

UBot Commando Guide

Command Reference

www.botsoftware.org



Version 3.0065

1. The Qualifiers
2. Action Commands
3. Flow Commands
4. Chosen Commands
5. Variable Commands
6. UI Elements
7. Constants
8. Additional Features

UBot

“the possibilities are almost endless”

**“put...commercial
scripts to shame”**

“very impressive”

“a thousand things I can use it for”

SEO
Keyword research
Standalone .exe exports

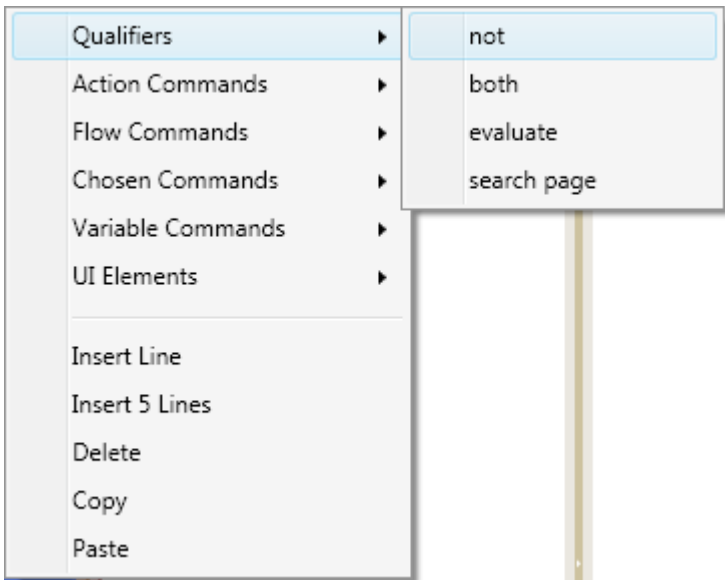
Site submissions
Account creation
Link building

Simple scripting
Create in minutes
Free bots included

**Automate
Everything.**

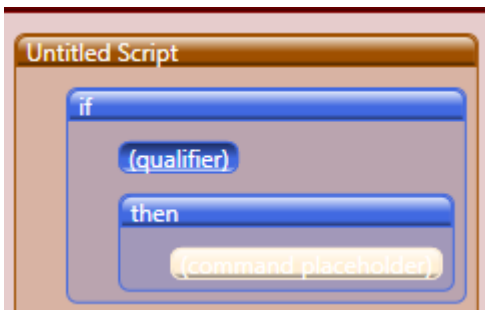
Watch the video at
www.botssoftware.org

1. The Qualifiers:



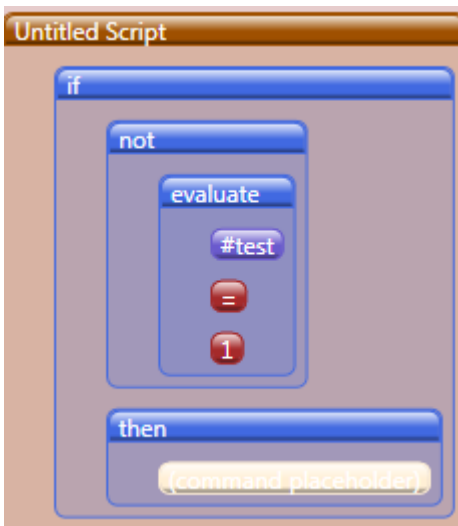
Qualifiers are used within conditional statements to direct the flow of a script.

If (qualifier) then (action).



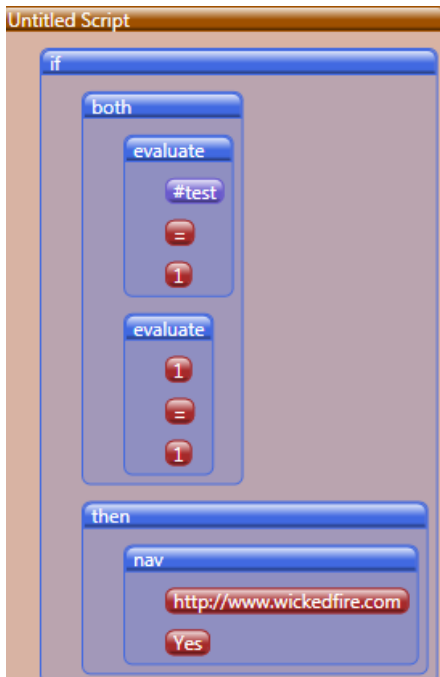
not:

This will return true when its subnodes do not return true. Using *not* combined with the *evaluate* qualifier allows you to compare variables or other portions of a script. In the below script, the *not* qualifier was added before the *evaluate* qualifier to determine if the variable #test is equal to the number "1". If it is *not* equal to 1, then the action (unspecified here) will be taken. Otherwise the script will continue past that action.



both:

This qualifier will return true only when all of its subnodes return true. In the following example, the script will only visit www.wickedfire.com if the variable #test contains the value "1". (The number of children beneath the "both" qualifier is unlimited.)

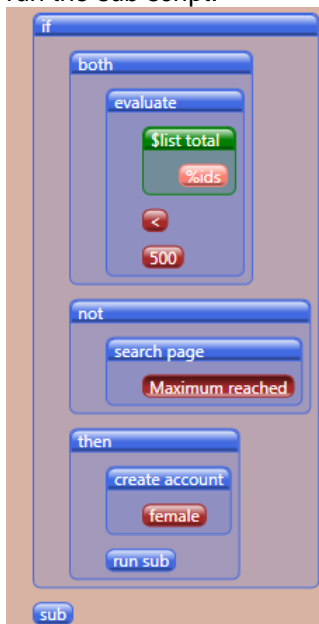


either:

Either will return true when any of its subnodes return true.

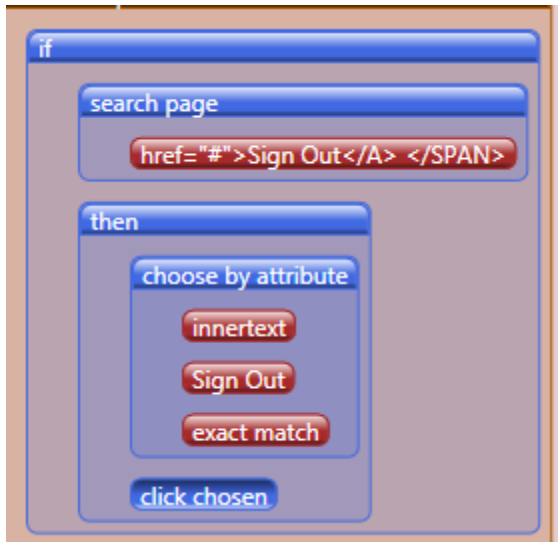
evaluate:

evaluate is the most common of all the qualifiers. The *evaluate* qualifier allows you to compare any two values, which will allow an action to run if they match (or don't match when using *not*). In the below example, the value of the list information contained in %ids must be less than 500, and the page must not contain the words "Maximum reached" for the action after the "then" command to fire. If both circumstances are true, then the script will run the create account command and run the sub script.

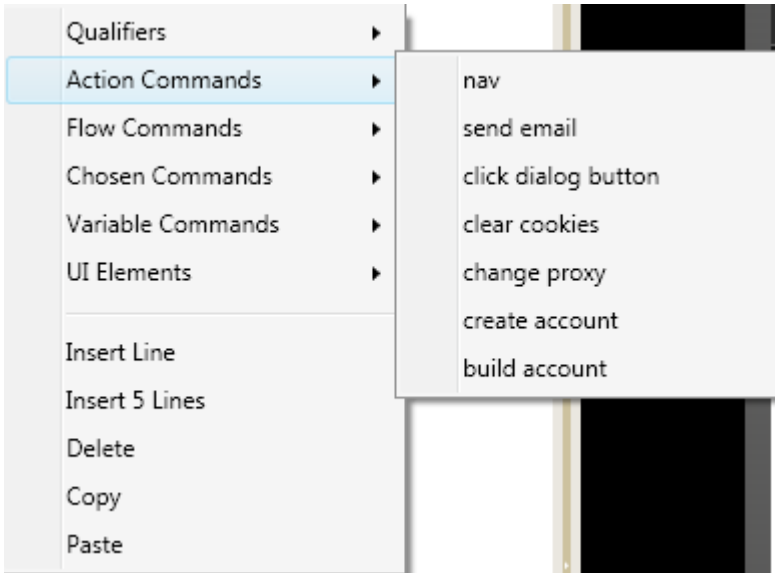


search page:

This is a qualifier that will return true if the specified text is present on the page. The format of the words being searched must match what is on the page, so it is best to highlight the information you want to search the page for and then bring up the "search page" command. This automatically fills in the values with the html of what you had highlighted, which will allow you to be sure that you don't misidentify the text that you are seeking (by leaving out a hidden html item between words or sentences, for example). The following will search the page for the code `href="#">Sign Out ` and if found, will click the "Sign Out" button.



2. Action Commands



nav:

The nav command simply visits whatever URL you choose. The only parameters are the url and Yes and No, which correspond to whether or not the script should wait until the site finishes loading before moving onto the next command. This is set to default and will be chosen most of the time.

send email:

The send email command allows UBot scripts to connect to smtp email accounts and create and send emails. These emails could contain information relevant to the script that's being run - for instance, you could email someone a list of scraped page information. Or you could scrape email addresses and email the client yourself! In the below instance, the script uses UBot's create account command and then sends an email containing a snippet of sales conversation to the recipient, some of which is determined by the randomized account that is created.



click dialog button:

The click dialog button command allows for the pressing of any (usually javascript) dialog button that appears on the screen. Sometimes these will be in the form of warnings or alerts (ie – “You are now navigating away from Nationalcitybank.com. Please press OK to continue.”), and other times they will be to tell you that something has been done incorrectly...Perhaps you’ve forgotten to fill in a form field on an account creation page. The command requires only one parameter – the text of the button that requires clicking. (This is not the window text, but the text of the button itself, usually something like “OK”.) The below script will visit an example page, bring up a dialog window, wait 5 seconds, and click it.

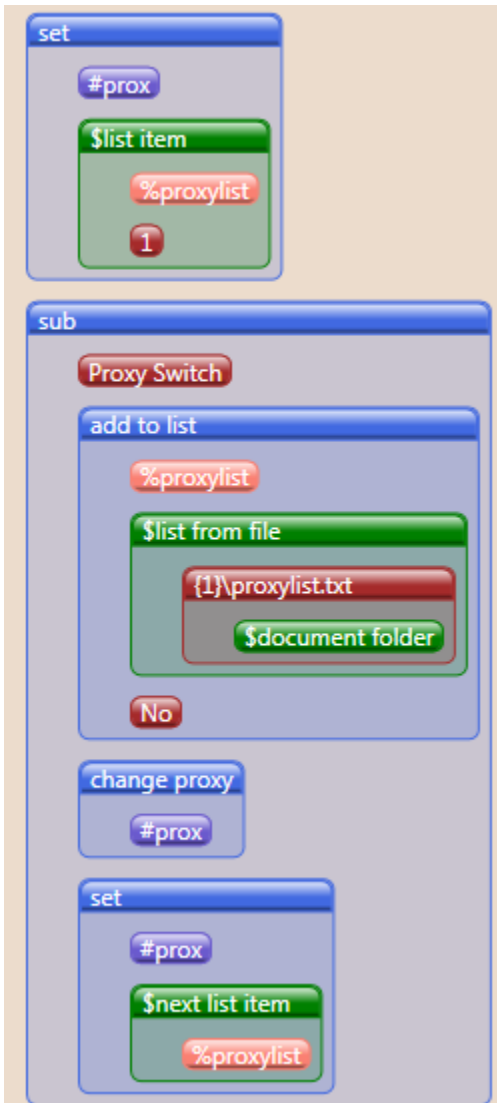


clear cookies:

This command is useful if you want to visit a site as more than one user but the site regularly records your visits in a cookie file.

change proxy:

This command lets you flip proxies. You can do this using a list from a file, or with a variable, or with specified proxies. You could, for example, search a page that contains proxies, test them, and implement them all in a single script. The script below contains a "sub" function that will load up a list of proxies and then pull one out from the first line and enter it into the variable (#prox). Then it will change the proxy currently used to that variable (#prox). After this it will pull a new proxy out so that when run again it will use a new proxy.



Build account:

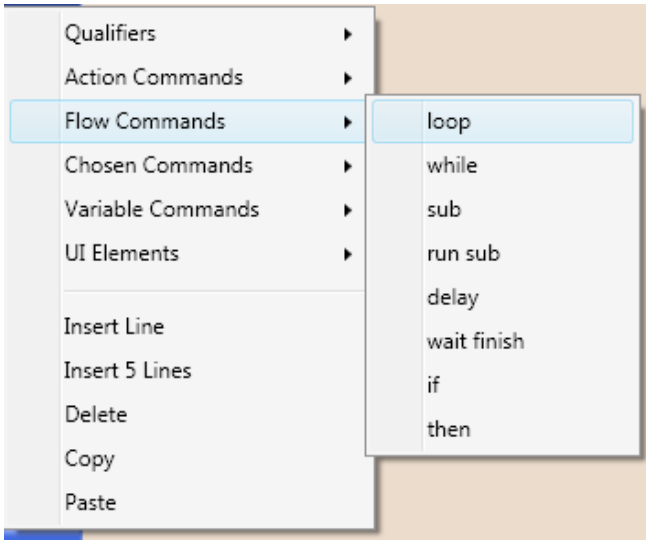
Build account works the same way as create account but allows you to specify the username, email address, and password. The below script will create the following, slightly different result as the one above:

The image shows a web interface for building an account. It consists of three main panels on the left and a search bar on the right.

- build account**: Contains buttons for 'female', 'myusername', 'myemail@email.com', and 'helloworld'.
- choose by attribute**: Contains buttons for 'name', 'as_q', and 'exact match'.
- change chosen attribute**: Contains a 'value' button and a list of attributes from 1 to 10: (1) (2) (3) (4) (5) (6) (7) (8) (9) (10). Below the list are buttons for '\$birth year', '\$birth month word', '\$birth day', '\$zip code', '\$password', '\$email', '\$username', '\$password', '\$last name', and '\$first name'.

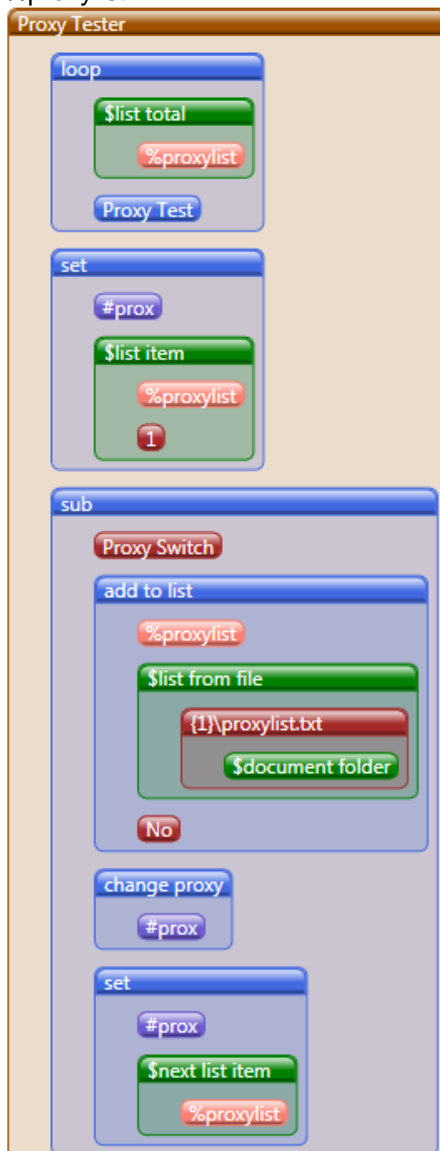
On the right, a search bar contains the text: '1987 November 18 81081 helloworld myemail@email.com myusername helloworld McMahon Nettie'. Below the search bar are two 'OR' operators and empty input fields.

3. Flow Commands



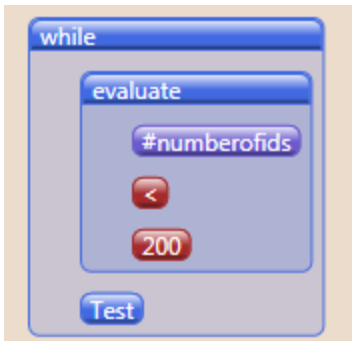
Loop:

The loop command allows anything contained within it to repeat a given number of times. That number can be a numeral (from 1 all the way up to infinity) or it can be the contents of a variable. Note that you can insert more lines above the command place holder. In the following example, the loop command will repeat as many times as there are lines in the list %proxylst.



While:

The while command functions similarly to loop. This command will run all contained commands for as long as its qualifier remains true. Note that you can insert lines above the command placeholder. In the below script, the sub script “Test” will repeat until the #numberofids is greater than 200. This command is ideal for any repeated action that needs to continue until a certain number is reached.

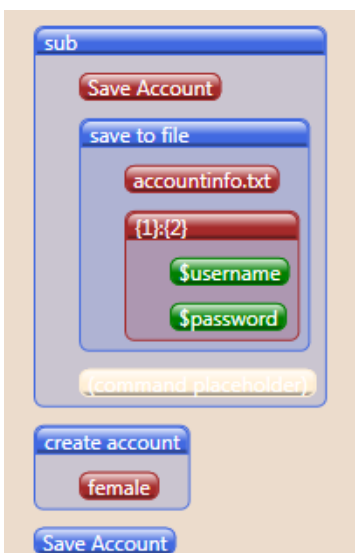


Sub / Run sub:

The sub command is one of the most useful features in UBot. A sub is like a self contained script within a script. It helps to create cleaner and more maintainable code. When you have a sub in your script, it will not run until you call it using the run sub command. Besides cleaner scripting and easier organization, it allows you to create an independent script that can be called by any other portion of the bot. Again—if you have a single bot that contains multiple scripts, any one of the scripts can call a sub routine that is in any of the other scripts. This allows you to, for example, create a separate script for each section of a keyword research tool, and then combine them all in a single script by calling each of the subs in the appropriate order.

To use the sub command, first choose “sub” in the flow command menu. Name your sub routine. Once the sub exists, you can either drag and drop previous script elements into the blank node inside the sub, or you can begin a new section of script there.

Anything contained within the “sub” command won’t occur on its own; rather, it must be called by the “run sub” command. Once a sub has been created, it will not run until you tell it to with this command. This command simply allows you to choose the sub you wish to run. In the below example, the sub routine “Save Account” will not occur until after the create account command, because only then is the sub run command executed.



Delay:

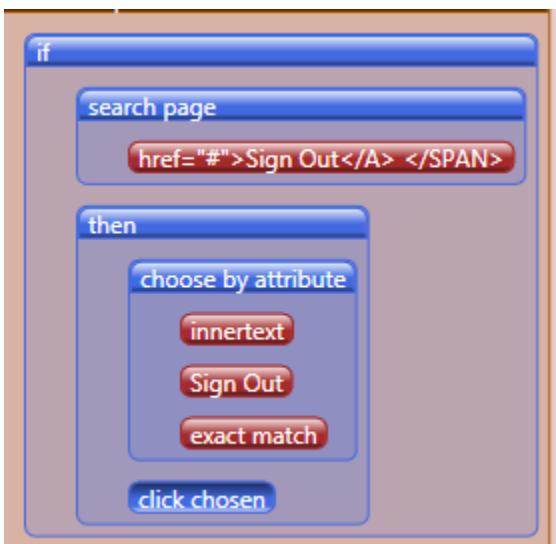
The delay command waits a specified number of seconds before continuing with the next command. – it suspends the script for the specified number of seconds. This is probably the most simple command in all of UBot.

Wait finish:

Wait finish suspends the script until the webpage finishes loading. This is NOT necessary after the NAV command, as nav has it built-in. But the command is useful after clicking a button. While this is generally not necessary, some specific commands that require a dialog button is pushed, for example, may require wait finish or the script would otherwise not pause for the results of the button to occur. If your script appears to be ending too quickly, try adding a Delay or Wait Finish.

If / Then:

The “if-then” command sequence is a fairly common way to determine the flow of events in scripting. In UBot it works very simply. A qualifier (see above in section 1) follows the “if” command, and the action specified follows the “then” command. In the below example, the qualifier “search page” determines whether or not the “choose by attribute,” “click chosen” commands will be run. When the result of the qualifier is false, the child nodes within the “then” node will not be activated.



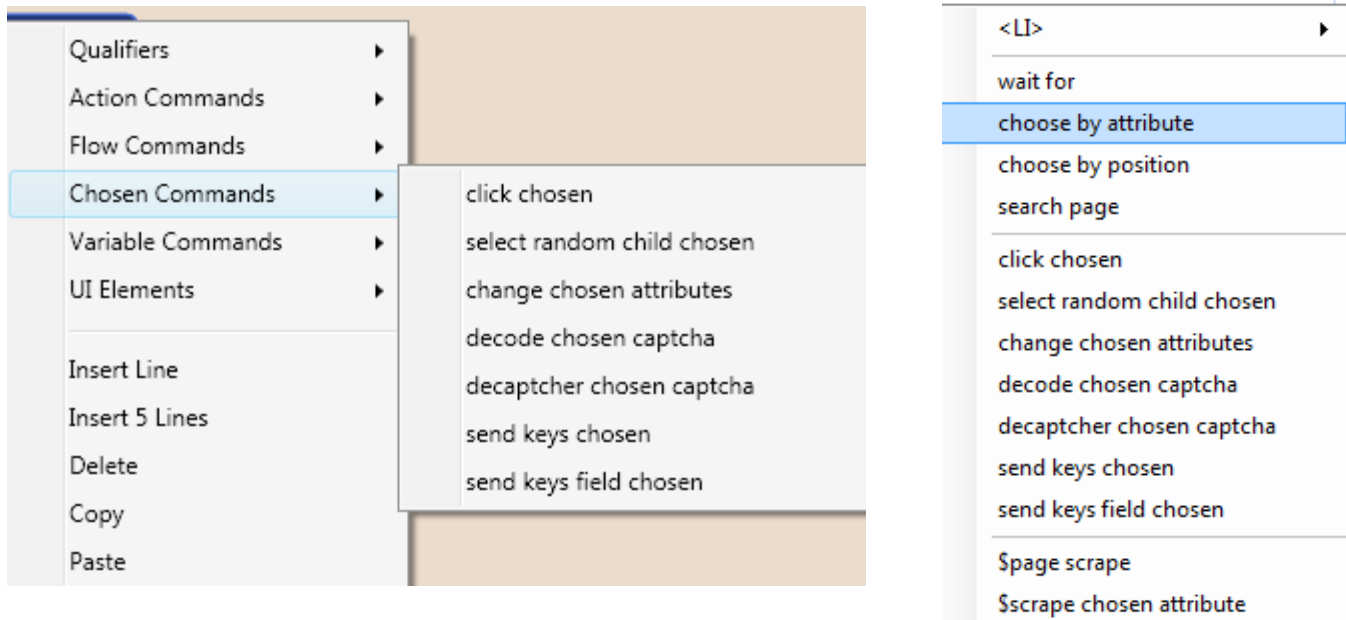
UBOT.

UNLEASH THE ONLY BOT YOU'LL EVER NEED

SEO
KEYWORD RESEARCH
UPLOADING

SITE SUBMISSIONS
ACCOUNT CREATION
LINK BUILDING

4. Chosen Commands



Wait For:

This command will suspend the script until the specified text appears on the web page. Works well with ajax.

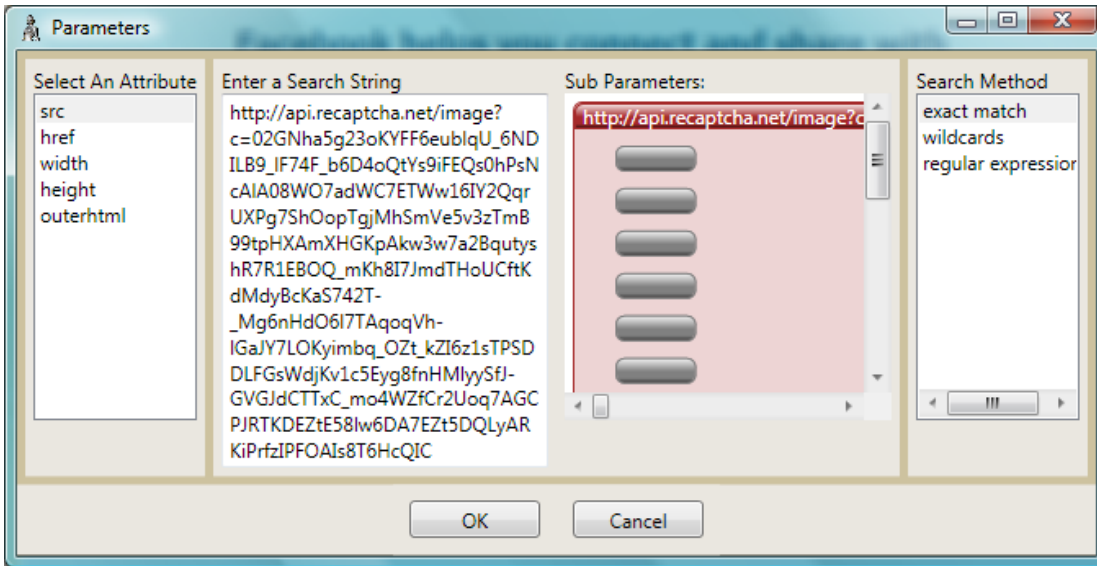
The chosen commands can be found when you right click a script node or when you right click in your browser window.

Choose by attribute:

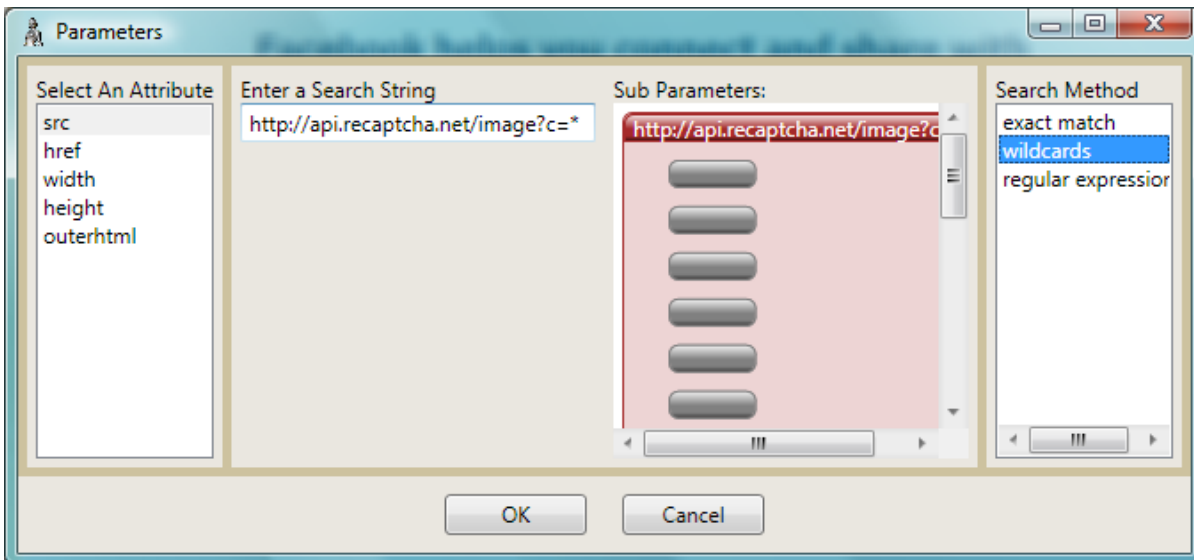
Choose by attribute is probably one of the most common commands that you will use in UBot, because it is part of the foundation of how UBot communicates with webpages. Before you can manipulate a part of a webpage, you must first choose it. You can choose an element of a page by any of its attributes. Selecting a value in the parameters window will display its current value under the search parameter. This is particularly useful if you need to choose something like a select box, or a checkbox. You can put it in the state that is most useful, and then when choosing it, the value will be ready to go. With checkboxes, use the "checked" attribute.

You can make the search value more flexible by using wildcards and regular expressions. Wildcard cards are * and ?. * will allow for any number of any character, while ? will allow for just one random character. Regular expressions are much more powerful search tools than wildcards, but are outside the scope of this explanation. Many tutorials for regular expressions can be found online. Most of the time, wildcards are enough. Because choosing elements can be simple sometimes and surprisingly tricky others, UBot has multiple methods for choosing elements by the attribute.

Let's say you wish to select an image on a page—perhaps a captcha that you'd like to solve. You have several options. If you are on Facebook's signup page and you right click their captcha image, and use the choose by attribute command, you will see the following:

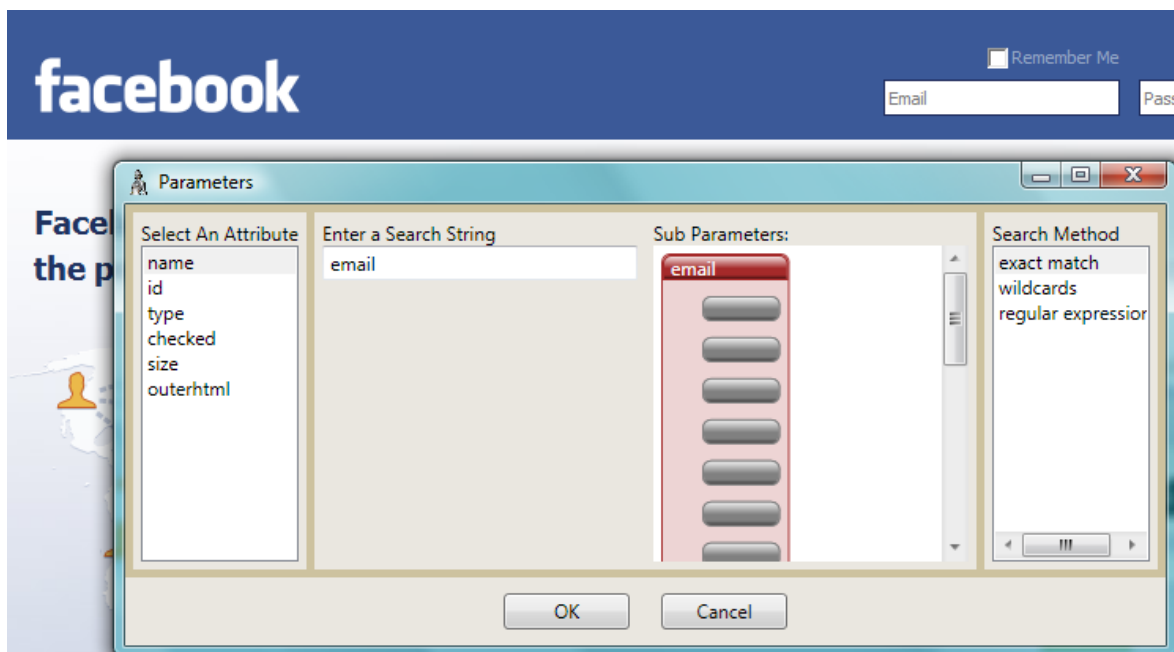


The best attribute for selecting a captcha is by “src”, but first you must limit the source because the URL is usually randomized. Changing the search method to “wildcards” and adding an asterisk (*) in place of the random portions of the search string, we can easily select the captcha every time it loads:



UBot recognizes all of the different available ways to select an element on the page, including by the element's name, id, type, size, innerhtml, outerhtml, height, width, and more. There should be almost no elements that UBot cannot select. Because some sites try to make portions of the page difficult to select, you will quickly find that the best attribute for selecting an element can vary from page to page.

Using the choose by attribute command on a text box will bring up a window like the following:



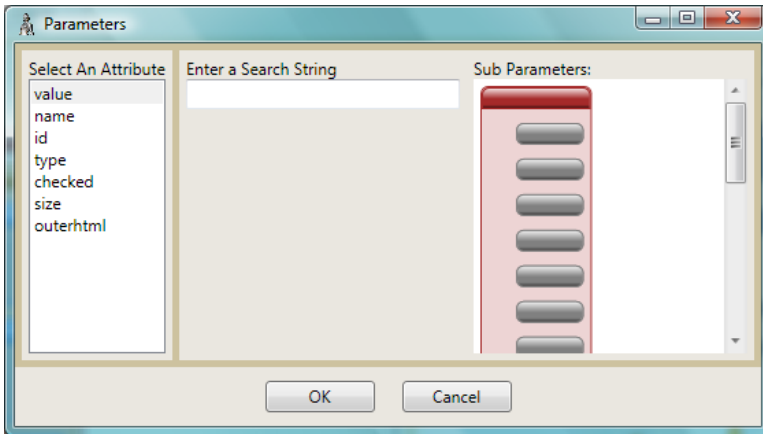
In this instance, you could use the “name” of the box as well as the “id”.

If one attribute isn’t working when you try to select it, there may be multiple elements on the page with the same attribute. With UBot’s many options, you will be able to find an attribute that works 99% of the time.

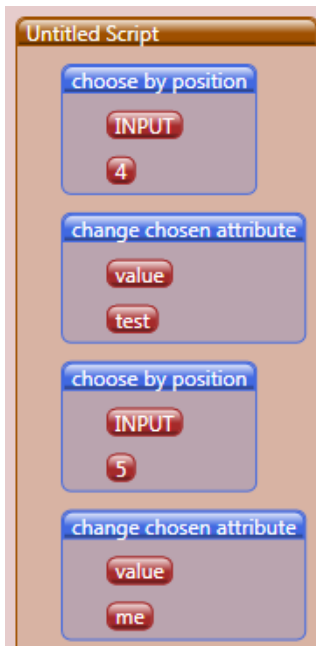
Choose by position:

Another option for choosing elements, choose by position will choose the element based on what the tag name is, and the order of said tags in the document. For instance, if the parameters are INPUT and 4, it will choose the 4th input tag on the page. Parameters are chosen automatically.

Change chosen attribute:



Your goal in many scripts will be to modify websites – whether by filling text boxes, clicking buttons, or choosing from drop-down lists. After choosing an element by attribute, you will often be using the “change chosen attribute” command to modify the page itself.

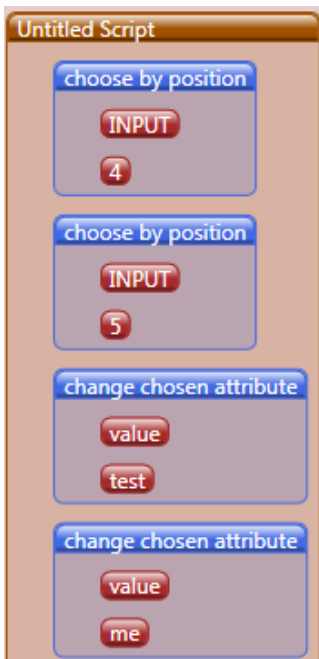


The change chosen attribute command allows you to modify any modifiable attribute of an element, regardless of which attribute you select. For example, you could select an attribute by its position and then modify the “value” attribute, as in the following example, which fills two separate text boxes with the words “test” and “me”.

(Special note about drop-down boxes: Frequently drop-down boxes “values” are not the same as the value that a user selects. For instance, a drop-down box with a list of months may have values of 1 through 12, even though the values say “Jan” or “Feb”. If you wish to choose a value in a drop-down list, you should change the value to the one you would like to use first, *then* use the “change chosen attribute” command. This will automatically fill UBot’s search string box with the appropriate attribute and aid you from having to guess what the value fields are.)

(Special note about checkboxes: The value you will want to change when selecting a checkbox will often be “Checked,” which has parameters of “true” for yes and “false” for no.)

(Special note about file uploading: Certain text boxes that allow you to browse files on your own hard drive are “protected” against their values being changed. Please see the “Send keys chosen” command for information on how to get around this.



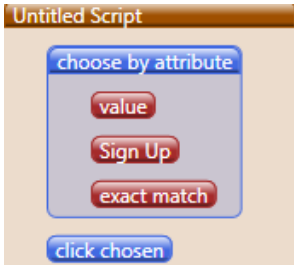
Remember: the most recently chosen element on a page will be the one that “change chosen attribute” modifies. This slightly modified script would only fill text field “Input 5,” and it would fill it first with the value “test,” then replace that value with the value “me”, leaving you with one empty text box, and another that contains the value “me”.

Select random child chosen:

Select random child chosen is extremely useful for drop-down boxes. Choosing this command after choosing a drop-down list by attribute randomly chooses any value within the drop-down box. So, for example, if you want to create an account and don't care which state will be selected, choose the state drop-down box by an attribute and use this command to choose any state within the value.

Click chosen:

This command will click a link or a button. You can also toggle check boxes with it.

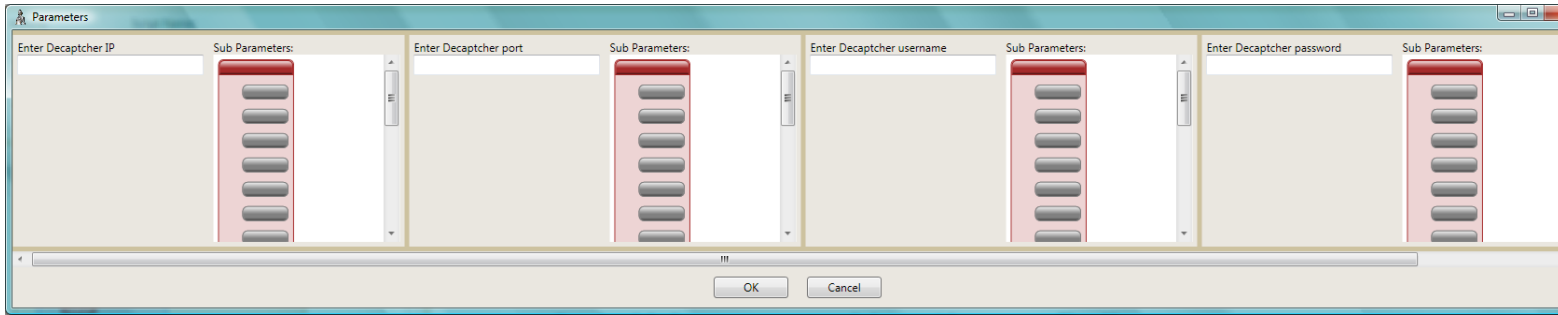


Decode chosen captcha:

After choosing a captcha image by attribute, using the "decode chosen captcha" command will bring the captcha image into a separate dialog box and the script will wait for a user to enter a value (the captcha) and press OK. (Saying "Yes" to the parameter "Use brute force" will allow the captcha window that appears to always appear in the same location). The user entered value is sent into a constant that can be called using the variable \$captcha. In the below script, the captcha is selected using choose by attribute, it is brought into focus by the decode chosen captcha command, and then the appropriate field is filled with whatever value the user enters into the captcha window. These four commands are all it takes to deal with nearly all captchas:



Decaptcher chosen captcha:



This command will use the decaptcher service to automatically decode captchas for you. To sign up with decaptcher, visit decaptcher.com. A user who wishes to use the decaptcher captcha entering service need only choose the captcha image by attribute and enter the appropriate decaptcher account information into UBot's secure parameters window to make a completely automated captcha-solving bot. This command will automatically interface with the decaptcher service in a matter of seconds and return a solved captcha.

Send keys chosen:

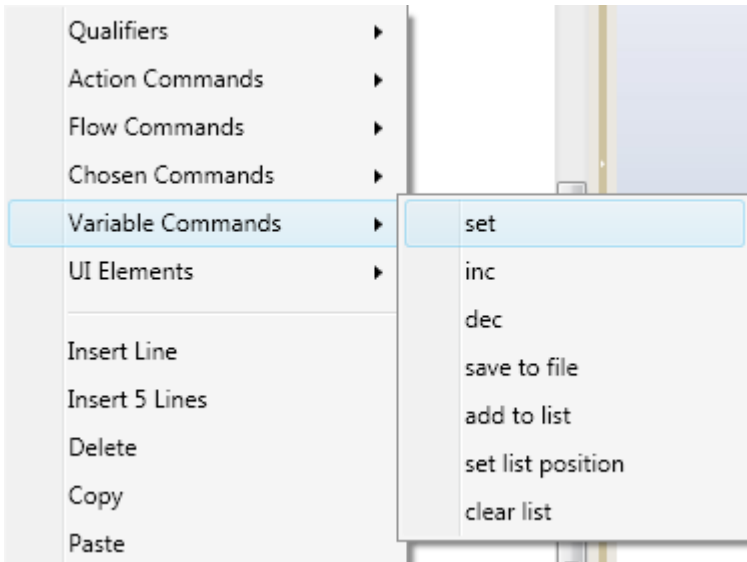
This command is used specifically for filling text into file input fields, as a means of getting around the lack of direct access to them. This command will not steal focus, and runs entirely in the background. To simulate typing on elements other than file fields, use send keys chosen field.

Occasionally a text box will be difficult to fill with a specific value. Usually text boxes that allow the user to browse their own computer or to upload files are "secure" in this way, and where one would normally use the "change chosen attribute" command with the "value" parameter, you will be required to use the "send keys chosen" command.

Send keys chosen field:

This command will simulate keystrokes to input fields other than file fields. This is useful in cases where the field reacts to keystrokes with javascript, and change chosen attribute does not trigger the javascript associated with the element.

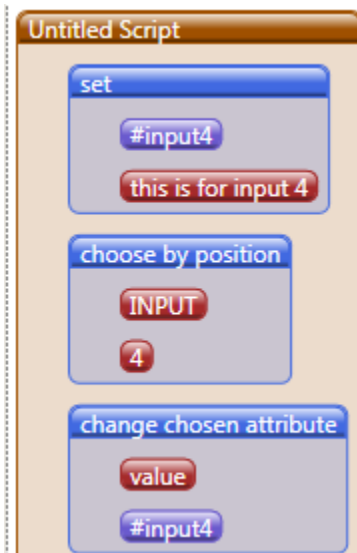
5. Variable Commands



Set:

Set allows you to create a variable and choose its value (it can be left blank until later if required). All variables begin with the pound sign (#) to distinguish them from other strings.

A variable can be filled with a simple string (a number, word, or phrase), or it can be filled with a constant. To fill the variable with a constant (or more than one if necessary), right click the gray parameter nodes and choose one from the constant menu (see “Constants” below for more info). In the below script, the variable #input4 is filled with a string and that variable is then used to fill a text field in Position 4.



Inc:

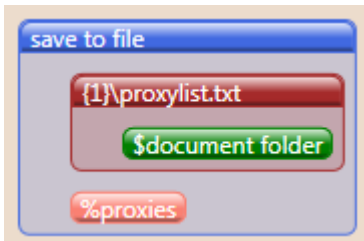
Increments a number variable by 1.

Dec:

Decrements a number variable by 1.

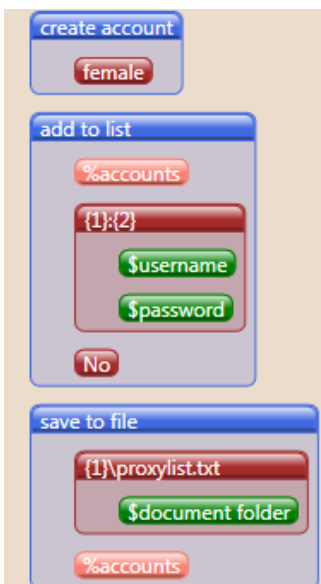
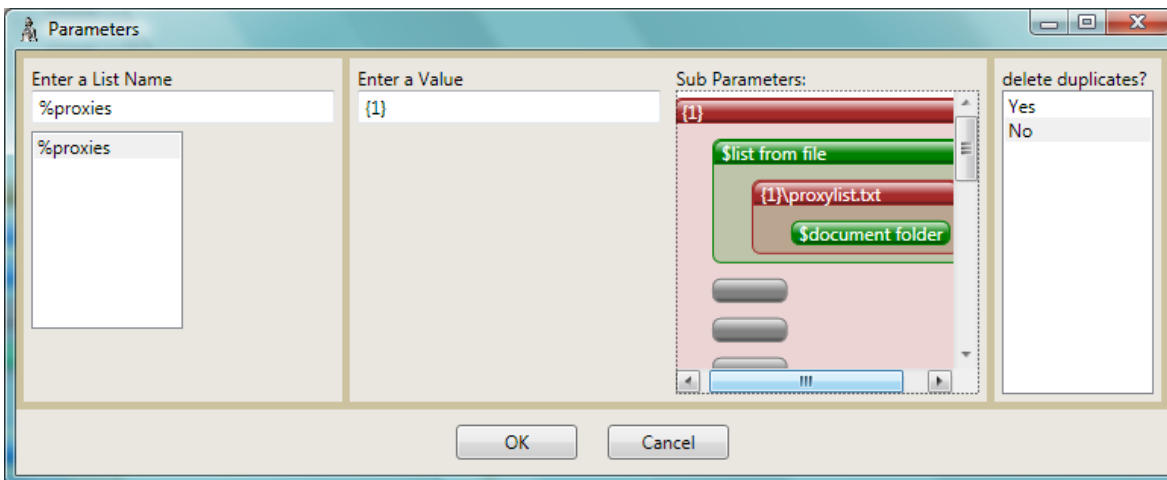
Save to file:

Saves text and lists to files. With text, it will save the text verbatim to the specified file. With lists, it will save one list item per line. To read files, use constants \$read file and \$list from file.



Add to list:

Add to list allows you to work with multiple strings or variables at one time, which is essential for page scraping or loading information from text files. Choosing add to list with a sub parameter of "\$list from file" will allow you to pull a text file into a UBot list (all lists are indicated by a % sign), which can then be modified or read on a line by line basis with a variety of commands. See below for an image of what this looks like in the Parameters window:



Choosing any other constant or string will simply save that file to a list. In the example on left, the username and password constants created by the create account command are saved to a list and then added to a text file for future use.

Set list position:

This sets the position of a list, which is used by \$next list item

Clear list:

Clear list empties the contents of a list.

UBot Affiliate Program

UBot Affiliate Program is your opportunity to make **money promoting the most powerful automation-bot creation software available in the market today**. You can earn up-to \$150 when someone buys from your affiliate link.

UBot is a extremely powerful software that lets you easily create bots to automate virtually every aspect of your business, including most aspects of marketing and business research. If you haven't already seen it, be sure to [check out the UBot features](#) and what it can do for you.

The potential of this software is endless and so is your potential to make money from our affiliate program. Click here to [sign-up now](#).

What's In It For You

High Commission: Earn up-to \$150 per sale. We believe in incentivizing your hard-work and the more sales you make every month, the higher your commission. Here is a chart showing your earning potential.

Monthly Sales Volume	Commission Earned
More Than 100	\$150 Per Sale
Between 75 and 100	\$125 Per Sale
Between 25 and 75	\$100 Per Sale
Between 5 and 25	\$75 Per Sale
Less Than 5	\$50 Per Sale

The more sales you refer each month, the higher your payout is. **We work with you at every step of the way to help you realize the full potential of your marketing efforts.** After all, our success depends on your success.

UBot

"the possibilities are almost endless"

"very impressive"

"put...commercial scripts to shame"

"a thousand things I can use it for"

SEO
Keyword research
Standalone .exe exports

Site submissions
Account creation
Link building

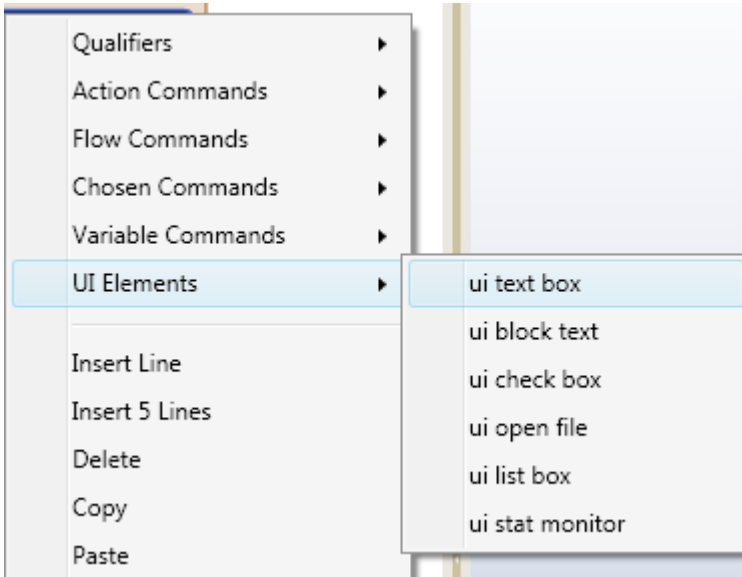
Simple scripting
Create in minutes
Free bots included

"Automate Everything."

Watch the video at
www.botsoftware.org

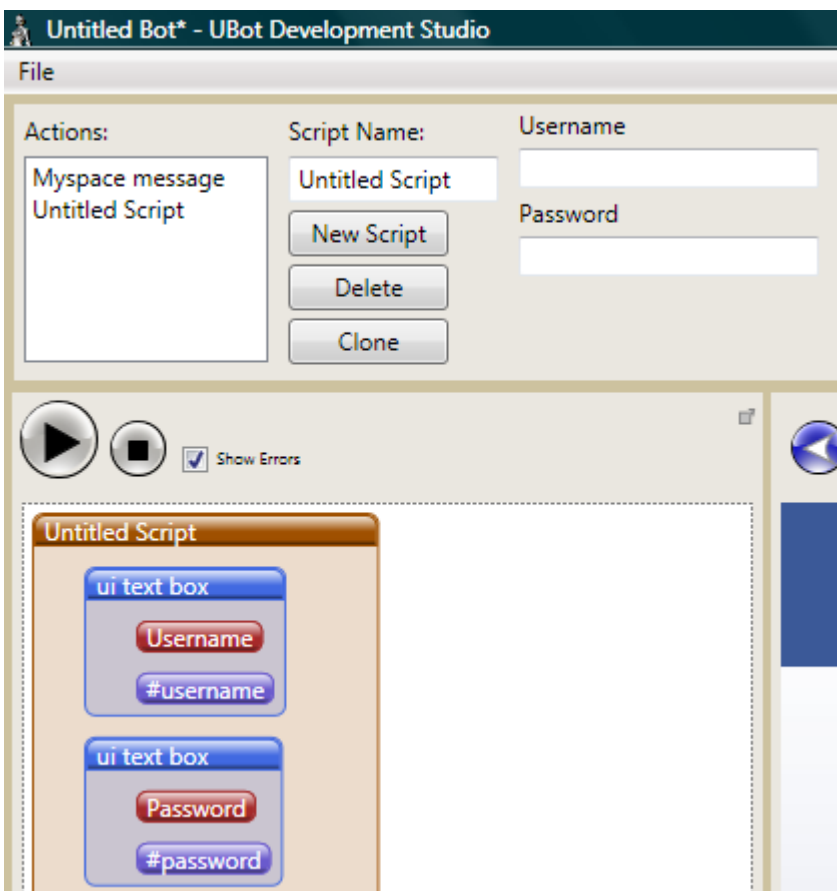
6. UI Elements

UI Elements create user-defined input boxes at the top of the UBot window. This allows users to enter information (variables, file locations, lists, or true-false checkboxes) when a bot first starts, or on the fly while a bot is running. The UI stat monitor also allows users to monitor variables and other information in realtime.



UI Text Box:

This creates a text box in the ui panel for the script. Any time text changes in the text box, the specified variable will change to that text. In the following script the user-defined textboxes are there for the user to enter the username and password for a site.



UI Block Text:

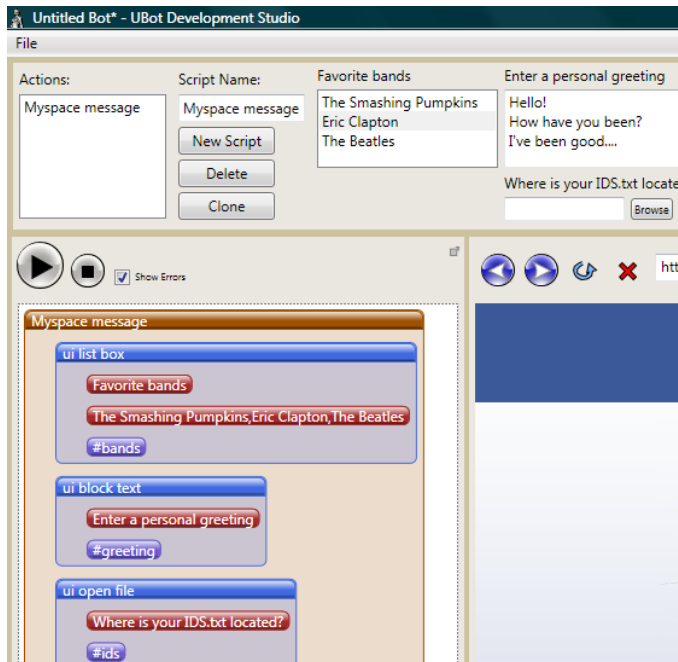
This creates a large text box in the ui panel for the script. Any time text changes in the text box, the specified variable will change to that text.

UI List Box

This creates a check box in the ui panel for the script. The specified variable will contain "True" if the box is checked, and "false" if it is not.

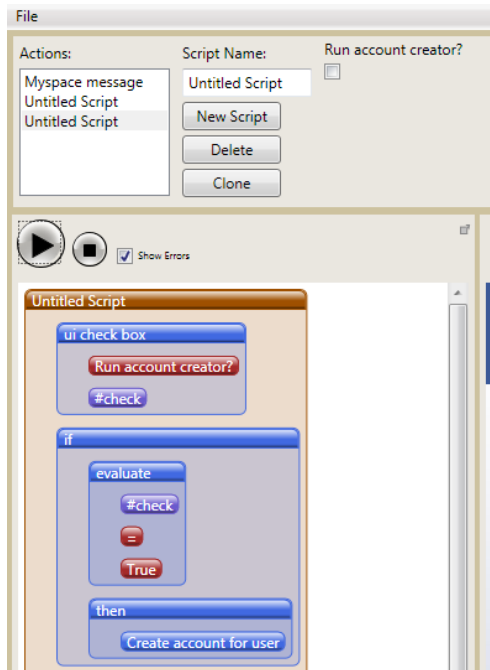
UI Open File:

This creates a file field in the ui panel for the script. Any time the field changes, the specified variable will change to that text.



UI Check Box:

This creates a check box in the ui panel for the script. The specified variable will contain "True" if the box is checked, and "false" if it is not. In the below script, checking the box causes a sub routine to run. Leaving it unchecked would not let the sub execute.

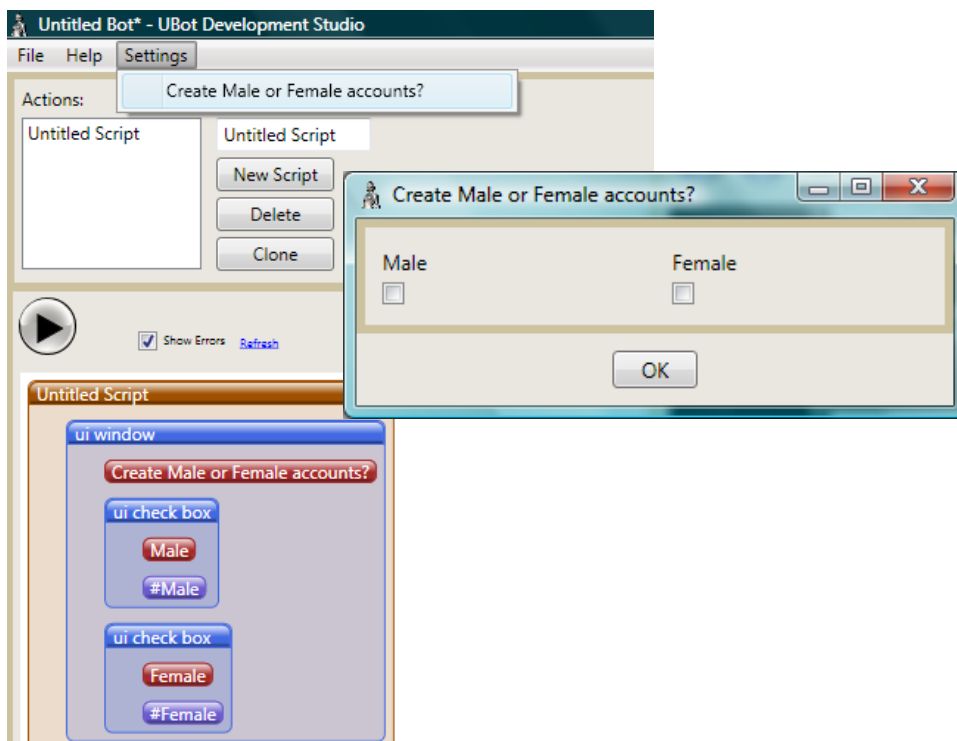


UI Stat Monitor:

This creates stat tracker in the ui panel for the script. Whenever the specified variable changes in the script, the stat panel will reflect that change. This parameter can be a variable, so that at a glance you can see such information as how many ids or keywords you've scraped, your current username and password, or the number of loops a loop command has completed.

UI Window:

This adds a UI element that is accessible through the Settings menu. When the menu item is clicked, a window will pop up showing all the UI items that exist within the command. This is useful for setting variables that you do not want to display in the top of the UBot screen at all times – use it as a normal “settings” menu might be used. See the script below for an example.



7. Constants

insert variable
insert list
insert string
insert file
\$new line
\$nothing
\$special keys down
\$document folder
\$captcha
\$find story
\$story title
\$rand
\$replace
\$eval
\$random list item
\$list position
\$list total
\$next list item
\$list item
\$read file
\$list from file
\$list from text
\$first name
\$last name
\$user name
\$email
\$password
\$zipcode
\$birth day
\$birth month
\$birth month word
\$birth year

Insert variable: Allows you to enter a variable into a parameter.

Insert list: Allows you to enter a list into a parameter.

Insert string: Allows you to enter a string into a parameter

Insert file: Allows you to enter a file into a parameter.

\$new line: Creates a line break in a list, in a parameter, or on a page. Because line breaks are not always easy to create in programming just with the “enter” key, this constant should help you create line breaks when you find that the enter key doesn’t work.

\$nothing: Allows you to enter an empty value.

\$special keys down: Allows you to enter the “arrow down” value into a parameter.

\$document folder: Shows the current directory of the computer’s document folder (usually this is where your scripts and text files are saved – this is shorthand to make it easier.)

\$captcha: For use with the decode captcha command, this constant displays the results of the information typed below the captcha field or the decaptcher results.

\$find story: The \$find story constant should return the “story content” on a page – that is, it automatically returns the main article and only the main article on a news or blog page. This is an experimental constant that does not always work correctly yet.

\$rand: Returns a random number between two specified numbers.

\$replace: Returns a piece of text modified from an original by replacing a piece of it with another. For example, if you started with the original text "blue dog democrats", and you replaced "blue" with "wiener", the result would be "weiner dog democrats"

\$story title: This returns the headline or title of the story – as in the \$find story constant, this command is not always completely accurate..

\$eval: \$eval returns the value of a mathematical function or javascript. This could be a simple math string (under \$eval the parameter 1+1 would return “2”), or it could be more complicated math or javascript involving variables.

\$random list item: This returns an entire line from a list (%), at random.

\$list position: This returns the current numerical position of UBot in a list. (ie, if \$random list item chooses line #10, \$list will return “10”).

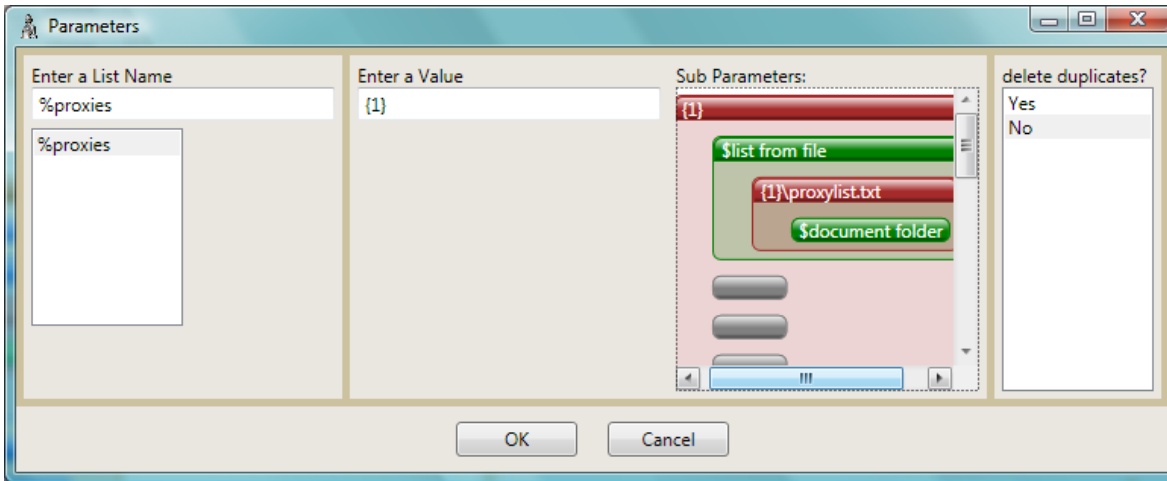
\$list total: Returns the total number of items in a list.

\$next list item: This constant will return the list item at the list's current position, and then increment the list's position.

\$list item: This constant will return the list item at the specified position.

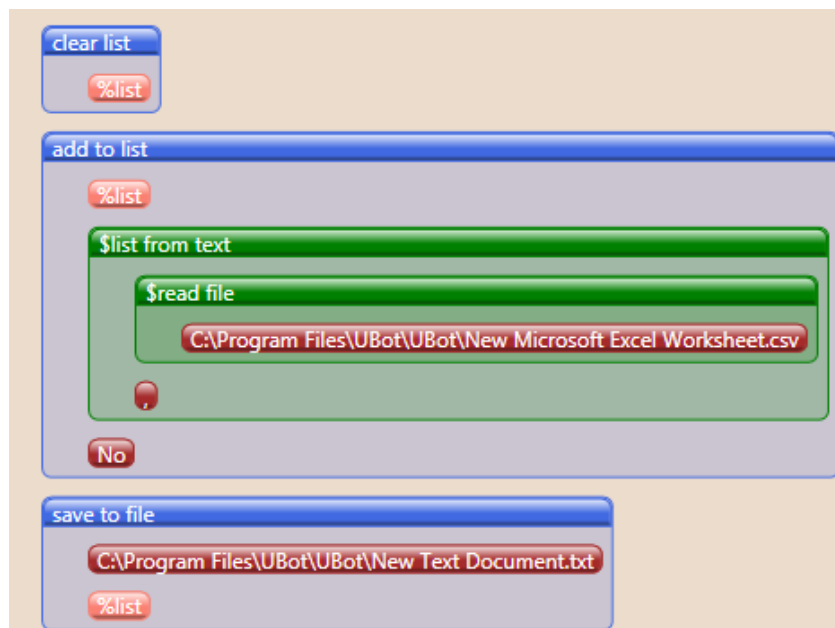
\$read file: This loads a file for use in a parameter.

\$list from file: Combine this command with add to list to pull information from a file into a list (%). In the below example, the contents of proxylist.txt is entered directly into the %proxies variable.



\$list from text: This command breaks a string of text into a list by delimiting the values based on a delimiter of your choosing. Where \$list from file creates a list that mimics the text file that you load, paragraph/line breaks and all, \$list from text allows you to work directly with CSV and other delimited files. In the below script example, the CSV file is loaded and the individual values between commas are transferred into a list with each value on a separate line. Basically, the string from the CSV file is converted from "this,is,a,CSV,file" to:

this
is
a
cSV
file



\$page scrape: Returns text or a list scraped from the current page. The two parameters will be the text on either side of the text you wish to scrape. Note that this is a constant, which means that to use it, you must select a parameter node, and not a blue command node. To set a variable with it, for example, create a set command, leaving the contents blank, which will create a placeholder. Click the placeholder and then choose \$page scrape.

\$scrape chosen attribute: Returns text or a list of the specified attribute from the chosen element(s). Like \$page scrape, this constant must take the place of a parameter node, and not a blue command node.

The remaining constants pull data from the “Create account” command. After using that command you can use these constants to retrieve the respective data from that specific account:

\$first name, \$last name, \$user name, \$email, \$password, \$zipcode, \$birth day, \$birth month, \$birth month word, \$birth year

8. Additional Features

As features are added to UBot this section will be updated.