Developer Console

USER MANUAL

Developed by South East Games

Contents

Description

The Developer Console script is a turn-key, generic developer console for use at run time to assist in dynamically adjusting variables and calling functions. Besides being an essential debugging tool, the console can easily be graphically adjusted and provided with alias terms for simple use by end users.

Similar in style and function to consoles found most typically in PC games such as first person shooters, the console can be set to appear at a variable rate on a given key press (Default: ~). Any public variable or method on any Component on any GameObject in the loaded scene can be accessed, set or called from within the Developer Console. All possibly entries are easily identified and entered with an auto-complete system similar to Microsoft Visual Studio's Intellisense.

Installation

The Developer Console is condensed into a single MonoBehaviour based script that needs only to be applied to a GameObject in a scene. Provided in the package is both the script and a Prefab object containing the script. Simply drag and drop the Prefab object into the scene and the Developer Console will be accessible from within the running game.

Setup

The Developer Console's inspector allows for a number of general settings to be applied (listed below), as well as settings for the size and look of the console and aliases.

Settings

Setting Name	Setting Description
`GUI Skin`	An optional GUI Skin for rendering the console
`Y Size Percent`	Height of the console in percentage of the screen size (0.0 to 1.0)
`X Size Percent`	Width of the console in percentage of the screen size (0.0 to 1.0)
`Text Field Height`	Height in pixels of the console's textbox
`Line Height`	Height of each text line in the console's output window
`Transition Time`	Time (in seconds) that the console takes to enter and leave the frame
`Autocomplete Box Max Size`	Maximum pixel size of the autocomplete window
`Autocomplete Selected Color`	Color of the selected autocomplete option
`Toggle Key Code`	The key used to toggle the display of the console
`Autocomplete Key Code`	The key used to trigger text autocompletion during typing
`Copy Log Output`	Whether or not to print standard log (Debug.Log) output to the Developer Console
`Show Cursor When Active`	Show the mouse cursor when the console is showing
`Only Accept Aliases`	Whether or not to only accept aliases and not the standard variable and function call syntax
`Alias List`	List of all aliases the console will utilise at runtime

Usage

The main three purposes of the console are

- Printing the value of a variable
- Setting the value of a variable
- Calling a method on a component

Retrieve Value

To print the value of a variable or result of a method, the name of the GameObject, Component type name and variable/method name need to be typed into the console with dots ('.') in between and following the word 'print'.

E.G. 1

```
‘print Main Camera.Transform.childCount’
```

E.G. 2

```
‘print Main Camera.Transform.GetChildCount’
```

The above examples will print to the console the number of child transforms the transform component on the Main Camera GameObject has. Example 1 shows printing the value of the Field/Property and Example 2 shows printing the value of a method.

Set Value

To set the value of a variable, you must enter the variable in the same way as above, but then provide the new value separated by a space as a string.

E.G.

```
‘GUI Text.GUIText.text NewText’
```

The above will set text of the GUIText component on the ‘GUI Text’ GameObject to “NewText”.

In order to allow for spaces in strings, double quotation marks (“) must surround the string.

E.G.

‘GUI Text.GUIText.text “New Text Now With Spaces”’

Call Method

To call a public method in component, the GameObject name, Component type name and method name need to be typed into the console in the same way as above.

E.G.

```
‘Main Camera.Transform.Rotate 0,90,0’
```

The above will rotate the transform of the Main Camera GameObject 90 degrees on the Y axis.

Note that the above example accepts a Vector3 by entering comma separated numbers. The below table lists the types that can be converted to in the console from a given string and the format in which the console accepts them. The console will automatically convert the entered string to the appropriate type if possible when either setting a variable of calling a function.

Accepted Type	Example String	
INT	0	
FLOAT	0.1	
DOUBLE	0.1	
BOOL	0 / 1	false / true
STRING	EGString	“EG String”
VECTOR2	45,0	
VECTOR3	45,0,45	
QUATERNION	0,0,0,1	
COLOR	1,0,0,0.5	

Aliases

An alias is a string that can be used instead of writing another full string.
E.G.

Alias Name: `sm_Scene`
Alias Value: `call SceneManagerOb.SceneManager.LoadScene`

Typing ‘`sm_Scene MainMenu`’ into the console would give the same result as typing ‘`call SceneManagerOb.SceneManager.LoadScene MainMenu`’.

Aliases are a great way of giving end users access to functions that are useful to them in a simple, intuitive way.

License Information

Purchasing this tool gives the purchaser a non-transferable license to install and use the tool on a single machine at a time for use by a single user (the purchaser).

Source code may be modified to suit the purpose of the purchaser, but all code and content, original and edited, may not be distributed in source form.

Following the above, the purchaser has the right to sell and distribute any number of built applications containing this tool in a compiled, binary form.