Hidden Object Game Template

Game documentation and HowTo guide.

This document contains:

Package Description and features

Hidden Object Game is a full Unity template ready for release. It is compatible with mobile as well as standalone and webplayer.

How to Play?

Try to find the hidden object in all the clutter. Find it quickly for a higher bonus.

Watch gameplay video

Try the webplayer

Current version 1.33

Update history

1.33 (03.08.2016)
- Support for Unity 5.3 and higher versions.
- Better support for UnityAds 5.2 and above.

1.31 (13.12.2015)
- Added support for Unity 5.3 SceneManager
- Uploaded several versions: 4.6.9, 5.1.4, 5.2.3 and 5.3

1.3 (15.09.2015)
- Fixed double click on hidden object that counts as two found objects.
- Fixed Global Music not registering in PlayerPrefs.
- Scattered objects across Z axis to prevent clipping on the lower left part of the game area.

1.28 (15.08.2015)
- Added Global Music script. Now the music flows seamlessly between scenes.
- Minor fixes to C# version.

1.26 (23.07.2015)

- Entire project converted to C#. The package now contains both JS and C#,
both fully featured.

1.15 (29.09.2014)
- Added a script to the camera which automatically changes the game area to fit all the preset aspect ratios (5:4,4:3,3:2,10:16,16:10) as well as any custom resolution you want. This allows you to build the game once and make it fit all possible devices.

1.1 (09.09.2014)
- Added an option to use icons for hidden objects instead of text. (Thanks to David Smith for the suggestion).
- Minor bug fixes, updated code comments and documentation.

Credits

The sounds are courtesy of the free sound project.

Credits go to these authors for their great sound samples: **justinbw, greencouch, and various others**

Music track is an edited clip from **Bassa Island by Incompetech Kevin MacLeod**

Please rate my file, I'd appreciate it

Overview of the game's library contents

Let's take a look inside the game files. Open the main HOGAssets folder using Unity3D 4.5 or newer. Take a look at the project library, usually placed on the right or bottom side of the screen. Here are the various folders inside:

- **Animations:** Holds the animation clips made with Unity's built-in animation system.

- **FLA:** Holds the object graphics made with Flash CS3. These are vector graphics than can be easily scaled without loss of quality and then exported as PNG to be used in Unity.

- **Fonts:** Holds the fonts used in the game. The fonts are AGENCYB, and AGENCYR.

- **Prefabs:** Holds all the prefabs used in the game. These are distributed to various folders for easier access, Buttons, Levels, Objects, etc

- **Scenes:** The first scene that runs in the game is MainMenu. From this scene you can get to any of the 3 levels.

- **Scripts:** Holds all the scripts used in the game. Each prefab contains one or more of these scripts.

- **Sounds:** Holds all the sounds used in the game. Click, Win, Lose, etc

- **Textures:** Holds all the textures used in the game which are used as sprites in Unity.

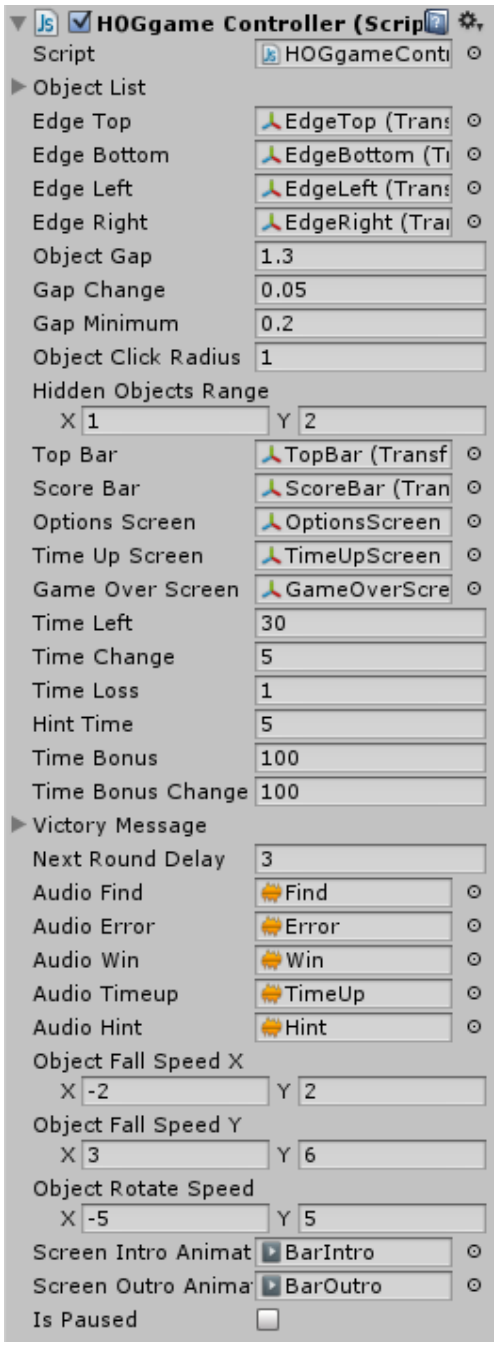
Customization guide

Getting started

HOG is considered a complete project, and as such is supposed to work as the starting point of your planned game, rather than an addition to an existing project. That said, you may of course pick and choose some of the scripts/models to import into you existing project, but HOG works best as a starter kit which you can customize any part of to your liking.

How a level is made

Each level in HOG is contained in a prefab and can be easily edited to make the level more challenging and varied.



Open the scene JSEasy from the folder JS_Assets > JS_Scenes and then click on the object GCEasy to see the game controller component. This component defines the entire game.

Object List: The first thing you can set is list of objects in the game. You can simply drag objects to add to the list.

Edge Top/Bottom/Left/Right: After that we must assign the the edges of the area within which the objects are placed. You can't start the game without assigning all 4 edges.

Object Gap, Gap Change, Gap minimum: These attributes define how a level looks; the smaller the gap between objects the more objects are placed within the game area. You can also set how much the gap changes after each level, and the minimum gap (A gap of less than 0.2 may cause performance issues).

Object Click Radius: The click area of a hidden object. Each hidden object has a clickable script as well as a circle collider which detects the click. This value defines the relative size of the circle collider. 1 is the default.

Hidden Object Range: How many objects we need to find to win this

level. At the start of a level, a random number is chosen between the X and Y values of this variable.

Show Icons On Top: If set to true, this will replace the text (“Find X object”) with the icon of the hidden objects itself. The number of icons displayed in the top bar represents the number of objects of that type that need to be found

Show Icons On Top: The gap between the hidden object icons in the top bar, starting from the looking glass icon.

Top Bar: This object contains the timer icon and text, as well as the "find object" text.

Score Bar: This object contains the level score, and total score.

Various Screens: These are the screens that show up when the timer reaches 0, or the game ends, or you click on the options button.

Time Variables: These are attributes for the timer, how much time we have left, how much time increases after each level, and how many seconds we lose when misclikcing. Also how many seconds to wait before showing a hint, and how many points to earn for each second on the timer at the end of a level.

Victory Messages: A list of messages that randomly appear when winning a level.

Next Round Delay: How many seconds to wait before starting the next level.

Object Fall Speed/Rotation: The speed at which objects fall and rotate at the end of a level.

Screen Animations: The animation of the top bar and score bar.

UnityAds Integration (Unity 5.2 +)

Since Unity 5.2 UnityAds integration has been simplified, here's how you can have full screen video ads in your game.

This video shows a quick process of integrating UnityAds into your project. In the example we used one of my templates, but it works on all my other templates too.

https://www.youtube.com/watch?v=EQNTgfV35DU

Here is what we did in the process:

1. Sign in to your Unity account in order to allow Unity Services such as UnityAds to be activated.
2. Open Build Settings and switch the platform to one of the supported ones (iOS, Android).
3. Download Puppeteer's UnityAds package from: http://www.puppeteerinteractive.com/freebies/PUPUnityAds.unitypackage
4. Drag the downloaded package into your Unity project, and import it. This UnityAds prefab can be used to display ads every several minutes.
5. Drag the prefab into any scene where you want ads to be shown. Make sure to save changes.
6. The time check is shared between all prefabs in all scenes, so you will never show too many ads.
7. The final step is to activate UnityAds services and get your unique project ID.
8. Open the services window and choose your organization, then click create.
9. Choose UnityAds from the list and turn it On.
10. Choose age group for your project (Will affect the nature of ads shown), and save changes.
11. While working on your project keep Test Mode activated. But when you are ready to release the final project, switch Test Mode off.

12. That's it! Now when you start the game, an ad will be shown after 3 minutes. The ad will never appear during gameplay or post-game screen. Instead, it will wait until the next level load (restart, main menu, etc) and then show the ad.

Before releasing a game, make sure you uncheck **Enable Test Mode.**

For more info about integrating UnityAds read this:

http://unityads.unity3d.com/help/monetization/integration-guide-unity

Frequently Asked Questions

Does this package work on mobile?

Yes, this package has been successfully tested on both Android and iOS devices. The mobile controls are based on the default mouse controls, meaning that a click on PC equals a tap on the mobile device.

My sprites are not showing on iOS

Sprite-based textures made with the new Unity 4.3 can sometimes disappear when working on the iOS platform.

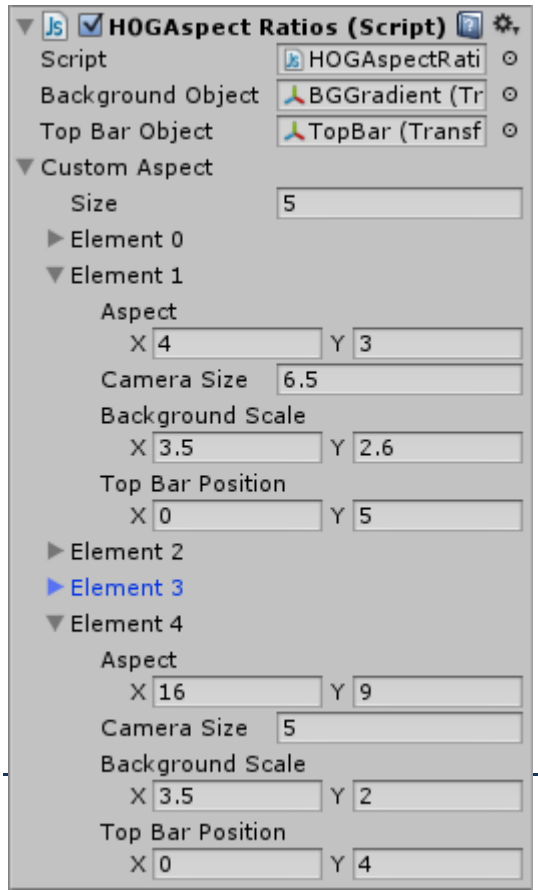
You can notice this by opening a scene playing it. When you switch from your current platform to the iOS platform the sprite textures become invisible.

To solve this we must change the texture compression format for iOS. Follow these steps:

1. Click on a texture in the project view.
2. Click on the override for iphone button on the right side.
3. Change the format to 16bit.
4. Click Apply.

How can I make my game fit multiple device resolutions/aspects?

In the 1.15 update I added a script that is attached to the camera, which automatically changes the game area to fit all the pre-set aspect ratios (5:4,4:3,3:2,10:16,16:10) as well as any custom resolution you want. This allows you to build the game once and make it fit all possible devices.



Open any of the scenes and click on the camera to see the HOGAspectRatios component which controls all of this. Here you can see a list of all basic aspect ratios and the changes in the game area for each.

In HOG there are 3 things that need to be changed to fit different aspect ratios, these are the background, the top bar, and the camera size.

Look at **Element 0**, which defines the changes when using a 4:3 aspect ratio (such as iPad). We set the **Camera Size** to 6.5 which will

zoom it out revealing more of the game area. We also changed the scale of the background object to make it fit the new game area. Finally, we change **Top Bar Position** to place it at the top of the screen.

Another example you can see in **Element 4**, which defines the game area when using a 16:9 aspect ratio. In this case we set **Camera Size** to 5, which reveal a smaller game area than the camera size we set in **Element 0**.

You can play around with the attributes and add your own elements with your own custom resolutions/aspects.

It is highly advised, whether you are a designer or a developer to look further into the code and customize it to your pleasing. See what can be improved upon or changed to make this file work better and faster. Don't hesitate to send me suggestions and feedback to puppeteerint@gmail.com

Follow me on twitter for updates and freebies!

Good luck with your modifications!