Device Console Documentation

DeviceConsole

Handles displaying logs during runtime and executing commands. The DeviceConsole can be opened by either pressing the '~' command on your keyboard or 3-finger tapping the screen on device.

Inspector Properties:

- **UI Container** - The GameObject that will be set active/de-active when the consoles visibility is toggled.
- **Log Container -** The GameObject which will become the parent of all log GameObjects.
- **Command Input Field** - The Unity.UI.InputField which will be used to enter commands.
- **Auto Focus Input Field** - If checked, every time the console is opened and every time after a command is executed the Command Input Field will gain focus.
- **Header Colour** - The colour of the header text.
- **Header Text** - The text that always appears at the beginning of the logs.
- **Log Prefab -** The prefab that will be used for simple logs.
- **Warning Log Prefab -** The prefab that will be used for warning logs.
- **Error Log Prefab -** The prefab that will be used for error logs.
- **Assert Log Prefab** - The prefab that will be used for assert logs.
- **Exception Log Prefab** - The prefab that will be used for exception logs.
- **Exception Stack Trace Prefab** - The prefab that will be used for exception statck traces.

DeviceLogUI

Simple script that is used by DeviceConsole to get the Unity.UI.Text to place the log message text in.

DebugLogs

The ScriptableObject that holds a List of all the logs during runtime. This class must be instantiated before logs will begain to appear in the DeviceConsole. DebugLogs is instantiated in DeviceConsoles Awake methods, however, it can be forced to be instantiated earlier by calling the static method DebugLogs.Touch().

DebugCommands

The ScriptableObject that holds all the commands that can be exectued by DeviceConsole. Commands can be added by calling the AddCommand method. DeviceConsole already comes with a number of commands, they are added in the DeviceConsoles Awake method.