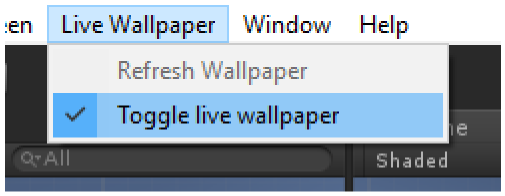
Live Wallpaper Documentation

1. ABOUT:

1.1. **About the asset** – This asset is an extension aimed to set a unity application as wallpaper with a minimum effort from the user as possible.
It's supported only on windows machines.

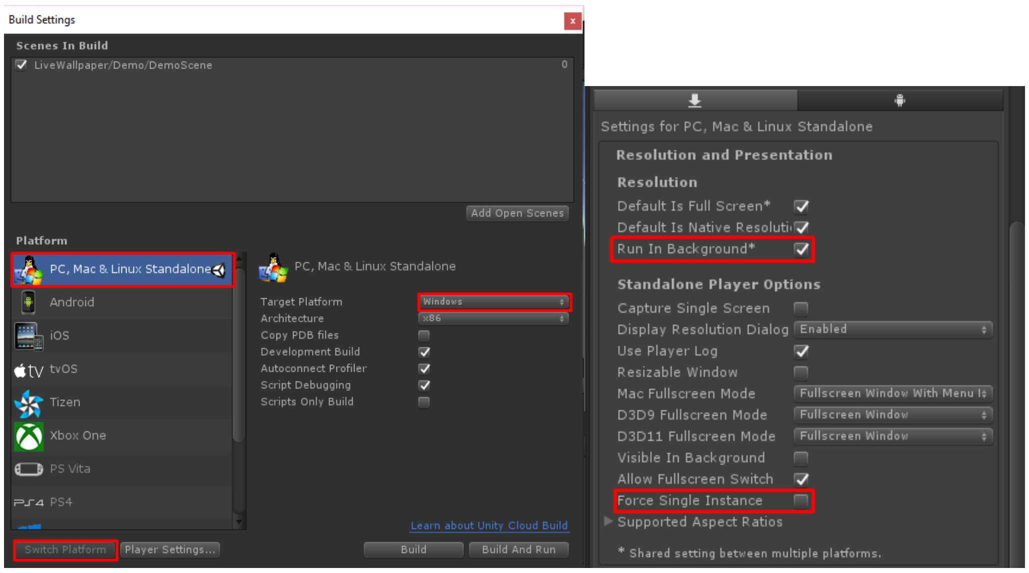
1.2. **Launcher** – It is a tool to directly start your application as a wallpaper, without the need of any code.
Once you've built your project the .exe file will appear next to your application .exe, it will have a name like this: "YourProjectName_wallpaper.exe" (it can be renamed to whatever you want).
You'll need to compile the launcher with a separated tool (VS, for example) if you want to customize its icon.

1.3. **Editor support** – Unity Editor is fully supported, you can use the same methods as the build or use one of the MenuItems, in the top bar of the Editor.



2. HOW TO USE + SETUP:

2.1. **Setup** – The target platform in build settings must be set to windows standalone (make sure to click "switch platform" if you change it), and you need to enable the "Run in Background" at player settings, otherwise, your application will freeze as soon as it loses focus. You may also want to enable the "Force single Instance" if you're sure your application will always be a live wallpaper, but that's optional.



Scenes In Build
☑ LiveWallpaper/Demo/DemoScene 0

PC, Mac & Linux Standalone

Target Platform	Windows
Architecture	x86
Copy PDB files	☐
Development Build	✓
Autoconnect Profiler	✓
Script Debugging	✓
Scripts Only Build	☐

Settings for PC, Mac & Linux Standalone

Resolution and Presentation

Resolution

Default Is Full Screen*	✓
Default Is Native Resoluti	✓
Run In Background*	✓

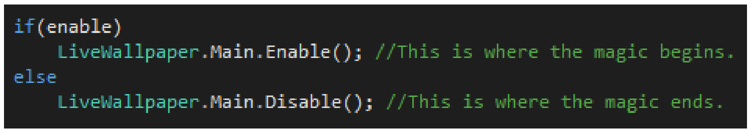
Standalone Player Options

Capture Single Screen	☐
Display Resolution Dialog	Enabled
Use Player Log	✓
Resizable Window	☐
Mac Fullscreen Mode	Fullscreen Window With Menu I
D3D9 Fullscreen Mode	Fullscreen Window
D3D11 Fullscreen Mode	Fullscreen Window
Visible In Background	☐
Allow Fullscreen Switch	✓
Force Single Instance	☐

Supported Aspect Ratios

* Shared setting between multiple platforms.

2.2. **Main usage (Live Wallpaper)** – You just need to call LiveWallpaper.Main.Enable(); to begin and LiveWallpaper.Main.Disable(); to end. Pretty simple, isn't it?



```
if(enable)
    LiveWallpaper.Main.Enable(); //This is where the magic begins.
else
    LiveWallpaper.Main.Disable(); //This is where the magic ends.
```

2.3. **Classes** – Briefly explanation of each class and what it is for, all the methods in classes listed below have XML comments explaining their usage, you can check them in the source code or in the tooltips while programming.
All the classes are in a namespace called "LiveWallpaperCore", the only exception is the LiveWallpaper class, that is in the global namespace.

- **LiveWallpaper** – Contains methods and properties for managing live wallpapers, in most cases you will only need the main live wallpaper, which can be accessed by LiveWallpaper.Main.

- **EditorLiveWallpaper<T>** – Same as the LiveWallpaper, but for the editor, it'll use an EditorWindow instead of a window handle.

- **Fullscreen** – Contains methods for setting a window fullscreen and/or borderless, it's used automatically by LiveWallpaper when enabling to make sure the window fits the entire screen.

- **Wallpaper** – Class for getting and setting the static wallpaper, the one with images, methods of this class will have no apparent effect while a LiveWallpaper is enabled.

- **HookedInput** – As you may know, if you read the asset store description, the Input class doesn't play well with this asset, this is due to the lack of focus in the wallpaper mode, the HookedInput is a workaround for this, it uses low-level hooks to get the inputs, its usage is basically the same as the normal Input class

- **HookedInputModule** – A class derived from StandaloneInputModule that uses the HookedInput to pass events to the UI and sprites.
  Replacing the StandaloneInputModule with this you'll be able to make your UI work while in live wallpaper mode.

The classes that are not listed here may not be relevant enough to have an explanation, with the classes listed above you can do pretty much everything you'd do with the non-listed ones.

2.4. **Multi-screen support** – On multi-screens setups, the window will be resized to fit all the monitors, and it will have "invisible parts" if the screens are not aligned, for example, if one screen is placed on the top-left and other in the bottom-right, then the top-right and bottom-left would be invisible but they would still be rendered as part of the screen, to override this behavior you can use manual handle parenting and manual fullscreen.

3. SUPPORT:

3.1. **Contact** – Don't hesitate in contacting me (email below), I'll do my best to explain something or solve your problem.

3.2. **Bug report** – For reporting a bug with the asset send me an email (samuelschultze@gmail.com) with the following things:

- A detailed explanation of the bug, how it happened and what you were trying to achieve.
- Your Windows version.
- Your Unity version + Platform (Unity Editor or Standalone)
- The code that started it.
- The error that was thrown, if any.
- Optional: Steps for reproducing the bug, if you know exactly how it happened.

Please understand that this kind of information is extremely important, it helps me figure out what's happening and fix it a lot faster.

4. KNOWN ISSUES:

4.1. A fullscreen window will be created when opening Unity Editor if you haven't disabled the live wallpaper before closing it, restarting Unity will fix this.

5. CHANGELOG:

Version 1.0.0 (28/07/2017):

- Initial release