# Speeder Run Game

## Game documentation and HowTo guide.


## This document contains:

## Package Description and features

Speeder Run is an action packed game full of challenge and fun. The game is ready to release straight out of the box, and it can also be easily customized to make it even more engaging to your players. The game supports PC/Mac, iOS, Android, and WinPhone. It can be played with the keyboard, gamepad, or touch controls!

**How to Play?**

Move with the arrow keys left and right. Collect orbs and avoid obstacles!

## Try the webplayer

**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.
- UnityAds support with integration guide.

**Current version 1.20**

## Update history

**1.20 (28.04.2018)** – Rebuilt project so it can be easily mixed with other Puppeteer templates.
**1.16 (30.08.2017)** - Update to Unity 5.5, 5.6, and 2017.
**1.15 (20.07.2016)**
- Support for Unity 5.3 and higher versions.
- Better support for UnityAds 5.2 and above.
- You can assign multiple items in a section, which will be randomly spawned.
- A new Shield item makes you invulnerable for a few seconds.
- A new Slow Motion item slows down time for a few seconds.
- A stage selector with unlockable stages based on the number of stars you have.
- A star rating system for each stage based on the score you received.

**1.0 (15.01.2016**)
- Initial version

# Credits

The font used is Aero by Nirmal Biswas

The sounds are courtesy of the free sound project.

Music is a clip from Drum n Bass K Megatrack by Frank Nora ( Public Domain )

Credits go to these authors for their great sound samples: **torn, qubodup, cydon, squareal**

**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 SRGAssets folder using Unity3D 4.6.9 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. Jump,Item, etc
- **Textures:** Holds all the textures used in the game which are used as sprites in Unity.

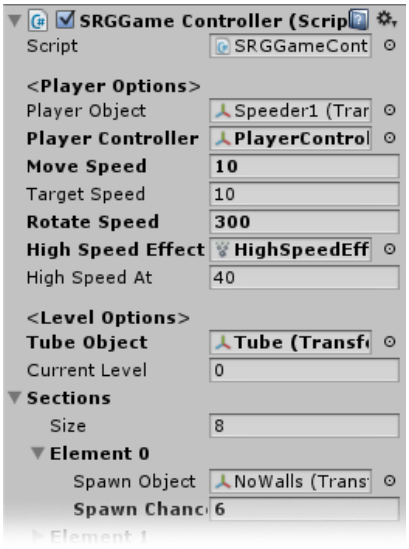
## Customization Guide

# Getting started

Speeder Run Game Template (SRG) 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 SRG 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 obstacles and items and checks the level up condition.




**Player Object –** This is the prefab of the player speeder. It should be assigned from the project pane, and it is created at the start of the game in the start position point.

**Player Controller –** This is the object that holds the player, it should be assigned from the scene. It moves forward and rotates based on player input.

**Move Speed –** The current speed of the player controller.

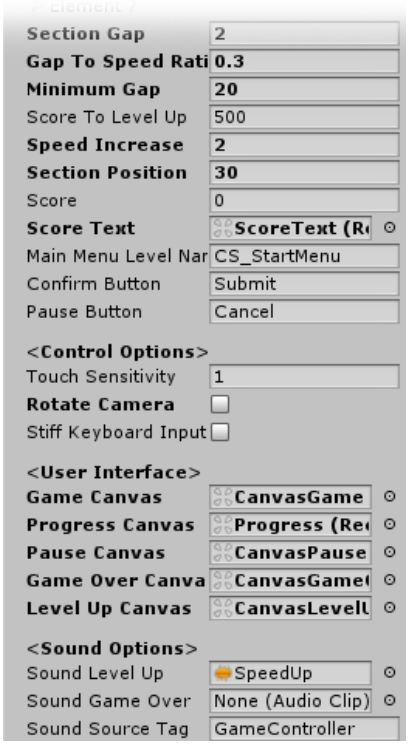
**Target Speed –** The speed that the player controller will get to over time.

**Rotate Speed –** The rotation speed of the player controller. Changing this will make it easier or harder to avoid obstacles.

**High Speed Effect –** The effect that appears when the player reaches a very high speed.

**High Speed At –** The speed at which the high speed effect appears.

**Tube Object –** The tube through which the player moves. The tube material is animated to give an illusion of movement.




**Sections –** A list of all the sections spawned in the game, and their spawn chance.

- **Spawn Object –** The prefab of the section that can be spawned.
- **Spawn Chance –** The chance of appearance for this section. This is relevant to other objects in the list.

**Section Gap –** The distance between each two wall sections.

**Gap To Speed Ratio –** The ratio between the current speed of the player and the distance between wall sections. This is used to control the game difficulty by making the gap smaller and smaller as we progress, but still taking into account the fact that we are moving faster and need more time to react.

**Minimum Gap –** The minimum allowed distance between two wall sections. This is to make sure there are no walls too close to each other that make it impossible to pass.

**Score To Level Up –** How many points we need in order to level up. When leveling up the speed and gap increase.

**Speed Increase –** How much the speed increases when we level up.

**Section Position –** The position of the first section in the tube.

**Touch Sensitivity –** How sensitive the rotation is when using touch controls.

**Rotate Camera –** Makes the camera rotate to follow the player movement.

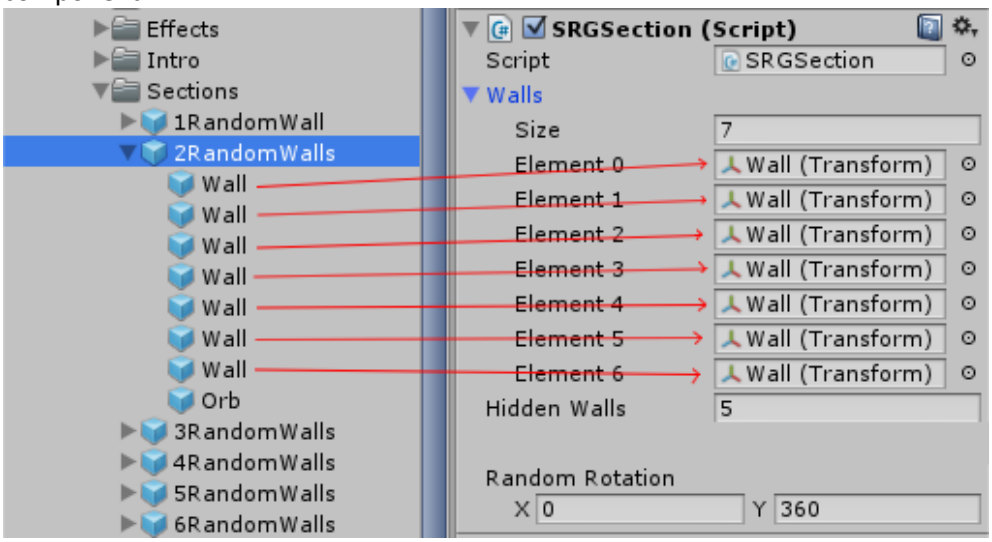
**Stiff Keyboard Input –** Use a stiff input for rotating. This makes the player move immediately left/right instead of easing into the motion.

**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.

## Editing Wall Sections

Wall sections are spawned in front of the player as he moves forward. Each section contains 7 wall pieces and an orb item. Here is what you can change in the section component:




**Walls –** The walls are the obstacles that kill the player when he touches them. Each section has 7 walls which can be hidden randomly. The walls are assigned to the wall list from the section itself.

**Hidden Walls –** The number of walls that will be hidden in this sections. The walls are chosen randomly each time a section is created.

**Random Rotation –** A random rotation for the section.

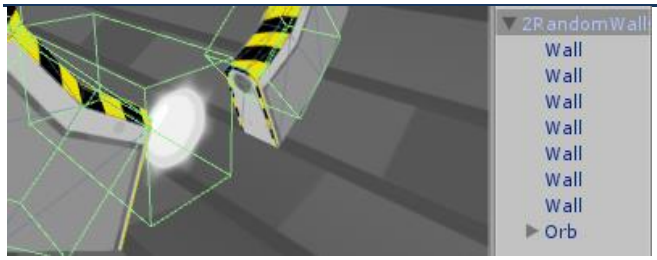
## Editing Walls and Orbs

Walls that kill you and Orbs that you can collect are both made of the same type of component, the Block. Blocks are components that trigger a function when they are touched by something.

When walls are touched they trigger a Die() function on the player that touched them.

Orbs trigger a ChangeScore() function with a value of 100, which means that when they are touched by the player they increase the score by 100.

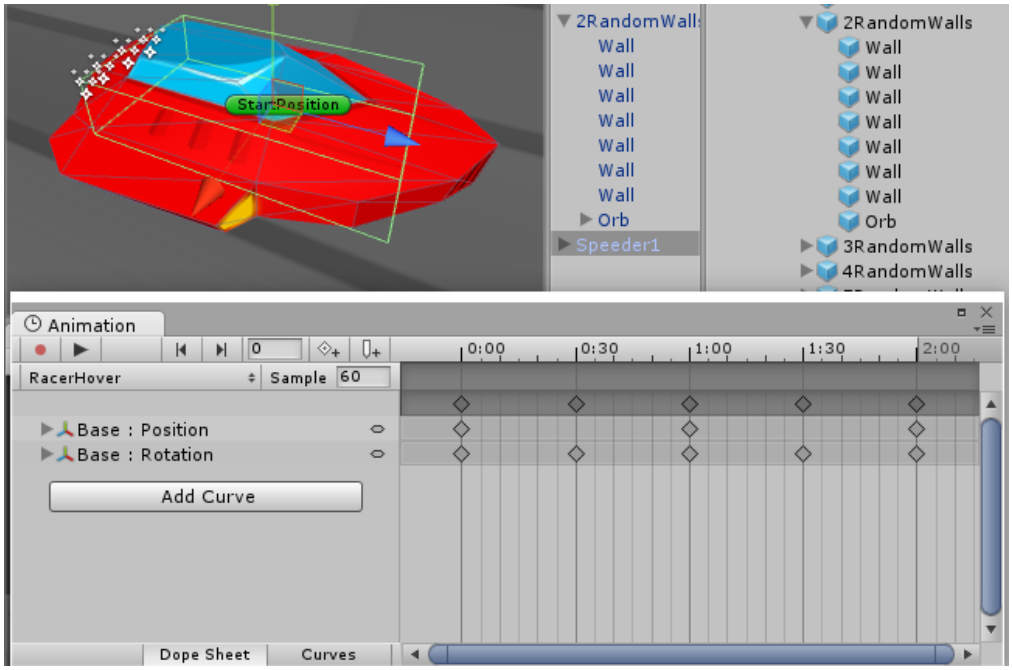
You can change the touch area of a wall or an orb by changing the collider size.




## Editing the Player

Walls that kill you and Orbs that you can collect are both made of the same type of component, the Block. Blocks are components that trigger a function when they are touched by something.

When walls are touched




The player can move and rotate within the tube, and is controlled using the PlayerController assigned in the GameController component. Here is what you can change in the player component:

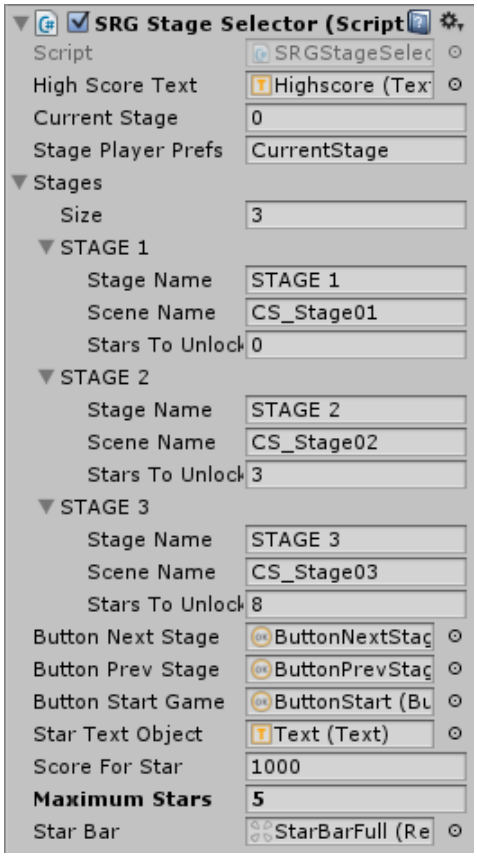
**Death Effect -** The effect that is created at the location of this object when it is destroyed.

**Animator –** This is the animated model that contains an Animator component. The Animator has all the animations of the player (Spawn,Hover,etc).

You can edit the animations of the player by selecting the Animator object and pressing Ctrl+6 to access the animation tab.

## The Stage Selector

The stage selector allows you to have several stages in the game. Each level must be unlocked by earning a certain number of stars. Each star can be earned when getting a certain score in a stage. You can customize all of these settings.




**High Score Text –** The text object that shows the high score we got in the current stage.

**Current Stage –** The current stage we are in. 0 is the first stage, 1 the second.

**Stage Player Prefs –** This is the name of the record that holds the current stage we are in. This is saved locally.

**Stages –** A list of stages you can unlock and play.

**Stage Name –** The name that appears when we select a stage. This is not the name of the scene that is loaded.

**Scene Name –** The scene that is loaded when we start this stage.

**Stars To Unlock –** The number of stars needed in order to unlock this level.

**Button Next/Prev/Start –** These buttons assigned from the scene let us switch stages and start the stage.

**Start Text Object –** The text inside the start button which also shows how many stars we need to unlock a level.

**Score For Star –** The number of points needed in a level in order to get a star. This is calculated based on the high score we got in the level, so for example if we have 100 highscore we get 1 star, and for 200 highscore we get 2 stars.

**Maximum Stars –** The maximum stars we can get in a stage.

## 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: 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.

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

## Frequently Asked Questions

### 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.

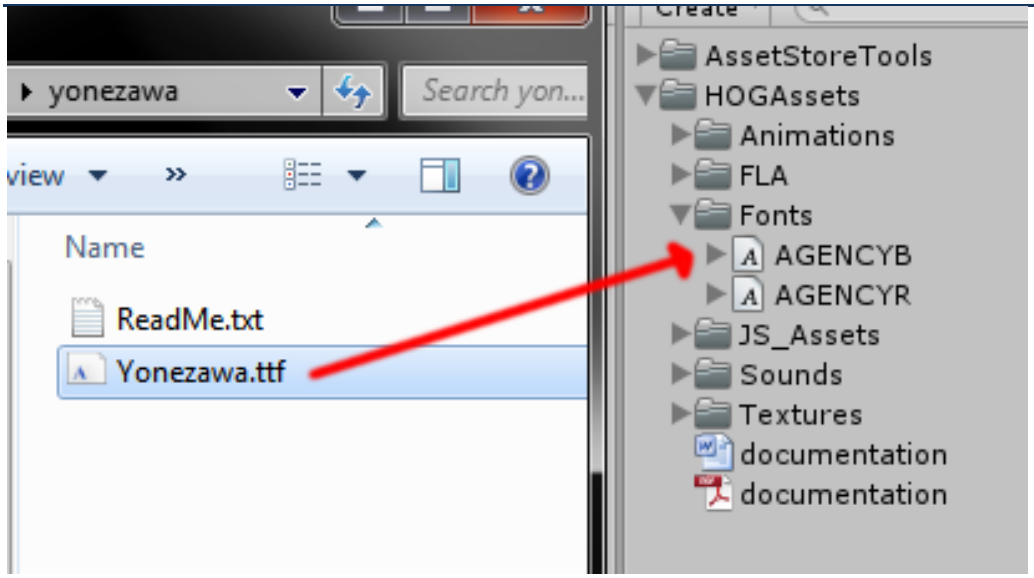
To solve this we must change the texture compression format for iOS. Follow these steps:

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

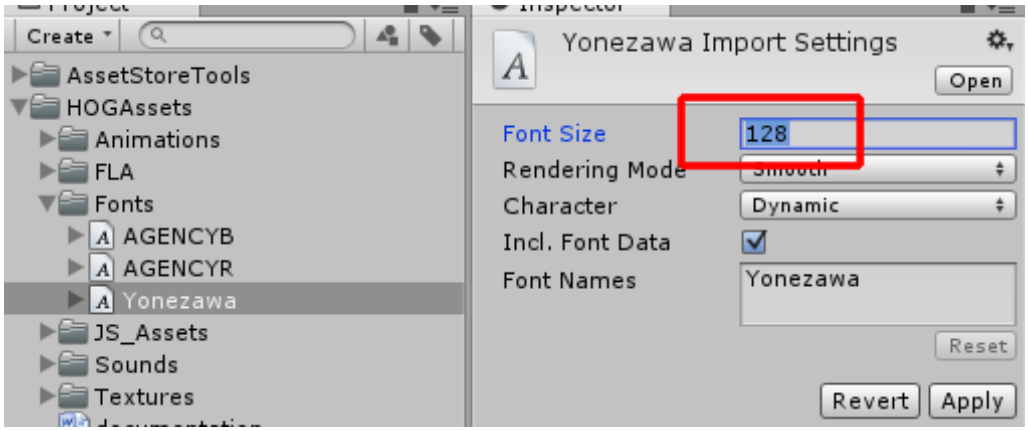
### How to change font in the game?

To change a font in the game do the following:

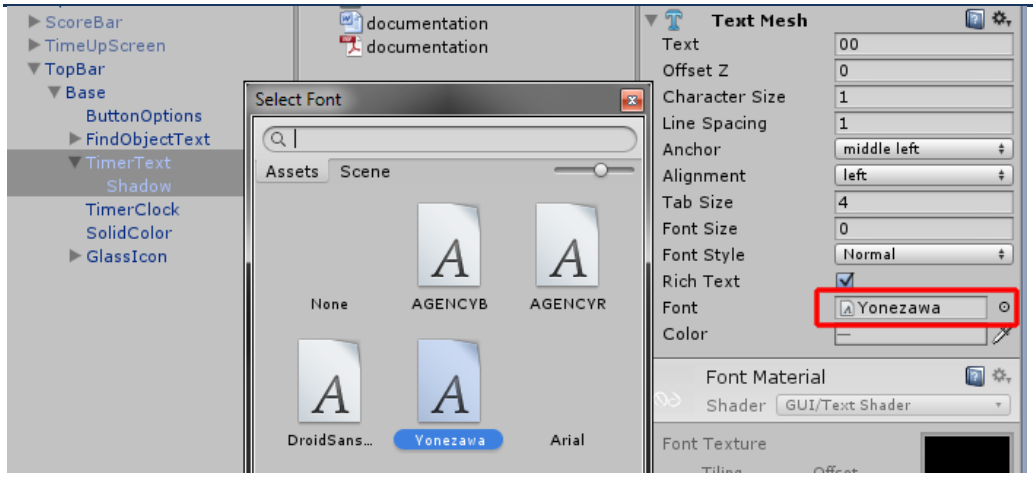
Find a font you like and drag the .ttf file over to the Fonts folder in your game.




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.




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.




## 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!