API - IMPLEMENTED

data output("DATA");

host

PC

0

Ip output($host->getIP());

lease output($host->getExpireDate());

band output("Deprecated, please use stats.band");

mac output($host->getMac());

name output($host->getName()); || output("Un-named PC");

stats output("Please Login");

redirect

Ottawa output("Redirection to Ottawa Complete");

Oshawa output("Redirection to Oshawa Complete");

currentRedirect output($network->getDestName());

theme output(“”);

account

login output('<i class="fa fa-thumbs-up"></i> Login Successful');

theme output(getUserTheme());

set output ("Theme has been updated");

numberOfPC output(sizeof($userComputer));

PC

#

Ip output($userComputer[$num]->getIP());

Mac output($userComputer[$num]->getMac());

Lease output($userComputer[$num]->getExpireDate());

Band output("Deprecated Please use stats.band");

Name output($userComputer[$num]->getName()); || output("PC ".($num + 1 ));

Set

<value> output('<i class="fa fa-thumbs-up"></i> Computer Name Was Successfully Updated');

removePCfromUser output('<i class="fa fa-thumbs-up"></i> This PC has been removed from your account');

statsmultiCache

MTU output($PCstats['MTU']);

Band output($PCstats['BW']);

Delay output($PCstats['delay']);

reliability output($PCstats['reliability']);

transmitLoad output($PCstats['transmitLoad']);

receiveLoad output($PCstats['receiveLoad']);

duplex output($PCstats['duplex']);

speed output($PCstats['speed']);

5minuteInputBS output($PCstats['5minuteInputBS']);

5minuteInputPS output($PCstats['5minuteInputPS']);

5minuteOutputBS output($PCstats['5minuteOutputBS']);

5minuteOutputPS output($PCstats['5minuteOutputPS']);

inputByte output($PCstats['inputByte']);

outputByte output($PCstats['outputByte']);

inputPacket output($PCstats['inputPacket']);

outputPacket output($PCstats['outputPacket']);

description output($PCstats['description']);

redirect

Ottawa output("Redirection to Ottawa Complete"); || output("No Redirection is Currently Being Applied");

Oshawa output("Redirection to Oshawa Complete"); || output("No Redirection is Currently Being Applied");

currentRedirect output("Device is Offline"); || output($network->getDestName()); || output($db->getRedirectLocation($userComputer[$num]->getMac()));

addPCtoUser output('<i class="fa fa-thumbs-up"></i> This PC has been added to your account');

removePCfromUser output('<i class="fa fa-exclamation-triangle"></i><strong></strong> This PC has been removed from your account');

register output("Missing Arguments"); || output("User Created Successfully"); || output("ERROR: Unable to Create User");

# FILES

# File myscripts.js

function **request**(cmd, id)

**Returns:** Nothing **Arguments:** String, String

**Notes:** Adds command and ID to a queuing system which indicates the command and HTML ID to store the response.

**Use:** table population, visual return of data

function **requestSp**(cmd, fct)

**Returns:** Nothing **Arguments:** String, String

**Notes:** Adds the command and function to a queuing system which indicates the command and function to call with the data. Multiple commands going to a function are indicated in the respected function calls by an opening brace and closing brace within the function name. IE commands with a function call “FUNCTION:functionName” only have 1 command associated with them, whereas multi-function calls are indicated by a square bracet. The first function call is “[FUNCTION:functionName”, the last “FUNCTION]:functionName”, and inbetween “FUNCTION:functionName”.

**Use:** Login/Logout, Setting variables

function **publish** (responseText)

**Returns:** Nothing **Arguments:** String

**Notes:** Takings the responding AJAX request and directs the output according to the queuing array as set within requests and reqest functions.

**Use:** Login/Logout, Setting variables

function **sendRequest** ()

**Returns:** Nothing **Arguments:** Nothing

**Notes:** Takes the request queue, send the commands and waits a predefined timeout value, if no response has been received within the timeout value, the commands are resent. This continues until a response is receive for every command (Publish removes them from the queue once it has finished processing the data).

**Use:** Queueing/Sending AJAX requests

function **pageLoad** ()

**Returns:** Nothing **Arguments:** None

**Notes:** currently not used. Used to be used

**Use:** call the page load

function **loginFunction**()

**Returns:** Nothing **Arguments:** None

**Notes:** Checks for username and password sends ajax request. Response sent to loginVerify().

**Use:** Starts the login process.

function **loginVerify**(login)

**Returns:** Nothing **Arguments:** AJAX response array

**Notes:** Verifies credentials and sets cookies, enables buttons.

**Use:** Do not call, used by loginFunction()

function **logoutFunction**()

**Returns:** Nothing **Arguments:** None

**Notes:** sets values to their default

**Use:** wipes log on data

function **register**()

**Returns:** Nothing **Arguments:** None

**Notes:** Allows for user registration. Grabs text input and sends a register request. Response is sent to registerVerify();

**Use:** user registration

function **registerVerify**()

**Returns:** Nothing **Arguments:** None

**Notes:** confirms user registration

**Use:**  Do not call, used by register()

function **removeComp**()

**Returns:** Nothing **Arguments:** None

**Notes:** Removes currently selected PC from active directory. Sends ajax request

**Use:** remove a pc

function **addComp**()

**Returns:** Nothing **Arguments:** None

**Notes:** Sends ajax request for addComp.

**Use:** add a PC

function **redirectFunction**()

**Returns:** Nothing **Arguments:** None

**Notes:** Sends ajax request for redirection

**Use:** Redirect current selected PC

function **tableLoadComplete**()

**Returns:** Nothing **Arguments:** None

**Notes:** called by tablePop()

**Use:** watches for user click for table selection

function **tablePop**()

**Returns:** Nothing **Arguments:** None

**Notes:** Populates the table according the AJAX repsonses

**Use:** Populates tables

function **detailPop**()

**Returns:** Nothing **Arguments:** None

**Notes:** called by tableLoadComplete();

**Use:** Populates detail table.

function **compName**()

**Returns:** Nothing **Arguments:** None

**Notes:** Send an AJEX request to update the current PC name

**Use:** Set the current PC name

function **setCookie**(cname,cvalue,exdays)

**Returns:**  **Arguments:** string, string, integer

**Notes:** takes a variable name, variable value and cookie expiration time in days.

**Use:** create a cookie

function **getCookie**(cname)

**Returns:** Nothing **Arguments:** string

**Notes:**

**Use:** gets the cookie of the specified name

function **checkCookie**()

**Returns:** Boolean **Arguments:** None

**Notes:** Checks for cookies and populates variables accordingly.

**Use:** check for cookies

# File interface.php

function **clean\_input**($data)

**Returns:** String **Arguments:** String

**Notes:** Sanitizes input.

**Use:** Sanitizes input.

function **getGateway**()

**Returns:** Array of String **Arguments:** None

**Notes:** Returns the gateway, in this case there is only 1 gateway and so it is statically assigned.

**Use:** Obtain gateway information.

function **getComputerArray**()

**Returns:** Array of Computer Object **Arguments:** None

**Notes:** Checked if cached information is available, if so it is used. If not a new connection is estabilished and the newly fetched information is stored in cache.

**Use:** Obtain current DHCP information

function **updateComputerGroup**($fcnComputerArray)

**Returns:** Nothing **Arguments:** None

**Notes:** Ensures Active Directory’s groups are in sync with the number of PCs on the network. IE a PC isn’t missing its group.

**Use:** Keeps AD synced with the DHCP table

function **getUserComputer**()

**Returns:** Array of Computer Object **Arguments:** None

**Notes:** Obtains the computer objects based on the AD groups the user belongs too. Each group is associated with a PC.

**Use:** Obtain a list of the users Devices.

function **getUserGroup**()

**Returns:** Array of String **Arguments:** None

**Notes:** Fetches group information from AD and stores it in its corresponding area.

**Use:** Obtains group information from AD

function **addUserToGroup**($fcnGroup)

**Returns:** Nothing **Arguments:** String

**Notes:** Adds a user to an AD group

**Use:** Adds a user to an AD group

function **removeUserFromGroup**($fcnGroup)

**Returns:** None **Arguments:** String

**Notes:** Removes a user from the specified group in AD

**Use:** Removes a user from a specified group.

function **addPCtoUser**()

**Returns:** Nothing **Arguments:** None

**Notes:** Adds the host PC to the users account.

**Use:** Adds the device to the users account.

function **removePCfromUser**($fcnComputer)

**Returns:** Nothing **Arguments:** Computer Object

**Notes:** Removes a specific PC from the users account.

**Use:** Removes a specific PC from the users account.

function **setUserTheme** ($fcnTheme)

**Returns:** Nothing **Arguments:** String

**Notes:** Sets the users theme group to the corresponding theme.

**Use:** Saves the users theme in AD.

function **getUserTheme** ()

**Returns:** String **Arguments:** None

**Notes:** Retrieves the users theme from

**Use:** Returns the users theme

function **getStats()**

**Returns:** Array of Strings **Arguments:**

**Notes:** Returns the array of output retrieved from the IOS device for the host PC.

**Use:** Obtains a list of stats from the gateway regarding a particular PC.

function **getPCStats**($fcnComputer, $fcnGatewayInfo)

**Returns:** Array of Strings **Arguments:** Computer Object, Array of Strings

**Notes:** Retrieves statistical information regarding a particular host. The function first checks if the information is cached, if so the cached information is retrieved. Otherwise the gateway is contacted.

**Use:** Obtain stats for a particular PC

function **getMultiPCStats**($fcnComputer, $fcnGatewayInfo)

**Returns:** Nothing **Arguments:** Array of Computer, Array of String

**Notes:** Takes a list of PCs and obtains statistical information from the gateway for each PC supplied in the array. All information is then cached for the getPCStats to access.

**Use:** Cache multiple PC ahead of the request

function **commaFilter**($value)

**Returns:** String **Arguments:** String

**Notes:** Filters out special characters and commas from the provided input.

**Use:** Cleans the input removing special characters and commas.

function **cleanCache**()

**Returns:** Nothing **Arguments:** None

**Notes:** Calls the DB clean cache function, which deletes rows that are older than a predefined timer.

**Use:** Remove stale entries from cache.

function **cleanCacheMac**($mac)

**Returns:** Nothing **Arguments:** String

**Notes:** Removes cached entries relating to a particular host.

**Use:** Clears cached entries relating to a particular host.

function **output**($value)

**Returns:** Nothing **Arguments:** String

**Notes:** Adds the input to the end of the output array, which is used to print information after all code has completed execution

**Use:** Adds the function input to the file to output

function **redirect**($fcnDestination, $pc)

**Returns:** Boolean **Arguments:** String, Computer Object

**Notes:** Setups the redirection for a particular PC and particular destination. Cache is first checked, if an entry is found, it is loaded. If no entry is found, a new redirection object is created. The redirection operations are then preformed then the networkRedirect object is saved to MySQL.

**Use:** Redirects the PC to a particular gateway.

CLASSES

# class computer – File: computer.php

function **computer** ($fcnIP , $fcnMac , $fcnExpireTimeStamp)

**Return**: Nothing **Arguments**: String, String, String

**Notes**: Computer class constructor

**Use**: Object setup

function **setIP** ($fcnIP)

**Return**: None **Arguments**: String

**Notes**: Sets the IP of the computer class (does not update MySQL).

**Use**: Update Information.

function **setMac** ($fcnMac)

**Return**: None **Arguments**: String

**Notes**: Set Mac to allow setting of the variable.

**Use**: Loading/Caching uses. Should not be used otherwise (unpredictable results)

function **setExpireTimeStamp** ($fcnExpireTimeStamp)

**Return**: None **Arguments**: String

**Notes**: Set ExpireTimeStamp variable

**Use**: Updating information

function **setName** ($fcnName)

**Return**: None **Arguments**: String

**Notes**: Sets the name variable

**Use**: Allow users to identify PCs based on names

function **setUser** ($fcnUser)

**Return**: None **Arguments**: String

**Notes**: Sets the User variable

**Use**: (not used yet) Device ownership would be possible

function **updateIP** ($fcnIP, $fcnExpireTimeStamp) - Public

**Return**: None **Arguments**: String

**Notes**: Updates both the IP and Expiration timer

**Use**: (not used yet)

function **getExpireTimeStamp** () – Public

**Return**: string of digits **Arguments**: None

**Notes**: Return the timestamp of the IP lease expiration

**Use**: To obtain Timestamp information, mostly used with the database class for writing to the database

function **getExpireDate** ()

**Returns**: string **Arguments**: None

**Notes**: Returns the date in H:M:S D/M/Y based on the predefined timestamp

**Use**: To obtain human readable date and time information

function **getUser**()

**Returns**: string **Arguments**: None

**Notes**: Returns username of device owner as seen in SQL DHCP table.

**Use**: (not used yet)

function **getName**()

**Returns**: string **Arguments**: None

**Notes**: Returns name of device, string

**Use**: Obtains the name of the device as seen in SQL DHCP table

function **getIP**()

**Returns**: String **Arguments**: None

**Notes**: Returns the ip of the device, string

**Uses**: Obtains the IP of the device

function **getMac**()

**Returns**: String **Arguments**: None

**Notes**: returns the MAC address of the device, string (all lower case, no dividers, IE 00ab1d50c00b00e0)

**Uses**: Obtains the MAC addresses of the device as seen in SQL DHCP table (used to identify the device)

function **PRINTCOMPUTER** ()

**Returns**: Nothing **Arguments**: None

**Notes**: Echo’s PC info (name, IP, Mac, ExpireDate, ExpireTS. Somewhat formatted)

**Uses**: Debugging/Development

function **getComputer**()

**Returns**: Array **Arguments**: None

**Notes**: Returns an array of the computer’s properties using the following indexes: Name, IP, MAC, ExpireDate, ExpireTimeStamp. All case sensitive

**Uses**: Caching

# class IOS\_device – File IOS\_device.php

function **IOS\_device** ($fcnIP, $fcnUsername, $fcnPassword, $fcnType, $fcnDHCP)

**Returns**: Array **Arguments**: None

**Notes**: Constructor, type is either IOS\_Switch, or IOS\_Router, DHCP indicates if the device is a DHCP server. Implementation of dynamic scanning of DHCP servers will follow. / Dynamic host to GW association

**Uses**: Class constructor

function **setPort** ($fcnPort)

**Returns**: None **Arguments**: Integer

**Notes**: Sets the SSH port on the SSH connection. Optional, default is 22.

**Uses**: Flexible port options/restoring from cache

function **getPort** ()

**Returns**: Integer **Arguments**: None

**Notes**: Returns the port of the SSH connection

**Uses**: Obtains the SSH port

function **setType** ($fcnType)

**Returns**: Nothing **Arguments**: String

**Notes**: sets the type variable, either IOS\_Switch or IOS\_Router. This is done as there is a slight difference in output formatting between the two platforms

**Uses**: Sets the device type

function **getType** ()

**Returns:** String **Arguments**: String

**Returns** device Type (IOS\_Switch or IOS\_Router)

**Uses**: Obtain the device type

function **setDHCP** ($fcnDHCP)

**Returns**: Nothing **Arguments**: String

**Notes**: sets the DHCP variable. True indicates the device is a DHCP server, false indicates the device is a transit device.

**Uses**: Sets DHCP variable, true/false 1/0

function **getDHCP** ()

**Returns**: Boolean **Arguments**: String

**Notes**: Returns the value of the DHCP variable

**Uses**: Return DHCP variable, true/false

function **connect** ()

**Returns**: Boolean **Arguments**: None

**Notes**: Calls the connect function of the sshconnection of the class (ssh class, connect)

**Uses**: Connects to the device

function **disconnect** ()

NOT CODE YET

function **getUsername** ()

**Returns**: String **Arguments**: None

**Notes**: Returns the username of the object

**Uses**: Obtain the username of the device, Caching

function **setUsername** ($fctUsername)

**Returns**: Nothing **Arguments**: None

**Notes**: Sets the username variable of the object. Should not be used

**Uses**: Sets the username of the device, used in the constructor

function **getPassword** ()

**Returns**: String **Arguments**: None

**Notes**: Returns the password variable of the object.

**Uses**: Obtains the username of the device, Caching

function **setPassword** ($fctUsername)

**Returns**: Nothing **Arguments**: String

**Notes**: Sets the password variable of the object

**Uses**: Sets the password, used in the constructor

function **setRID** ($fctRID)

**Returns**: Nothing **Arguments**: String

**Notes**: sets the RID, the main interface that will be used to identify the device, in addition, this IP Address will be used to communicate with the device.

**Uses**: Sets the RID, used in the constructor

function **getRID**()

**Returns**: String **Arguments**: None

**Notes**: Returns the RID, the main interface that will used to identify the device

**Uses**: Obtains the RIP/IP of the device, used for network redirection

function **addIP** ($fctIP)

**Returns**: Nothing **Arguments**: String

**Notes**: Adds an IP address to the list of IP Addresses assigned to each device. This list is used within the Network redirect object.

**Uses**: Used within the getInterface function

function **getIP** ()

**Returns**: Array of Strings **Arguments**: None

**Notes**: Returns array of IP addresses assigned to the device.

**Uses**: Used to obtain all the IP addresses that a device posses

function **getName** () //Does this return the SQL hostname (TH-Switch) or Location as the name

**Returns**: String **Arguments**: None

**Notes**: Returns the name of the device

**Uses**: Get the device name.

function **setName**($fcnName)

**Returns**: Nothing **Arguments**: String

**Notes**: Sets the hostname variable

**Uses**: Sets the name of the device, used within the constructor

function **sendCMD** ($cmd)

**Returns**: String **Arguments**: Array of Strings

**Notes**: Outdated, migrate to new sendMultiCMD. This function uses ssh2\_exec, and needs to create a new SSH connection for each command issued.

**Uses**: Issues commands to the device, and returns the output in a single string

function **sendMultiCMD**($cmd)

**Returns**: Array of Strings **Arguments**: Array of Strings

**Notes**: Sends multiple commands to a device over a single SSH connection, uses ssh2\_shell. More efficient then sendCMD

**Uses**: Communicate with the device

function **initilizeDevice** ()

NOT IMPLEMENTED CORRECTLY DO NOT USE. Future use: will allow for dynamic provisioning of type, ip, dhcp, and name variable by pulling the running configuration.

function **getClient** ()

**Returns**: Nothing **Arguments**: None

**Notes**: Queries the device for DHCP bindings and returns an array of computers who have been assigned an IP address by this device. Querying the devices is costly in time and should be done sparingly

**Uses**: Pulls information from the DHCP server allowing for updating of IP/MAC maps and adding new hosts to the databases

function **getStat**($fctComputer)

**Returns**: Array of port statistics **Arguments**: Computer Object

**Notes**: Returns an array with the following indexes: MTU, BW, delay, reliability, transmitLoad, receiveLoad, duplex, speed, 5minuteInputBS, 5minuteInputPS, 5minuteOutputBS, 5minuteOutputPS, inputByte, inputPacket, outputPacket, outputByte and description. This should be run sparingly as it is costly in time.

**Use**: Retrieves statistics about a particular host from the switch

function **getStatMultiComp** ($fcnComputers)

**Returns**: 2 dimensional Array of port statistics **Arguments**: Array of Computer Object

**Notes**: Returns a 2 dimensional array of statistics with the following indexes: MTU, BW, delay, reliability, transmitLoad, receiveLoad, duplex, speed, 5minuteInputBS, 5minuteInputPS, 5minuteOutputBS, 5minuteOutputPS, inputByte, inputPacket, outputPacket, outputByte and description. This should be run sparingly as it is costly in time.

**Use**: Retrieves statistics about a particular host from the switch

function **getOutput**($fctOutput)

**Returns** String **Arguments**: Strings

**Notes**: Outdated function. GetStats and getstatsMultiComp return the arrays that are accessible through this function.

**Use**: Retrieves values in the output array, using the argument as an index.

function **getInterface**()

**Returns**: Nothing **Arguments**: None

**Notes**: Queries the device and obtains IP addresses that are assigned to the device.

**Use**: Obtains associated with the device for use in networkRedirect path

function **getNextHop**($fcnIP)

**Returns**: String **Arguments**: String

**Notes**: Queries the device for the next hop IP Address of the argument address.

**Use**: Backtracking/adding devices that need to be updated to redirect a user’s traffic.

# class database – File database.php

        function **database** ($fctUsername, $fctPassword, $fctAddress, $fctDbName)

**Returns** Nothing                                                      **Arguments**: String, String, String, String

**Notes**: Creates the DB class and initializes the connection

**Use**: Class constructor

        function **connect** ()

**Returns**: Nothing **Arguments**: None

**Notes**: Connects to the mySQL server. If the connection is unsuccessful a die command is issued.

**Use**: Called during the database class set, should not be called outside of the database class

        function **multiQuery** ($fctMultiQuery)

**Returns:** Array of results                                              **Arguments**: Array of Strings

**Notes**: Calls the query function in a looped fashion, taking the SQL result objects and pushing them to the end of an array

**Use**: Issuing a series of SQL commands to the database, typically only called from within the class.

function **query** ($fctQuery)

**Returns:** SQL results object                                      **Arguments**: String

**Notes**: Issues the provided command to the SQL database returning an SQL result object

**Use**: Issues SQL commands to the database, typically the multiQuery function is called.

function **close** ()

**Returns:** None  **Arguments**: None

**Notes**: Closes the SQL connection

**Use**: Closes the SQL connection

function **PRINTDB**()

**Returns:** String **Arguments**: None

**Notes**: Prints/Returns DB IP address

**Use**: Development/Debugging

function **addComputer** ($fctComputer)

**Returns:** SQL Result object  **Arguments**: Computer Object

**Notes**: Adds a computer object to the SQL DHCP table, no checks are performed. Checks should be performed by the caller.

**Use**: Called in the updateComputers function. Should not be called outside the class

function **updateComputers** ($fctComputers)

**Returns:** Array of Computer Objects **Arguments:** Array of Computer Object

**Notes:** Algorithm to detect and update changes within the network hosts. Firsts completes is update of the DHCP table, once complete, the function grabs additional information that is not yet present in the provided array (IE. device Name) from the DHCP table. This information is then added to the array. The updated array is then returned.

**Use:** Updates/returns a complete version of hosts with a valid DHCP binding.

function **getComputerMAC** ($fctMac)

**Returns:** Computer Object **Arguments:** String

**Notes:** Searches the SQL database for a host matching the MAC address provided. A Computer object is then initialized/populated based on the information found within the SQL table

**Use:** Obtain the Computer Object of a host based on its MAC address.

function **getComputerIP** ($fctIP)

**Returns:** Computer Object **Arguments:** String

**Notes:** Searches the SQL database for a host matching the IP addresses provided. A computer object is then initialized/populated based on the information found within the SQL table.

**Use:** Obtain the Computer Object of a host based on its IP address.

function **updateComputer** ($fctComputer)

**Returns:** Nothing **Arguments:** Computer Object

**Notes:** Takes a computer object and updates its MySQL row. The MAC addresses is used as a unique identifier (IE. Primary key).

**Use:** Update device information, IE IP address changes.

function **formatIP** ($fcnIP)

**Returns:** String **Arguments:** Array of Strings

**Notes:** Takes an array of strings as input and concatates them into a single string where each element of the array is separated by a colon (:). Will be switching over to a json encode/decode scheme.

**Use:** For storing IP information within MySQL tables.

function **addNetworkDevice** ($fcnDevice)

**Returns:** Nothing **Arguments:** IOS\_Device Object

**Notes:** Creates a new MySQL entries for a supplied IOS\_Device Object.

**Use:** Used to populate the networkDevice MySQL table.

function **getNetworkDevice** ($fcnIP)

**Returns:** IOS\_Device Object **Arguments:** String

**Notes:** Searches the MySQL networkDevice table to find a matching IP addresses and returns a initialize populated IOS\_device based on the information obtained from the MySQL query.

**Use:** Obtains the IOS\_device object based on the supplied IP address.

function **cacheCmd**($fcnID,$fcnCmd,$fcnData,$fcnTimestamp)

**Returns:** MySQLi result object **Arguments:** String, String, String, String

**Notes:** Checks if the command has already been stored, if so update the command by updating the data and timestamp. If the command is currently not cached, then a new row is added/populated for the cached command.

**Use:** Updates the caching table (netCache).

function **cacheCheck**($fcnID, $fcnCmd)

**Returns:** String/Boolean **Arguments:** String, String

**Notes:** Checks if a cached version of the command with the corresponding ID exists. If the cache entry exists it returns a string with the corresponding data. Otherwise false is returned.

**Use:** Retrieves cached information.

function **cleanCache**($fcnTimestamp)

**Returns:** MySQLi result object **Arguments:** String

**Notes:** Clears all cached commands who’s timestamp is greater then the current, minus the predefined expiration time.

**Use:** Removes stale cached entries.

# class ssh – File SSH.php

function **ssh** ($fcnUsername, $fcnPassword, $fcnAddress)

**Returns:** Nothing **Arguments:** String, String, String

**Notes:** Initializes the class.

**Use:** Initial setup of the object.

function **setUsername** ($fcnUsername)

**Returns:** Nothing **Arguments:** String

**Notes:** Sets the username variable used within the SSH session. Should not be called outside of the constructor, as it may lead to unpredictable results.

**Use:** Sets the username used within the SSH session.

function **setPassword** ($fcnPassword)

**Returns:** Nothing **Arguments:** String

**Notes:** Sets the password used within the SSH session. Should not be called outside the constructor, as it may lead to unpredictable results.

**Use:** Sets the password used within the SSH session.

function **setAddress** ($fcnAddress)

**Returns:** Nothing **Arguments:** String

**Notes:** Sets the IP Address used within the SSH session. Should not be called outside the constructor, as it may lead to unpredictable results.

**Use:** Sets the IP Address used within the SSH session.

function **setPort** ($fcnPort)

**Returns:**  **Arguments:** String

**Notes:** Sets the Port used within the SSH session. Should not be called outside the constructor, as it may lead to unpredictable results.

**Use:** Sets the Port used within the SSH session.

function **getPort** ()

**Returns:** Integer **Arguments:** None

**Notes:** Returns the port variable value.

**Use:** Obtain the port being used within the SSH session.

function **getAddress**()

**Returns:** String **Arguments:** None

**Notes:** Returns the IP Address variable value.

**Use:** Obtain the IP Address being used within the SSH session.

function **getUsername**()

**Returns:** String **Arguments:** None

**Notes:** Returns the port variable value.

**Use:** Obtain the port being used within the SSH session.

function **getError** ()

**Returns:** String **Arguments:** None

**Notes:** Returns the Error variable value which is set after an error occurred during the SSH connection setup.

**Use:** Obtain the last error message triggered by the SSH connection setup.

function **PRINTSSH** ()

**Returns:** Nothing **Arguments:** None

**Notes:** Prints information relating to the SSH session in a formatted manner.

<username>:<password>@<IP Address>:<port>

**Use:** Development/troubleshooting only.

function **connect**()

**Returns:** Boolean **Arguments:** None

**Notes:** Establishes the SSH connection to a device with the specified parameters provided in the class constructor. Returns true if the connects was successful and false if the connection was unsuccessful. getError() can be used to retrieve error information if the session establishment was unsuccessful.

**Use:** Establishes a connection to the device.

function **close** ()

**Returns:** Nothing **Arguments:** None

**Notes:** Closes the stream to the device. This function is being deprecated.

**Use:** Closes the stream to the device.

function **cmd** ($cmd)

**Returns:** String **Arguments:** Array of Strings

**Notes:** Issues commands found within the supplied array. These commands are executed using the SSH2\_Exec which executes each command in a new SSH connection. Output is returned as a string with a line break between the output of each command supplied. This function is being deprecated.

**Use:** To execute single, none linked/dependent commands.

function **multiCmd**($cmd)

**Returns:** Array of Strings **Arguments:** Array of Strings

**Notes:** Issues commands found within the supplied array. These commands are executed using the SSH2\_SHELL which executes each command within a single SSH connection. Output is them pushed to an array that returns the output in the same order the commands were received/executed.

**Use:** To execute single or multiple commands that are linked/dependent.

# class redirect – network.php

function **redirect** ($fcnComputer, $fcnSource, $fcnDestIP, $fcnDb)

**Returns:** Nothing **Arguments:** Computer Object, String, String, Database Object

**Notes:** Class constructor, initializes the class with the provided values.

**Use:**

function **setDb** ($fcnDb)

**Returns:** Nothing **Arguments:** Database Object

**Notes:** Sets the database variable. Should not be called outside of the constructor.

**Use:** Sets the database variable.

function **getDb** ()

**Returns:** Database Object **Arguments:** None

**Notes:** Returns the Database object of the networkRedirect object.

**Use:** Obtain the database object used within the networkRedirect Object.

function **setComputer** ($fcnComputer)

**Returns:** Nothing **Arguments:** Computer Object

**Notes:** Sets the source computer used within the network redirection.

**Use:** Sets the source of the redirection.

function **getComputer** ()

**Returns:** Computer Object **Arguments:** None

**Notes:** Returns source computer object.

**Use:** Returns source computer object.

function **setSource** ($fcnSource)

**Returns:** Nothing **Arguments:** String

**Notes:** Sets the IP Address of the source gateway.

**Use:** Sets the IP Address of the source gateway.

function **getSource** ()

**Returns:** String **Arguments:** None

**Notes:** Obtains source ip address information.

**Use:** Obtains source ip address information.

function **setDestIP** ($fcnDestIP)

**Returns:** Nothing **Arguments:** String

**Notes:** Sets the destination IP variable

**Use:** Sets the destination IP

function **getDestIP** ()

**Returns:** String **Arguments:** None

**Notes:** Returns the destination ip variable

**Use:** Obtains the destination IP of the redirect

function **getDevicePath**()

**Returns:** Array of IOS\_Device Objects **Arguments:** None

**Notes:** Returns the array of IOS\_Devices in a sequential order from start to finish of the network path between the source PC and destination gateway. IE the chain of devices packets will travel thought

**Use:** Obtain the Device path of a network redirect

function **getNextHopPath**()

**Returns:** Array of Strings **Arguments:** None

**Notes:** Returns the array of next hop interfaces that form the path from source PC to the destination gateway. IE the next hop IP address of each hop.

**Use:** Obtain the path of the direct to be used to convert to DevicePath in a load/save

function **findPath** ()

**Returns:** Boolean **Arguments:** None

**Notes:** Finds the shortest path between two points in a network by leveraging the IGP/Routing of the network. By recursively call a network devices getNextHop (equivilate to issuing show ip route <destination> and parsing the output to return only the next hop IP). Then finding the nexthop device based on the IP address obtained. And repeat until a connection interface is found. Maximum path length is set to 10 to prevent loops

**Use:** Finding the shortest path between two points

function **applyPath** ()

**Returns:** Nothing **Arguments:** None

**Notes:** Moving through the device path, the route-maps ACL is modified to include the requesting device.

**Use:** Applies the redirection to devices within the network path

function **removePath**()

**Returns:** Nothing **Arguments:** None

**Notes:** Moving through the device path, the route-map ACL is modified to remove the requesting device.

**Use:** Removes the redirection on devices within the network path