

## Table of Contents

<b>1. Table of Contents</b>		<b>1</b>
<b>2. Introduction</b>		<b>2</b>
<b>3. Identifiers (names)</b>		<b>2</b>
3.1. Labels		3
3.2. Strings		3
3.3. Numbers	3	
3.4. Variables	3	
3.5. Predefined Variables		4
<b>4. Commands</b>		<b>5</b>
4.1 Internal Commands		5
4.1.1 Program Termination Commands		5
4.1.1.1 End	5	
4.1.1.2 Abort	5	
4.1.2. Branching Commands		6
4.1.2.1. Gosub	6	
4.1.2.2. Goto	6	
4.1.2.3. Return	7	
4.1.3. Conditional Execution		8
4.1.3.1. If	8	
4.1.3.2. Then	8	
4.1.3.3. Else	8	
4.1.3.4. Endif	9	
4.1.4. Iterative Execution		9
4.1.4.1. Repeat	9	
4.1.4.2. Until	9	
4.1.4.3. While	9	
4.1.4.4. Do	10	
4.1.4.5. Wend	10	
4.1.4.6. For	10	
4.1.4.7. To	10	
4.1.4.8. Next	10	
4.1.5. Variable Commands		11
4.1.5.1. String		11
4.1.5.2. Number		11
4.1.5.3. Destroy		11
4.1.6. Display Commands		11
4.1.6.1. Print		11
4.1.6.2. Trace		12
4.2. External "Run-Time" Commands		12
4.2.1. Configuration Commands		12
4.2.1.1. CfgSetValue		12
4.2.1.2. CfgGetValue		12
4.2.2. COM Read/Write commands	13	
4.2.2.1. CommSend		13
4.2.2.2. CommWaitFor		14
4.2.2.3. CommReadIPAddr	14	

## *Quarterdeck Winsock Script Language*

4.2.3. Miscellaneous commands	15
4.2.3.1. SetTimeout	15
4.2.3.2. UserInput	15
4.2.3.3. GetPassword	15
4.2.3.4. Delay	15

## **2. Introduction**

Script files are simple programs in the QWinsock script language. This language is similar to a simplified version of the BASIC programming language with some additions to support login requirements of various Service Providers.

**NOTE:** This document is not intended to provide instruction in programming fundamentals and concepts. Its intent is to provide a brief overview of the Qwinsock script language and reference to its keywords and commands. If you are not familiar with programming concepts, it will be helpful if you refer to a book on BASIC programming.

## **3. Identifiers (names)**

*Identifiers* are used to represent variables, labels, and commands. Identifiers can be any combination of alphanumeric characters and the underscore character and are not case sensitive. For example, the identifier “myname” is the same as “MyName” and “MYNAME.”

The script language reserves a number of names for its language definition. These *internal commands* can not be used as Variable names. These reserved names include the following:

abort	goto	then
destroy	if	to
do	next	trace
else	number	until
end	print	wend
endif	repeat	while
for	return	
gosub	string	

A number of commands, also called *run time commands*, may not be used as Variable names in login scripts. These reserved command names include the following:

CfgSetValue	CommWaitFor	GetInput
CfgGetValue	CommReadWord	GetPassword
CommSend	CommReadIPAddr	SetTimeout

## *Quarterdeck Winsock Script Language*

### **3.1. Labels**

*Labels* specify a point in the script program that can be used as the target of a GOTO or GOSUB command. A Label begins with a colon and is followed by a series of alpha or numeric characters. Labels must appear in the first column of a line.

The following are examples of valid Labels:

```
:Label  
:500  
:ErrorRoutine
```

Labels are not case sensitive. As a result, the following two Label definitions are equivalent:

```
:LabelA  
:labela
```

Defining two Labels with the same or equivalent name will cause an error.

### **3.2. Strings**

Constant strings are made up of printable characters enclosed within double quotes. The length of a string is limited to 65535 bytes or your editor's line limit, whichever is less. You can also include the punctuation and non-printing characters listed in the table below.

%r	carriage return	
%n	line feed	
%t	tab	
%b	backspace	
%"	double quote	
%%		percent sign (%)

### **3.3. Numbers**

Constant numbers are integer values comprised of one or more digits. The integer range of Numbers is -2,147,483,648 to 2,147,483,647.

### **3.4. Variables**

Two types of variables are supported by the script language, *String* and *Number*. Variables are defined by specifying the variable type (String or Number) followed by the name to assign the variable. In addition, a variable may be assigned the value of a constant or previously defined variable.



## Quarterdeck Winsock Script Language

assign a value to the variable RESULT, which you can test to see whether the command was successful.

Note: only the *run-time* commands modify the RESULT variable (see the second table on page 2); the *internal* commands do not. Each run-time variable uses the RESULT variable in its own way; be sure to see the documentation for these run time commands to determine their use, if any, of the RESULT variable.

### 4. Command Reference

The next section is a reference guide for the available Winsock script commands. Some conventions used in the reference guide:

- Commands are commented in italics. That is, next to each line of Winsock script commands is a short explanation in italic type; the italicized words are not part of the command. For example:

<u>Command</u>	<u>Explanation</u>
End	<i>this command ends the program</i>

- The vertical bar in a command's "Syntax" separates a list of possible command options from which you should select *one*. For example:

**Syntax:**           Trace ON | OFF

This means that the command line should read "Trace ON" or "Trace OFF" — but **not** "Trace" (alone) or "Trace ON OFF".

#### 4.1. Internal Commands

##### 4.1.1. Program Termination Commands

###### 4.1.1.1. *End*

**Description**       Ends execution of the script program and indicates to the calling program that the script completed successfully.

**Syntax**            End

**Example**           End

**See Also**          Abort

###### 4.1.1.2. *Abort*

**Description**       Ends execution of the script program and indicates to the calling program that the script did not complete successfully. The ABORT command may optionally display an error message to the output window using the same parameter format as the PRINT command.

**Syntax**            Abort [ *String* | *stringvariable* ]



## Quarterdeck Winsock Script Language

**Example**

```
String Username define string variables
String Password

CfgGetValue "Username" Username get the user name from the config file
if RESULT = 0 then if 0 (no user name) then jump to the label
    label
    Goto ReadError "ReadError"
endif

CfgGetValue "Password" password get the password from the config file
if RESULT = 0 then if 0 (no password) then jump to the label
    Goto ReadError "ReadError"
endif

.
.
:ReadError label "ReadError"
Abort "Error reading username or password" abort with error message
```

**See Also** Gosub (Use Gosub when you want to jump to a label but soon thereafter to return to the same place.)

### 4.1.2.3. Description

**Return**  
Terminates the GOSUB commands by passing control (that is, "returning") to the command immediately following the GOSUB command. If the GOSUB command is not terminated with a RETURN, the script is terminated with an error.

**Syntax** Return

**Example**

```
Gosub ReadPassword jump to the label "ReadPassword"
.
.
.
:ReadPassword label

String Password define the string variable "Password"

CfgGetValue "Password" Password get the password from the config file
if RESULT = 0 then if 0 (no password) then abort with an error
    Abort "Error reading password" message
    endif
Return return to the line after the Gosub command
```

**See Also** Gosub





## Quarterdeck Winsock Script Language

### 4.1.3.4. *Endif*

#### Description

Terminates the IF command statement. Each IF command block **must** end with the Endif command. If the IF command is not terminated with an ENDIF command, the script is terminated with an error.

#### See Also

If, Then, Else (For a notated example, see the **If** command on page 8.)

### 4.1.4. Iterative Execution

The following commands provide methods for executing a group of commands or expressions multiple times.

#### 4.1.4.1. *Repeat*

#### Description

Allows a group of commands to be executed until a condition resolves to TRUE. The group of commands will be executed *at least one time*. The Repeat command is terminated by the Until command. Repeat commands may be nested.

#### Syntax

Repeat  
    one or more commands or expressions  
Until *expression*

#### See Also

Until, While

#### 4.1.4.2. *Until*

#### Description

Terminates the Repeat command statement. If the Repeat command is not terminated with the Until command, the script is terminated with an error.

#### See Also

Repeat

#### 4.1.4.3. *While*

#### Description

Allows a group of commands to be executed while a condition is TRUE.

The While command differs from the Repeat command in that the condition is evaluated *before* the group of commands executed. As a result, the condition must be TRUE *before* any commands are executed.

The While command's condition must be followed by the **Do** command. The While command is terminated by the **Wend** command. While commands may be nested.

#### Syntax

While *expression* Do  
    one or more commands or expressions  
Wend

#### Hint:

To make a command repeat indefinitely, use the expression "While 1 = 1".

#### See Also

Do, Wend, Repeat

## *Quarterdeck Winsock Script Language*

### **4.1.4.4. Description**

#### ***Do***

The Do command indicates the beginning of the command(s) to execute if the While expression resolves to TRUE. A While command without a Do command will cause an error.

**See Also**            While, Wend

### **4.1.4.5. Description**

#### ***Wend***

Terminates the While command statement. All commands between the Do and Wend commands will get executed if the While expression resolves to TRUE.

If the While command is not terminated with the Wend command, the script is terminated with an error.

**See Also**            While, Do

### **4.1.4.6.**

#### ***For***

Repeats a group of commands a fixed number of times. The FOR command uses a Number variable as a counter to keep track of the current iteration and requires a *starting value* to initialize the counter and an *ending value* to compare against the counter to determine if the correct number of iterations have been performed.

```
FOR counter = start TO end  
  one or more commands or expressions  
NEXT
```

<i>counter</i>	<i>Previously defined</i> Number variable used as a loop
counter	
<i>start</i>	Number constant used as the initial value of counter
TO	keyword separating <i>start</i> and <i>end</i>
<i>end</i>	Number constant used as the final value of counter

### **Example**

Number I	<i>define the counter variable</i>
For I = 1 to 10	<i>repeat this ten times</i>
Print "This will print ten times."	<i>print a message</i>
Next	<i>ends the for...next block</i>

### **4.1.4.7.**

#### ***To***

Keyword used to separate the *start* and *end* values of the FOR command.

### **4.1.4.8.**

#### ***Next***

Ends the FOR loop and causes the *counter* to be incremented by one. All commands between the FOR and NEXT command are considered. If the FOR command is not terminated with a NEXT command, the script is terminated with an error.

## *Quarterdeck Winsock Script Language*

### **4.1.5. Variable Commands**

The following commands allow creating and destroying variables.

#### **4.1.5.1. *String***

Creates a named variable for storing alphanumeric text.

**Syntax:**

STRING *variablename*

#### **4.1.5.2. *Number***

Creates a named variable for storing integer numbers.

**Syntax:**

NUMBER *variablename*

#### **4.1.5.3. *Destroy***

Destroys a previously defined String or Number variable.

**Syntax:**

DESTROY *variablename*

### **4.1.6. Display Commands**

#### **4.1.6.1. *Print***

The PRINT command allows you to display quoted strings (such as “Press Return please”) as well as Constant or Variable identifiers. When displaying multiple Constant or Variable identifier, each must be separated by a comma or a semicolon. The *semicolon* will insert a blank between the two values and the *comma* will insert a Tab between the two values.

PRINT *identifier* [, | ; *identifier* ] [, | ; *identifier* ] ... [, | ; *identifier* ]

#### **Examples**

Print command

Print “Good morning.”

String User = “John.”

Print “Good morning”, User

Prints as

Good morning.

Good morning, John

See the table in the section on Strings on page 3 if you need to insert non-printing characters, quotes or percent signs in a PRINT command.

## Quarterdeck Winsock Script Language

### 4.1.6.2. Trace

The TRACE command causes all commands or expressions executed to be displayed to the output window as their are executed.

**Syntax:** TRACE ON | OFF

## 4.2. External “Run Time” Commands

### 4.2.1. Configuration Commands

#### 4.2.1.1. CfgSetValue

**Description** The CfgSetValue allows saving a value for the provider to the INI file. This command is used for setting values read from the COM port such as the IP address, DNS server, etc.

**Syntax** CfgSetValue *variablename* | *String identifier2*

**Example**

```
String IPAddress define the variable "IPAddress"

CommWaitfor "ipaddress =" listen to the COM port for the string "ipaddress ="

CommReadIPAddress IPAddress read from the COM port into the variable IPAddress

if RESULT = -1 then if -1 (problem reading the address)
  Abort "Error reading IP address from host" abort with a message
endif end the "if" block
```

*NOTE: If the program gets this far, there was no error reading from the COM port.*

```
CfgSetValue "IPAdddress" IPAddress set the IPAddress config file directive with the contents of the variable "IPAddress"
```

#### 4.2.1.2. CfgGetValue

**Description** The CfgGetValue reads a provider value from the INI file. This command is used for reading values such as the Username or Password for later tranmission out the COM port.

**Syntax** CfgGetValue *variablename* | *String stringvariable*

**RESULT variable** The length of the data retrieved from the INI file. A result of 0 means that no data was read from the INI file.

## ***Quarterdeck Winsock Script Language***

<b>Example</b>	String Uname	<i>declare some string variables</i>
	String Pword	
	CfgGetValue "Username" Uname	<i>get the Username information from the config file and assign it to the variable "username"</i>
	if RESULT = 0 then	<i>if 0 (problem) then</i>
	Abort "User name not set"	<i>abort with an error message</i>
	endif	<i>end the "if" block</i>
	<i>otherwise...</i>	
	CfgGetValue "Password" Pword	<i>get the config file password information and assign it to the variable "Pword"</i>
	if RESULT = 0 then	<i>if 0 (problem) then</i>
	Abort "Password not set"	<i>abort with an error message</i>
	endif	<i>end the "if" block</i>

### **4.2.2. COM Read/Write commands**

#### **4.2.2.1. CommSend**

**Description** The CommSend commands allows sending a variable or constant data to the COM port.

**Syntax** CommSend *variablename* | *String* | *Number*

**RESULT variable** The RESULT variable will be set to the 0 if all data was sent or -1 if an error occurred.

<b>Example</b>	String Uname	<i>declare a variable</i>
	CfgGetValue "Username" Uname	<i>get the Username parameter from the config file</i>
	if RESULT = 0 then	<i>if 0 (problem getting the parameter)</i>
	Abort "Username not set"	<i>abort with an error message</i>
	endif	<i>end the "if" block</i>
	<i>otherwise...</i>	
	CommWaitfor "login:"	<i>listen at the COM port for the string "login:"</i>
	CommSend Uname	<i>send the contents of</i>
	<i>the variable "Uname"</i>	<i>through the COM port</i>

## Quarterdeck Winsock Script Language

```
if RESULT = -1 then                                if -1 (problem sending) then
  Abort "Error sending Username" abort with an error message
endif                                              end the "if" block
```

### 4.2.2.2. *CommWaitFor*

**Description** The CommWaitFor command allows the script to pause until a specific character string has been received on the COM port. All data up to and including the wait string is discarded.

**Syntax** CommWaitFor *variablename* | *String*

**Example**

```
String Uname                                     declare a variable

CfgGetValue "Username", Uname get the Username value from the config
                               file and assign it to the variable
"Uname"

CommWaitFor "login:"                               listen at the COM port for the string
"login:"                                           "login:"

CommSend Uname                                     send the contents of
the variable "Uname"                               through the COM port
```

### 4.2.2.3. *CommReadIPAddr*

**Description** The CommReadIPAddr command allows reading the next word from the COM port as an IP Address.

**Syntax** CommReadIPAddr *stringvariable*

**RESULT variable** The length of the IP address string on success and zero on failure.

**Example**

```
String ipaddress                                 declare a variable to hold the IP
address

CommReadIPAddr ipaddress                       read an IP address from the COM port

if RESULT > 0 then                               if the length of the string just read
  CfgSetValue "IPAddress" ipaddress exceeds 0, set the IPAddress config file
  variable                                       parameter to the contents of the
                                               "ipaddress"
endif                                           end the "if" block
```

## Quarterdeck Winsock Script Language

### 4.2.3. Miscellaneous commands

#### 4.2.3.1. *SetTimeout*

**Description** The SetTimeout command resets the timeout for establishing the connection. This command is used in cases where the default ConnectTime setting may be too short to allow the script to complete successfully. The value specified is the number of seconds to continue connecting before timing out.

**Syntax** SetTimeout *Number* | *numbervariable*

**Example** SetTimeout 60

#### 4.2.3.2. *GetInput*

**Description** This command is used to prompt the user for input and store the result in a variable

**Syntax** GetInput “promptstring” variable

Note that the prompt string must be in double quotes.

**RESULT variable** The length of the text typed by the user.

**Example**

String Uname	<i>declare a variable</i>
GetInput “Enter your name” Uname	<i>ask the user to enter a username and store the user’s input in the variable “Uname”</i>
if RESULT > 0 then	<i>if the length of the text just entered exceeds 0 then</i>
Print Uname “Uname”	<i>print the contents of the variable</i>
endif	<i>end the “if” block</i>

#### 4.2.3.3. *GetPassword*

**Description** This command is used to prompt the user for input and store the result in a variable. This function is identical to the GetInput command with the exception that the input field in the dialog box displays an astericks for each character typed instead of the actual character

**Syntax** GetPassword “promptstring” variable

Note that the prompt string must be in double quotes.

**RESULT variable** The length of the text typed by the user.

### ***Quarterdeck Winsock Script Language***

**Example**

String Uname	<i>declare a variable</i>
GetPassword "Enter your password" Uname <i>store</i>	<i>ask the user to enter a username and the user's input in the variable "Uname"</i>
if RESULT > 0 then <i>exceeds</i>	<i>if the length of the text just entered 0, then</i>
Print Uname <i>"Uname"</i> endif	<i>print the contents of the variable end the "if" block</i>

**4.2.3.4.**

***Delay***

**Description**

The Delay command introduces a pause in the execution of the script for a specified number of seconds.

**Syntax**

Delay *seconds*.