User Reference

Selenium – Retrieve Web GUI Tables

Date: 2/8/2015

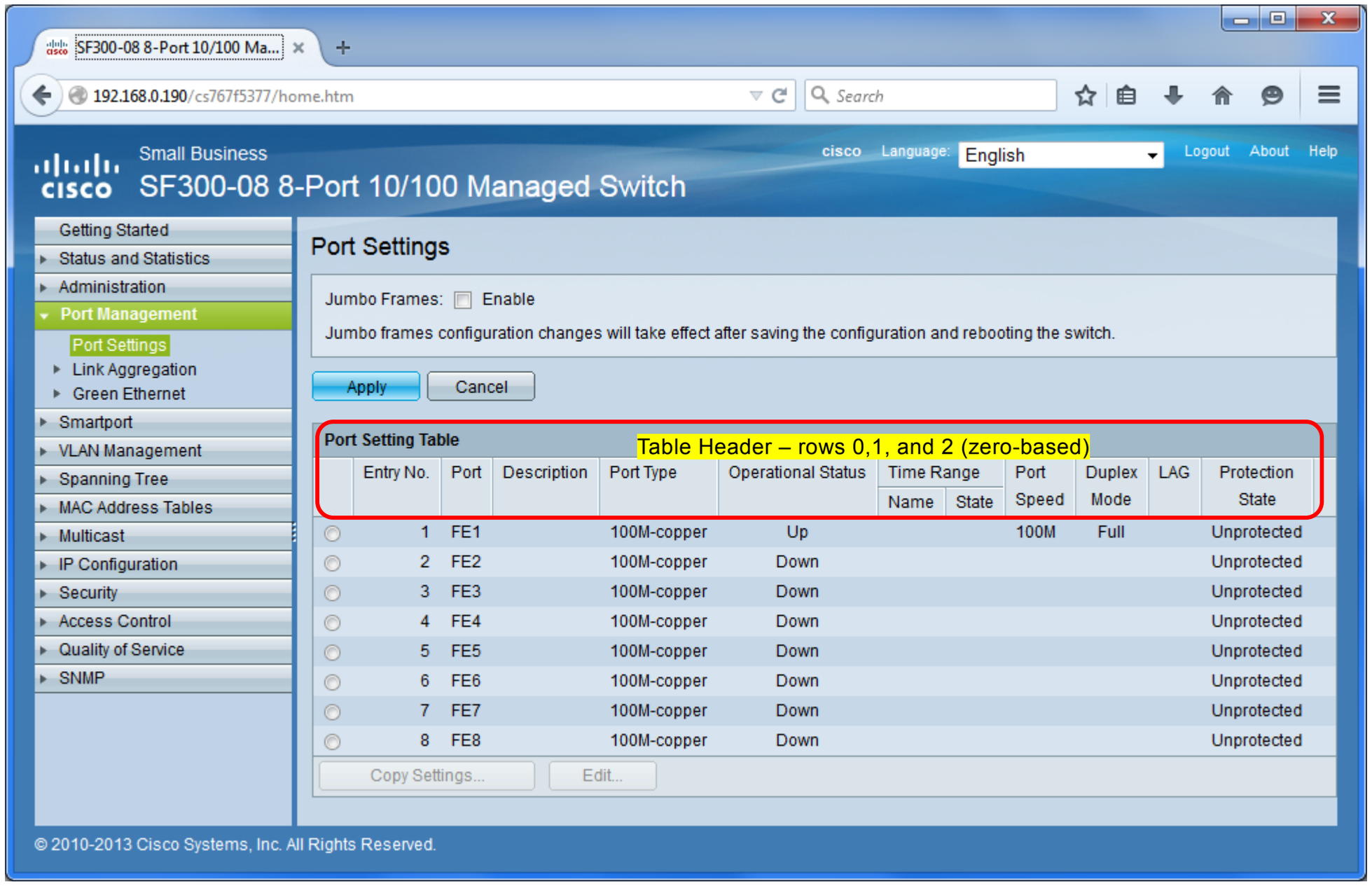
Revision: 1.0

Overview

In this document we will focus on retrieving information from a web GUI that is presented in table format. Please review other tutorials for Selenium installation, connection, and basic usage, where we discuss how to connect and login to our DUT. In this tutorial we will open a web GUI session with our Cisco switch and retrieve the contents of tables. These we will store in TAZ *ETables* to evaluate the contents. The *TestCase* **SeleniumDemo_TC_RetrieveTableData.txt** and the *Equipment Table* file **SeleniumDemoEquip2.txt** are both included in the accompanying **zip** file as well as logs from the run.

Open the GUI

Let's connect to our Cisco switch through the GUI and use TAZ Selenium commands to go to **PORT MANAGEMENT>>PORT SETTINGS**. Note that the first 3 rows of the **Port Setting Table** is the header.



Retrieve the Port Settings Page

We can use

Send{1|selectWindow&name=mainFrame|Sel} to select the **PORT MANAGEMENT>>PORT SETTINGS** page and use

Send{1|getBodyText|Sel} to return the entire page contents.

Here is the *Return*. TAZ has formed it as Send1 Selenium result **<OK,**Text**>**:

```
Send1 Selenium result <OK,Port Settings Jumbo Frames: Enable Jumbo frames configuration changes will take effect after saving the
configuration and rebooting the switch. Apply Cancel Port Setting Table

 Entry No. Port Description Port Type Operational Status Time Range Port

Speed Duplex

Mode LAG Protection

State Name State 1 FE1 100M-copper Up 100M Full Unprotected 2 FE2 100M-copper Down Unprotected 3 FE3 100M-copper Down Unprotected 4 FE4
100M-copper Down Unprotected 5 FE5 100M-copper Down Unprotected 6 FE6 100M-copper Down Unprotected 7 FE7 100M-copper Down Unprotected 8
FE8 100M-copper Down Unprotected Copy Settings... Edit...>
```

This is difficult to read and **Xpect** on, so let's retrieve the table information in another way.

Capture the GUI Table as a TAZ ETable

We'll use a loop to retrieve and enter the data in a TAZ *ETable*. But first, let's create column headers.

Create Column Headers for ETable1

The GUI table has column headers spread over two rows, so we will create our own. One way to do this is to put the column names into the *PassOut Table*, and then add the *PO Table* as the first row of our *ETable*.

PassOut{Insert|1|Selection}

PassOut{Insert||Entry}

PassOut{Insert||Port}

PassOut{Insert||Description}

PassOut{Insert||Type}

PassOut{Insert||Stat}

PassOut{Insert||Name}

PassOut{Insert||State}

PassOut{Insert||Speed}

PassOut{Insert||Duplex}

PassOut{Insert||Lag}

PassOut{Insert||Protection} We can add *ArgF*, **ExcludeBoth** to each **Send{…}** to unclutter our log

Once our header names are entered in the *PO Table*, we can use

PassOut{Show|All|Add|ETable1((NextRow))} to add our *PO Table* as the first row. We'll use this same command to add each subsequent row of data.

Capture the GUI Table Data

Now let's add our data. We will use 2 nested loops. The outer loop (Loop1) will iterate row by row, while the inner loop will capture the values for each column and add them to the *PO Table*. For more detail on the **Loop{…}** command, see the TAZ Command Manual.

Outer

Loop{1|8|%x=3} We will capture eight rows of data, starting on row 3. Rows 0,1, and 2 are part of the GUI table header (see screenshot)

PassOut{Clear} We make sure that the *PO Table* is empty before adding values

Inner

Loop{2|12|%y=0} we iterate through twelve columns (zero-based) in this inner loop for each outer loop row

Send{1|getTable&tblRepeatedPortConfig.%x.%y|Sel}

Here the **getTable** Selenium command does the work. It takes an argument of the form:

<HTML TableName>**dot**<row number>**dot**<column number> which returns the value at those coordinates. By using variables %x (row number) and %y (column number) in this inner loop we will retrieve the value of each cell for the current row. We will capture the cell value and put it in the *PO Table* with

PassOutBetween{<OK,|>||||||((Empty))|ExcludeBoth} TAZ returns the data from the previous command as

<OK,"the cell value"**>**

This **PassOut** command will look between **<OK,** and **>** and place what it finds (the cell value) in the *PO Table*. The **((Empty))** parameter says that regardless whether a value is found or not, add an element to the *PO Table*. This keeps our table columns ordered.

EndLoop{2} end this inner loop, which is one complete row

PassOut{Show|All|Add|ETable1((NextRow))|ExcludeBoth} Now that the inner loop has put a value for each of the 12 columns in the *PO Table*, take the entire *PO Table* and add it as the next row of the *ETable*

EndLoop{1}

Result

When we use **ETable{1|Show}** we see that our *ETable* values match the **Port Setting Table** of the GUI.

Let's use **XpectET{…}** to check the value of the Stat column for Port FE1.

ETable1

Selection	Entry	Port	Description	Type	Stat	Name	State	Speed	Duplex	Lag	Protection
	1	FE1		100M-copper	Up			100M	Full		Unprotected
	2	FE2		100M-copper	Down						Unprotected
	3	FE3		100M-copper	Down						Unprotected
	4	FE4		100M-copper	Down						Unprotected
	5	FE5		100M-copper	Down						Unprotected
	6	FE6		100M-copper	Down						Unprotected
	7	FE7		100M-copper	Down						Unprotected
	8	FE8		100M-copper	Down						Unprotected

XpectET{ET[1,FE1,Stat]|=|Up}

ArgA – **ET[1**,FE1,Stat**]** , replace with the cell value where

ETable = **1**

Row = **FE1**

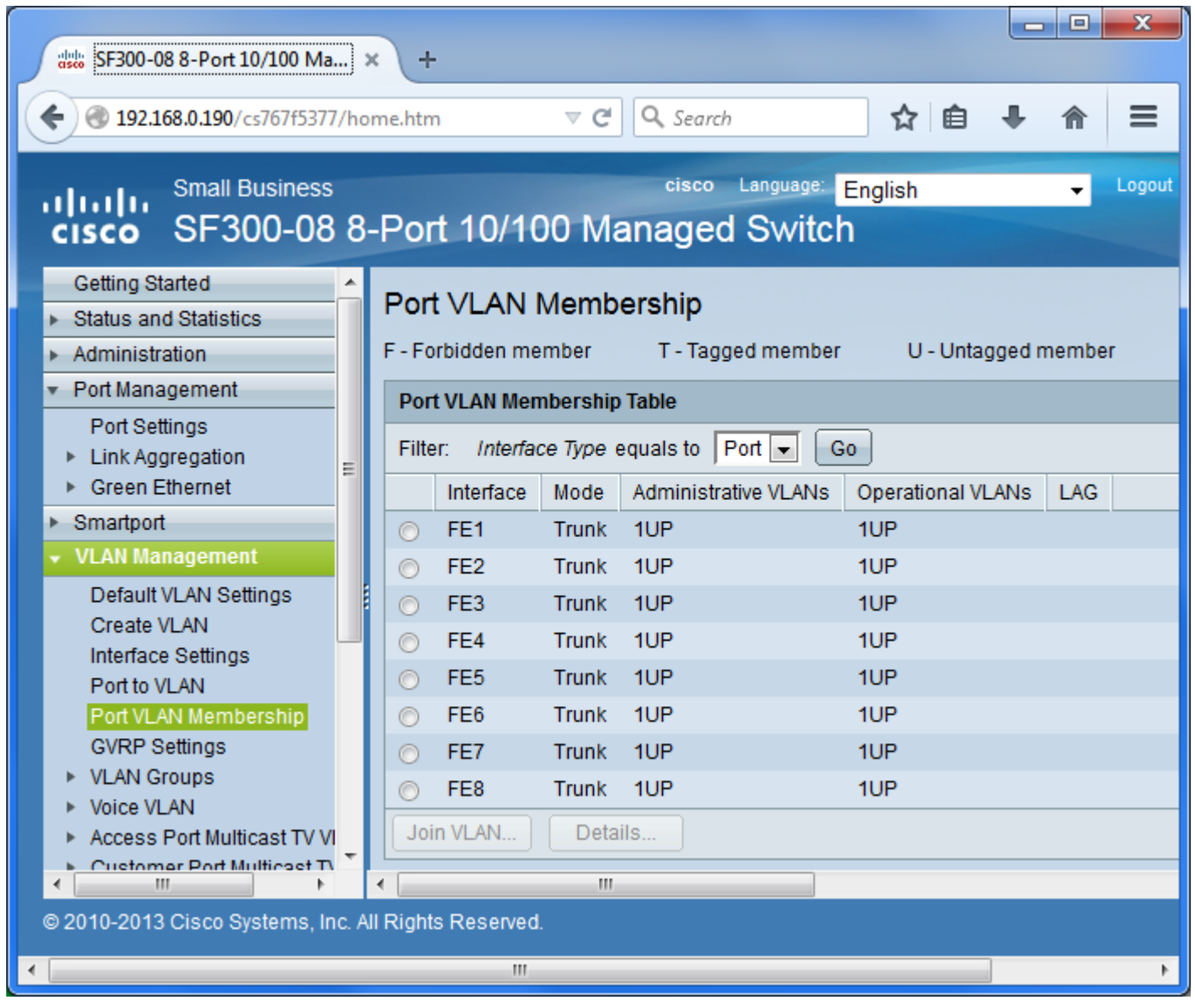
Column = **Stat**

ArgB – **=** , equals

ArgC – **Up** , the text “Up”

The VLAN Membership Table

Our *TestCase* navigates to the **VLAN Membership Table**.



Column Headers and Data for ETable2

This table is more orderly than the first, so we can capture the headers and data directly from the GUI table. We will iterate through 9 rows, the header row plus 8 rows of data. We start at row 1 (zero-based).

TAZ Loop{1|9|%x=1} 9 iterations get the header row and 8 rows of data

TAZ PassOut{Clear}

TAZ Loop{2|6|%y=0} there are 6 columns of data in this table

TAZ Send{1|getTable&oPVM.%x.%y|Sel}

TAZ PassOutBetween{<OK,|>||||||((Empty))|ExcludeBoth}

TAZ EndLoop{2}

TAZ PassOut{Show|All|Add|ETable2((NextRow))|ExcludeBoth}

TAZ EndLoop{1}

And here is *ETable2* from **ETable{2|Show}**. The first column, the selection buttons, does not have a header name.

ETable2

Interface	Mode	Administrative VLANs	Operational VLANs	LAG
FE1	Trunk	1UP	1UP	
FE2	Trunk	1UP	1UP	
FE3	Trunk	1UP	1UP	
FE4	Trunk	1UP	1UP	
FE5	Trunk	1UP	1UP	
FE6	Trunk	1UP	1UP	
FE7	Trunk	1UP	1UP	
FE8	Trunk	1UP	1UP	

Third Party Disclaimer

This disclaimer pertains to all third-parties, including but not limited to:

- Spirent Communications plc
- Ixia
- Shennick
- Ameritec
- Mozilla
- Selenium
- Net-SNMP.org
- Microsoft
- Cisco
- APC (Schneider Electric)
- ControlByWeb
- 3G Store
- Net-SNMP
- Adobe
- TestLink
- Oracle

AZTAZ Software, LLC is not affiliated, associated, authorized, endorsed by, or in any way connected to any third-party or third-party products referenced in any AZTAZ Software, LLC products, including but not limited to our website, software, marketing, videos, documentation or communications.

AZTAZ Software, LLC does not control, endorse, or accept responsibility for any materials, products, websites, or services offered by third parties that we may reference.

All product and company names are trademarks™ or registered® trademarks of their respective holders. Use of them does not imply any affiliation with or endorsement by their respective owners.