VBScript Cheat Sheet

A quick reference guide to VBScript, covering essential syntax, objects, and functions for automating tasks in Windows environments.

VBScript Basics

Data Types

Boolean	True or False
Byte	Integer between 0 and 255
Integer	Integer between -32,768 and 32,767
Long	Integer between -2,147,483,648 and 2,147,483,647
Single	Single-precision floating-point number
Double	Double-precision floating-point number
String	Sequence of characters
Date	Represents dates and times
Object	An OLE Automation object

Variables

Variables are declared using the `Dim`, `Public`, or `Private` keywords.

Example:

```
Dim myVariable
myVariable = "Hello, World!"
```

To explicitly declare the type of a variable, use `As` keyword.

Example:

```
Dim myInteger As Integer
myInteger = 10
```

Constants are declared using the `Const` keyword.

Example:

```
Const PI = 3.14159
```

Operators

Arithmetic	+, -, *, /, \ (integer division), Mod (modulus), ^ (exponentiation)
Comparison	=, <>, <, >, <=, >=
Logical	And, Or, Not, Xor
String Concatenation	&
Assignment	=

Control Structures

Conditional Statements

If...Then...Else

```
If condition Then
  'Statements to execute if condition is true
ElseIf condition2 Then
  'Statements to execute if condition2 is true
Else
  'Statements to execute if all conditions are false
End If
```

Select Case

```
Select Case expression
  Case value1
    'Statements to execute if expression = value1
  Case value2
    'Statements to execute if expression = value2
  Case Else
    'Statements to execute if expression doesn't match any value
End Select
```

Looping Structures

For...Next

```
For i = start To end [Step step]
  'Statements to execute
Next
```

For Each...Next

```
For Each element In group
  'Statements to execute
Next
```

Do While...Loop

```
Do While condition
  'Statements to execute
Loop
```

Do...Loop While

```
Do
  'Statements to execute
Loop While condition
```

While...Wend (Legacy, avoid using)

```
While condition
  'Statements to execute
Wend
```

Error Handling

On Error Resume Next

Continues execution even after an error occurs.

Example:

```
On Error Resume Next
'Code that might cause an error
If Err.Number <> 0 Then
  'Handle the error
End If
On Error GoTo 0 'Disable error handling
```

Err Object

Provides information about runtime errors.

Properties: `Number`, `Description`, `Source`

Methods: `Clear`, `Raise`

Functions and Procedures

Functions

Functions are blocks of code that perform a specific task and return a value.

Syntax:

```
Function FunctionName(parameter1, parameter2)
  'Statements
  FunctionName = returnValue
End Function
```

Example:

```
Function Add(a, b)
  Add = a + b
End Function
```

Functions are called by using their name and passing arguments, if any.

Example:

```
result = Add(5, 3)
WScript.Echo result 'Output: 8
```

Subroutines (Procedures)

Subroutines are blocks of code that perform a specific task but do not return a value.

Syntax:

```
Sub SubroutineName(parameter1, parameter2)
  'Statements
End Sub
```

Example:

```
Sub Greet(name)
  WScript.Echo "Hello, " & name
End Sub
```

Subroutines are called using the `Call` keyword or by simply using their name.

Example:

```
Call Greet("Alice")
Greet "Bob"
```

Built-in Functions

`MsgBox`	Displays a message box.
`InputBox`	Displays a prompt for user input.
`Len(string)`	Returns the length of a string.
`UCase(string)`	Converts a string to uppercase.
`LCase(string)`	Converts a string to lowercase.
`Trim(string)`	Removes leading and trailing spaces from a string.
`Left(string, length)`	Returns a specified number of characters from the left side of a string.
`Right(string, length)`	Returns a specified number of characters from the right side of a string.
`Mid(string, start, length)`	Returns a specified number of characters from a string.

Working with Objects

Creating Objects

Objects can be created using the `CreateObject` function.

Syntax:

```
Set objectVariable = CreateObject("ProgID")
```

Example:

```
Set fso = CreateObject("Scripting.FileSystemObject")
```

Late Binding vs Early Binding:

- **Late Binding**: Object type is determined at runtime (using `CreateObject`). More flexible but can be slower.
- **Early Binding**: Object type is determined at design time (requires referencing a type library). Faster but less flexible.

FileSystemObject (FSO)

`CreateTextFile(filename, [overwrite], [unicode])`	Creates a new text file.
`OpenTextFile(filename, [iomode], [create], [format])`	Opens an existing text file.
`FolderExists(folderpath)`	Checks if a folder exists.
`FileExists(filepath)`	Checks if a file exists.
`CreateFolder(folderpath)`	Creates a new folder.
`DeleteFile(filespec, [force])`	Deletes a file.
`DeleteFolder(folderspec, [force])`	Deletes a folder.

WScript Object

`WScript.Echo message`	Displays a message.
`WScript.Quit [exitcode]`	Terminates the script.
`WScript.Arguments`	Collection of command-line arguments.
`WScript.FullName`	The full path of the current script.