



User Reference

What You Can Do With TAZ Test Automation

Date: 10/12/2014

Revision: 2.4

This controlled document is the proprietary and confidential property of AZTAZ Software, LLC and may be viewed only with the express written permission of Aztaz Software, LLC.

Any duplication, reproduction, or transmission to unauthorized parties without the express written permission of AZTAZ Software, LLC is prohibited.

Copyright © 2014 AZTAZ Software, LLC. All rights reserved.

Overview

This is an introduction to the many TAZ features. The list is not complete and the features that are included are not covered in detail. If you have a testing need and can't see the solution here, please ask. TAZ can probably do it. And if not, we are continually adding functionality. support@aztazsoftware.com

Also, TAZ was designed for customization. If you have special testing circumstances that require customization you will be surprised how quickly and inexpensively this too can be done.

What You Can Do with TAZ Test Automation

- TAZ Commands can control any device, instrument, Windows, or Linux computer

Here is an example of the basic TAZ **Send{...}** command sending a command to a switch:

Send{mySwitch|show version|CLI}

- Connection information for **mySwitch** was previously entered in an *Equipment Table*.
- We've chosen to send the native command **show version**
- We are using the **CLI** language. TAZ supports multiple languages for each piece of equipment.

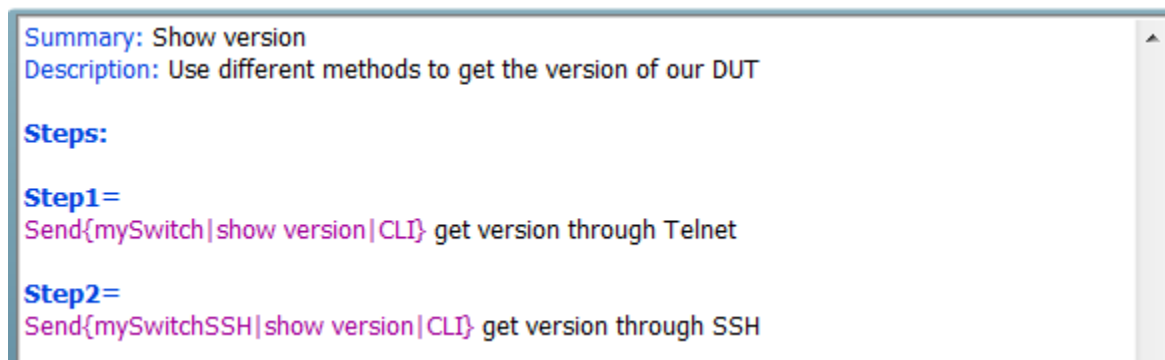
- Equipment communication protocols:
Telnet, SSH, SNMP, Serial, TCP/IP, HTTP, SQL, FTP, DOS, TCL, GPIB, USB, SCPI

We previously created a connection named **mySwitch** in the TAZ *Equipment Table* and chose **Telnet** as the protocol. Let's create another connection to the same physical switch, choose **SSH** as the protocol, and name it **mySwitchSSH**. We can have multiple *Equipment Table* connections to the same piece of equipment, but they must have unique names.

For example:
Send{mySwitchSSH|show version|CLI} will go over **SSH**.

- What a TAZ TestCase looks like

The above commands in a TAZ *TestCase* look like this:



```
Summary: Show version
Description: Use different methods to get the version of our DUT

Steps:

Step1=
Send{mySwitch|show version|CLI} get version through Telnet

Step2=
Send{mySwitchSSH|show version|CLI} get version through SSH
```

TAZ error checks *TestCases* as they are written with color-coding. Valid commands are **purple** or **green**. Remarks are **black**. Structural elements are **blue**.

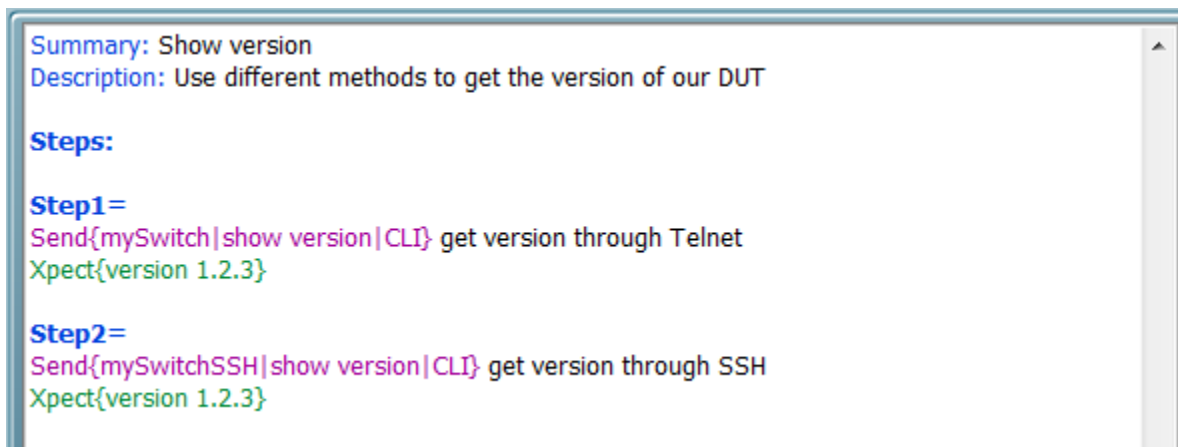
- Pass/Fail evaluation of TestCases - Basic

Each **Send{...}** produces a **Return**. Now we need to evaluate if the correct data is contained in that **Return**.

Here we will use **Xpect{...}** in its most basic form.

We know, in our example, that the response to our **show version** command should contain the text “**version 1.2.3**” so we will evaluate with **Xpect{version 1.2.3}**.

If the **Return** from our **Send{...}** command contains text that includes “**version 1.2.3**”, **Xpect{version 1.2.3}** will evaluate to **Pass**.



```
Summary: Show version
Description: Use different methods to get the version of our DUT

Steps:

Step1=
Send{mySwitch|show version|CLI} get version through Telnet
Xpect{version 1.2.3}

Step2=
Send{mySwitchSSH|show version|CLI} get version through SSH
Xpect{version 1.2.3}
```

In this **TestCase** we are confirming that the software version is 1.2.3 and that retrieval through **Telnet** or **SSH**, gives the same result. In a real world TAZ **TestCase** we would treat the version number as a variable. This would make the **TestCase** flexible for all versions.

PLEASE NOTE: Xpects in TAZ are very powerful commands with abilities to evaluate any text in any **Return**.

We have only demonstrated the most basic here.

- **Web GUI control of equipment**

TAZ uses Selenium for web control. When we connect to equipment in the *Equipment Table* with **GUI** as the protocol, TAZ opens a Selenium web session to the equipment.

For example:

Let's enter information to the same physical switch, but name our equipment **mySwitchGUI** with **GUI** as the protocol. Connecting will open a Selenium web session. **Send{mySwitchGUI|click&show version|Sel}** would "click" the button called "show version" on the web session. We could then send a Selenium command to retrieve the version information.

- **Capture and use runtime variables**

There are two methods to capture and use runtime variables.

- **PassOuts** – To capture an entire *Return* or any part of one.

For example:

PassOut{Insert|1|Return} will put the *Return* from a test step into the *PassOut Table*.

- **Etable** – When data is returned in table format, the *Etable* commands easily capture the data.
- Once in a TAZ table there are many different ways to manipulate the data. **Xpects** can compare elements and evaluate numerical expressions. Tables can be saved to file. And table elements can be used in other commands.

For example:

The **PO[1]** in **Send{mySwitch|reset CPU PO[1]|CLI}** will be replaced at runtime with its *PassOut Table* value.

- **Data driven testing – the Variable Table**

With a **Variable Table** you can parameterize your **TestCases** so that each run uses a different set of variables. A **Variable Table** is like an Excel spreadsheet where each row is a run and each column is a variable.

- **Run existing scripts in any programming language**

Whether you have existing legacy scripts written in other programming languages or are creating new ones, TAZ can run them for you.

Here is the TAZ command to run an example TCL script named **Setup VLANs.tcl**:

```
Script{myTclShell|Setup VLANs.tcl|TCL}.
```

When the script completes, TAZ captures the **Return**. This **Return** can then be evaluated with **Xpect**.

- **Direct instrument control of: Spirent TestCenter, Ax4000, Smartbits, Ixia, IxNetwork, DiversifEye, APC, Fortissimo, Crescendo, T-Berd, Agilent VSA**

Although TAZ can connect and send commands to any instrument, TAZ has been optimized to work with these.

For example, there are special, instrument independent commands that when included in a **TestCase** will work interchangeably on any of the listed like equipment. Choose a TestCenter and the command works on TestCenter. Choose an Ixia and the same command works on Ixia.

- **Power cycle and reconnect equipment**

Power cycling is just a simple command in TAZ and TAZ is smart about reconnecting. It monitors equipment and knows to reconnect only when the equipment is back online. This is particularly helpful for software updates, backup and restore, and system testing.

- **Simulate fiber cuts and cable pulls**

A single command “cuts” and “reconnects”.

- **Runtime debugging**

TAZ offers exceptional runtime control from the **RUN TESTS** screen. On **RUN TESTS** you have a complete visual display and control of test execution. This includes:

- Equipment communication – communicate with any connected equipment. Just pause a step and TAZ provides a client window for real time communication to the running session.
- Launch debug scripts – on many types of triggers
- Send emails – on many triggers
- Pause – on many triggers
- Skip steps
- Rerun steps

- **Save results to a database**

TAZ will interface with custom databases or commercial products like HP Quality Center, Contour, Jira, TestLink, or will be customized to work with a product of your choice.

- **TestCases are portable over any test bed**

A key feature of TAZ is portability. **TestCases** are not dependent on any specific equipment. If a tester is writing a **TestCase** for his DUT with his IP Address, the same **TestCase** can run without changes on any like DUT. There are many other ways that TAZ makes tests portable.

- **Save and recall entire testing setups - Configs**

All aspects of a testing setup are saved to file with a single button. This includes all tests, test options, all necessary equipment connections, file paths, search and replace tables, variable tables, etc. At any time a testing setup, called a **Config**, can be recalled.

- **Test Automation Management**

You saw above how entire testing setups, called **Configs**, can be saved and recalled. You or colleagues can create as many of these **Configs** as needed and then select one or more to run.

- They are portable – even though a colleague with different equipment created a **Config**, you can still run it with your equipment.
- Schedule – Instead of running immediately, **Configs** can run at a scheduled date/time. TAZ can also launch automation when a new software build is created.
- Command line – launch from a Windows command line

- **Create and view logs in many formats**

Each time TAZ executes tests, a log is created. This full log is a detailed account of every command sent, responses, date/times, testing parameters, etc. You can view the color coded log in its entirety or in several summary formats that can be saved and opened by Excel. You can also define and create your own logs.

- **Use Loops, Conditionals, call other TestCases**

Although TAZ is command-based and does not require programming, some programming constructs are very helpful and are included. Even these are implemented as commands that make them user friendly.

- **Other Features**

There are numerous other features. If you don't see what you need, please ask. support@aztazsoftware.com.

Third Party Disclaimer

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

- Spirent Communications plc – TestCenter, Ax4000, Smartbits
- Ixia – IxNetwork
- Shennick – DiversifEye
- Ameritec – Fortissimo, Crescendo
- Mozilla
- Selenium
- Net-SNMP.org
- Microsoft
- Linux
- Cisco
- Schneider Electric – APC
- ControlByWeb
- 3G Store
- Net-SNMP
- Adobe
- TestLink
- Oracle
- Google
- Perle
- Agilent – VSA
- JDSU – T-Berd
- Hewlett-Packard (HP) – Quality Center
- Jira
- Contour

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 company names and products are trademarks™, registered® trademarks, or copyrights of their respective holders. Reference to them does not imply any affiliation with, responsibility for, endorsement of, or endorsement by them.