spins up a web service using http://localhost:8080

Test cases using curl (same commands and output as the windows version):

POST:

curl -X POST -d "{\"title\": \"My test post\", \"body\": \"This is my test blog post\"}" http://localhost:8080/post

output:

{"post_id":1,"title":"My test post","body":"This is my test blog post"}

curl -X POST -d "{\"title\": \"My second test post\", \"body\": \"This is my second test blog post\"}" http://localhost:8080/post

output:

{"post_id":2,"title":"My second test post","body":"This is my second test blog post"}

GET:

curl -i http://localhost:8080/posts

output:

HTTP/1.1 200 OK

Content-Type: application/json; charset=UTF-8

Date: Tue, 14 Feb 2017 04:13:00 GMT

Content-Length: 160

[{"post_id":1,"title":"My test post","body":"This is my test blog post"},{"post_id":2,"title":"My second test post","body":"This is my second test blog post"}]