Web Macros Manual

Velocityscape, LLC

October 31, 2007

Contents

Chapter 1

About Web Macros

1.1 What is Web Macros?

Simply put, Web Macros is automated site navigation that can be produced rapidly by anyone who can use a web browser. If you're new to web automation and have no programming experience, you won't be confronted with an insurmountable learning curve. If you're a programmer who has struggled with getting HTTP requests to exactly mimic a browser, then Web Macros will be a huge time saver.

The guts of Web Macros is an engine that allows you to record and play back actions you perform within the browser. All you have to do is click the record button and then begin navigating as you would in any other browser. Your actions will be recorded and displayed in our own scripting language. From there your script or macro can be saved, modified, and played back whenever you like.

1.2 Why use Web Macros?

There are a few common uses of Web Macros. First and foremost is web data extraction or web scraping. This is the act of taking information from a website, parsing it, and storing it in a database or excel spreadsheet. Product catalogues, membership directories, and job listings are some of the most common types of information people seek, but the options are only limited by the data you can find online in a consistent format. So pretty much anything. There is just one important distinction to make. Web Macros automates the site navigation, but not the actual data extraction. To do that you should look at using the Web Macros Data Extraction Edition or Web Scraper, a separate Velocityscape product.

Site testing is another great application of Web Macros. You can create a macro to test key functionalities of your web site and then sit back and watch it do the checking for you. Or even better, don't watch at all. Go get some coffee and have the macro save the pages

it visits as it goes. Then you can check those files at your leasure or pass them to another program for further processing. By streamlining your site testing process, with Web Macros you can become confident that after every update, you have not introduced any bugs to your website.

Do you have any other monontenous tasks online? This could be online searches (especially when saved searches are not availible), logins, or other form filling tasks. Whatever it is, you can record it with Web Macros. From then on the task will be reduced to a single click on the "Play" button. Whatever you end up using Web Macros for, the results will be the same. A small amount of work up front, leading to a lot of saved time later.

1.3 Who Are We?

Here at Velocityscape, we are all things data extraction. Founded in 1999 by Michael J. Roberts, we have grown steadily year after year, even through the dotcom collapse, and continue striving to provide better software, better services, and better data.

Velocityscape's core product, Web Scraper Plus, has been on the market since 2001 and has sold over 1,000 licenses. Until the release of Web Macros, Web Scraper Plus and Web Scraper Lite were the entirety of our off the shelf products, and the foundation of our consulting services for web data extraction. Web Macros was created to supplement and improve all our previous products and services. A detailed comparison of the two products can be found at the end of this section.

Aside from offering our own software and services to help others extract data, we've also managed to extract some interesting data ourselves. The fruits of our labor can be seen at http://www.spyfu.com. That's SpyFu, a combination of "Spy" and "Kung Fu". It is a compilation of competitve intelligence on Google Adwords. Millions of companies, domains, and terms are refreshed every thirty days. For more information see the site.

1.4 Other Web Macros Products

There are four Web Macros products in total ranging from free up to the enterprise edition. Below is an explanation of each:

1.4.1 Web Macros Free

The full Web Macros browser with support for everything covered in this manual except for those things specifically marked as only availible in the full version.

1.4.2 Web Macros

The full version of Web Macros will have one very important distinction, silent mode. You will be able to run macros from the command line without any browser window every becoming visable. You can silently save the files you are interested in for later review or further processing.

Another noteworthy feature is the ability to use table variables. So you can link input from a database to a macro via whatever query you want.

1.4.3 Web Macros Data Extraction Edition

Our most popular Web Macros options so far. This product combines the power of Web Scraper's Datapage Editor with Web Macros. This means you can extract structured data from a website directly to an excel spreadsheet or database. You do so by using an additional EXTRACT macro command that comes with this product.

1.4.4 Web Macros Enterprise

The enterprise version will allow Web Macros to run as a distributed application. The core engine will be bundled as a web service that can be run on any machine. Pages can be saved locally on the server doing the automation, or submited to an entirely different web service for whatever further processing you need. The end result is the ability to have multiple users running multiple macros simulateously on any number of machines. Scaling is no longer a concern.

1.5 Web Scraper vs. Web Macros

1.5.1 Web Scraper Overview

Web Scraper is a product whose sole purpose is to take information from a website and extract it to a database or excel file. In order to do this you must create templates that map parts of a web page to different columns in a table. This template is called a datapage. Creating a datapage is a point and click experience for the user and is documented via video tutorials on velocityscape.com. The other big part of Web Scraper other than the datapage is the package. A package is Web Scraper's means of automatic site navigation. You can add tasks that perform logins, crawl sites, and follow links that you've scraped with other datapages. Building packages is also a point and click experience to an extent but also often involves typing SQL queries for advanced site navigation.

1.5.2 Web Macros Overview

Web Macros is dedicated to the task of automated site navigation. User's are able to record their actions by browsing normally and then play them back. While a data extraction edition is availible for Web Macros, it is not a part of its core functionality and will never be a part of Web Macros Free.

1.5.3 Compare and Contrast

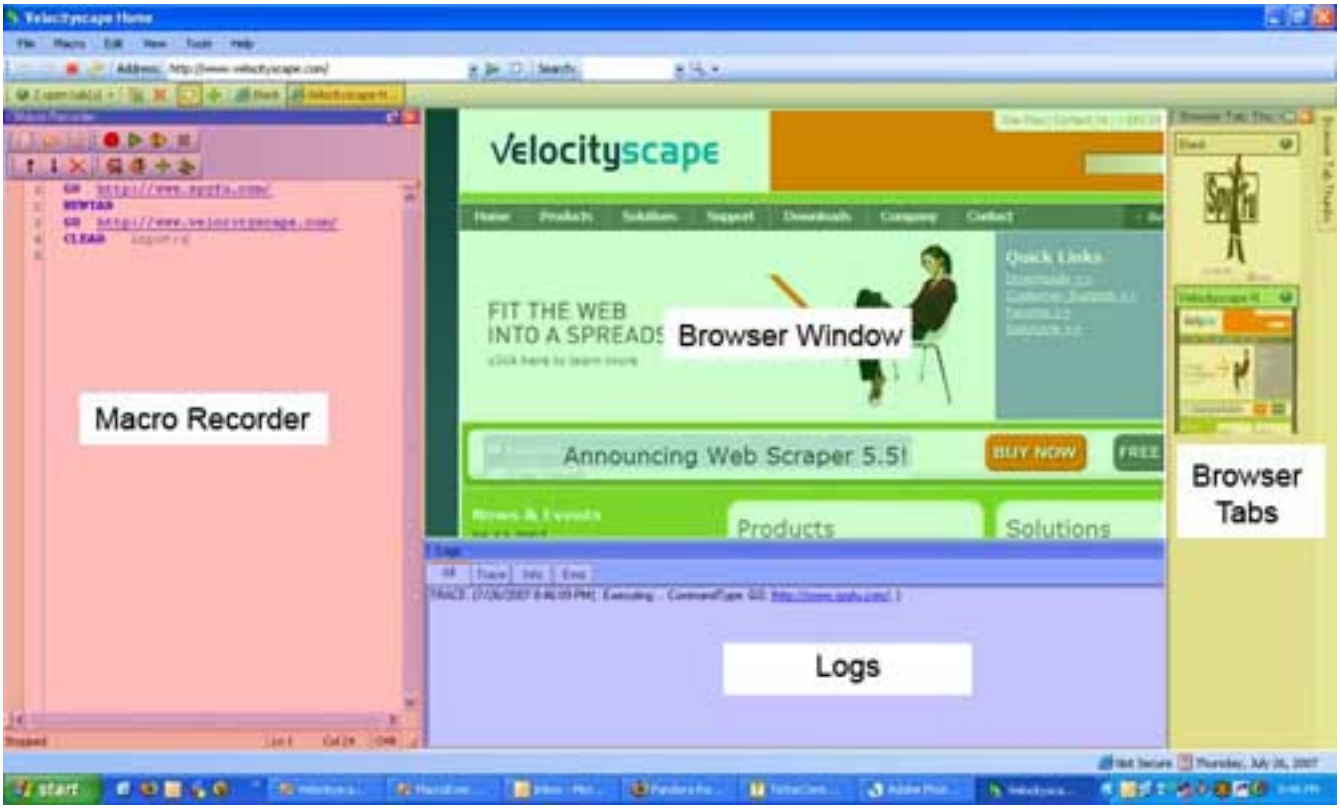
So Web Scraper does data extraction and site navigation, while Web Macros focuses only on site navigation. What makes it worth while is *how* Web Macros does site automation. The ability to record your actions, and debug your work by actually watching it playback in front of you is a feature that is unique to Web Macros.

Also, Web Macros records the actual clicks and keystrokes of your mouse and keyboard. Web Scraper works at a much lower level. To automate a site with Web Scraper, you must first be able to explore the site. This could involve understanding the requests certain JavaScript links make, determining whether a form uses POST data or a query string, or finding what session variables are maintained. In Web Macors, all the work is done behind the scenes. If a link executes JavaScript, Web Macros will take care of it; you just have to tell it which link to click. Web Macros will know if a form uses the GET or POST method; you just have to click to submit the form. Everything is simplified.

Web Macros is such an easier interface for web automation that it does things that for all intents and purposes Web Scraper can't. The best example is the ViewState that is found on all ASP.NET pages (.aspx pages). Microsoft uses a large encoded form variable to hold information on how a current page should be displayed. The more you click around, the bigger the ViewState typcially gets, and the more difficult it is to use with Web Scraper. But to the casual internet user, the ViewState is invisible behind the scenes. The same is true for Web Macros users. Forms that use ViewStates and other complicated form requests are no longer a problem.

Chapter 2

Exploring the Web Macros Browser



2.1 The Browser Window

The browser window is where you actually see the websites you're navigating to. It is no different from what you would see with Internet Explorer. In fact, the guts of this window are built off of mshtml.dll, the same dll that internet explorer uses.

2.2 The Macro Editor

The macro editor is the one window that makes the Web Macros browser more than just another browser. It is located on the left hand side by default and resembles a text editor. In the following sections we'll take a look at the functionality of this tool.

2.2.1 Recording

To begin recording, click the circular red record button in the macro editor's toolbar. You should notice the status bar at the bottom of the macro editor window changing from "Stopped" to "Recording". Once recording you will see new lines of macro code appear at the end of the current macro as you navigate. Try searching and filling out different forms to see the type of code Web Macros generates.

To stop recording click the square stop button. But be aware that you can play back what you have recorded without leaving recording mode. Web Macros will automatically stop recording, perform the play back, and then return to recording mode.

2.2.2 Playback

Play

The play button (solid triangle icon) will play the current macro from the very beginning regardless of where the cursor is and continue down to the end macro or the next breakpoint. Breakpoints are explained later in this section. As the macro plays, you will see the executing line highlighted, as well as see the browser navigate on its own in the main window.

All previous variables will be cleared out when the play button is clicked. That is why the Resume/Play button exists (a series of triangles overlapping). This button will start playing from wherever the cursor is placed, and it will remember whatever variables you have declared from playing a macro earlier.

Step

The step button (triangle with a vertical dash to the left of it) will execute just the current line of macro code. The current line is whatever line the cursor is on. When finished, the cursor will be moved down one line, allowing you to continually tap the step button and move down the current macro.

Variables will not be erased as you step through your code. So you can step through a DECLARE command and still reference that command later.

Breakpoints

Breakpoints are a debugging tool that have been around a long time. In Web Macros they are illustrated as a red circle next to whatever line of code they're marking. If you start playing a macro above a breakpoint, rather than continuing to the end, the macro will pause just before executing the command with the breakpoint. To add a breakpoint click on the grey box that runs to the left of the text editor portion of the Macro Editor at the desired line. You should see a red circle appear in that grey box after you click.

2.2.3 Macro File Management

Save

Once you have a macro worth hanging on to you can save that macro by clicking the save button in the macro editor toolbar(a disk icon). The file will be a text file with the extension wmacro (for web macro).

Open

You can open previously saved macros with the open button(a folder icon). You can only have one macro open at a time so you will be prompted to save changes to any previous macros you were editing.

New

The new macro icon (a page icon) will close any open macro (again prompting to save) and create a new untitled one.

2.3 The Log Window

The log window is a tool to help you trouble shoot any problems with a macro trying to run. There are three types of log messages; error, info, and trace. Each is written to its own tab while the tab labeled "All" catches all messages that Web Macros writes.

Logs are not saved automatically or archived. However, you can copy and paste information out of the log window or right click in a log window to save the contents as a text file.

You can hide or bring back the log window by selecting View → "Log Window" from the drop down menu. There is also a setting under Macro $->$ Settings that allows you to turn off loggin altogether.

2.4 Browser Tabs

Web Macros supports tabbed browsing much the same way that IE 7 and firefox do. The interface to the tabs is just a little bit more unique. In the tab toolbar at the top of the browser, you have the option to add or remove tabs. You can select the tab you want to view via the drop down menu on the left or by clicking the appropriate button from the list that expands along the right of the bar.

Separate from this toolbar at the right are the browser thumbs. By default this view is collapsed on the right hand side of the browser. It consists of a thumbnail image for each browser window you have open. You can click the thumbnails to switch between tabs.

It is also worth noting that you can record the opening and switching between tabs. Start recording and navigating between tabs and you'll see the NEWTAB and SWITCHTAB commands record in the macro editor. See the command reference for a full description of these commands.

2.5 Macro Settings

From the macro drop down menu, if you select settings you will get a separate window with some advanced options. One example is "SuppressClearCommands". When true the macro recorder will never record a CLEAR command. This setting exists since CLEAR commands are often recorded unnecessarily because search box often grab the focus of the cursor. So when you click on something else, the focus leaves the empty search box and Web Macros thinks you purposely entered nothing into the box.

As you select a given setting, an explanation of what it is is provided in the box at the bottom of the settings window. For that reason, and because the settings may change frequently, all the settings will not be documented here.

2.6 ObjId Finder

To see the Object Id finder click View − > ObjId Finder from the drop down menu. You will see it appear below the macro recorder in the left hand pane. The purpose of the obj id finder is to help you come up with alternatives for identifiying an object on a page and also for sanity checks to make sure a given ObjId is selecting what you think it is.

One quick example you can try is visiting http://www.velocityscape.com and then entering 'a;5' in the text box provided by the ObjId Finder and clicking 'Find ObjId Synonymns'. You'll get back a;Solutions in addition to what you put in. So the link with the text solutions is the sixth link on that page (a;0 is the first). So try putting in a;0. Now you see no alternatives. This tells us that the first link must be surrounding an image or some other object with no id, name, or text.

2.7 My Information

To see the 'My Information' form click View − > 'My Information' from the drop down menu. This will launch a new form where you can fill in some of your own contact information. This information is stored in an xml file by web macros and loaded every time the program starts. You can use this information for easy form filling via a set of global variables. So rather than having to say:

```
ENTER input;addr "My address information"
```

You can just use:

```
ENTER input;addr #ADDRESS
```

This way you can update your contact information in one spot and not have to edit all macros that use that information.

Note: If you are using Web Macros Free you are required to fill out some of this information and it is submitted to us when you first use the program. It is not resubmitted to us as you update your information. If you own Web Macros then your information is never submitted to us.

Chapter 3

Getting Started

Chapter 4

The Web Macros Language

This section will go over all the commands that exist in the Web Macros language, as well as explain the common pieces that make up any command. Using this information you can tweak commands after recording or add new commands altogether to make your macros loop or branch based on certain conditions. This is where you'll find what you need to become a Web Macros power user.

4.1 Anatomy of a Web Macros Command

At most a command can have three parts. Consider the following command:

```
ENTER input;q ‘‘Web Macros’’
```

This command contains all three parts. They are separated by whitespace (one or more tabs and/or spaces). The first part is the command name, the second is the object id, and the last part is a value string.

4.1.1 The Command Name

The command name indicates to Web Macros what type of action to take. In this case, the command ENTER indicates that text is to be entered in some text or password box. See the command reference later in this document for a complete list of all commands. Even though the command name is always shown in all caps, it is not case sensitive. The capitalization is just a naming convention. However, command names will never contain whitespace.

4.1.2 Object Ids

The object id of the above command is input;q. The object id tells the command which object on the current page to operate on. Different parts of the object id are separated by semicolons. The most common form is this:

$< tagname >; < id|name|index >$

"input;q" is in the above form. It is referring to the first input tag with the name or id of "q". This happens to be the textbox that you enter your search into on www.google.com. input;0 would refer to the very first input tag to appear on a given page. Another popular form of object id is below:

$< tagname >; < text >$

This is the default format of an object id that is recorded when a link is clicked. This tends to make macros more readable. For example, a;"Next Page" will refer to the link on the page that contains the text "Next Page". Notice that an object id can contain whitespace if it is surrounded by double quotes. Also, the default object id is not necessarily what you want to use. If you know that the text of the link is unimportant to you, but you just want to click the first link, then use a;0 instead of a;"First Link Text".

Finally, there is one more form that is specific to links:

$< tagname >; < href >$

where href is the URL that the link leads to. This isn't a very useful form, since in most cases 'CLICK a;/mypage.htm' and 'GO http://www.mysite.com/mypage.htm' are the same thing. But it does come in handy when Regular Expressions are used in an object id. This is discussed later.

Duplicate Object Ids

It is possible for the same object id to identify more than one object on a page. For example if two links have the exact same text. By default, the first occurence of a match is used, but it is possible to specify exactly which instance you want. To do so use this form:

$< tagname >; < tagmodifier >; < zerobasedindex >$

So a;"Next Page";0 is the first link that has the text "Next Page" and a;"Next Page";1 is the second.

Regular Expressions

One last feature of object ids that can be extrememly powerful is the ability to use regexes. These are explained in general a little later in this section. But one common example of using regexes with object ids is following links. Say you want to follow each link on a page

of 10 search results. You notice that each link has the text “details.aspx” in the url. You might want to click

```
a;~".*details\.aspx";0
```

and then

```
a;~".*details\.aspx";1
```

Later in the Moving Forward section we will tie variables into the mix to do some really powerful web automation.

4.1.3 Value Strings

A value string is text that the command will either enter into a text box or use to further identify what to select. In the above example the value string is “Web Macros”. If no whitespace is in the value text then double quotes are not needed.

4.1.4 Regular Expressions (Regexes)

A regular expression is a way of expressing a pattern that can be applied to text to check for a match. ‘[0-9]’ is the same as saying “a single character that is either 0,1,2,3,4,5,6,7,8, or 9”. A ‘.’ is the same as saying any single character. A ‘*’ indicates that the thing before it should appear 0 or more times. So say you had the regex ‘[0-9].*’. This would match ‘0’, ‘2d’, or ‘938189aidklakekfi’. There are an abundance of online tutuorials on regexes that can be found online. It is not something unique to Web Macros.

When you are using a regex in Web Macros you must surround a pattern with ~“”. Currently, regexes can be inserted into object ids and also into the SourceContains format of the IF command.

4.1.5 Variables Overview

Web Macros supports three types of variables. Just like you might use ‘x’ as a variable in an equation, you can use variables in Web Macros to represent some data. This data is always stored as text, but may be interpreted as an int depending on the context. See the IF command and string vs integer comparison for more on this.

All variable types in Web Macros must never contain whitespace

Standard Variables

Standard variables must start with an @ and are used to store a single string of text or an integer. You can use them to hold a piece of text that you use in several commands throughout a macro for easy maintainability or as a way of keeping track of how many errors have occurred, or how many iterations of a loop to perform. See the DECLARE, SET, and INCREMENT command for more examples.

Table Variables

Table variables hold the result of a query to a database or other data source. They must start with a $ instead of an @. You can index different rows and columns in a table variables with square brackets. For example $myTableVar[0][1] would be the first row, second column of the table. Also $myTableVar.Count will return the number of rows the table is holding. See the DECLARE and SET command for more information.

The syntax for the connection string used with Web Macros are OLEDB connection strings. So for example, to connect to MSSQL (The Local SQL Instance that comes with Web Scraper in particular) you would use:

```
"Provider=SQLOLEDB;Password=machupicchu;Persist Security Info=True;
User ID=providus_sys;Initial Catalog=hamiltonmunicipalcourt;
Data Source=(local)\ProvidusStd"
```

The exception to this is MySQL or any ODBC connection. Here, you still have the PROVIDER listed as if an OLEDB connection string, but then just add the connection string as you normally would. So an example MySql connection string would be:

```
"Provider=MSDASQL.1;Driver={MySQL ODBC 3.51 Driver};
Server=localhost;Database=; User=root;Password=12345;Option=3;"
```

and not

```
"Provider=MSDASQL.1;Password=12345;Persist Security Info=True;
User ID=root;Extended Properties=\"Driver={MySQL ODBC 3.51 Driver};
Server=localhost;Option=18475;\";Initial Catalog=;"
```

like you would expect with ODBC. We apologize for the quirk in sytax but in the end it actually translates to better performance.

Gloabl Variables

Global variables work the same way as standard variables, except there is no way to declare them. They already are defined and are availible for you to use at any time. They always

start with a #. The global variables currently availible are:

#NEWID Generates a random GUID (Global Unique IDentifier). An example would be, c5f3c793-691b-4898-8770-a462b034cf47. Good for saving a large number of files that don't need descriptive names.

#CURRENTTIMESTAMP The current date and time. More formatting options for this should be coming soon.

#ALERTTEXT The text of the last JavaScript pop up alert that occured. Web Macros kills these pop ups to avoid stopping automatic playback, but this text allows you to still analyze what happened

#ALERTCAPTION The caption of the last JavaScript pop up alert that occured. The caption is what appears in the title bar of a pop up message.

#FIRSTNAME The first name as entered into 'My Information'

#LASTNAME The last name as entered into 'My Information'

#EMAIL The email as entered into 'My Information'

#COMPANY The company as entered into 'My Information'

#ADDRESS The address as entered into 'My Information'

#CITY The city as entered into 'My Information'

#STATE The state as entered into 'My Information'

#ZIP The zip as entered into 'My Information'

#COUNTRY The country as entered into 'My Information'

#COMMENTS The comments as entered into 'My Information'

4.2 Command Reference

The following is a list of all possible Web Macro commands by category.

4.2.1 Navigation Commands

These commands are directly related to interaction with browser buttons such as GO and BACK.

BACK

Usage *BACK*

Not Implemented. Coming in a later verison.

Description The equivalent of hitting the "Back" button on a browser.

FORWARD

Usage *FORWARD*

Not Implemented. Coming in a later verison.

Description The equivalent of hitting the "Forward" button on a browser.

GO

Usage *GO < Url >*

Description Navigates to the specified URL. It is the equivalent to typing a URL in the browser and clicking the GO button or hitting Enter.

Parameters:

- Url - The fully qualified URL you wish to visit.

Example

```
GO http://www.velocityscape.com
```

NEWTAB

Usage *NEWTAB*

Only applicable to Web Macros Free. Does not work in silent mode.

Description Opens a new tab in the Macro Browser and sets the focus to this tab. Any following commands will be applied to the newly opened tab.

Example

```
//This example will open up the two main sites that Velocityscape publishes.
//Each will have its own tab.
//*************************************************************
GO http://www.velocityscape.com/
NEWTAB
GO http://www.spyfu.com/
```

REFRESH

Usage *REFRESH*

Description The equivalent of clicking the refresh button in the browser.

Example

```
//This example will get the latest headlines from Google News
//every half hour and save the results
//*************************************************************
GO http://news.google.com/
LABEL again
//wait a half hour (in ms)
WAIT 18000000
//get the latest news
REFRESH
//save the news with a random id in the file name
SAVE C:\WebMacros\p_#NEWID.htm
//loop forever
JUMP again
```

SESCLEAR

Usage *SESCLEAR*

Description Clears all cookies from the current browser status. In the background, this gets a completely new MSHTMLDocument object.

SWITCHTAB

Usage *SWITCHTAB < index >*

Only applicable to Web Macros Free. Does not work in silent mode.

Parameters

- index - The index of the tab to switch the focus to. 1 is the first tab.

Description Switches the focus of the browser to the specified tab. All following commands will be applied to this tab unless NEWTAB or another SWITCHTAB command is called.

Example

```
//This example will open up the two main sites that Velocityscape publishes.
//It will then switch between tabs every 5 seconds.  Not that useful, but
//it should demonstrate the command.
//*********************************************************
//open up a couple of cool sites
GO http://www.velocityscape.com/
NEWTAB
GO http://www.spyfu.com/
//constantly switch between the two tabs every 5 seconds
LABEL loop
SWITCHTAB 1
WAIT 5000
SWITCHTAB 2
WAIT 5000
jump loop
```

4.2.2 Browser Action Commands

These command are related to actions you would take within the actual browser window. Primarily, this includes filling our forms.

CLEAR

Usage $CLEAR < ObjId >$

Description Removes all text from any form of text box.

Parameters

- ObjId - The id of the text box on the page to be cleared.

Example

```
//If you step through this example, the text ‘‘Web Scraping’’ will
//appear and then be cleared from the search box on SpyFu.com
//*********************************************************
GO http://www.spyfu.com
ENTER input;ctl00_SearchBox1_SearchTextBox ‘‘Web Scraping’’
WAIT 1000
CLEAR input;ctl00_SearchBox1_SearchTextBox
```

Valid HTML Elements

- input.text
- input.password
- input.hidden
- textarea

Events Thrown

- focus
- keyup

- keydown
- keypress
- blur

CLICK

Usage $CLICK < ObjId >$

Description Clicks an object on the current page. While most commonly a button, this can be anything that you can form an object id for.

Parameters

- ObjId - The id of the object to be clicked when no value is specified. Otherwise, this is some parent of the object to be clicked.

Examples

```
//Here is an example without using the optional value parameter
//It will search for the term ‘‘Web Scraping’’ on SpyFu.com
//*******************************************************
GO http://www.spyfu.com
ENTER input;ctl00_SearchBox1_SearchTextBox ‘‘Web Scraping’’
CLICK  input;ctl00_SearchBox1_SearchButton
```

Valid HTML Elements

- anything

Events Thrown

- focus
- click
- blur

DOWNLOAD

Usage $DOWNLOAD < Path >< Url >$

Description Intended to save binary files such as reports or executables. Saves the indicated binary file to a given file path.

Parameters

- Path - The local file path and name to store the downloaded file at.
- Url - The Url that points to the file to download

Example

```
//This will save a copy of WebScraperPlus.exe, another Velocityscape product.
//Note that the first two commands actually aren't necessary, but reflect what you would
//probably record
//**********************************************************
GO http://www.velocityscape.com/
CLICK a;Downloads
DOWNLOAD C:\Downloads\WebScraper.exe http://www.velocityscape.com/download/WebScraperPlu
```

Valid HTML Elements Right now, only links are valid (a tags). This should eventually work with buttons and dynamic reports as well.

Events Thrown

- focus
- click
- blur

ENTER

Usage $ENTER < ObjId >< Value >$

Description Enters the text specified in the value parameter into the specified object on the page.

Parameters

- ObjId - The id of the text box to be modified.
- Value - The text to be entered into the text box.

Example

```
//This will search for the term ‘‘Web Scraping’’ on SpyFu.com
//*******************************************************
GO http://www.spyfu.com
ENTER input;ctl00_SearchBox1_SearchTextBox ‘‘Web Scraping’’
CLICK  input;ctl00_SearchBox1_SearchButton
```

Valid HTML Elements

- input.text
- input.password
- input.hidden
- textarea

Events Thrown

- focus
- keyup
- keydown
- keypress
- blur

SELECT

Usage $SELECT < ObjId > [< Value >]$

Description Selects the specified radio button, check box, or drop down list item. The Value parameter is used for drop down elements and check boxes but is omitted in the case of radio buttons.

Parameters

- ObjId - The id of the object to be selected
- Value - In the case of a drop down list, the text or value of the item in the drop down list. In the case of a check box, the value or text of the desired check box.

Example

```
//This macro checks out what Amazon has in the ping pong department
//we select the Toys and Games category first to help narrow our search.
//*********************************************************
GO http://www.amazon.com/
SELECT select;0 "Toys & Games"
ENTER input;twotabsearchtextbox "ping pong"
CLICK input;2
```

4.2.3 Branching/Conditional Commands

These commands are related branching or jumping withing a macro. Using these commands you can make a macro loop or skip over certain segments of code based on certain conditions.

IF

Usage $IF < VarName > [BEGIN...END]$
$IF < ObjId > [BEGIN...END]$
$IF[!]SourceContains < SourceCode > [BEGIN...END]$
$IF < LeftHandValue >== \mid = \mid ! = \mid < \mid > \mid <= \mid >=< RightHandValue > [BEGIN...END]$

Description The IF command will execute the next line in the macro if the statement evaluates to true. Otherwise, it will skip the next line in the macro. Alternatively, you can follow the IF command with BEGIN and END command to bracket more than one line that you want to execute if the statement evaluates to true.

There are four current forms. The first is passing a variable name to the IF command. If the variable is defined then the statement will evaluate to true.

The second is passing an object id to the IF command. If that object is found on the page then true is returned.

The third is using the "SourceContains" keyword. The will return true if that supplied piece of source code appears anywhere in the current page. If the optional ! is used then the

meaning is reversed and it is the equivalent of NotSourceContains. Both Regexes and value strings can be used in the SourceContains form. See Anatomy of a Web Macros Command above for more information.

The last is either a string or an integer comparison. If both the left hand and right hand arguments can be converted to integers then an integer comparison will occur. Otherwise, the two values will be compared as string. Either variable or hard coded values can be used on either side of a comparison. Currently only equality and inequality are valid for string comparison. All other operators are valid for integer comparison. Both == and = are valid equality operators to allow flexibility for programmers and non programmers.

Parameters VarName - Any valid variable name, whether it is declared or not. This means it begins with an @ symbol and contains no whitespace. SourceCode - A piece of source code to be searched for on the current page LeftHandValue - The left hand value of the comparison RightHandValue - The right hand value of the comparison

Examples

```
//First we'll look at two examples that show how the first form of the if command works
//*************************************************************
//Example 1: no variable defined, skip the command that enters text into the search box
DECLARE @MyVar
GO http://www.spyfu.com/
IF @MyVar
ENTER input;ctl00_SearchBox1_SearchTextBox "Web Scraping"
CLICK input;ctl00_SearchBox1_SearchButton

//Example 2: the same idea, but here the variable is defined.  So we actual enter the te
DECLARE @MyVar
GO http://www.spyfu.com/
IF @MyVar
ENTER input;ctl00_SearchBox1_SearchTextBox "Web Scraping"
CLICK input;ctl00_SearchBox1_SearchButton

//*************************************************************
//Now, let's use the IF command to check if an object is on a page
//Note: here we use the BEGIN and END command to perform
//multiple commands if true is returned
//*************************************************************
//this macro will go to google, but first check that the text box is on
//the page before performing a search
GO http://www.google.com
```

```
IF input;q
BEGIN
ENTER input;q "My Search"
CLICK input;btnG
END


//************************************************************
//Next we'll look at how the SourceContains keyword works
//************************************************************
//Say velocityscape has been fiddling with a few differnt link names
//, but we know we always want to follow the Contact link.  So take a
//few guesses at the spelling
GO http://www.velocityscape.com/
IF SourceContains ">Contact Us</a>"
CLICK a;"Contact Us"
IF SourceContains ">Contact</a>"
CLICK a;"Contact"
```

JUMP

Usage $JUMP < LabelName >$

Description Causes the execution of the macro to branch to the next command after the label name

Parameters itemize

LabelName - The name of a label somewhere else in the same macro.

Example

```
//This macro should skip over yahoo.com
//and go straight from google.com to velocityscape.com
//************************************************************
GO http://www.google.com
JUMP Vel
GO http://www.yahoo.com
LABEL Vel
GO http://www.velocityscape.com
```

LABEL

Usage $LABEL < LabelName >$

Description Marks a location in the macro with a Label, to be used by the JUMP command.

4.2.4 Variable Declaration/Modification Commands

These are commands that have to do with declaring and modifying variables to be used within a macro.

DECLARE

Usage $DECLARE < VarName > [< Value >]$ $DECLARE < TableName > [< ConnectionString >< Query > [< TimeOut >]]$

Description Adds either a variable or a table to the current macro's state. Either can be left empty or optionally initialized. A varaible is initialized by some text (could be number as well) while a table is initialized with the results of a database query. Tables are only availible in the full version of Web Macros.

Parameters itemize

VarName - A valid variable name (beginning with "@" and not containing whitespace

Value - An optional initializing value for the new variable. When used, it is the equivalent of doing a DECLARE immediately followed by a SET.

TableName - A valid table name (beginning with "" and not contianing whitespace

ConnectionString - the connection string for the database to run the query against

Query - the actual query to run against the databse. The results will be stored in the table variable. A value string, so whitespace is allowed if the entire string is surrounded with doubel quotes.

Timeout - an optional time out in seconds for the query being executed.

Example

```
DECLARE @myvar ‘‘Web Scraping’’
GO http://www.spyfu.com
ENTER input;ctl00_SearchBox1_SearchTextBox @myvar
CLICK  input;ctl00_SearchBox1_SearchButton
```

INCREMENT

Usage $INCREMENT < VarName >$

Description Adds 1 to the current value of the variable indicated by VarName. The variable being modified must be an integer.

Parameters

- VarName - The variable to be incremented.

Example

```
//this will search for ‘1’ instead of ‘0’
DECLARE @myvar 0
INCREMENT @myvar
GO http://www.spyfu.com
ENTER input;ctl00_SearchBox1_SearchTextBox @myvar
CLICK  input;ctl00_SearchBox1_SearchButton
```

SET

Usage $SET < VarName >< Value > SET < TableName > [< ConnectionString >< Query > [< TimeOut >]]$

Description When a varname is given, assigns the indicated value to the variable identified by VarName. Otherwise, will fill a table with the results of the indicated query. Tables are only availible in the full version of Web Macros.

Parameters itemize

VarName - A valid variable name (beginning with “@” and not containing whitespace

Value - The value the variable will take on.

TableName - A valid table name (beginning with “” and not contianing whitespace

ConnectionString - the connection string for the database to run the query against

Query - the actual query to run against the databse. The results will be stored in the table variable. A value string, so whitespace is allowed if the entire string is surrounded with doubel quotes.

Timeout - an optional time out in seconds for the query being executed.

Example

```
//This is just a variation on the DECLARE example.
//Here we set the value in a separate command
DECLARE @myvar
SET @myvar ‘‘Web Scraping’’
GO http://www.spyfu.com
ENTER input;ctl00_SearchBox1_SearchTextBox @myvar
CLICK  input;ctl00_SearchBox1_SearchButton
```

4.2.5 Console Commands

These commands all start with CONSOLE. and are designed specifically for running a macro in silent mode with WebMacrosConsole.exe. WebMacrosConsole only comes with the full version of Web Macros.

CONSOLE.EXIT

Usage $CONSOLE.EXIT[< ExitCode >]$

Description Ends the macro and exits the entire process.

Parameters itemize

ExitCode - The integer return code returned to the operating system. 0 if not specified (no errors).

Example

```
//just a quick example of how you might check a condition and return either
//success (0) or failure (100 in this case)
//*********************************************************
GO http://www.velocityscape.com/
IF SourceContains "Site Down"
BEGIN
  CONSOLE.EXIT 100
END

CONSOLE.EXIT 0
```

CONSOLE.WRITE

Usage $CONSOLE.WRITEText[< Stream >]$

Description Writes the text specified to standard output, or standard error if that is specified.

Parameters itemize

Text - The text to be written

Stream - STDOUT for the standard output stream. STDERR for standard error. STDOUT is the default if not specified.

Example

```
//report a loop iteration
//*********************************************************
GO http://www.velocityscape.com/
DECLARE @i 0

LABEL loop
REFRESH
CONSOLE.WRITE "Refreshed @i times"
INCREMENT @i

IF @i < 10
BEGIN
```

```
JUMP loop
END
```

4.2.6 Other Commands

RUNSQL

Usage $RUNSQL < ConnectionString >< Query > [< Timeout >]$

Description Runs the query specified. Use this command for all database commands you want to perform that do not return a table (Update, Insert, Delete, Truncate, etc)

Parameters itemize

ConnectionString - the connection string for the database to run the query against

Query - the actual query to run against the databse. The results will be stored in the table variable. A value string, so whitespace is allowed if the entire string is surrounded with doubel quotes.

Timeout - an optional time out in seconds for the query being executed.

EXEC

Usage $EXEC < CommandLineText > [SYNC[timeout]|ASYNC]$

Description Executes the supplied text at the command line. By default processes are started asynchronously (meaning that the macro will continue running and not wait for the process to finish). Alternatively, you can explicitly indicate synchronous or asynchronous execution.

Parameters itemize

CommandLineText - The text to be run at the command line.

timeout - the timeout in ms for the SYNC option of the EXEC command. 30 min is the default.

Example

```
//will start the calculator and then navigate to Velocityscape.com
EXEC calc
GO http://www.velocityscape.com

//will not go to Velocityscape.com until calculator is closed
EXEC calc ASYNC
GO http://www.velocityscape.com
```

SAVE

Usage $SAVE < FileName >$

Description Saves the current web page (source code only) to the specified file path.

Parameters itemize

FileName - The target file to save the current page to. The directory must exist before hand.

Example

```
//This will search for the term ‘‘Web Scraping’’ on SpyFu.com
//and save the results to the C drive
//*********************************************************
GO http://www.spyfu.com
ENTER input;ctl00_SearchBox1_SearchTextBox ‘‘Web Scraping’’
CLICK  input;ctl00_SearchBox1_SearchButton
SAVE C:\WebScrapingResults.htm
```

WAIT

Usage $WAIT < milliseconds >$

Description Causes the current executing macro to wait the specified number of milliseconds.

Parameters itemize

milliseconds - An integer that is the number of milliseconds you want the macro to wait.

Example

```
//waits 5 seconds before going to Velocityscape.com
GO http://www.spyfu.com
WAIT 5000
GO http://www.velocityscape.com
```

Chapter 5

Moving Forward