Scripting Tutorials 2012

Scripting Basics

9/2/2012

Kirby_422

Table of Contents

Script Types **1**

Startup 1

Continuous 2

Dormant 2

Static 4

Stub 5

Global Variables **6**

Saving Scripts & Other Basics **6**

Script File Type 6

Save Location 7

Compile Scripts 7

HS_Doc.txt 7

Console & Dev Mode 7

Common Errors **8**

Unmatched Parenthesis 8

Expected 'Script' or 'Global' 8

Not a valid _____ 9

Script Types

Startup

These scripts execute once, when the map has finished loading. Common uses are setting items up in a map, and for linear single player maps.

Example 1

```
(script startup detect_host
 (if (!= (unit_get_health random_vehicle) -1) (object_create_containing vehicle))
 ;;If a named vehicle exists, this proves they are the host. If they are the host,
 ;;create every other object that has the word 'vehicle' in their name.
)
```

Example 2

```
(script startup mission_assault
 (cinematic_start) ;;Cutscene
 (camera_set_dead enemy_commander) ;;Camera 3p around enemy character
 (camera_control 1) ;;Allows the camera to do that
 (sleep 120) ;;Displays for 4 seconds
 (cinematic_stop) ;;End cutscene
 (camera_control 0) ;;Return camera to normal
 (fade_in 0 0 0 60) ;;Fade in from black, 2 seconds.
 (sleep 60) ;; Wait 2 seconds before continuing.
 (ai_place enemy_squad) ;;Place enemy encounter
 (ai_attach enemy_commander enemy_squad) ;;Make that enemy character part of the AI
 (sleep_until (< (ai_living_count enemy_squad) 3) 150) ;;Wait until less than 3 of that AI are alive.
;;Checks every 5 seconds.
 (ai_place backup_squad) ;;Spawns more AI of a different group.
 (object_create_anew dropship) ;;Creates the dropship
 (vehicle_load_magic dropship "" (ai_actors backup_squad)) ;;Loads them into a dropship
 (recording_play_and_hover dropship entry_recorded_animation) ;;Plays a recorded animation.
;;When the animation ends, the ship hovers in place.
 (sleep (max 1 (recording_time dropship))) ;;Wait for the duration of the dropship animation.
;;If there was an error, wait at least 1/30th of a second, to prevent the script from breaking.
 (vehicle_unload dropship "") ;;Get the AI out of the ship.
 (sleep 150) ;;Wait 5 seconds.
 (vehicle_hover dropship false) ;;Stop floating idle
 (recording_play_and_delete dropship exit_recorded_animation) ;;Fly away and remove the dropship.
 (sleep_until (= (ai_living_count backup_squad) 0) 300) ;;Wait until AI are all dead.
;;checks every 10 seconds
 (game_won) ;;Tells the map that you have won. If the map name is one of the default ones,
;; it will progress to the next level. Otherwise... nothing.
)
```

Continuous

These scripts will rerun themselves over and over again. These are used for things you expect to happen more than once. You can, however, stop their execution if you so choose.

Example 1

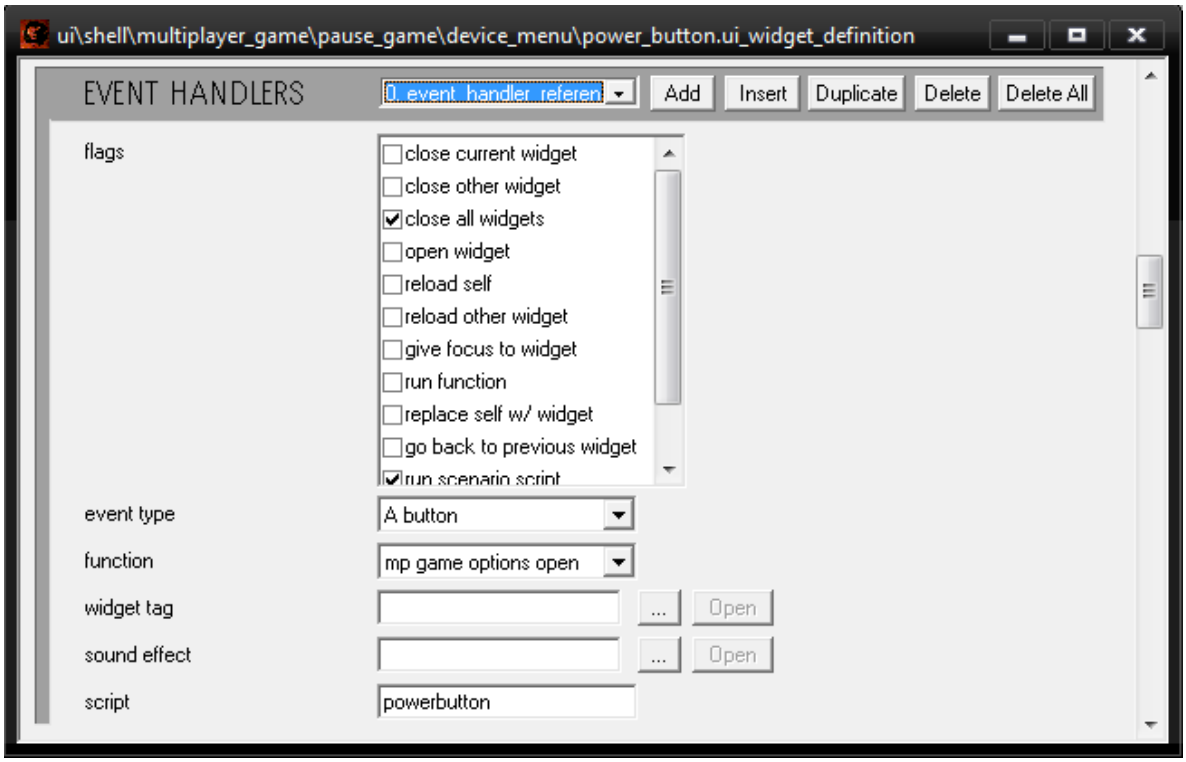
```
(script continuous basic_vehicle_teleporter
  (if (volume_test_object trigger_volume_a vehicle_01) (object_teleport vehicle_01 flag_b)
  (if (volume_test_object trigger_volume_b vehicle_01) (object_teleport vehicle_01 flag_a)
  ))
  ;;This script checks if they are in volume a, if they are, teleport to cutscene flag b.
  ;;If they are not (and only if they are not), check if they are in B, and if so, teleport to A.
  ;;(if <boolean> <then> <else>)
  ;;The way this script is, you cannot place flag_b inside trigger_volume_b, same with A.
)
```

Example 2

```
(global long run_count 0) ;;See Global Variable section; Think variables.
(script continuous run_500000_times
  (begin_random ;;Execute the following in a random order.
    (effect_new "effects\custom\firework" flag_01)  ;;creates that effect tag, at flag_01
    (effect_new "effects\custom\firework" flag_02)
    (effect_new "effects\custom\firework" flag_03)
    (effect_new "effects\custom\firework" flag_04)
    (effect_new "effects\custom\firework" flag_05)
    (effect_new "effects\custom\firework" flag_06)
    (sleep 60)  ;;Wait 2 seconds.
    (sleep 60)
    (sleep 60)
    (sleep 60)
    (sleep 60)
    (sleep 60)
  )
  (if (> 500000 run_count) (set run_count (+ run_count 1)) (sleep -1))
;;If the run_count is less than 500,000, increase run_count by 1.
;;Otherwise, stop execution of this script.
)
```

Dormant

When used in scripts, they are essentially the same as a startup, except they execute when you tell them to rather than automatically. These however, can also be executed from User Interface Widgets; When ran from a widget, they can be re-executed as many times as wanted.



Note: Default versions of Guerilla will have limited ability to edit widget tags. Try Kornman00v2, or OSGuerilla.

When using a widget, the script will only run on the PC who triggers it. Unless it is something that clients only use for their own personal preferences that do not effect others, it should be restricted so only host can click it. "mp game options open" is an effective way of making it disabled on clients.

Example 1

```
(script dormant powerbutton ;;To go with the picture
  (device_set_power my_device 1);;Set a device's power on.
)
```

Example 2

```
  (global boolean cliff_wave false) ;;See Global Variable section.
  (global boolean river_wave false)
  (script dormant cliff_waves
    (ai_place cliff_wave_1) ;;Spawn AI
    (sleep_until (< (ai_living_count cliff_wave_1) 3) 150) ;;Wait until less than 3 are left alive.
    (ai_place cliff_wave_2)  ;;Spawn AI
    (sleep_until (< (ai_living_count cliff_wave_2) 3) 150) ;;Wait until less than 3 are left alive.
    (set cliff_wave true) ;;Set global variable to True.
  )
  (script dormant river_waves
```

```
  (ai_place river_wave_1) ;;Spawn AI.
  (sleep_until (< (ai_living_count river_wave_1) 3) 150) ;;Wait until less than 3 are left alive.
  (ai_place river_wave_2) ;;Spawn AI.
  (sleep_until (< (ai_living_count river_wave_2) 3) 150) ;;Wait until less than 3 are left alive.
  (set river_wave true) ;;Set global variable to true
)
(script startup missions
  (ai_place first_enemies) ;;Spawn AI.
  (sleep_until (= (ai_living_count first_enemies) 0) 150) ;;Wait until 0 are left alive.
  (wake cliff_waves) ;;Begin execution of dormant script, cliff_waves
;;This does not effect this scripts execution, it continues while cliff_waves executes.
  (wake river_waves) ;;Begin execution of formant script, river_waves.
  (sleep_until (and cliff_wave river_wave) 150) ;;Waits until global variables,
;; cliff_wave and river_wave, are both true.
  (game_won)
)
```

Static

These scripts are basically shortcuts. If you need to do a certain thing a number of times, it is easier to make a static script that does it.

Example 1

```
(script static unit player0
  (unit (list_get (players) 0)) ;;Returns the first player.
;;The play can now just type (player0) instead of the above.
)
(script startup idk
  (sleep_until (< (unit_get_shield (player0)) 1) 5) ;;Wait until P0's shields aren't full
  (sv_say "Player 0's shields are damaged!") ;;Have the server declare it in the chat.
;;... idk why either. I just need a quick example *shrug*
)
```

Example 2

```
(global short var1 0) ;;Global Variable ;;Input 1
(global short var2 0) ;;Global Variable ;;Input 2
(global real result1 0) ;;Global Variable ;;Result as a decimal number
(global short result2 0) ;;Global Variable ;;Correct result.
(script static short modulus ;;Returns a 'Short' (Non-decimal number, within 2^16)
  (set result1 (/ var1 var2)) ;;Get decimal of input1 / input2
  (set result2 (/ var1 var2)) ;;Get non-decimal copy of input1 / input2
  (set result1 (- result1 result2)) ;;Difference between decimal and non-decimal (less than 1)
  (set result2 (* result1 var2)) ;;Difference * input 2, without decimals
  (set result1 (* result1 var2)) ;;Difference * input 2, with decimals
  (if (!= result1 result2) (set result2 (+ result2 1))) ;;If data was lost in decimals, restore rounding.
```

```
  (+ result2 0) ;;Return result2 variable.
)
(script startup idk
  (set var1 16) ;;set input1 as 16.
  (set var2 (list_count (players))) ;;Input2 as the number of players (Counting from 1)
  (fade_in 0 0 0 (modulus)) ;;Fade in from black, at the speed of '16 mod (player count)'
)
```

Stub

These scripts are made to be overwritten. With this, you can perform recursion. These scripts are the only scripts that can share names with other scripts.

Example 1

```
(global short in0 0)
(global short in1 0)
(script stub short modulus
;;This here is garbage, it's just to let sapien know
;;that a script called modulus that returns a short, exists.
;;Without this, the static script would have an error that 'modulus' is undefined.
  (+ 0 0) ;;Returns a number. This'll always return 0.
)
(script static short modulus
  (if (< in0 in1) (+ 0 in0) ;;If in0 is smaller than in1, return in0.
   (begin
     (set in0 (- in0 in1)) ;;Otherwise, reduce in0 by in1.
     (modulus) ;;Then repeat this.
  ))
)
```

Example 2

```
(script stub unit player
  (unit bob)
)
(script startup whatever
  (unit_kill (player))
)
(script static unit player
  (unit (list_get (players) 0))
)
```

Global Variables

These store data for scripts to read and write into. You can make almost anything into a global, the few exceptions being stuff like BSPs or skies. You can even have a global for a script, although there wouldn't be much point. They can be checked and altered via devmode. Halo has some globals by default, such as cheats like cheat_deathless_player.

Example 1

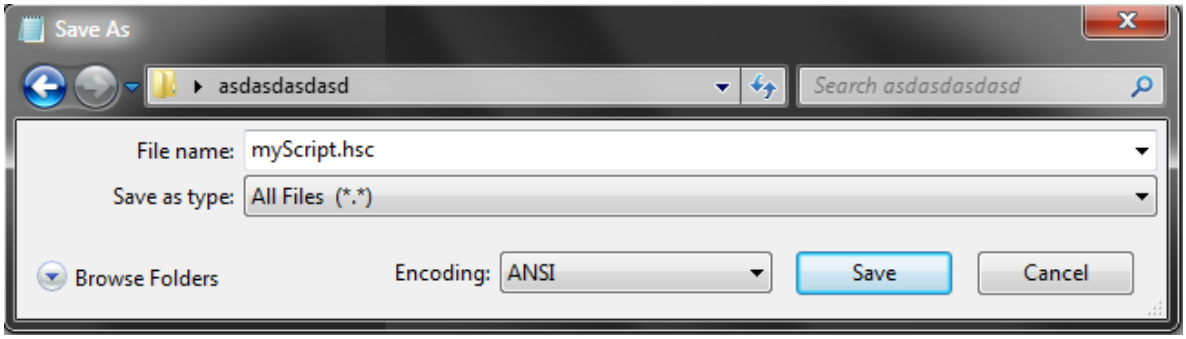
```
(global short number 0) ;;Global named Number; Initial value is 0.
(script continuous BSP stuffz
  (sleep 30000) ;;wait one thousand seconds
  (set number (+ number 1)) ;; increment the global variable, number.
  (switch_bsp number) ;;Set the BSP to the amount stored in number.
)
```

Example 2

```
(global boolean debugging false)
(script continuous something
  (sleep_until (volume_test_objects base_01_volume (players)) 5) ;;Wait until players enter volume.
  (device_set_position base_01_defenses 1) ;;Activate device fully (Move to open position)
  (if debugging (sv_say "debug: defenses on")) ;;If debugging variable is true, announce debug info.
  (sleep_until (not (volume_test_objects base_01_volume (players))) 5) ;;Wait until players exit.
  (device_set_position base_01_defenses 0) ;;Deactivate device (move to closed positon)
  (if debugging (sv_say "debug: defenses off")) ;;If debugging variable is true, announce debug info.
)
```

Saving Scripts & Other Basics

Script File Type



In your text editor (Such as Notepad which comes with Windows), enter a name followed by .hsc, and select "Save as type" as All Files. The file encoding must be ANSI, if you select Unicode, sapien will return symbols as an error.

Save Location

If your scenario tag where located as follows:
<Halo Custom Edition>\tags\levels\test\bloodgulch\bloodgulch.scenario

You would name your script (with whatever name you choose) in:
<Halo Custom Edition>\data\levels\test\bloodgulch\scripts\<yourscript>.hsc

When compiling hsc files, it does it in alphabetical order. If you have more than 1 HSC, and one references something in another one, you must alphabetically sort them so the one that gets referenced compiles first, or else it won't know it exists.

Compile Scripts

Open the scenario in sapien, File, Compile Scripts.

HS_Doc.txt

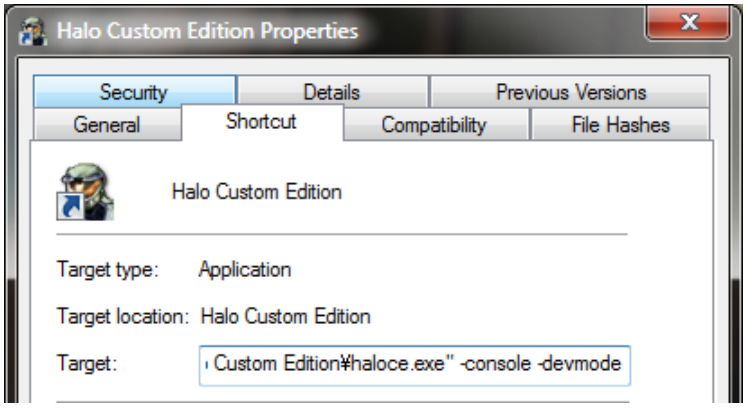
This file lists all available script commands (It does not, however, display any globals) To generate this file, open sapien, press the ~ key in the top left corner of your keyboard, and type script_doc, then press enter. If you are using a default version of sapien, hs_doc.txt will appear in your halo custom edition folder. If you are using an OpenSauce version of sapien, it will appear in:
<User Name>\Documents\My games\Halo CE\OpenSauce\Reports\hs_doc.txt

If you want a list of globals, refer to the bottom of this document **Here**.

For OS specific commands, please refer to the OpenSauce wiki page **Here**.

Console & Dev Mode

This is a feature in-game, that allows you to perform script functions anywhere. This is especially useful for debugging. To start the game in devmode, create a shortcut to Halo Custom Edition, and add both -console and -devmode to the shortcut target like so:



With devmode enabled, you will not be able to play online. With just console enabled, you will not be able to perform any commands that do not start with sv, such as sv_say. If you have OpenSauce installed, however, you will be able to use every command with just console enabled, while still being able to play online.

A good use of Dev Mode, falls back on the second Global example. That debugging variable; the script never changed that on its own. Through devmode, you can type 'set debugging true', and it will inform you when things are happening. Like-wise, you can make a short for debugging, that different parts of the script set to different numbers. With 'inspect debugging', you will be able to figure out where the script is in its execution, like when a script freezes for some reason.

Common Errors

Unmatched Parenthesis

If you receive this error AFTER you receive another error, ignore it until after you fix the other error and try recompiling. When the first error is encountered, it stops compiling. Since it is not finished compiling, it hasn't checked if you did this correctly.

What this error means (when you get it by itself) is that the number of "(", do not match the number of ")" in your script. Each script command is encased in its own (), missing one of these, or the one for the script itself can be the cause. If you receive this error, read through the script to make sure each script is closed correctly. Anything commented out will not be read either, so if you've commented out a), that's why it's not being noticed.

Examples

```
(script continuous whatever
  (unit_kill bob)
 ;;This script is missing the ) to finish the '(script continuous whatever'.
(script startup whatever
  (sv_say "Hello Everyone!"
  (sv_say "You will be baked, and then there will be cake")
) ;;The first sv_say is missing a ), it will give you an error about the number
  ;;of arguments here since sv_say is incomplete.
```

Expected Script or Global

If you write something outside of a script or global variable, you'll receive this error. This sometimes happens by someone putting too many ")" in the middle of their script, finishing off the script before they meant to. This will also happen if you misspell script or global. If you saved your file as unicode, you'll also get this error.

Examples

```
(srcipt startup myscript ;;It's not declaring a script, its declaring a 'srcipt' whatever that is.
  (sv_say Haiii)
)
(script continuous killpeople
  (unit_kill bob) )
  (unit_kill joe) ;;This is outside of the script.
)
```

not a valid _____

If you try to compile a script, that references an item, tag location, camera point, etc. that does not exist in the scenario (or, tag folder for tag location), it will give you this error. Simply correct it by placing an item of that type in the scenario, correct the script, etc. If you get an issue about a script not being valid, yet it exists in your hsc, please check the ordering of your HSC. Before a script can be referenced, it has to be compiled. Reorder your hsc file so that things such as global, dormant, static, and stub, are above where they are first called from, like the top of the script.

You should be able to figure errors out by yourself as long as you understand English.

Kirby_422