Reference Document for Flash Scripting

What is scripting ?

It is a line of code with appropriate syntax to perform a user defined task. Script is the coded version of the flow of your program. The flash file behaves, responds as per the script written. Flash script provides a simple yet basic way of writing your code, but before you proceed, you should exactly know what you want and how are you going to achieve it, i.e. ***the logic***. Once you have the logic, flash provides its inbuilt functions, operators, and property's to convert that logic into an error free executable code, the output.

Where is the script written?

Action script can be written on Buttons, Keyframes and Blank Keyframes.

- On the buttons where the script will be executed on some event performed by the user i.e. mouseevent.
- Secondly, On the Keyframe / Blank Keyframe – this script is executed as soon as the play head reaches that particular Keyframe / Blank Keyframe.

Why scripting should be done?

Scripting can be used for various purposes like storing variables, setting object properties for controlling animation.

Consider for an example, an object is moving (tweened) from left to right direction, and after it reaches a particular location a message welcoming the user is to be displayed can be done through scripting.

- Find out the location i.e. the co-ordinates (Getproperty)at which the message should be displayed.
- After finding the co-ord, the co-ords will be confirmed
- Confirm the co-ords by a conditional checking (IF) and then message will displayed.(Set variable)

How the above mentioned can be done through script ;

The script is written in the frame property where the animation stops, movie clip name is "try"

```
Set Variable: "xpos" = GetProperty ( "/try", _x )
Set Variable: "ypos" = GetProperty ( "/try", _y )
        If (xpos eq "442" and ypos eq "181")
                Set Variable: "display" = "WELCOME"
        End If
```

Here the above two lines of script finds the x, y co-ords. Then they are compared with a predetermined value of (x, y) using the ***If…Endif***. If co-ords matches, then display message. Since the above script is written in Keyframe Property, it will be executed immediately as soon as that Keyframe is reached.

Following is an example of how the script will be written on the button property which the user clicks.

```
On (Release)
        Set Variable: "xpos" = GetProperty ( "/try", _x )
        Set Variable: "ypos" = GetProperty ( "/try", _y )
            If (xpos eq "442" and ypos eq "181")
                Set Variable: "display" = "WELCOME"
            End If
End On
```

Above, *On (Release)* is the user performed mouse operation i.e. after clicking the button the remaining code will be executed.

- Set Variable :

A script is executed only when it has some values / contents in it. These values / content are stored in a script through variables, Thus a variable stores value or content. It behaves either in a very dynamic or a static way.

- Dynamic - as a variable can have varying values or contents being assigned to it, through frame properties or mouseevent.
- Static because the variables tends contain the same value / content throughout the program unless and until changed.

A variable is a container that holds information. The container itself is always the same, but the contents can change. By changing the value of a variable as the movie plays, you can record and save information about what the user has done, record values that change as the movie plays, or evaluate whether some condition is true or false.

Variables are characters that can be created as placeholders for numerical and string data. They can be updated and accessed dynamically. This provides a great deal of flexibility and interactivity in a Flash movie. Variables can be set, accessed and updated anywhere in a Flash movie. They can be set by Keyframe or button actions (using the action Set Variable) or by values entered in a text field.

Variables can hold any type of data: number, string and Boolean. The type of data a variable contains affects how the variable's value changes when it is assigned in a script.

Syntax

Set Variable: Name_of_variable = *"Value / content"*……String Literal
Set Variable: Name_of_variable = *Value/ content* ……Expression

An example to understand variables

```
On (Release)
        Set Variable: "num1" = "10"
        Set Variable: "num2" = "20"
        Set Variable: "total" = num1+num2
End On
```

Variable num1 & num2 contains value 10 & 20 resp. Variable total contains the sum of num1 & num2.

- **What is String Literal and Expression ?**

A string is a collection of characters. A variable, if set as String Literal will be displayed as it is in the output. "" are used to indicate a string

```
Set Variable: "total" = "num1+num2"
```

The above line of script will display total as num1+num2.

An expression will be evaluated or it identifies what is to be done with the value.

```
Set Variable: "total" = num1+num2
```

The above line of script will display total as the sum of num1 & num2. i.e 30

Note:

If a variable is set as *STRING*, its value will be displayed as it is.
If a variable is set as *EXPRESSION*, its value will be evaluated and resultant will be displayed.

When making a script for comparison remember the following :

1. When two variables are being compared, do not enclose the variables in double quotes
   e.g. **if(Salary > lowsalary)**

2. When comparing variable with a value, enclose only the value in double quotes
   e.g. **if(clicked eq "x")**

- Textfield & Textfield Properties :
  Textfield is a user defined area which will accept some input from the user or will display something in it during runtime i.e. when the SWF file is being played.
  - Textfield can accept any character from the keyboard (Special character also). A Textfield property can be restricted or controlled to accept input.
  - Textfield if made, first of all has to be identified so that it can used in the script. Textfield will always have a **unique name**(Variable Name). Textfield name i.e. variable name can be anything, but a relevant name will help you in scripting.

Some properties of Textflield :

Draw background and border	If enabled, the text field does not show a background and border
Password	Accepts contents without displaying what is being typed, replaces each with an * while typing
Multiline	If no. of lines are to be typed in a textfield.
Word wrap	It is available only when Multiline feature is enabled. If only multilane is enabled, the text continues to be typed in the same line , whereas Word wrap fits the text one below the other the entire textfield
Restrict text length to "n" characters	"n" indicates no. of characters to be accepted in the text field
Disable editing	The text displayed can be changed or edited to avoid it, enable this option
Disable selection	It is available only when Disable Editing is enabled. This does'nt the text to get selected
Outlines :	The text accepted in a textfield is shown as non-anti aliased. Keeping this option enable for all outlines increases the file size, hence advisable to use only those outline which are reqd. viz Include only specified outlines only - UPPERCASE, lowercase, numbers, punctuations (, . / ?) and characters.

- Relationship between Textfield and Variables :

Anything that is to be accepted / displayed has to be done in a textfield. But how the script identifies where to display and from where to accept .i.e. a variable.

The name given to the textfield is the variable name

- Symbol Property **:**

A symbol is a reusable image, animation, or button. An instance is an occurrence of a symbol on the Stage or nested inside another symbol.
Symbols can make editing a movie simpler as changes to repeating elements need only be made to the symbol and Flash updates all instances.

Using symbols in your movies :

- Dramatically reduces file size.
- Saves several references to a symbol requires less storage space than saving a complete description of the element for each occurrence.
- You can reduce the file size of your movies if you convert static graphics such as background images into symbols.
- Symbols can also speed movie playback because a symbol needs to be downloaded to a browser only once.

Graphic	Button	Movie Clip
Graphic can be created for objects which are to be re-used. Duplicating Graphics doesnot increase file size with comparison to objects. Any object, can be converted to a graphic symbol. If changes are made, changes will be applied in all the frames where that graphic symbol is appearing.	Buttons are objects which initiate action to be performed on various mouseevents(On Press, On Release, On Rollover, etc.).Buttons provide interactivity to the flash file.	Movie clips have their own multiframe Timeline that plays independent of the main movie's Timeline—think of them as mini-movies inside a main movie that can contain interactive controls, sounds, and even other movie clip instances. You can also place movie clip instances inside the Timeline of a button symbol to create animated buttons.

- **Button :**

Buttons are used for interactivity purposes. Action which will performed or script will be executed on any mouseevent.

A button is a type of symbol that can display a different image for each of the button's possible states and carry out a specific action when user interacts with the button using the mouse. You specify the different states of a button by creating keyframes in a four-frame Timeline.

Up state	Represents the button whenever the pointer is not over the button i.e. face of the Button
Over state	Represents the button's appearance when the pointer is over it
Down state	Represents the button's appearance as it is clicked.
Hit state	Defines the area in which the button will be activated. This area is invisible in the movie.

Note : you can also place a movie clip in the Over & Down State to make an animated button

- Various Mouseevents Are As Follows :

Press	Mouse button is pressed down
Release	Mouse button is pressed down and released
Release Outside	Mouse button is pressed down and released outside the button area
Roll Over	Mouse pointer is rolled over the button area
Roll out	Mouse pointer is rolled outside the button area
Drag Over	Mouse pointer is pressed & dragged over the button area
Drag Out	Mouse pointer is pressed & dragged out the button area
Key press	though its not a mouseevent, to enable the button to interact through keyboard

- Navigation in Frame / Scene by using buttons :

Buttons provide a very simple and a visible way to navigate in frames / scene. There are several ways to do the same.

```
On (MouseEvent)
    Go to and Stop (1)
End On
```

MouseEvent can be anything the list seen earlier depending on what is the users requirement, the action i.e. **Go to** and Stop (1) can be given in the following ways also

Go to	Scene	To goto a particular scene
	Frame number	To goto a particular frame
	Frame label	Label is the name given to identify a frame. At times when no: of frames used are large enough, giving frame number becomes difficult as you have to count, instead give a label and directly goto that frame
	Expression	A value is evaluated and passed on
	Next Frame	When requirement is to move just to next frame
	Previous Frame	When requirement is to move just to previous frame
Go to and Stop(5) – to goto frame no: 5 and stop the animation Go to and Play(5) – to goto frame no: 5 and play the animation from there onward		

Movie Clip :

A movie clip is mini-flash file inside the existing file as it has its own timeline. When creating a movie clip, it is very important to have an **Instance Name** given, in order to identify the movie clip through script.

- **Controlling a movie clip :**

By controlling a movie clip we mean stopping it, playing it and going to any frame of that movie clip. For this we use "**Tell Target**" see syntax below

```
On (Release)
    Begin Tell Target ("/try")
        Go to and Stop (5)
    End Tell Target
End On
```

Movie clip having instance name "try" is targeted and stopped at frame no : 5. "***/***" indicates main timeline, the syntax translates as target the movie clip having instance name "try" on the main timeline.

Note : "/" indicates the main timeline.

- **To Target main time line :**

```
Begin Tell Target ("/")
    Go to and Play (15)
End Tell Target
```

This particular script will target the 15 frames on the main timeline.

- **To set a variable from the movieclip on the main timeline :**

  Set variable: **"/:display"**="Please enter correct value"

Here – **"/"** is indicating the main timeline, whereas **":"** is attached to the variable name.

- **To set a variable from main timeline in the movieclip :**

  Set variable: **"/movieclipname:variablename"**="Please enter correct value"

A variable's name can depend on where it exists. To access a variable in another timeline, use an **absolute** or **relative** path.

- Absolute paths to variables begin with a forward slash ("/"), indicating that the path starts from the main timeline. This is similar to accessing a root directory on a server.
- Relative paths can refer to timelines in the context of the current timeline. If the instance of a movie clip exists on the current timeline, a variable named x inside of it could be simply referred to as mc:x, using its relative path name. The absolute path would be /mc:x.

Relative paths starting with a dot ("..") indicate one level up from the current timeline. If a variable called x is set in the main timeline from a movie clip, it can be referred to as ..:x (relative path) or /:x (absolute path). The forward slash at the beginning is used to indicate the main timeline.

Property :

A property is an element, state or a condition of an object. In short it is the characteristics of an object. All the objects have certain set of Properties which are changeable. The property of an object can be changed through the frame (i.e. as soon as the flash file is loaded) and through actions (button).

- Property of an object can be *SET* to a pre-determined value e.g. An object should be situated at a particular co-ord & rotated by certain degrees as the flash file is loaded.
- Property of an object can also be *GET* (got) from its existing state for performing changes in its property.
- Properties can be SET or GET only for a movie clip.
- During runtime if the property of an a MovieClip is to be changed then it is important to know (*GET*)its existing property then new / changed property can be *SET*.

Some commonly used properties listed below :

e.g

```
Set Property ("/box", X Position) = 200
Set Property ("/box", Y Position) = 125
```

Sets the x & y location of the movie clip "box" to (200,125)

SetProperty

Syntax : SetProperty ("target", property)

X position Y position	Sets the position of the movieclip corresponding to value assigned as x / y co-ord
X scale Y scale	To scale the MovieClip along the x axis / y axis. Value 100 is the normal scale, value more than hundred increases the size & value less than hundred decreases the size
Alpha	Sets the transparency of the movieclip. Range is between 0 – 100. Value 0 makes the movieclip completely transparent whereas value 100 makes it opaque.
Visibility	Sets the visibility, either the object is visible or invisible, Range is 0 & 1
Rotation	Rotates the MovieClip to the angle set (0 through 360 degrees)

GetProperty

Syntax : GetProperty ("target", property)

_x _y	Gets the x / y position
_width _height	Gets width / height
_rotation	Gets the angle of rotation of the object
_xscale _yscale	Gets the x scale / y scale
_alpha	Gets the level of transparency
_visibility	
_droptarget	Gets the drop location of the drag object
_currentframe	Gets the current frame No.
_totalframes	Gets the total No. of Frames in the entire animation
_framesloaded	Gets the total No. of frames loaded

A small example to illustrate the use of Set & Get Property

```
On (Release)
        Set Property ("/try", X Position) = (GetProperty ("/circle", _x )) +5
        Set Property ("/try", X Scale) = int(GetProperty ( "/try", _xscale))*1.1
        Set Property ("/try", Y Scale) = int(GetProperty ( "/try", _yscale))*1.1
End On
```

- Other Action Script Related Commands :

Play	To play the animation	
Stop	To stop the animation	
GetURL	Opens a specified URL	
	Get URL ("http://www.dpsl.net,", window="_parent") Window – is the target area in which the specified URL should open	
FS Command	Issues an FS Command to Player	
	FS Command ("fullscreen", "true")	SWF file opens in maximized mode
	FS Command ("allowscale", "true")	Allows the SWF to scale
	FS Command ("showmenu", "true")	Displays Menubar, & menus on right click of mouse button
Load/Unload Movie	Load / unloads SWF files into the existing file at runtime. Decreases original file size, as other SWF are loaded on requirement	
	Load Movie ("URL", Level/target) Level – has numerical value beginning from 1 to n, level indicates the hierarchy in which the movies should be loaded. Target – is target area in which the specified URL should be loaded **[Note : If level is same, movie loaded first is replaced with the movie loaded having the same level.]** **Unload Movie(Level)** – unloads the movie loaded at a particular level	
If Frame is Loaded	Determines whether a specified frame is loaded. Action is often used in early frames as a preloader	
Duplicate MovieClip	Duplicates a movie during runtime **Duplicate movie clip("/movieclip_name", "new name",level)** Level – has numerical value beginning from 1 to n, level indicates the hierarchy in which the movies should be loaded. Removes movie clip a movie which is a result of earlier mentioned Duplicate movie clip command **Remove movie clip("/movieclip_name")**	
Drag Movie Clip	Makes a movie draggable. **[To make a movie draggable, create a button inside it where the draggable code is written]** **Start Drag("movie_clip_name", constrain to rectangle, lock mouse to centre)** e.g. Start Drag ("/green", L=50, T=5, R=05, B=05, lockcenter) /green - movie clip name constrain to rectangle - defines the draggable area lockcenter - drags the movieclip from the center point Button will have on Press mouseevent, for drag to happen **Stop Drag** – Stops the drag action of the movieclip	

Conditional Action Script :

An expression is any phrase that Flash can evaluate to a value. Flash has three types of expressions: string expressions, numerical expressions, and comparison expressions. You create an expression by combining operators and values.

- **LOOP :**

Use the Loop statement to set up a series of statements that run repeatedly while a specific condition remains true. A common use of looping is to use a variable as a counter and perform an action while the counter is less than a specified value. At the end of each loop, you increment the counter. To use the Loop statement effectively, you should be familiar with creating expressions that evaluate conditions.

1) The following action script displays the use loop. The script is executed five times (loop), each time it extracts one character from variable “name”.

```
On (Release)
        Loop While (ctr <= 5)
              Set Variable: "display" = Substring ( name, 1, ctr )
               Set Variable: "ctr" = ctr+1
        End Loop
End On
```

2) Loop within a Loop : Here the first loop is used to duplicate a movie clip, as soon as one movieclip is duplicated in positive x-direction, it enter into another loop to duplicate the movie clip postive y-direction, movieclip is duplicated 10 times and goes back to the outer loop to increase one more movieclip in x-direction and again enters the second loop. End result is a matrix of 10 * 10 of duplicate movieclip.

```
Loop While (i < 11)
      Set Variable: "d" = d+1
      Duplicate Movie Clip ("/deep/aa", "aa"&d, d)
      Set Property ("/deep/aa"&d, Y Position) = GetProperty ( "/deep/aa"&i, _y ) + 8.5
            Loop While (a < 11)
                  Set Variable: "b" = b+1
                  Duplicate Movie Clip ("/deep/aa"&d, "aa"&d&b, d&b)
                  Set Property ("/deep/aa"&d&b, X Position) =
                  GetProperty ("/deep/aa"&d&a, _x ) + 8.5
                 Set Variable: "a" = a+1
            End Loop
     Set Variable: "a" = 0
     Set Variable: "b" = 0
     Set Variable: "i" = i+1
End Loop
```

- **IF :**

If statement is used to set up statements that run only when a certain condition exists. i.e a particular set of statements / commands are executed only if the conditions set is true and if not true another set of statements / commands are executed.
e.g.

```
if (username eq "inter")
        Set Variable: "message" = "Allowed"
Else
        Set Variable: "message" = "Not Allowed"
End If
```

The above action script display the use of "if" statement. Here a variable *"username"* is checked to contain value *"inter"*, if value is present, another variable *"message"* displays Allowed else displays Not Allowed.

The "If" statement can be used in various combination, as stated below

1) If password is flash123, then a message "Allowed "should be displayed

```
if (password eq "flash123")
        Set Variable: "message" = "Allowed"
End If
```

2) Condition to give classes depending on the percentage i.e more than 75 Distinction, more than 60 Firstclass, more than 60 is pass class & less than 50 is fail.

```
if (percent >= 75)
        Set Variable: "disppercent" = "Distinction"
Else if (percent >=60)
        Set Variable: "disppercent" = "First Class "
Else if (percent >=50)
        Set Variable: "disppercent" = "Pass Class "
Else
        Set Variable: " disppercent " = "Fail"
End If
```

3) Condition for dragging a Movieclip and checking the area where it is dropped, If it is in the proper location a message “Correct” is diplayed otherwise “Wrong” message is displayed, and the draggable movieclip goes to the original position .

```
If (GetProperty ( "/circle", _droptarget ) eq "/tcircle" or
        GetProperty ( "/circle", _droptarget ) eq "/tcircle1")
        Set Property ("/circle", X Position) = GetProperty ( "/tcircle", _x )
        Set Property ("/circle", Y Position) = GetProperty ( "/tcircle", _y )
        Set Variable: "/:display" = "correct"
        Go to and Stop (1)
Else
        Set Variable: "/:display" = "wrong"
        Set Property ("/circle", X Position) = cx
        Set Property ("/circle", Y Position) = cy
End If
```

Note : In the above “If” statement GetProperty & SetProperty are used.]

4) Condition to check if question is 1 and the question is not attempted previously and to check whether the answer is correct, if correct increment the score & increment the attempt.

```
If (question_no eq “1”)
        If (visited1 eq "FALSE" and Clicked ne "x")
                If (Clicked eq correct)
                    Set Variable: "score" = score+1
                End If
        Set Variable: "visited1" = "TRUE"
        Set Variable: "attempted" = attempted+1
        End If
End If
```

5) if within a loop :

```
On (Release)
        Loop While (ctr <= 5)
            Set Variable: "display" = Substring ( name, 1, ctr )
                        If ctr eq “3”
                            Set Variable: "display"= "Use of Substring"
                        End if
             Set Variable: "ctr" = ctr+1
         End Loop
End On
```

- **Numeric Functions used In Flash Scripting :**

Int	Converts a number to integer
	e.g Set Variable: "display" = int(average)
Random	Generates a series of random numbers
	e.g *Set Variable: "rannum" = random(4)+1* Generates 5, Starts generating random numbers from 0 onwards, hence plus 1.

- **String Functions used In Flash Scripting :**

Substring	Extracts specified no: of characters from a string. It can be used in a "Loop" also.
	e.g Set Variable: "display" = Substring("international",6,6) Beginning from 6th character 6 characters are extracted from string "international"
Length	Calculates length of a string
	e.g Set Variable: "totalchar" = length(username)
Chr	Converts ASCII code to a character
	e.g Set Variable: "display" = chr(65)
Ord	Converts character to an ASCII code
	e.g Set Variable: "display" = Ord("A")