SWE Project Report

Gregor Holli

Introduction

This report aims to compensate for previous assignments. It contains both a description of the project itself as well as my gained experiences and learnings.

URL

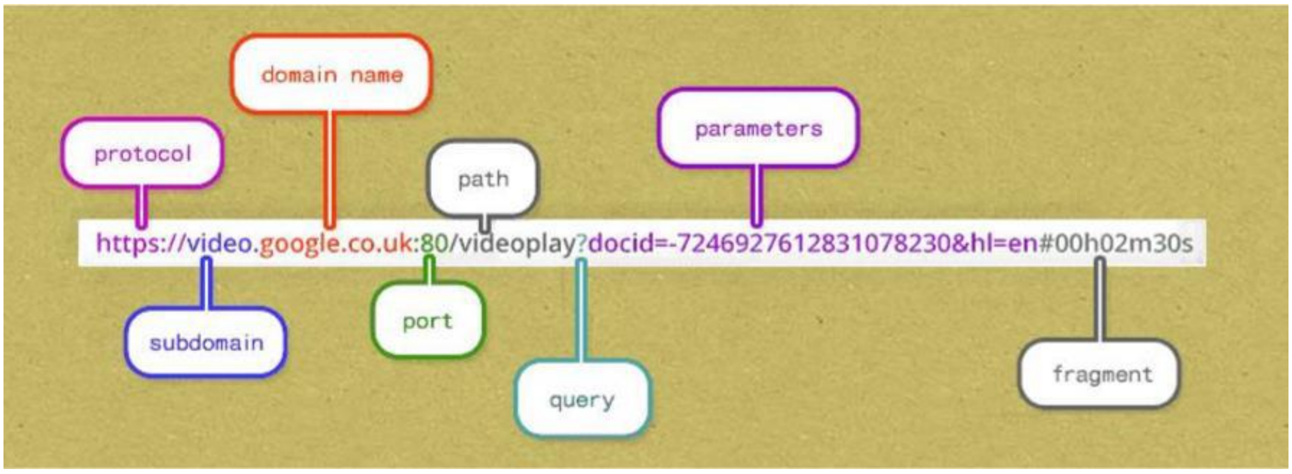
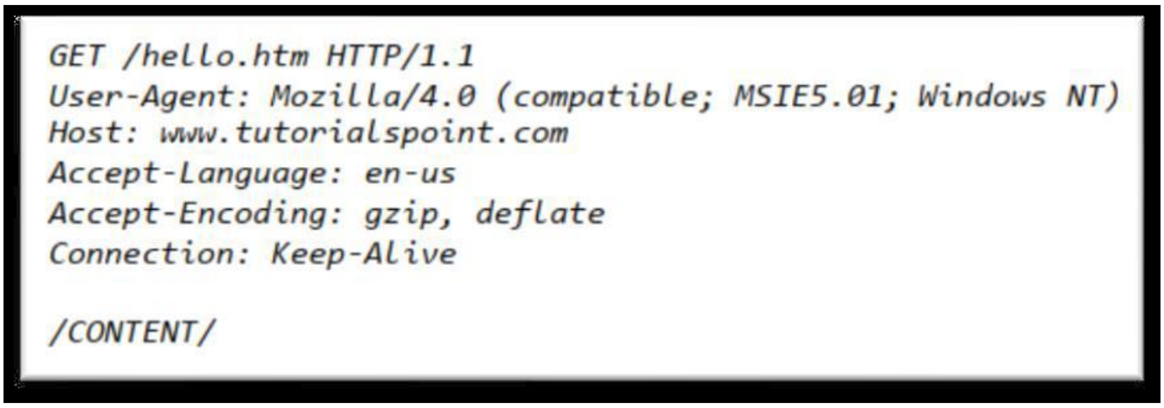

domain name
protocol
path
parameters
https://video.google.co.uk:80/videoplay?docid=-7246927612831078230&hl=en#00h02m30s
subdomain
port
query
fragment

Figure 1 Example of a URL

The first assignment was related to the deconstruction of a URL into its separate components. From the displayed graphic, the separate elements, which make up a URL can be easily visually determined.
The separation of the raw URL was achieved in this project, using five distinct methods.
These first extract the protocol and fragments. Next, if present, the parameters are separated to be extracted then extracted in a separate method. The segments, which are separated via a '/' are also extracted. Finally, through various getter functions these stored values may be accessed for other parts of the code, namely the plugins.

Request and Response

Both are implemented using their respective interfaces.



```
GET /hello.htm HTTP/1.1
User-Agent: Mozilla/4.0 (compatible; MSIE5.01; Windows NT)
Host: www.tutorialspoint.com
Accept-Language: en-us
Accept-Encoding: gzip, deflate
Connection: Keep-Alive

/CONTENT/
```

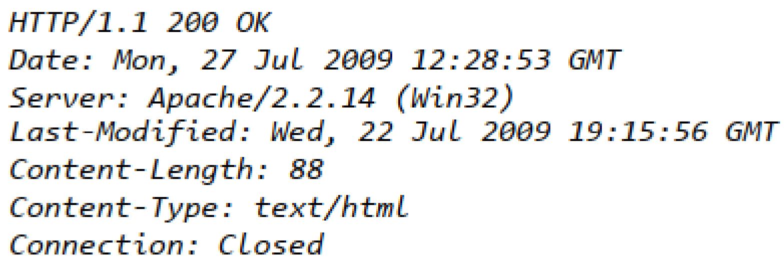
Figure 2 Request Example

The request constructor requires a stream, through several methods, this stream is then dissected.

```
public Request(Stream request)
        {
            sr = new StreamReader(request);
            getFirstLine();
            if (isValid)
            {
                extractHeaders();
                getPostContent();
            }
        }
```

- private void getFirstLine() - splits the request to extract the tokens (f.e. GET / HTTP / 1.1) and then constructs a new URL
- private void checkIfIsValid() - confirms that one of the tokens(method) is GET or POST
- private void extractHeaders() - reads and stores the request header
- private void getPostContent() - if a POST method occursm the content is extracted

The dissected values are stored and made accessible via various getter functions.



```
HTTP/1.1 200 OK
Date: Mon, 27 Jul 2009 12:28:53 GMT
Server: Apache/2.2.14 (Win32)
Last-Modified: Wed, 22 Jul 2009 19:15:56 GMT
Content-Length: 88
Content-Type: text/html
Connection: Closed
```

Figure 3 Response Example

The response offers an initial standard constructor and allows for ContentType, StatusCode, ServerHeader and content, via various types of input, to be set with its methods. It is also possible to add headers and finally to send the response back to the network stream.

Client and Server

The Client is established via a web interface, it calls up the relevant index.html and home.js files, which will allow graphical interaction with the plugin functionalities via REST API. Simply put the Client code handles the URL as well as the subsequent request and notifies the necessary plugins. Subsequent requests to the plugins will, usually, generate and deliver a new html page by the plugin. Only exception being the ToLower Plugin which returns the value via a JavaScript alert.

The Server adds the plugins to the plugin manager, listens to port 8080 and establishes a connection to the database. If the client connects, a new thread is started and the client is handled.

```
            PluginManager.Instance.Add(new TempPlugin());
            PluginManager.Instance.Add(new NaviPlugin());
            PluginManager.Instance.Add(new ToLowPlugin());
            PluginManager.Instance.Add(new FilePlugin());
```

```
if (client.Connected)
{
    Thread t = new Thread(start: () => handeClient(client));
    t.Start();
    threads.Add(t);
}
```

Pluginmanager and IPlugin

The plugin manager simply contains a list of plugins which may be added or cleared. It also returns a singleton instance of the plugin manager ensuring only one plugin manager at a time may be initiated and used.

```
public static PluginManager Instance
        {
            get
            {
                if (instance == null)
                {
                    instance = new PluginManager();
                }
                return instance;
            }
        }
```

Every base plugin must contain two methods:

- float CanHandle(IRequest req); - returns a value of 0 or 1 depending on, if the relevant parameter is contained within a requests segements.
- IResponse Handle(IRequest req); - Called by the server when the plugin should handle the request.

```
        public float CanHandle(IRequest req)
        {
            foreach (string seg in req.Url.Segments)
            {
                if (seg == "tolow")
                {
                    return 1;
                }
            }
            return 0;
        }
```

As exemplified by the ToLower plugin implementation, if the segment “tolow” is recognized, a true value of 1 can be returned, otherwise a value of 0 is returned, indication, that the request does not reference this plugin.

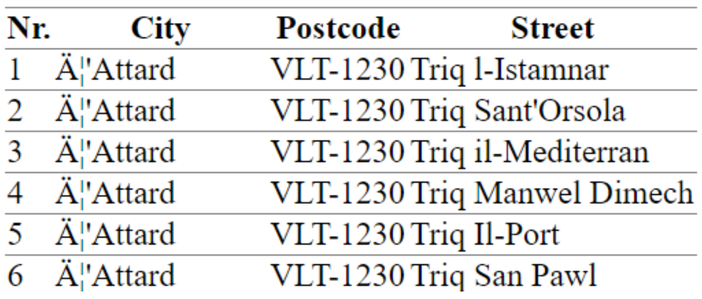
File Plugin

Helps to navigate to the directory of the html file and determine file extensions. In essence it returns the requested file, if the URL path is '/' then index.html is returned. An additional handleExt() method allows for certain file extensions to be recognized, such as PNGs, JPGs and of course HTML files.

```
if (req.Url.Path == "/")
{
    fileInfo = new FileInfo(Path.Combine(dPath + "/wwwroot", "index.html"));
    response.AddHeader("Content-Type", handleExt(req.Url.Extension));
}
```

Navi Plugin

It is possible to search the osm data which is loaded into a database via chosen street, postcode or city. Depending on which parameters are given a relevant SQL command is constructed and executed to select the wanted data from the database. This information is then returned via a streamwriter variable, in which a new html page for the results is constructed.



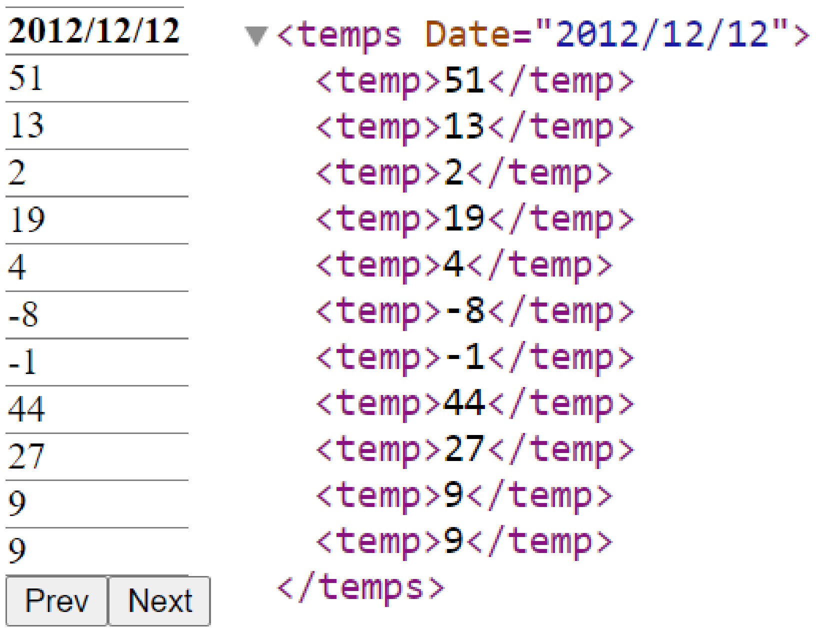
Nr.	City	Postcode	Street
1	Ä¦'Attard	VLT-1230	Triq l-Istamnar
2	Ä¦'Attard	VLT-1230	Triq Sant'Orsola
3	Ä¦'Attard	VLT-1230	Triq il-Mediterran
4	Ä¦'Attard	VLT-1230	Triq Manwel Dimech
5	Ä¦'Attard	VLT-1230	Triq Il-Port
6	Ä¦'Attard	VLT-1230	Triq San Pawl

Figure 4 Navi Search Result

Temperature Plugin

Temperatures may be generated and stored in the database at will and can be called up and even navigated via "next" and "prev", which in turn changes the Parameter in the request. If the search does not find a result, "No result found" will be the output.

Additionally it is possible to display the search result as XML by using the URL without using parameters.



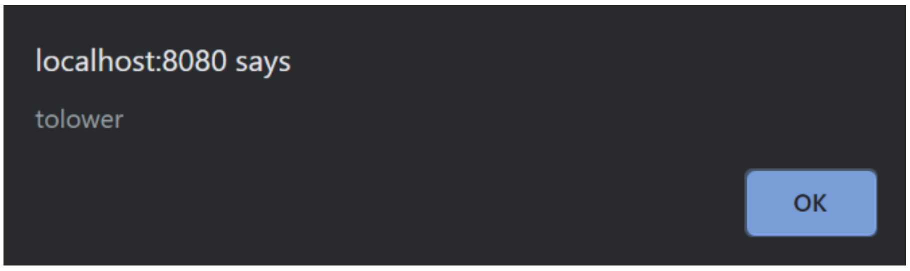
2012/12/12
51
13
2
19
4
-8
-1
44
27
9
9

```
<temps Date="2012/12/12">
  <temp>51</temp>
  <temp>13</temp>
  <temp>2</temp>
  <temp>19</temp>
  <temp>4</temp>
  <temp>-8</temp>
  <temp>-1</temp>
  <temp>44</temp>
  <temp>27</temp>
  <temp>9</temp>
  <temp>9</temp>
</temps>
```

Figure 5 Temperature Result in HTML and XML

ToLower Plugin

The resulting lowercase string will be returned via a Javascript alert.



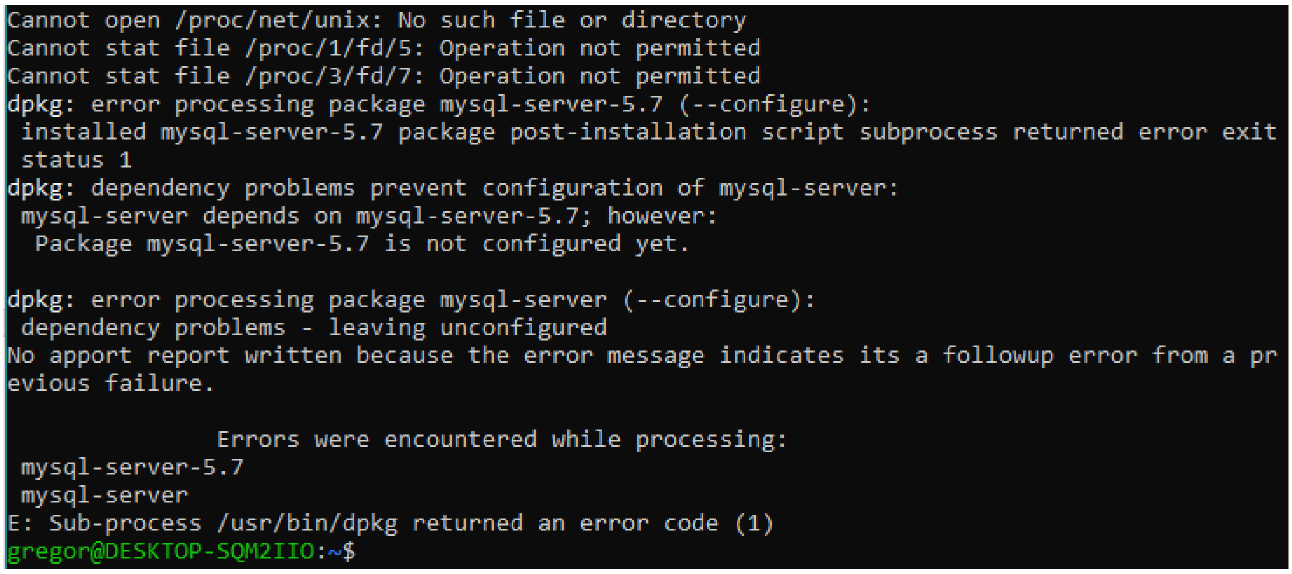
localhost:8080 says

tolower

Figure 6 ToLower Alert Example

Database Connection

Along the way I faced a few challenges trying to establish a database connection. When trying to set up a MYSQL Server I ran into errors using the Ubuntu Subsystem for Win10.



```
Cannot open /proc/net/unix: No such file or directory
Cannot stat file /proc/1/fd/5: Operation not permitted
Cannot stat file /proc/3/fd/7: Operation not permitted
dpkg: error processing package mysql-server-5.7 (--configure):
 installed mysql-server-5.7 package post-installation script subprocess returned error exit
 status 1
dpkg: dependency problems prevent configuration of mysql-server:
 mysql-server depends on mysql-server-5.7; however:
  Package mysql-server-5.7 is not configured yet.

dpkg: error processing package mysql-server (--configure):
 dependency problems - leaving unconfigured
No apport report written because the error message indicates its a followup error from a pr
evious failure.

               Errors were encountered while processing:
 mysql-server-5.7
 mysql-server
E: Sub-process /usr/bin/dpkg returned an error code (1)
gregor@DESKTOP-SQM2IIO:~$
```

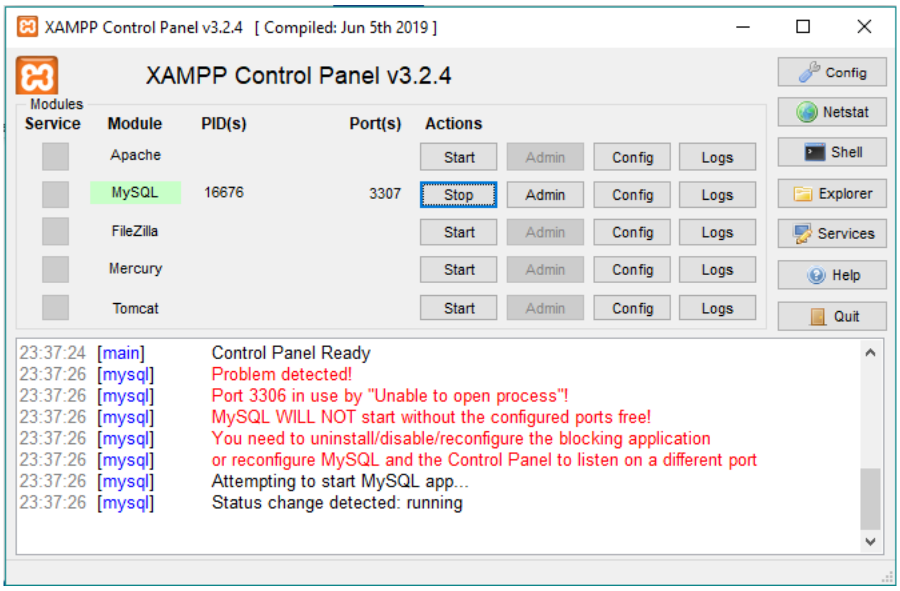
Figure 7 Error installing mysql-server on Linux Subsystem

Attempting a secure installation proved to be unsuccessful as well.

```
sudo mysql_secure_installation
```

```
Error: Can't connect to local MySQL server through socket '/var/run/mysqld/mysqld.s
ock' (2)
```

After hunting down solutions, I ended up just using XAMPP instead as I had never attempted to connect to a Database running on a VirtualBox before.



XAMPP Control Panel v3.2.4 [Compiled: Jun 5th 2019]

XAMPP Control Panel v3.2.4

Service	Module	PID(s)	Port(s)	Actions			
	Apache			Start	Admin	Config	Logs
	MySQL	16676	3307	Stop	Admin	Config	Logs
	FileZilla			Start	Admin	Config	Logs
	Mercury			Start	Admin	Config	Logs
	Tomcat			Start	Admin	Config	Logs

```
23:37:24  [main]    Control Panel Ready
23:37:26  [mysql]   Problem detected!
23:37:26  [mysql]   Port 3306 in use by "Unable to open process"!
23:37:26  [mysql]   MySQL WILL NOT start without the configured ports free!
23:37:26  [mysql]   You need to uninstall/disable/reconfigure the blocking application
23:37:26  [mysql]   or reconfigure MySQL and the Control Panel to listen on a different port
23:37:26  [mysql]   Attempting to start MySQL app...
23:37:26  [mysql]   Status change detected: running
```

Figure 8 XAMPP running on Port 3307

After editing the config file to have the server run on the still available port 3307 I was able to set up my database using SQLWorkbench.
The SQLConnection class itself used a singleton pattern to ensure only one instance of the database would be running at any given time.