The PageUnit Framework

A User Guide

Ian F. Darwin, http://www.darwinsys.com/

Introduction

PageUnit is an open-source testing framework for testing web applications. We offer the following draft documentation in hopes that it will be found useful in describing the easy-to-use features of this web testing framework:

- Very simple input language
- No XML (this is a feature!)
- Small footprint – only a few Jar files (7 at last count)
- Fetch most pages by relative URL; host and port can be specified in several ways
- Match text in whole page or in selected HTML elements (text or regex match)
- Find link or form, fill in form parameters, submit form or link
- Support for Java EE login screens
- Built-in link checker
- Extensibility by Java-based plug-in mechanism.

There is a web site for the project, at http://pageunit.darwinsys.com/ .

Input Language

The input language is documented in the following table. Each non-comment line has one command and zero or more operands, as shown in the table. Optional operands are denoted by [...].

Table 1. The PageUnit Commands

Cmd	Operands	Meaning
#	Anything	Comment line, printed but ignored.
A	username [password]	run As, e.g., set new user
B	URL	Base URL; Change base of url e.g. http://www.foo.com/
C	config-name	Choose one tuple {username,pass,baseurl,port} from a Config (preferred over B,A,H,P) (not working yet)
D	t|f	Debugging

Cmd	Operands	Meaning
E	any	Echo operands to standard output
F	formName	find form in page
G	url	GoTo link found by L
H	hostname	Set Hostname for subsequent tests
I	(unused)	Reserved for 'IsItThere?' (HEAD HTTP-method test)
J	URL	Get Java EE “container managed login” page
K	(unused)	-
L	text	Locate Link in page containing text
M	regex	Match regex in current page
N	-	New browser session (discard previous)
O	port	Hard-code pOrt number (see H above)
P	rURL	Get Page
Q	(unused)	-
R	name=value	set paRameter in form
S	\[buttonName]	Submit form
T	tag text	At least one tag of the given class in the current page must contain the text.
U	(unused)	Reserved for “timeoUt and other options”
V	URL	Verify Links - Run Link Checker
W	(unused)	-
X	className	Add plug-in
Y	className	Remove plug-in
Z	(unused)	-

Cmd	Operands	Meaning
=	name value	Set variable for use in other commands (e.g., E, R, ...); value must not contain \ or $ characters.
<	filename	Proposed feature: file inclusion

Variables

Variables can be set at any time with the = command, and remain in effect for the run (even across multiple input files). Operands should not be quoted, but must be set off from the variable name by a space character. Variable substitution occurs in all input lines, on all non-null text following the command; the command's operands need only be valid after substitutions are done. Variables are substituted by use of ${...}, for example:

```
= testerName Robin Smith
E Test run by ${testerName}
```

The following variables are pre-defined; their names are in all capitals:

Table 2. PageUnit Pre-defined variables

Name	Meaning	Set By
USER	User name for login	Properties, C, or A
PASS	Password for login	Properties, C, or A
HOST	Host name (or IP)	Properties, C, or H
PORT	Port number	Properties, C, or O

Variables as Extraction Rules: The Match command (M) sets groups as variables. The variable M0 is set to whatever matched; if the regex pattern contains capture groups, they are set in M1, M2, ... Mn. This allows you to select values in one page and use them in another, for example, find the primary key assigned to a newly-inserted record, and use it in a search.

Building With Maven

PageUnit is in Maven Central so you don't have to build it if you don't want to!

PageUnit is available in source code from the github repository. Check out the code and build it; all Jar files are downloaded. Java 8 is the current Java and is recommended. The GitHub repository is at http://github.com/IanDarwin/pageunit.git

Build, test and package it using Maven in the normal way: `mvn package install`

Optional: Create a file called ".pageunit.properties" in your home directory (~ or $HOME on UNIX; on other Oses, whatever Java thinks is ${user.home}. It should contain these default parameters: login=someUserName password=anyOldPassword host=localhost port=8080 Of course, the login and password should be those to log you in to the "main" web application that you want to test, and the "host" and "port" should be the server's web address.

Getting Started with JUnit

PageUnit is in Maven Central. All you have to do to use it with Maven is add this to the `<dependencies>` section of your pom.xml:

```
<dependency>
    <groupId>com.darwinsys</groupId>
    <artifactId>pageunit</artifactId>
    <version>1.0.1</version>
</dependency>
```

Then you can write a test like this one:

```
import org.junit.Test;

import pageunit.ScriptTestCase;

public class PageUnitDemoTest {

    @Test
    public void demo() throws Exception {
        new ScriptTestCase("src/test/pageunit/oreilly.txt").run();
    }
}
```

Run it as usual!

Getting Started under Eclipse

Check out the git repo "pageunit" as above, as an Eclipse project. This will create a Java project named pageunit. Then, create a second Java project called SiteTest (substituting the name of your site); have it depend on the pageunit project so it can find the needed class files. Create a non-source folder therein called "tests", and create at least one test file thereunder (a good minimal test file that will allow you to run the test framework is a file called tests.txt (the filename extension txt is what it looks for) containing just the one line:

```
# This will be a test file.
```

Next, make a Java Application run configuration (Run→Run→New Configuration); set the “main class” name to pageunit.PageUnit, and change the “Run in” directory by adding “/tests” to it. Now you should be able to run the new Test Configuration and see a message with some text like “0 tests run, 0 failures”.

Getting Started Command Line Usage

Get the Git project as above. Build it with one extra step:

```
mvn package assembly:single
```

This creates an ”uber jar” with all needed dependencies in one jar file, which can be run with java -jar. You probably want to copy this to your home directory:

```
mkdir ~/lib
cp target/pageunit-1.0.1-SNAPSHOT-jar-with-dependencies.jar ~/lib
```

You don’t have to put it in ~/lib, but the default script we provide assumes so. Either way, you should then add the directory with the scripts dir to your PATH setting, or copy the script into a directory that’s in your PATH. Assuming you’ve done both steps (and/or edited things to match), you should be able to say:

```
pageunit search.txt
```

and see the output showing success.

Contributing

This is an open source project, and contributions from users are welcome. Code patches should be in the form of “git diff -u” or Eclipse Team→Create Patch. The web site and thus the documentation are in a different git repo; sadly, documentation patches should be sent in email for now. There is a lot of work still to do. For ideas on some smaller things that need doing, see the TODO file in the docs directory; for some larger ideas, see the “Future Directions” of the academic report.