

# LabLANreadme

Final year project by Anthony Baudinette and Jarred Paola

<b>Setup summary</b>	<b>2</b>
Client setup	2
Server setup	2
<b>Definitions</b>	<b>2</b>
<b>External dependencies and pre-installed requirements</b>	<b>3</b>
Server dependencies	3
Client dependencies	3
<b>Network settings and static routing (start trouble-shooting here)</b>	<b>4</b>
Server static IP settings	5
Client static IP settings	6
Switch IP settings and forwarding	6
LAN side address configuration	6
WAN side address configuration	7
Port forwarding settings	7
Forwarding visa devices	7
<b>'config.txt' format</b>	<b>8</b>
Client side 'config.txt'	8
Multi-server and 'changeConfigs.py' usage	8
Server side 'config.txt'	9
<b>Client direct scripts functions</b>	<b>9</b>
Low level python functions	9
MATLAB-Python interface	9
High level matlab functions	9
<b>Client MATLAB GUI</b>	<b>10</b>
Setup via installer	10
Modifying source code in MATLAB inbuilt app designer	10
<b>TCP keystings</b>	<b>10</b>

# Setup summary

## Client setup

1. Copy simpleUploadAndCapure into desired folder or run MATLAB GUI installer
2. Ensure dependencies are installed, MATLAB 2019 and Python 3.7
3. Set static IP address as shown in network settings section of this document
4. Set config.txt with the IP and port to match that of the server used

## Server setup

1. Copy 'server' folder into the desired location on the server
2. Install dependencies (see instructions on page 2)
3. Setup static IP addresses (as shown in network settings section of this document)
4. If using switch between client and server configure the port forwarding on the router
5. **Allow python.exe through server firewall**, the location of python.exe changes depending on the options chosen during setup, it is usually in 'users/appdata/local/' or 'Program Files'
6. Set config.txt, the IP and port should match and point the client to the server. The 'first hop' address should be used for the IP, e.g. if the switch is used, it will be the wan side switch IP(192.168.0.1 in network map on page 4), otherwise if the client and server are directly connected the local IP of the server is used. Also ensure the File\_Path setting is an existing folder otherwise uploading files will cause an error.
7. Edit runAll.bat with the location of the soft front panel exe.
8. Create & name a shortcut to runAll.bat

## Definitions

TCP: A plain text transmission protocol used by the server and client to transmit the command strings

Static IP: In most networks an end device e.g. laptop will automatically ask the router it is connected to for an address, robust networks for server hosting such as in labLAN override this automation and set static/unchanging IP address.

Switch/Router: Device which manages/directs the traffic, the flow of the traffic is done by port forwarding and is explained in the network settings section.

Port forwarding: since all clients on the LAN (server/awg/oscilloscope side) appear as the same IP on the WAN (client side) due to NAT on the switch, different tools on the LAN are contacted on different TCP ports. The port forwarding section of the switch settings defines how messages are transmitted to their intended recipient.

VISA: Virtual Instrument Software Architecture simplifies the various serial, ethernet, and on-board bus communication protocols into a uniform interface, VISA software is installed on the server and performs actions as tasked by the client.

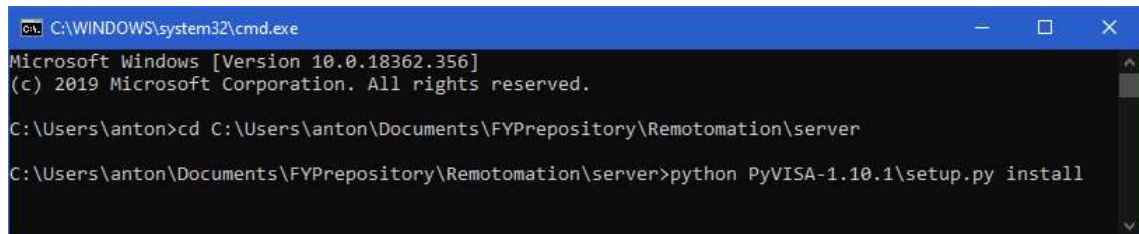
## External dependencies and pre-installed requirements

### Server dependencies

- pyVisa/visa  
This library provides the tools to consolidate the instrument interfaces, the two branches of the server code rely on different libraries the main recommended branch uses python 3 and pyvisa 1.9.1 however for compatibility with existing lab systems server\_py2branch/ uses python 2 and pyvisa 1.5 (note that list devices functionality is not operational through the server\_py2 branch).

Install instructions:

For latest update of pyvisa use pip install or for offline install the latest version at release of labLAN is included in server folder through command line using the instructions as shown below (note: installation requires python installed first)



```
C:\WINDOWS\system32\cmd.exe
Microsoft Windows [Version 10.0.18362.356]
(c) 2019 Microsoft Corporation. All rights reserved.

C:\Users\anton>cd C:\Users\anton\Documents\FYPrepository\Remotomation\server
C:\Users\anton\Documents\FYPrepository\Remotomation\server>python PyVISA-1.10.1\setup.py install
```

- Python  
As with above the python dependency varies with branch the main branch runs on python 3.7.  
The latest version of python is available from <https://www.python.org/downloads/>
- SFP  
The keysight AWG soft front panel GUI is required to be running and in the connected state to communicate with the AWG. note that communication to other devices does not require soft front panel to run however in this case the program should be started through \_\_run\_\_.py instead of run.bat.

### Client dependencies

- MATLAB,  
The low level processes are independent of matlab and therefore the low level functions can be utilised without matlab installed but for ease of use the high level

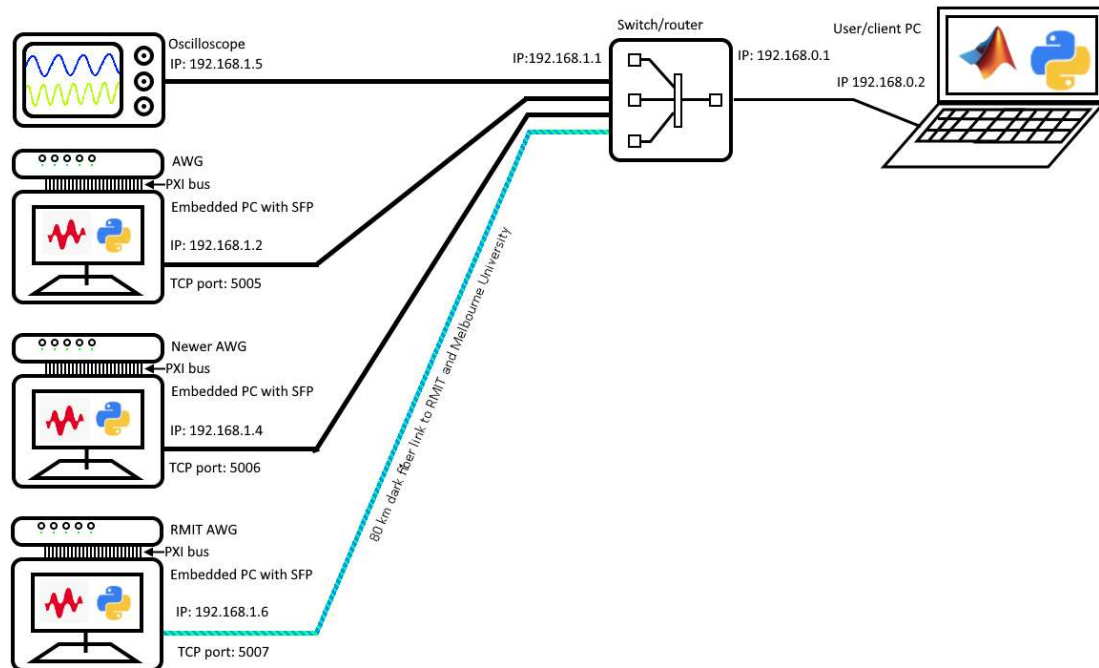
functions are run in MATLAB.

- Python

The fundamental network interface of the program is formed by inbuilt libraries and has been written to be compatible with all versions of python 2 and python 3 with no extra libraries.

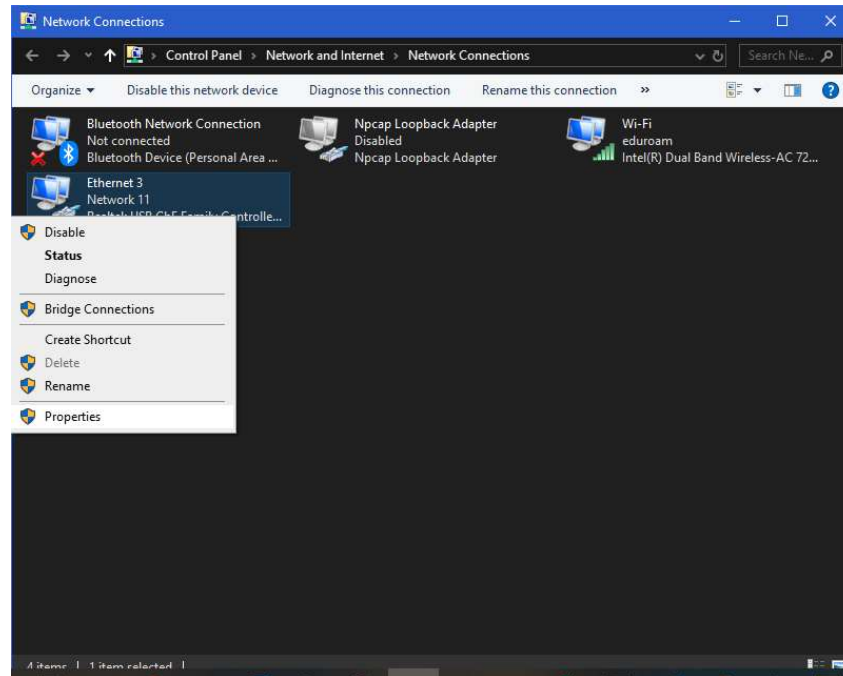
The latest version of python is available from <https://www.python.org/downloads/>

## Network settings and static routing (start trouble-shooting here)

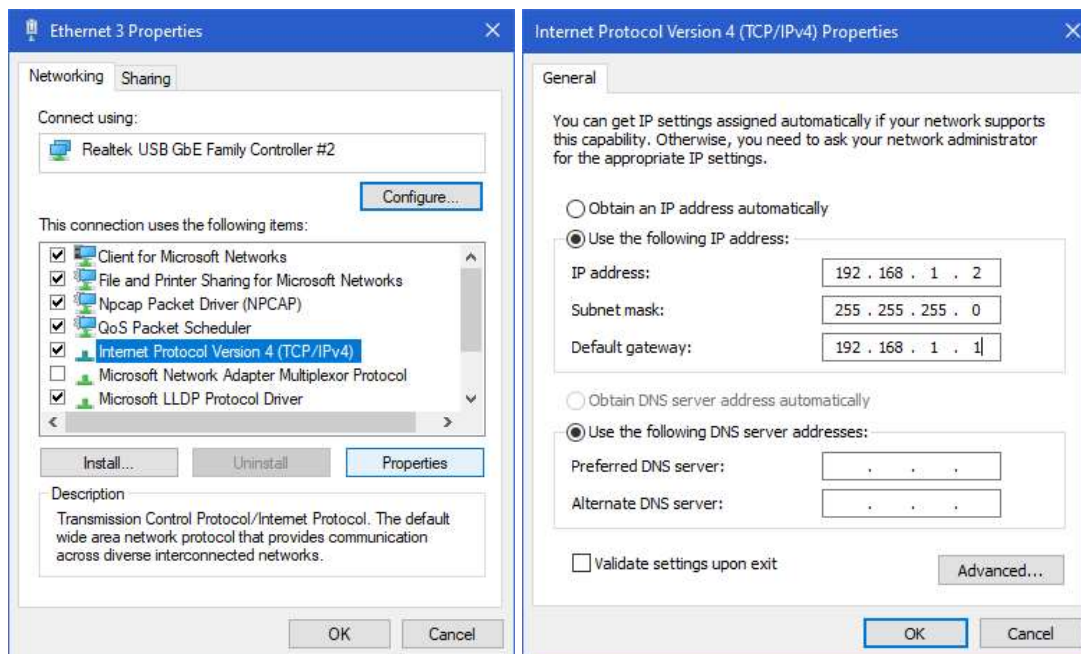


## Server static IP settings

To set a static IP access the network adapters by run > ncpa.cpl or through the control panel to show the window shown below.

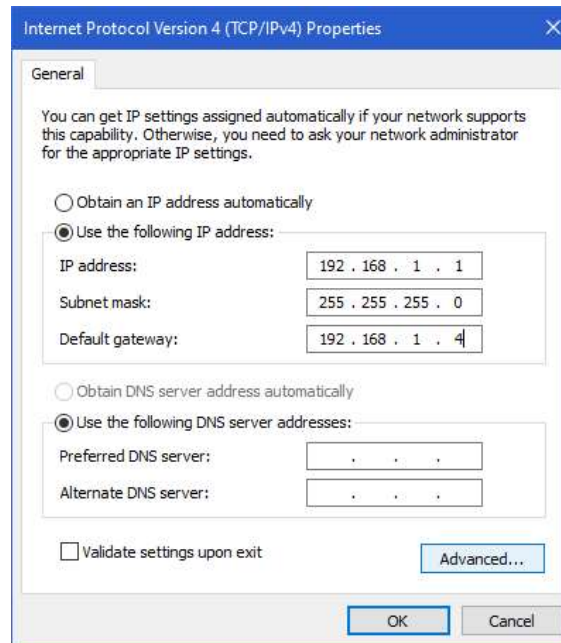


From here select properties and the IPv4 properties to set the local IP address with the default gateway as the address of the switch, shown here is the setting of the AWG address. If the client is connected to the server without the switch between the default gateway would be the client address.



## Client static IP settings

Follow the same run > ncpa.cpl > ethernet properties > IPv4 > properties, as shown in server static IP settings above. Once again set IP address to the local address and default gateway to the switch address (192.168.0.1). The window below shows the client set to be connected to the newer AWG (from network map above) without the switch between them.



Note: setting the ethernet driver to a static IP will prevent it from connecting to another network e.g. UNI internet through the same adapter. Windows trouble shoot wizard will reset this setting to automatic but will need to be re-configured before use again.

## Switch IP settings and forwarding

Since the method of applying these settings differs between router manufacturers this section will focus on the settings applied rather than the method of using the switch. The configuration panel of the router can be accessed by typing the default gateway into a browser window.

### LAN side address configuration

Network Setup	
Router IP	Local IP Address: 192 . 168 . 1 . 1
	Subnet Mask: 255.255.255.0 ▼
Network Address Server Settings (DHCP)	DHCP Server: <input checked="" type="radio"/> Enable <input type="radio"/> Disable
	Starting IP Address: 192.168.1.100

The DHCP controls the automatic IP setting that the previous section bypasses, the importance of this setting is just to prevent interference with the static IPs manually set.

WAN side address configuration

Internet Setup

Internet Connection Type

Static IP

Internet IP Address:19216801

Subnet Mask:2552552550

Gateway:19216802

This setting is the opposite of the client static setting (i.e. swapped IP and gateway addresses).

Port forwarding settings

Firstly one rule needs to be set for each server needs to have its port (as set by server/config.txt) forwarded to its static IP.

External Port	Internal Port	Protocol	To IP Address	Enable
---	---	---	192.168.1.0	<input type="checkbox"/>
---	---	---	192.168.1.0	<input type="checkbox"/>
---	---	---	192.168.1.0	<input type="checkbox"/>
---	---	---	192.168.1.0	<input type="checkbox"/>
---	---	---	192.168.1.0	<input type="checkbox"/>
5005	5005	Both ▾	192.168.1.2	<input checked="" type="checkbox"/>
5006	5006	Both ▾	192.168.1.4	<input checked="" type="checkbox"/>

Forwarding visa devices

While a raspberry pi virtual here solution is recommended for the future the immediate solution to forwarding visa devices through the switch is to forward the range of ports that are not used in servers to the visa devices IP, the main limitation of this solution is that it is only able to forward one visa device onto the WAN connection.

There are two ways to achieve this either setting the device as DMZ host, or forwarding the entire port range the device IP. an example of each is shown below.

DMZ

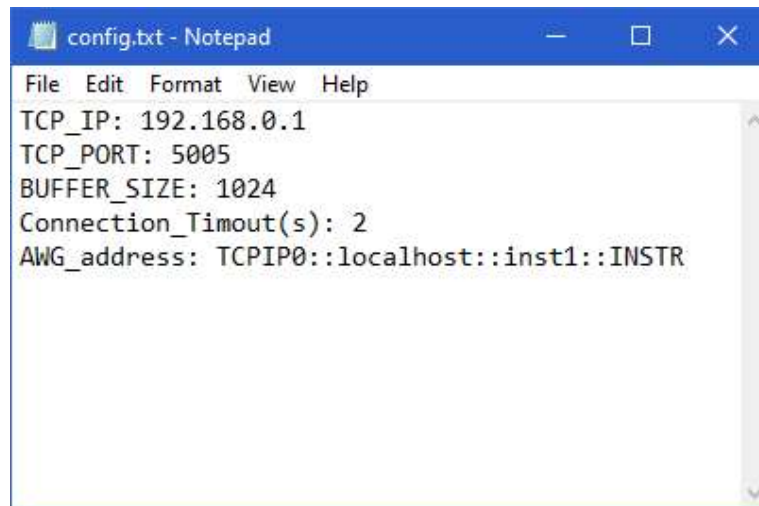
☒ Enable ☐ Disable

DMZ Host IP Address :192.168.1.5

Port Range					
Application	Start	End	Protocol	IP Address	Enable
	1	to 60000	Both ▼	192.168.1.5	<input checked="" type="checkbox"/>

## ‘config.txt’ format

### Client side ‘config.txt’



```

File Edit Format View Help
TCP_IP: 192.168.0.1
TCP_PORT: 5005
BUFFER_SIZE: 1024
Connection_Timeout(s): 2
AWG_address: TCPIP0::localhost::inst1::INSTR

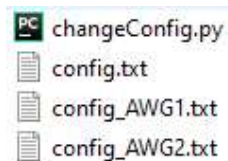
```

The IP address and port set in client/config.txt point the python functions to the server, here the IP points to the switches address and because of the port specified the switch sends the connection onto the desired server. BUFFER\_SIZE and ConnectionTimeout(s) are used by the TCP library but shouldn't need changing. AWG\_address specifies the visa address to be used when the python functions are called with the argument of 'AWG' in place of visa address.

N.B. the config.txt files on both client and server side should both point toward the same IP address (the server/AWG) when connecting the server and client directly.

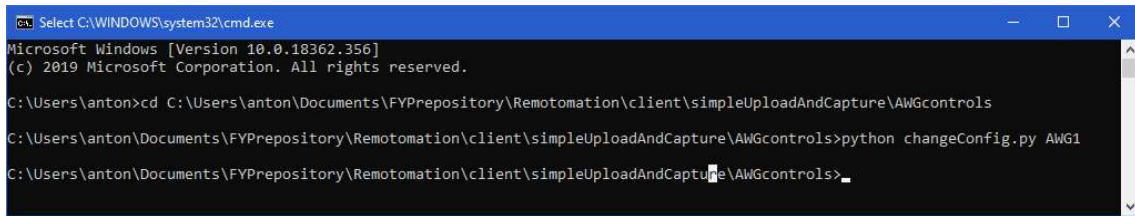
## Multi-server and ‘changeConfigs.py’ usage

To support multiple servers the client can change between servers by changing config.txt. As shown below there is a config\_\*\*\*\*\*.txt for each server.



To change between servers changeConfig.py overwrites the existing config.txt with the contents of that of the other txt files. The window below shows config.txt being overwritten with config\_AWG1.txt using the command `python changeConfig.py AWG1`.

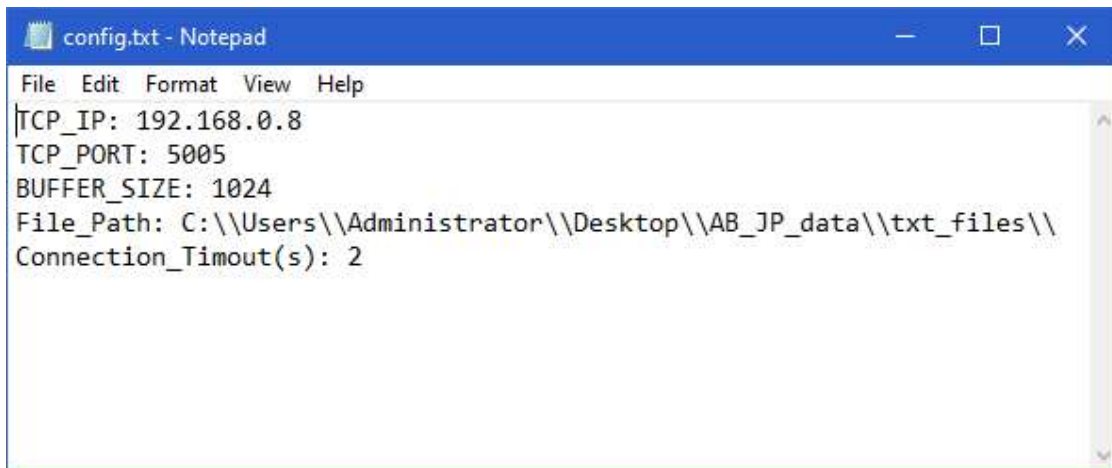




```
Select C:\WINDOWS\system32\cmd.exe
Microsoft Windows [Version 10.0.18362.356]
(c) 2019 Microsoft Corporation. All rights reserved.

C:\Users\anton>cd C:\Users\anton\Documents\FYPrepository\Remotomation\client\simpleUploadAndCapture\AWGcontrols
C:\Users\anton\Documents\FYPrepository\Remotomation\client\simpleUploadAndCapture\AWGcontrols>python changeConfig.py AWG1
C:\Users\anton\Documents\FYPrepository\Remotomation\client\simpleUploadAndCapture\AWGcontrols>
```

## Server side 'config.txt'



```
config.txt - Notepad
File Edit Format View Help
TCP_IP: 192.168.0.8
TCP_PORT: 5005
BUFFER_SIZE: 1024
File_Path: C:\\Users\\Administrator\\Desktop\\AB_JP_data\\txt_files\\
Connection_Timout(s): 2
```

The IP and port specifies what the server listens on to wait for a connection and must be the same as the servers static IP setting, note removing or commenting TCP\_IP will default the IP to the PC's own address but may cause issues if multiple network adapters are enabled. As with client BUFFER\_SIZE and Connection\_Timout(s) are used by the TCP library. File\_Path must point to an existing folder and is used for storing files to be uploaded to the AWG, the \ characters are replaced with \\ due to the decoding of strings in the python code, e.g. \n becomes a newline character and \\ becomes \.

## Client direct scripts functions

### Low level python functions

The connection between the server and client is run on a set of python scripts, these scripts can be either ran standalone by calling with arguments as shown in the CMD window below or can be imported as python functions. If the client receives a bad response or the server indicates an error the python functions return a string err: followed by a description of the error.

Importing the functions into another python script:

```
1 import write
2 import query
3
4 write.write("AWG", ":INIT:IMM")
```

Running manually from a CMD window, this formatting is also used in the MATLAB-python interface shown below:

```
C:\Users\anton\Documents\FYPRepository\Remotomation\client\simpleUploadAndCapture\AWGcontrols>python write.py AWG :INIT:IMM
```

## MATLAB-Python interface

MATLAB and python communicate by passing strings through the `system()` function, In the example shown below “AWG” and “:INIT:IMM” are python input arguments and on the server responding with a success message system will return with `status=0` and `cmdOut=""`.

Calling scripts from MATLAB with `system()`:

```
16 -  
17 - [status,cmdOut] = system("python write.py AWG :INIT:IMM");  
18 - if status~=0  
19 -     warning("system error: "+cmdOut)  
20 - end
```

## High level matlab functions

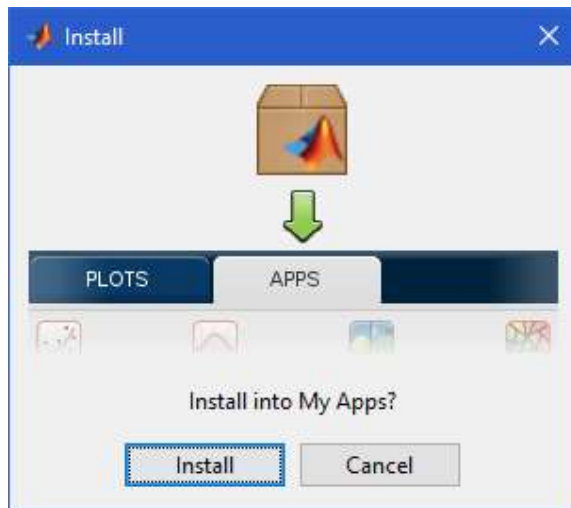
To further abstract the visa interface and specific instructions the user can call a set of MATLAB functions with the command strings pre-programmed. e.g. calling `AWGrun()` sends the command to run the outputs as shown below:

```
1  function AWGrun()  
2  % starts signal generation on enabled outputs  
3  % Usage:  
4  %   AWGrun();  
5  % Inputs:  
6  %   none  
7  % Outputs:  
8  %   none  
9  
10  
11 - AWGadd = "AWG";  
12 - Command = ":INIT:IMM";  
13  
14  % in the form of ">python (python_command) (device) (device_command)"  
15 - cmdStr = "python write.py " + AWGadd + " " + Command;  
16  
17 - [status,cmdOut] = system(cmdStr);  
18 - if status~=0  
19 -     warning("system error: "+cmdOut)  
20 - end  
21 - end
```

# Client MATLAB GUI

## Setup via installer

The GUI can be installed into MATLAB by running '*client/GUI/labLAN.mlappinstall*'. Note the app by default runs the scripts from the MATLAB installed apps folder, if the location app is run without the python low level function files present the app will need to be pointed to them.



## Modifying source code in MATLAB inbuilt app designer

The MATLAB app designer can be started by `>>appdesigner` or selecting apps>design App, note: the app cannot be modified by some earlier versions of MATLAB.



## TCP keystreams

The strings received by the server and client are parsed as an array of arguments separated by ", " characters. The valid keystreams are shown below in the case of some responses such as "visa, write, instID, ..." extra array elements beyond the number expected are recombined into one argument.

## Client to server commands

arg[0]	arg[1]	arg[2]	arg[3]	arg[4]	description
ping					
file	txRequest	Destination FileName			Upon receiving "file, rxReady" client is to send the .txt contents to be saved onto the server
file	upload	FileName	ChannelNumber	InstID	Server instructs the AWG to pull the .txt file sent to the server using the above function
visa	write	instID	message		
visa	query	instID	message		
visa	read	instID			
visa	listResources				

## Server to client responses

arg[0]	arg[1]	arg[2]	arg[3]	description
pong				
err	Unrecognised_cmd / command_error			
file	rxReady			
file	writeResult	errorFlag	errorDescription (optional)	
file	uploadResult	errorFlag	errorDescription (optional)	
visa	writeResult	errorFlag	errorDescription (optional)	
visa	readResult	errorFlag	errorDescription / instrument response	