Scrambled Image Game

Game documentation and HowTo guide.

MORE GAMES
SCRAMBLED IMAGE GAME
Swap the tiles in the scrambled images and get them back in order!
GO!

This document contains:

Package Description and features

Scrambled Image Game is a full Unity template that gives you a quick and easy way to create a scrambled image game, along with a timer, levels, varying bonuses, and some nice audio-visual feedback.

How to Play?

Click on one of the tiles and then click on another tile to swap them. Reorganize the image to its correct form to win!

Try the Demo

Features:

- Game ready for release straight out of the box, just build and play!
- Works on all platforms, PC, Mac, iOS, Android, etc
- Supports multiple resolutions and aspect ratios, automatically.
- Supports Mouse, Keyboard, Gamepad, and Touch controls.
- Easily customizable with lots of options to control game difficulty.
- Great learning resource with commented scripts and documentation.
- All assets included: graphics, sounds, and code.

Current version 1.05

Update history

1.05 (23.04.2018)
- Added more options to the gamecontroller: You can set the grid size increase with each level, the delay time after an image is solved, and the option to reset the timer after a level.

1.0 (22.08.2017)
- Initial version

Credits

The main font used is Fava Black by Themnific

The sounds are courtesy of the free sound project.

Music is Waterford by Kevin MacLeod (Public Domain)

Credits go to these authors for their great sound samples: **xyzr-kx, isaac200000, harris85, speedygonzo, wagna, jimhancock, sforsman**

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 SIGAssets folder using Unity3D 5.3.0 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 font used in the game.
- **Prefabs:** Holds all the prefabs used in the game. These are distributed to various folders for easier access, Buttons, Enemies, Objects, etc. It also holds all the canvases in the game which are used to hold buttons and other UI elements.
- **Scenes:** The first scene that runs in the game is MainMenu. From this scene you can get to the Game scene.
- **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. Correct, Wrong, etc
- **Textures:** Holds all the textures used in the game which are used as sprites in Unity.

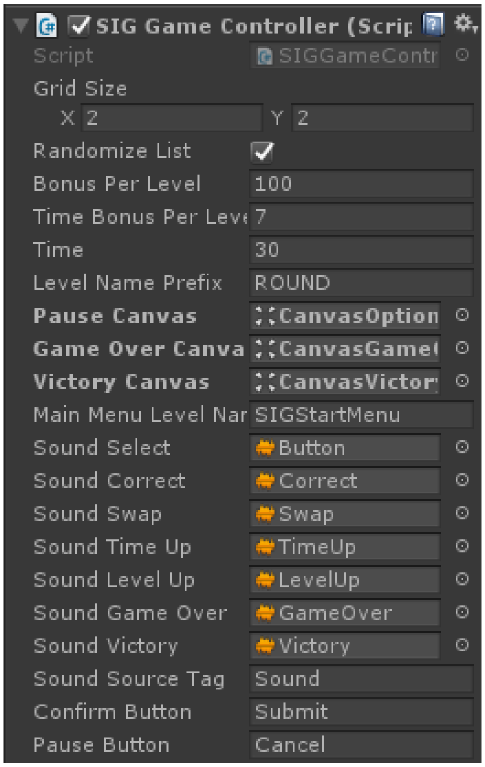
Customization Guide

Getting started

Scrambled Word Game (SIG) 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 your existing project, but SIG works best as a starter kit which you can customize any part of to your liking.

The Game Controller

The Game Controller is the main prefab that controls all the progress of the game from start to finish. It controls the UI of the game, creates scrambled words and checks the level up condition.



SIG Game Controller (Scrip

Script	SIGGameContr
Grid Size	
X 2	Y 2
Randomize List	☑
Bonus Per Level	100
Time Bonus Per Leve	7
Time	30
Level Name Prefix	ROUND
Pause Canvas	CanvasOption
Game Over Canva	CanvasGame(
Victory Canvas	CanvasVictor
Main Menu Level Nar	SIGStartMenu
Sound Select	Button
Sound Correct	Correct
Sound Swap	Swap
Sound Time Up	TimeUp
Sound Level Up	LevelUp
Sound Game Over	GameOver
Sound Victory	Victory
Sound Source Tag	Sound
Confirm Button	Submit
Pause Button	Cancel

Grid Size – The number of rows and columns that the image will be sliced to. For example 2x2 means we slice the image to 4 tiles.

Randomize List – Randomize the list of images so that we don't get the same images each time we start the game.

Bonus Per Level – How many points we get for each image in the level. This value is multiplied by the number of the level we are on. Ex: Level 1 gives 100 points, Level 2 gives 200 points.

Time Bonus Per Level – How many extra seconds we add to the timer in the level. This value is multiplied by the number of the level we are on. Ex: Level 1 gives 5 seconds at the start, Level 2 gives 10 seconds at the start.

Time – How many seconds are left before game is over.

Canvases – These UI screens are assigned from the scene for each level.

Main Menu Level Name – The level of the main menu that can be loaded after the game ends.

Confirm Button – The keyboard/gamepad button that will restart the game after game over.

Pause Button – The keyboard/gamepad button that pauses the game.

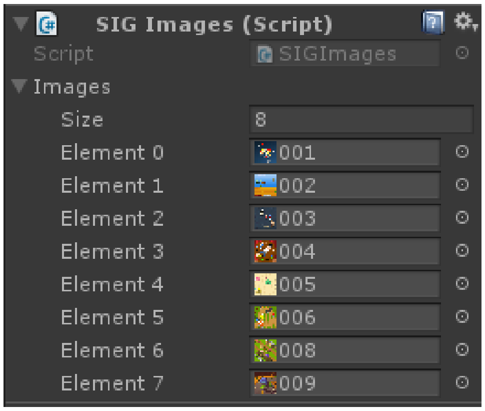
User Interface – Various canvases for the UI, assign them from the scene.

Sounds – Various sounds that play during the game.

Sound Source Tag – The audio source from which the Game Over sound plays.

The Images List

The Images list script contains all the images in the game, and must be attached to a gamecontroller in order to work. Simply drag and drop your images into this list.



SIG Images (Script)

Script	SIGImages
Images	
Size	8
Element 0	001
Element 1	002
Element 2	003
Element 3	004
Element 4	005
Element 5	006
Element 6	008
Element 7	009

UnityAds Integration

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://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

Does this package work on mobile?

Yes, this package has been successfully tested on both Android and iOS devices. The scripts for each lock type include controls for mobile that are detected automatically based on the platform it's built on.

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.

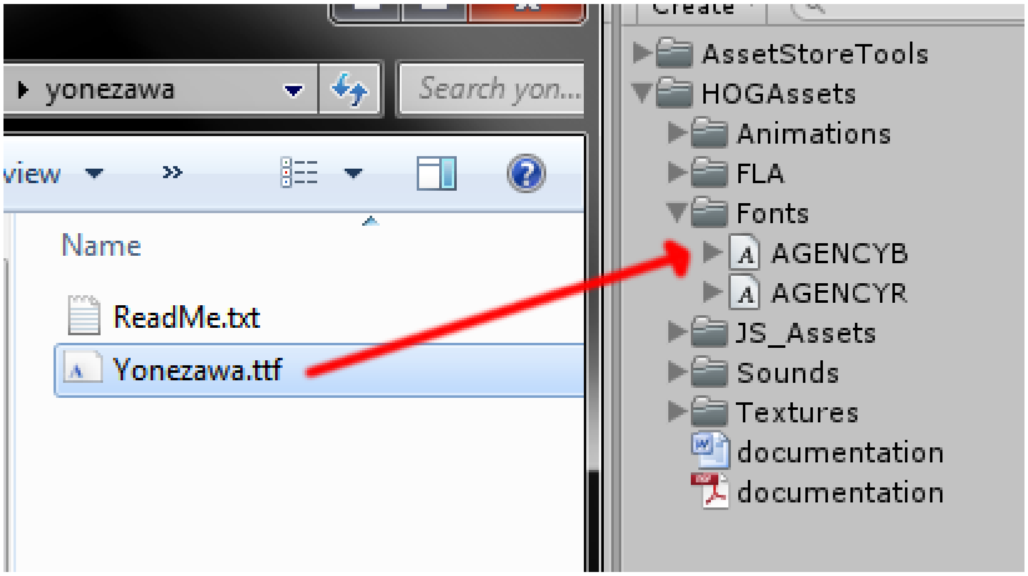
To solve this we must change the texture compression format for iOS. Follow these steps:

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.

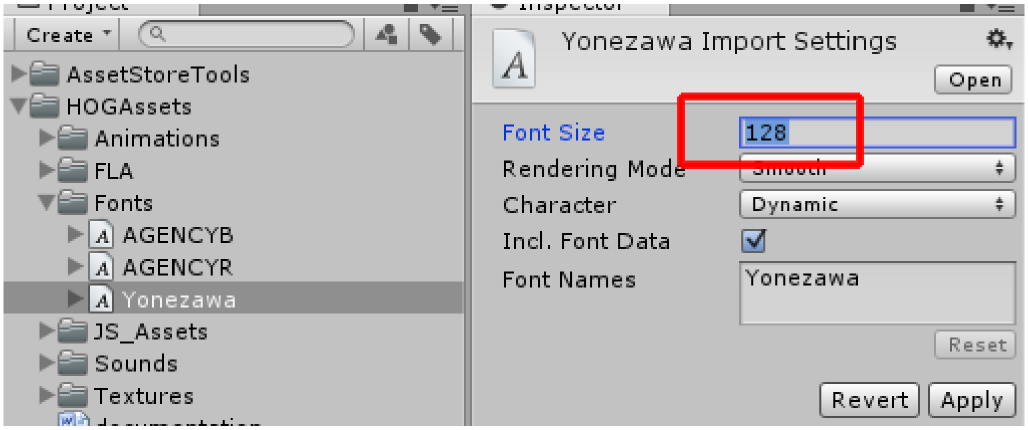
How to change font in the game?

To change a font in the game do the following:

Find a font you like and drag the .ttf file over to the Fonts folder in your game.



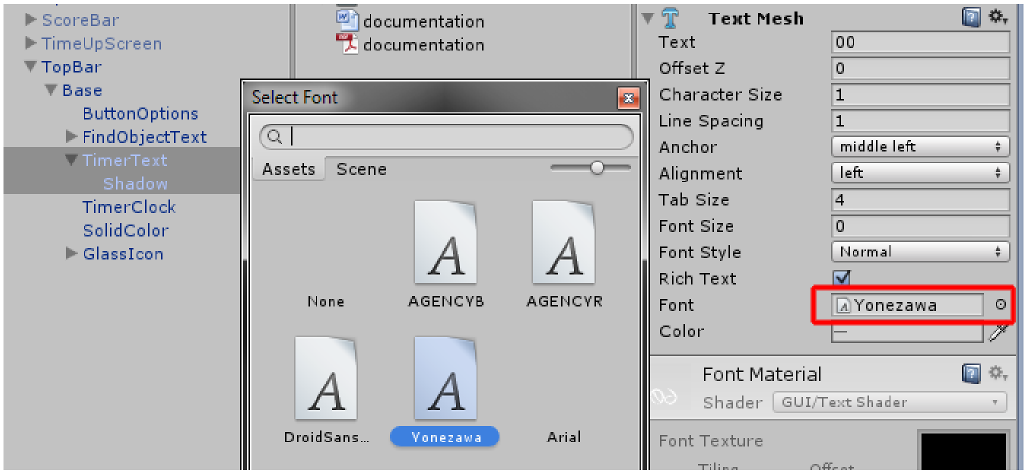
Click on the font you added and edit its attributes. I personally set all my fonts to a high number (and then scale the text object down) so that they look crisper in-game.



Yonezawa Import Settings

- Font Size: 128
- Rendering Mode: Smooth
- Character: Dynamic
- Incl. Font Data: ☑
- Font Names: Yonezawa

Select any text object in the game and change its font to the new font you have. Sometimes the text might disappear, but it's normal. Just write something in the text box above and it will refresh. Also, make sure you change the text for the shadow; you can select both the main text and its shadow and edit them together.



Text Mesh

Text	00
Offset Z	0
Character Size	1
Line Spacing	1
Anchor	middle left
Alignment	left
Tab Size	4
Font Size	0
Font Style	Normal
Rich Text	☑
Font	Yonezawa
Color	

Font Material

Shader: GUI/Text Shader

More games by Puppeteer

Click here to see the full catalogue of Asset Store files!

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!