Fruit Slicing Game

“A Fruit Ninja Style Complete Project Template”

I would like to start by saying thank you for downloading this Complete Project Template. Please rate it if you get the time… its mean a lot to me. I would also like to give you a little background on HBB and myself... I still consider myself very new to Unity, and C#. I have no classical programming training, though I do get some scripting/batching sorts of jobs competed at work on a Technical helpdesk without issue. I am quite fond of Unity. I had always enjoyed working on little game mods when I was younger, and the idea of game making. Unity Technologies has kind of made that dream a reality for me. You may read some places that when it comes to indie game development… “Fake it, till you make it”. That is an idea I have had to subscribe to whole-heartedly as a solo developer, with no experience, and no classical training. I can do all of the jobs necessary to bring an idea to life, but because I have to do all of the jobs the end product suffers. If I would have been able to spend all of my time devoting myself to a “single discipline” like modeling, texturing, animation, game-design, or scripting I would be much better off. Since It can be hard to find individuals as devoted to / or passionate about a specific project or idea I continue to slowly continue learning/tinkering.

My main goals for releasing these kit are twofold. 1.) I wanted a pipeline to deposit some of my partially completed projects, while still trying to find some people to work with. It helps me to continue to learn, and practice. 2.) I am always learning/reading about Unity… I remember when I was starting out about a year ago, that it was INCREDIBLY useful for me to take apart other people’s work. I would scour the Asset Store and I had just about every free complete project I could find. I think that in addition to the UT projects that there could definitely be more FREE Complete Project Templates. So I plan on releasing

several of the projects I have laying around on my computer as free kits. Beyond that I will probably sell some too, but they will be super cheap.
If you have gotten this far into my ramble then I thank you. Most of the scripts in the project have pretty thorough/lengthy comments. I wish they were a little more concise, but… c'est la vie. <04/11/2016> … still relevant!!!

Update - 6/20/2016 – FNC_AboutAndReadMe

I would like to thank all of you who downloaded the Color Switch Clone Complete Project Template. I was very happy to hear (at least from the people that commented, or reviewed) how it was received. It made me feel really good. A lot of work does go into projects like this, and while it is a labor of love… it is still a great deal of work. For everyone that did review it, or reach out to me "Thank You"! This Fruit Ninja Style Game took a little longer to do… The prototype was completed before the CSC prototype was, BUT it required a lot more refactoring and optimization than CSC did. The art assets are heavily decimated from their original forms, but I did my best to make them as nice as I could, in a limited amount of time. There were also performance considerations, when it came to assets. My phone is pretty old, and it will play this game fine, not great, but fine. WebGL and Standalone builds obviously run well enough (depends on pc)… but for Galaxy S3 grade phones it is a little heavy. The textures could be smashed down further, but not without significant loss. They are already pretty compressed. All tests I have done on more recent devices have been promising. Just as stated last time, these Free Kits are kind of released "as-is", I have been talking to someone about doing the FAQ on the website (the primary support form for the free asset), but if you need help you can e-mail me… as long as spam doesn't grab it, I WILL get to it as soon as I can. Work/Paid support inquiries have to come first. I believe I got back to everyone last release within a couple days… I do my best as one person.

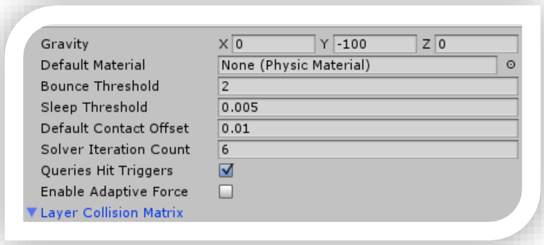
Best Regards

-Brian🚀🚹

1.) Setup

Setting Up Gravity, Tags, Scene Build Order, Sorting Layers, Time Manager, and Layers

- **Set Gravity to -100 m/s^2 on the Y Axis.** *See Figure 1.*

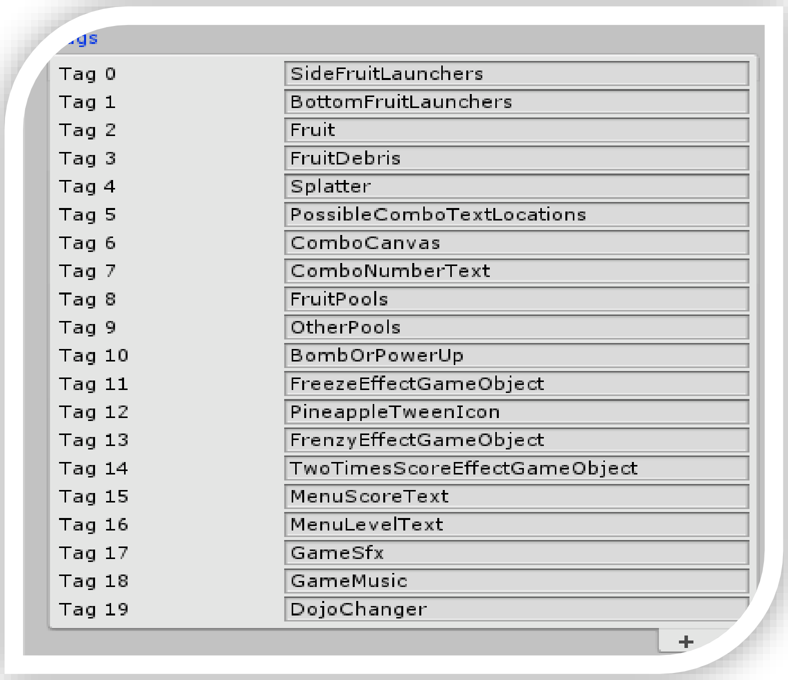


Gravity	X 0 Y -100 Z 0
Default Material	None (Physic Material)
Bounce Threshold	2
Sleep Threshold	0.005
Default Contact Offset	0.01
Solver Iteration Count	6
Queries Hit Triggers	✓
Enable Adaptive Force	☐

▼Layer Collision Matrix

Figure 1- No other changes are necessary in the Physics Manager.

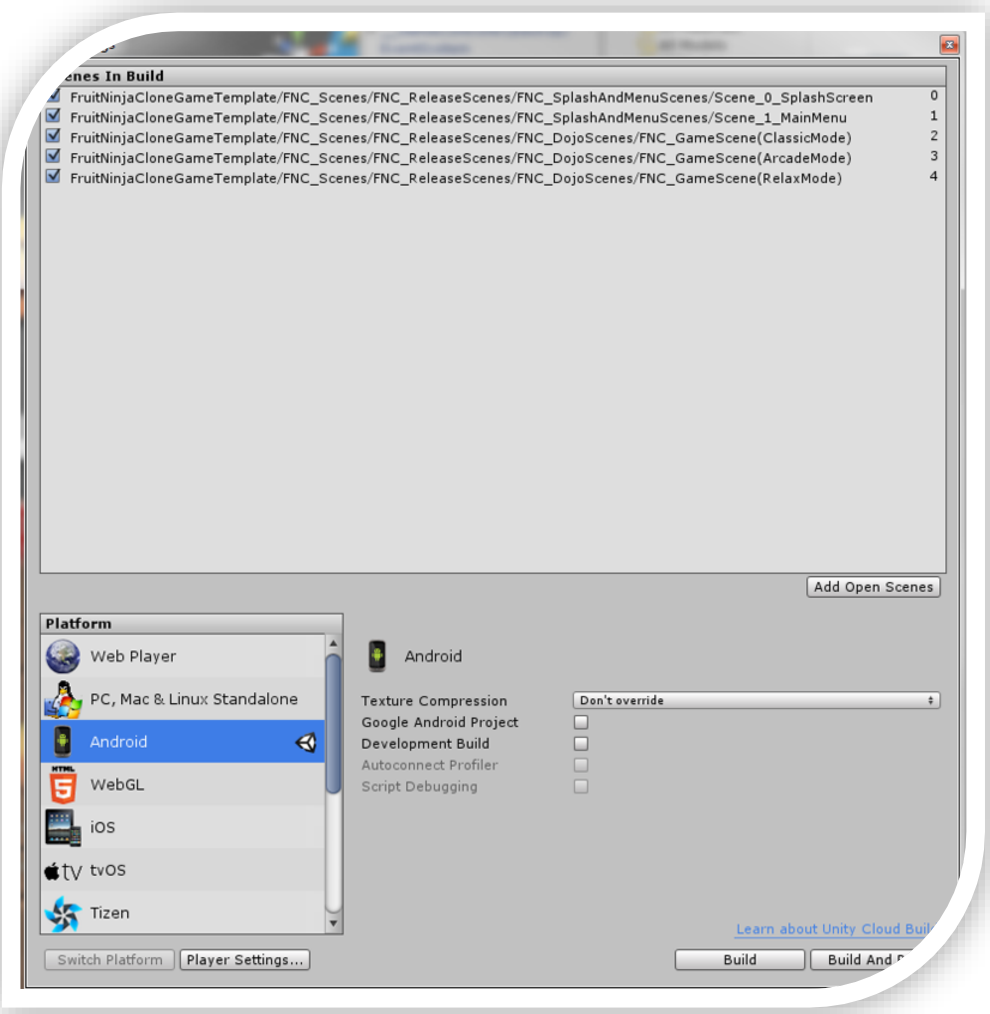
- **There are Several Tags used.** *See Figure 2.*



Tag 0	SideFruitLaunchers
Tag 1	BottomFruitLaunchers
Tag 2	Fruit
Tag 3	FruitDebris
Tag 4	Splatter
Tag 5	PossibleComboTextLocations
Tag 6	ComboCanvas
Tag 7	ComboNumberText
Tag 8	FruitPools
Tag 9	OtherPools
Tag 10	BombOrPowerUp
Tag 11	FreezeEffectGameObject
Tag 12	PineappleTweenIcon
Tag 13	FrenzyEffectGameObject
Tag 14	TwoTimesScoreEffectGameObject
Tag 15	MenuScoreText
Tag 16	MenuLevelText
Tag 17	GameSfx
Tag 18	GameMusic
Tag 19	DojoChanger

Figure 2 - Several Tags

- **Make sure these Scenes are added to the Build Settings. In this order.** *See Figure 3a.*



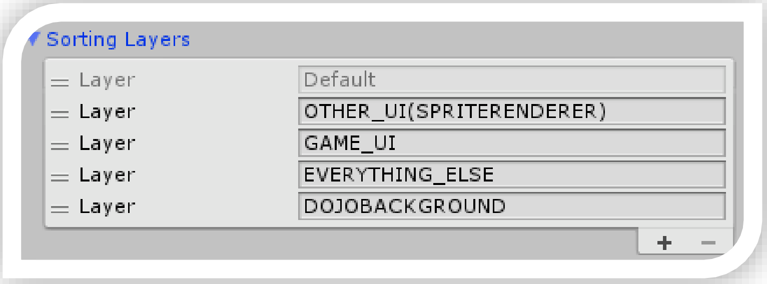
enes In Build

```
FruitNinjaCloneGameTemplate/FNC_Scenes/FNC_ReleaseScenes/FNC_SplashAndMenuScenes/Scene_0_SplashScreen      0
FruitNinjaCloneGameTemplate/FNC_Scenes/FNC_ReleaseScenes/FNC_SplashAndMenuScenes/Scene_1_MainMenu          1
FruitNinjaCloneGameTemplate/FNC_Scenes/FNC_ReleaseScenes/FNC_DojoScenes/FNC_GameScene(ClassicMode)         2
FruitNinjaCloneGameTemplate/FNC_Scenes/FNC_ReleaseScenes/FNC_DojoScenes/FNC_GameScene(ArcadeMode)          3
FruitNinjaCloneGameTemplate/FNC_Scenes/FNC_ReleaseScenes/FNC_DojoScenes/FNC_GameScene(RelaxMode)           4
```

Platform: Android

Texture Compression: Don't override
Google Android Project: ☐
Development Build: ☐
Autoconnect Profiler: ☐
Script Debugging: ☐

Figure 3a - Splash, Menu, Classic, Arcade, and Relax



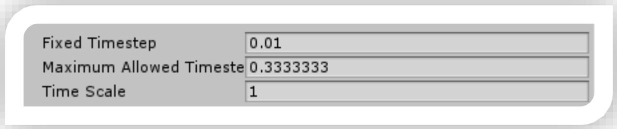
Sorting Layers

Layer	Default
Layer	OTHER_UI(SPRITERENDERER)
Layer	GAME_UI
Layer	EVERYTHING_ELSE
Layer	DOJOBACKGROUND

Figure 3b - Sorting Layers used for different UI elements

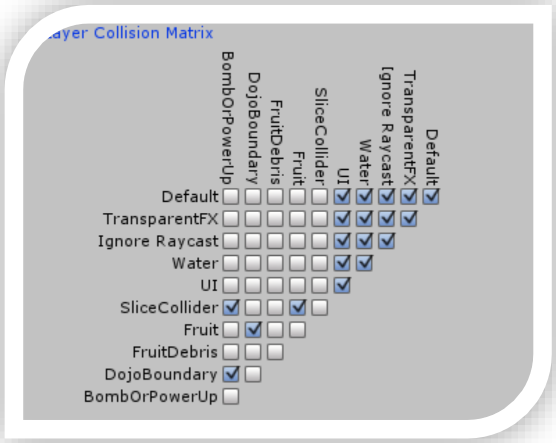
- **SortingLayers for Various UI Elements** *See Figure 3b.*

- **New Layers added for FNC.** *See Figure 4a/b/c.*



Fixed Timestep	0.01
Maximum Allowed Timeste	0.3333333
Time Scale	1

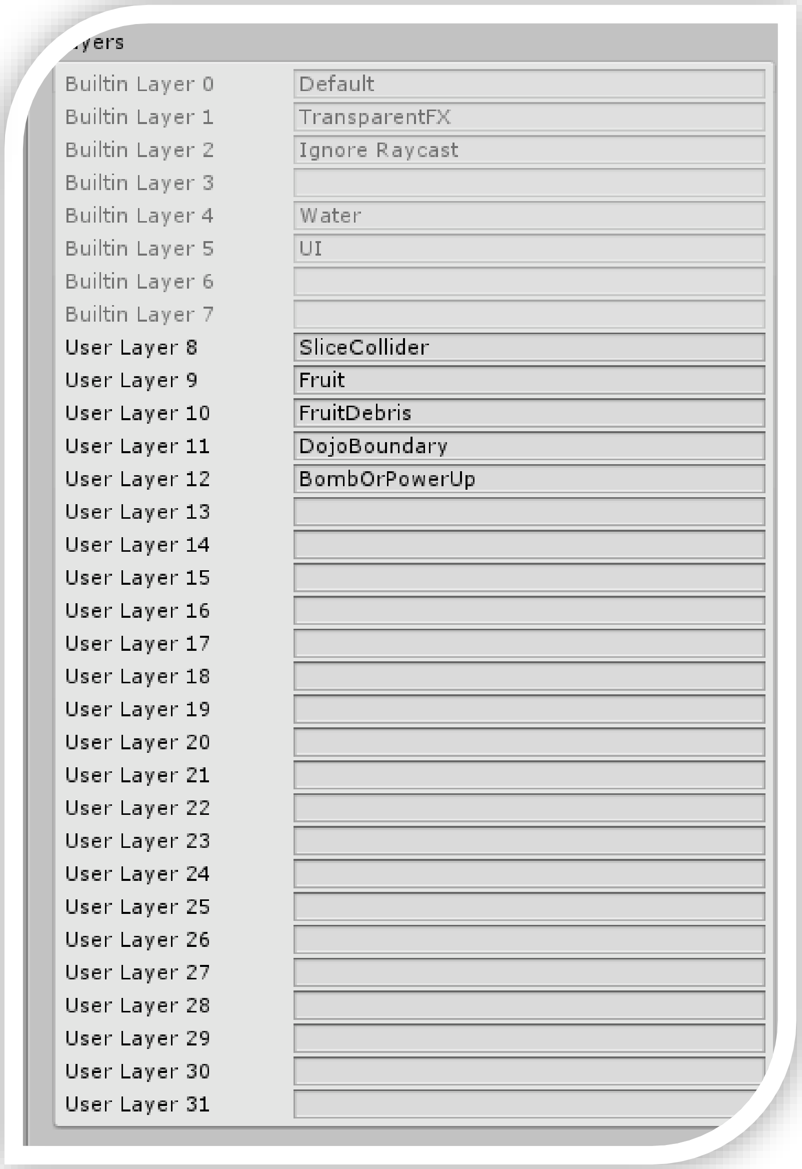
Figure 4a - Time Manager showing an adjusted Fixed Time Step



ayer Collision Matrix

	BombOrPowerUp	DojoBoundary	FruitDebris	Fruit	SliceCollider	UI	Water	Ignore Raycast	TransparentFX	Default
Default	☐	☐	☐	☐	☐	✓	✓	✓	✓	✓
TransparentFX	☐	☐	☐	☐	☐	✓	✓	✓	✓	
Ignore Raycast	☐	☐	☐	☐	☐	✓	✓	✓		
Water	☐	☐	☐	☐	☐	✓	✓			
UI	☐	☐	☐	☐	☐	✓				
SliceCollider	✓	☐	☐	✓	☐					
Fruit	☐	✓	☐	☐						
FruitDebris	☐	☐	☐							
DojoBoundary	✓	☐								
BombOrPowerUp	☐									

Figure 4b - Layer Matrix



yers

Builtin Layer 0	Default
Builtin Layer 1	TransparentFX
Builtin Layer 2	Ignore Raycast
Builtin Layer 3	
Builtin Layer 4	Water
Builtin Layer 5	UI
Builtin Layer 6	
Builtin Layer 7	
User Layer 8	SliceCollider
User Layer 9	Fruit
User Layer 10	FruitDebris
User Layer 11	DojoBoundary
User Layer 12	BombOrPowerUp
User Layer 13	
User Layer 14	
User Layer 15	
User Layer 16	
User Layer 17	
User Layer 18	
User Layer 19	
User Layer 20	
User Layer 21	
User Layer 22	
User Layer 23	
User Layer 24	
User Layer 25	
User Layer 26	
User Layer 27	
User Layer 28	
User Layer 29	
User Layer 30	
User Layer 31	

Figure 4c - Listed Layers

2.) Things to Know

The meat, potatoes, and nuances…

This section we will cover "Things to Know". I will give a brief rundown of some of the main scripts. This section will be long… my apologies ;).

The General Stuff…

If you are attempting to play in the editor make sure the game is started at Scene 0 (the "Splash Scene"). When the project is built that point becomes kind of a "Non-Issue". As long as the "Build Settings" have the scenes in the proper order everything should work properly. It is worth mentioning, that the 3 "game scenes" are really the same scene. Prior to making the scene prefabs separate and in separate folders, they were the same "linked" prefabs, but two of the scenes had unapplied changes. Saving the scene was sufficient. Example…

"I *Initially I made only the Arcade Mode Scene, and then I did "save as", and named it the Classic Mode Scene. Then I did the same for the "Relax Mode" Scene. Now, since the "Arcade Mode" scene had the game controller set to "Arcade"… All that we had to do was go into the "Classic-Mode" scene and change the GameController drop down to "Classic", and save the scene. Then do the same for the "Relax-Mode" scene. With how the scenes were originally created, changes made to a prefab in any game scene were reflected on all of the game scenes. I decided to separate the Scenes & Scene Prefabs so that no one accidently applied changes to other scenes."*

While the game "should" be started in scene 0, there should not be any errors when started elsewhere. The cameras in the game have a FaderCaller.cs attached to them, so testing from a specific scene should be error free. The "FaderCaller" class instantiates the fader canvas (if one does not exist). If you run into any issues or exceptions try starting the game from scene 0 and see if the problem(s) persist.

You may notice that the gravity is set to an extremely high value. Initially I was having some performance issues. The "Cutting" part of the project has been refactored 3 times. There was a version with actual mesh deformation(turned out to heavy for my mobile), a physics driven version(still… ended up being about 45 rigidbodies on screen at one time), and then I finally decided to make a very simple "cut" method, and use the animation system to fulfill the "falling gibs" portion. Early in the project I noticed that cutting those tiny fruit with a raycast or collider didn't work all that well. Even with some of the interpolation settings turned on. The distance in between frames did not always catch a fruit. My answer was to dial up the scale quite considerably. I made sure that other things were proportionate to those changes… I.e. camera size, and gravity. Now that the system is in a pretty optimal state (at least from a performance standpoint), the scale could probably be returned.

The current implementation leaves a lot to be desired, but I am hoping to revisit the Mesh Deformation version to see about increasing performance. That will be later down the road though. As it stands, this project is a very basic, and simple way to achieve the effect of "cutting fruit". Since I am trying to make these simple, and to be used as a learning resource, which is good enough for me. As long as the setup bullet points are hit then you are ready to go.

NOTE

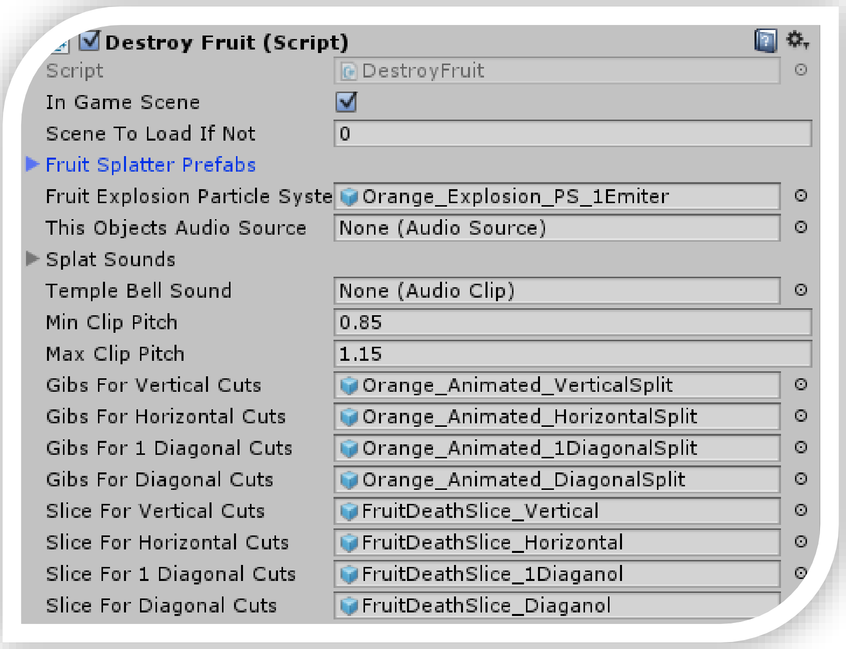
There are DoxyGen Doc pages with a search and navigation pane in the form of a webpage, and of course the scripts themselves are heavily commented. There are probably too many comments… so strip the class/method comments down if you need! My only intention was to be thorough for beginners. Between this Document, Doc Pages, and Comments in the Scripts themselves you shouldn't have too many questions. If you refer to the various scenes and the written material everything should be pretty clear. As with CSC the Free Complete Project Templates are basically released "As-Is"… I should have a FAQ on the website soon, and you can e-mail me, but it may take me a while to get back to you. I stay pretty busy with work and stuff like this. If you are patient, and my spam folder did not grab your email, I will get back to you.

Paid assets will have a higher priority when it comes to e-mailed support inquiries.

(DoxyGen Doc pages grabs "///" comments, but not the line by line "//" comments)… so that means that there is more detail in the actually script files.

<u>The Difference between Menu Fruit and Game Fruit…</u>

There are basically a few different categories when it comes to the Script/Classes in the Fruit Ninja Style Game. We load the game via the splash screen, and then it loads the Main Menu. The "Menu Fruit" that load the different scenes are almost identical to the "Game Fruit", they just have some values modified in the inspector. *See Figure 5*. The "In Game Scene" Boolean value at the top is checked for fruit in the game scenes. If you are at the main menu, then the value is unchecked. If it is unchecked the variable below it ("Scene To Load If Not") needs to be given a value between 0 and the max number of scenes there are. This integer is used like… *SceneManager.LoadLevel (sceneToLoadIfNot);* So, we pass that value to the LoadLevel function. There are some other small differences at the end of the "CutFruit" Method that "destroys" fruit. For instance we don't Increment "Fruit Missed" if a fruit is chopped in the Menu Scene. We make sure all colliders are immediately deactivated in the Menu Scene so we don't have double level loads. These are the types of the differences, and something to keep in mind.



Destroy Fruit (Script)

Script	DestroyFruit
In Game Scene	✓
Scene To Load If Not	0
▶ Fruit Splatter Prefabs	
Fruit Explosion Particle Syste	Orange_Explosion_PS_1Emiter
This Objects Audio Source	None (Audio Source)
▶ Splat Sounds	
Temple Bell Sound	None (Audio Clip)
Min Clip Pitch	0.85
Max Clip Pitch	1.15
Gibs For Vertical Cuts	Orange_Animated_VerticalSplit
Gibs For Horizontal Cuts	Orange_Animated_HorizontalSplit
Gibs For 1 Diagonal Cuts	Orange_Animated_1DiagonalSplit
Gibs For Diagonal Cuts	Orange_Animated_DiagonalSplit
Slice For Vertical Cuts	FruitDeathSlice_Vertical
Slice For Horizontal Cuts	FruitDeathSlice_Horizontal
Slice For 1 Diagonal Cuts	FruitDeathSlice_1Diaganol
Slice For Diagonal Cuts	FruitDeathSlice_Diaganol

Figure 5 - Destroy Fruit Class. Make sure every destructible fruit has one.

<u>Re-Using the ScreenFaderSingleton from the Color Switch Clone…</u>

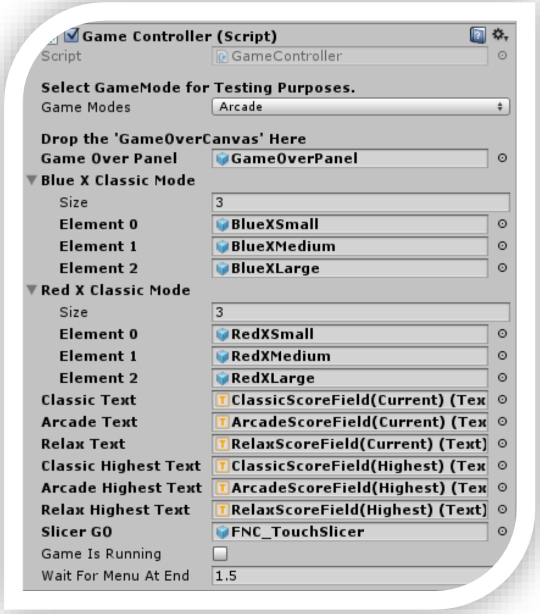
This game uses the ScreenFaderSingleton just like ColorSwitchClone. There have been a couple of additions to it. The “BackGround” Music/Settings object is a child of the Fader Canvas (so that it persists through the scenes with the Fader Canvas). Unlike in CSC, where the “Background Settings/Music” Gameobject was a separate persistent object. Since CSC had just 3 scenes the setup was a little simpler. We needed a method that would accept a parameter and load the parameter value, so that we could request specific scenes to load. Not just move in a linear scene loading fashion. There are a few Classes that are returning from the CSC complete project template. FaderCaller, FaderReferenceSetup, Singleton, ScreenFaderSingleton, RotateObject (little more verbose), ChromaticAberration, SimpleCameraShake, GameVariables, AnimationCurveMover, ObjectPoolScript, ObjectFollow, and Destroy/DisableGameObject have all been reused. They are very close (if not identical) to their versions in CSC. We also added another Static Class named “Tags” because there are definitely more tags in FNC. So, if you got somewhat familiar with those Classes from CSC they have remained mostly the same, and that should be helpful. I get tired a re-writing the same scripts over and over, so I have a starter template that saves me some time, and those scripts are part of it. They will be in a majority of the kits I publish.

<u>The GameController gameObject…</u>

The game scenes contain a gameObject named “__GameController (BackUp)”. The GameController consists of a few separate Classes…

- ***GameController.cs***
- ***LaunchController.cs***
- ***CountdownTimer.cs***
- ***DojoBoundaryController.cs***

➢ GameController.cs



Game Controller (Script)

Script	GameController
Select GameMode for Testing Purposes.	
Game Modes	Arcade
Drop the 'GameOverCanvas' Here	
Game Over Panel	GameOverPanel
▼ Blue X Classic Mode	
Size	3
Element 0	BlueXSmall
Element 1	BlueXMedium
Element 2	BlueXLarge
▼ Red X Classic Mode	
Size	3
Element 0	RedXSmall
Element 1	RedXMedium
Element 2	RedXLarge
Classic Text	ClassicScoreField(Current) (Tex
Arcade Text	ArcadeScoreField(Current) (Tex
Relax Text	RelaxScoreField(Current) (Text)
Classic Highest Text	ClassicScoreField(Highest) (Tex
Arcade Highest Text	ArcadeScoreField(Highest) (Tex
Relax Highest Text	RelaxScoreField(Highest) (Text)
Slicer GO	FNC_TouchSlicer
Game Is Running	☐
Wait For Menu At End	1.5

Figure 6 - The Game Controller in Arcade Mode

The Game Controller controls the active game mode. *See Figure 6.* It calls the Game-Over Panel to activate when the round ends. Depending on the game mode it does a few different things. In Classic Mode it Activates 3 blue X's in the top right hand corner. If a fruit is "missed" then it removes the blue X, and activates a red X at the position. The X's have an animation that plays when they are activated, this causes them to wiggle.

If we are in Arcade, or Relax Mode the Game Controller just calls for the Game-Over Panel when the timer has run out. It also keeps the current / best scores updated on the top left hand side of the screen. Note: best scores display will be updated the following round, for that game mode.

At the end of a round the game controller will disable the FNC Touch Slicer (the "sword/swipe object that cuts fruit, and is followed by a trail renderer). The last two visible variables on the Game Controller inspector are "Game Is Running" and "Wait For Menu At End". The Boolean stays true if

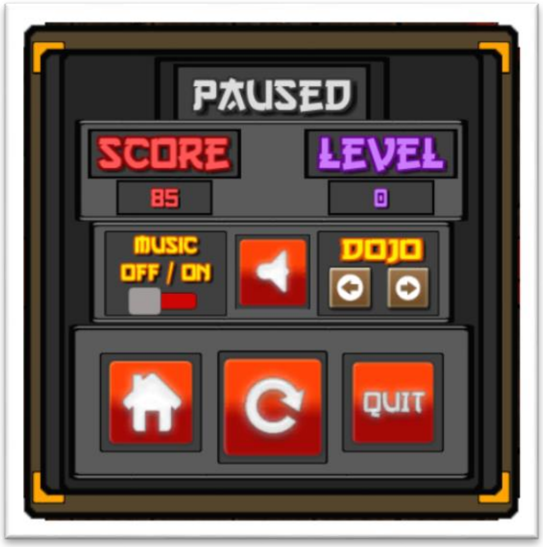
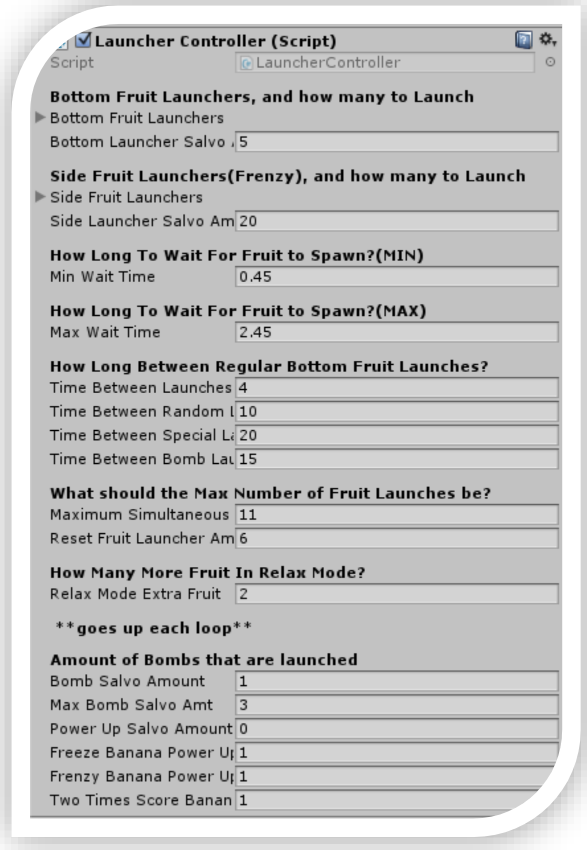


Figure 7 - Settings And Pause Menu that slides up from the bottom of the screen.

the round is still active, and false if it is over… it is only public for visual reference. The Float "WaitForMenuAtEnd" is the amount of time after the Game-Over Panel is called that the "Settings/Pause Menu" will slide on to the screen. *See Figure 7.* 1.5f seemed like a reasonable setting to me. The Game Controller obviously does more than just that, but this is a pretty good explanation of the inspector side. The Game Controller repeatedly asks the "Launcher Controller" to launch fruit, bombs, and/or power-ups depending on the game mode, while the "gameIsRunning".

➢ LauncherController

LauncherController.cs is responsible for calling "Launch" on all of the "FruitLauncher.cs's" in the scene. *See Figure 8.* Launch Controller handles all of the amounts, and timing of the Launches. So let's go through each of the public fields. The two Lists labeled "Bottom Fruit Launchers", and "Side Fruit Launchers" can be ignored for now. Those are populated via the script at round start. The "Bottom Launcher Salvo Amt" is incremented after each salvo. It should fall in between the "Max Simultaneous Fruit Launches", and the "Reset Fruit Launcher Amt". So that once it is launching 10, 15 or however many the "Max Simultaneous Fruit Launches" is set to, then it will reset itself to the "Reset Fruit Launcher Amt" and then start incrementing "Bottom Launcher Salvo Amt" again. This is so that the level can get progressively more difficult. You could set the reset amount to 1 and the Max amount to 20, and it would increase by one each launch. There are some random launches mixed in to break up that monotony but otherwise it would increment by one. The "Side Launcher Salvo Amt" is how many fruit will be launched when a Frenzy Power Up is hit in Arcade mode. Set to



Launcher Controller (Script)

Script	LauncherController
Bottom Fruit Launchers, and how many to Launch	
▸ Bottom Fruit Launchers	
Bottom Launcher Salvo	5
Side Fruit Launchers(Frenzy), and how many to Launch	
▸ Side Fruit Launchers	
Side Launcher Salvo Am	20
How Long To Wait For Fruit to Spawn?(MIN)	
Min Wait Time	0.45
How Long To Wait For Fruit to Spawn?(MAX)	
Max Wait Time	2.45
How Long Between Regular Bottom Fruit Launches?	
Time Between Launches	4
Time Between Random L	10
Time Between Special La	20
Time Between Bomb Lau	15
What should the Max Number of Fruit Launches be?	
Maximum Simultaneous	11
Reset Fruit Launcher Am	6
How Many More Fruit In Relax Mode?	
Relax Mode Extra Fruit	2
****goes up each loop****	
Amount of Bombs that are launched	
Bomb Salvo Amount	1
Max Bomb Salvo Amt	3
Power Up Salvo Amount	0
Freeze Banana Power Up	1
Frenzy Banana Power Up	1
Two Times Score Banan	1

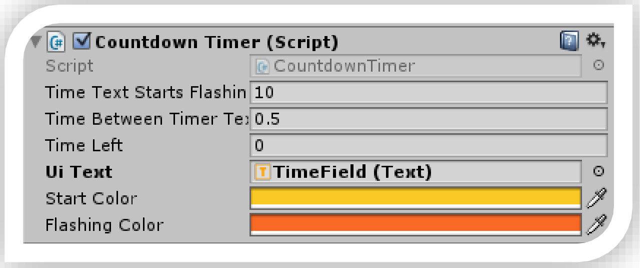
Figure 8 – Launch Controller Inspector

20 by default. The “min Wait Time”, and “max Wait Time” is how long it will take the fruit to launch when a salvo is requested. Set these values so that the fruit do not all launch at the same time. Set the “Time Between xxxxxxxx Launches” variables to the frequency in which Fruit, Random Fruit, Power-Ups (special), and Bombs should spawn. That way you can control the spawn rates of different items. “Relax Mode Extra Fruit” var is a multiplier for the fruit launched… since Relax Mode does not have any bombs, or special launches it will increase the number of regular fruit launched. Especially because there is no penalty for missing fruit in Relax Mode. Then the Amount of Objects That Are Launched portion at the bottom of the inspector (last 6 fields) you enter the amount of the objects you want to start with on the first launch. They will increment themselves after that. The max Bomb salvo amount is one of the bottom fields as well.

Note: Special Bananas do not get incremented… Just bombs and fruit. The Special Bananas will occur at the denomination of the bottom fields at the rate of “Time Between Special Launches” (center of inspector);

➢ CountdownTimer

CountdownTimer.cs is the round timer. *See Figure 9.* You can set at what time the clock (UI Text Component) should start flashing. You can also set what color it should change to once it starts flashing. The “timeBetweenTimerTextFlashes” is how long the time blink is. How many seconds the display is OFF. “Time Left” (controlled via script, it is just public for visual reference). The “uIText” is the text component of the canvas or panel that contains the timer. Then of course… the start and flashing color. Set them as desired.



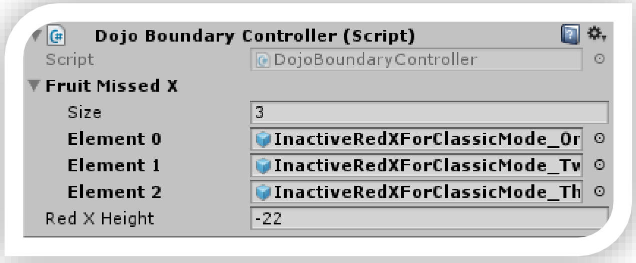
Countdown Timer (Script)

Script	CountdownTimer
Time Text Starts Flashin	10
Time Between Timer Te:	0.5
Time Left	0
Ui Text	**TimeField (Text)**
Start Color	🟨
Flashing Color	🟧

Figure 9 – Round Countdown Timer

➢ DojoBoundaryController

The DojoBoundaryController.cs is responsible for making fruit, bombs, and powe-ups that pass into its trigger area inactive (returning them to their respective pools). *See Figure 10.* Whenever a fruit passes into the trigger it increases the "fruitMissed" var. The static variable that all of the classes access. There are 3 Red X gameobjects in the scene that are inactive. When we are in classic mode and a fruit is lost, it will move one of the inactive red X's to the lost fruit's position and then to the height of -22 on the Y-Axis (so that it's just above the bottom of the screen). Then after a few seconds it goes inactive again. When 3 fruit have been missed, all three of the red X's have been used.



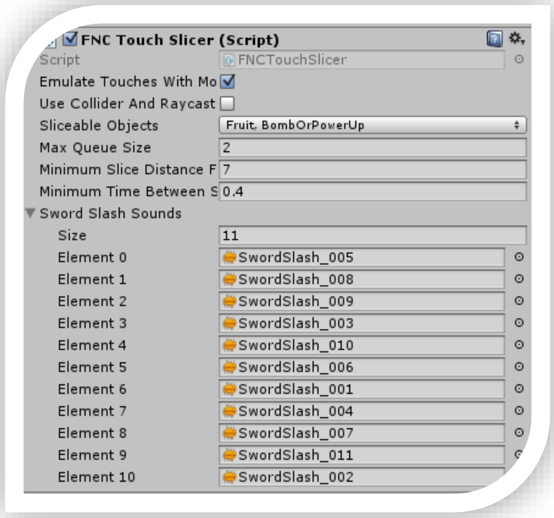
Dojo Boundary Controller (Script)	
Script	DojoBoundaryController
Fruit Missed X	
Size	3
Element 0	**InactiveRedXForClassicMode_Or**
Element 1	**InactiveRedXForClassicMode_Tw**
Element 2	**InactiveRedXForClassicMode_Th**
Red X Height	-22

Figure 10 - Dojo Boundary Controller

**** Game Controller Section Ends****

The FNCTouchSlicer

The FNCTouchSlicer is the class that gets input from mouse position or touch position. *See Figure 11.* While a finger is on the screen, or the mousebutton(0) is held down the script fires a raycast into the scene at that position. If "emulateTouchesWithMouse" is selected then the mouse will power all of the Inputs. Input.getMouseButtonDown(0) & Input.mousePosition. If it is N**ot** checked then we use Input.touches. Checking for "touch.phase == TouchPhase.Began, Moved, and Ended" while recording the "touch.position". Since the kit will work on standalone, mobile, or web… I usually leave the "Emulate Touches with Mouse" checked. There is another check that you can decide to do simultaneously (catches some fringe cases). You can select



FNC Touch Slicer (Script)	
Script	FNCTouchSlicer
Emulate Touches With Mo	✓
Use Collider And Raycast	☐
Sliceable Objects	Fruit, BombOrPowerUp
Max Queue Size	2
Minimum Slice Distance F	7
Minimum Time Between S	0.4
Sword Slash Sounds	
Size	11
Element 0	SwordSlash_005
Element 1	SwordSlash_008
Element 2	SwordSlash_009
Element 3	SwordSlash_003
Element 4	SwordSlash_010
Element 5	SwordSlash_006
Element 6	SwordSlash_001
Element 7	SwordSlash_004
Element 8	SwordSlash_007
Element 9	SwordSlash_011
Element 10	SwordSlash_002

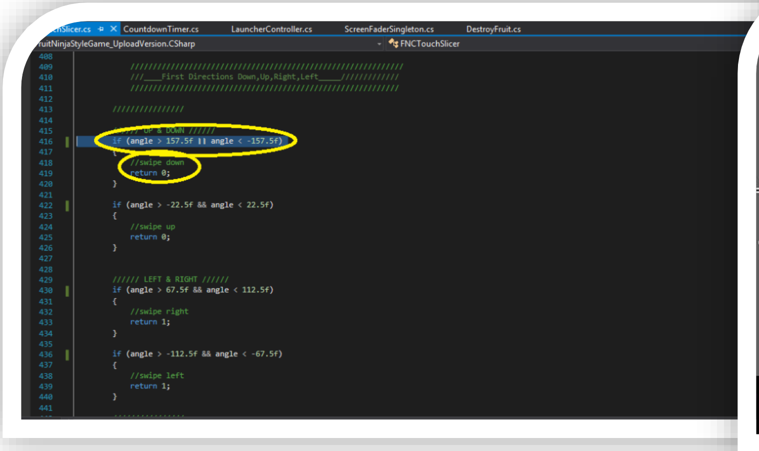
Figure 11 - FNCTouchSlicer

“useColliderAndRaycast” towards the top of the Touch Slicer Inspector. That will activate an elongated box collider that protrudes into the scene, allowing for raycast hits, AND collider hits. This collider is obviously larger than the raycast line. This means that there are more “close” hits than with the raycast alone. The majority of the time, when you are moving quickly the raycast will be responsible for the cuts. To test/see this. Add a “Debug.Log” to the Raycast Destroy, and to the “OnTriggerEnter” destroy, set the console to collapse, and count or watch for the difference. For this reason I set the “Bombs” colliders to be slightly smaller than the fruits (proportionately). Make sure that you select the layers that contain your fruit, and bombOrPowerUps for the layers that the raycast/collision will interact with. The “maxQueueSize” is a queue that stores the most recent “transform.positions” of the Touch Slicer. Depending on how high/low the “maxQueueSize” is, will determine the swipe angle accuracy. Setting this to 2 is probably ideal. When we are trying to get the direction vector we will subtract our current position from the position 2 frames earlier. Note: a single frame was not providing enough delta, and so I implemented this queue to help. “minimumSliceDistanceForAudio” and ”minimumTimeBetweenSwipes” both are in regard to the “swordSlashSound”. The MinimumSliceDistanceForAudio is the length a swipe has to be to trigger the slash sound. The minimumTimeBetweenSwipes” is how often swordSlashSound can play (we don’t want them spamming constantly). The array of sound clips at the bottom of the inspector are the sounds that will randomly play when we swipe.

Below is a unit square that we will refer to for understanding the angle calculation (what decides which cut prefab to instantiate). In this game there are several fruit. Each fruit has a halved equivalent with cuts going from top to bottom, left to right, diagonally up-right to down-left, and diagonally up-left to down-right. When we hit a fruit we compare our current position to where we were coming from(2 frames earlier), but this time we use Mathf.Atan2 and we feed it our xPos – our storedXPos, and our yPos – our storedYPos. Since we get radians back from Mathf.Atan2 we will multiply our radians by 57.2957795f (instead of using Mathf.Rad2Deg… it’s the same thing anyway….

Float angle = Mathf.Atan2(currentPos.x – storedPos.x, currentPos.y – storedPos.y);

Then we multiply the returned radian value by 57.2957795f and we get our angle in degrees. Now… with that angle we create several conditionals. See the UnitCircle below and note the degrees I



```
        ////////////////////////////////////////////////////////////
        ///____First Directions Down,Up,Right,Left____////////////
        ////////////////////////////////////////////////////////////

    ///////////////

    ////// UP & DOWN //////
    if (angle > 157.5f || angle < -157.5f)
    {
        //swipe down
        return 0;
    }

    if (angle > -22.5f && angle < 22.5f)
    {
        //swipe up
        return 0;
    }


    ////// LEFT & RIGHT //////
    if (angle > 67.5f && angle < 112.5f)
    {
        //swipe right
        return 1;
    }

    if (angle > -112.5f && angle < -67.5f)
    {
        //swipe left
        return 1;
    }
```

Figure 12 – Visual Studio FNCTouchSlicer

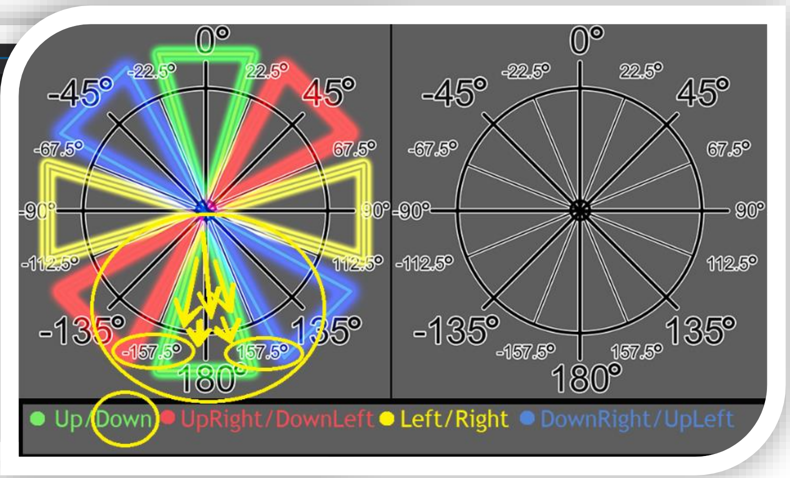


Figure 13 – Unit Circle showing swipe angles

added to it, and how they enclose an area that will help you visualize our "swipeDirection". I have also pasted some code of the conditionals. Note the angles... So, the first conditional states that

IF ("angle" is greater than 157.5f OR "angle" is less than -157.5f)

I circled the degrees on the unitCircle, and the Conditional in Visual Studio with yellow ellipses. Notice the smaller font I added to the unitCircle are the evenly spaced degrees that are used in the Conditionals. We could make a 4 direction version, or an 8 direction (what we used here). You could also do a 16, or a 3… it's up to you. Just ask the right question. As an added exercise… what would the angles be for a 4 direction setup? Here is a hint… those degrees are listed on the unitCircle as well. The Answer is…

We would use Less than 45 AND Greater than -45 for UP

We would use Greater than 135 OR less than -135 for DOWN

We would use Greater than 45 AND less than 135 for RIGHT

We would use Less than -45 AND Greater than -135 for LEFT

Note… You could also decide that you didn't want to make the angles looser… you could use the smaller angles from the 8way, BUT if you wanted to make good use of the space you'd make forgiving "casual half tilted phone" swipes work.   For that you would use the full range in between -45 and 45 for up… instead of requiring the tighter -22.5 and 22.5 for up. Which would be a lot tighter.

This FNCTouchSlicer returns an integer when we ask it to calculate the angle of our most recent swipe.

It returns 0 for an up or down swipe.

1 for a left or right swipe,

2 for a diagonal up-Right or down-Left swipe, or finally…

3 for an up-Left or down-Right.

We pass this integer to the DestroyFruit.cs's "CutFruit" Method that way it knows which halves to instantiate. This is kind of a tricky/hacky way to do this. We could get even crazier and make it a 12 or 16 directional swipe and then create even more mesh variations, but this isn't the way we program these sorts of things. Fewer lines of code, and minimal number of repeated art assets are ideal… not the opposite. That fruit mesh file is getting pretty big, and the whole thing requires a fair amount of inspector setup. It is not a maintainable or elegant solution. I will however state that low powered systems can benefit from this sort of thinking/system creation, especially with "Destruction" and using "non-physics gibs" to make things look very breakable/real, but save the complex mesh deformation, and the rigidbody count/physics calculations… You can make pretty believable explosions by doing mesh swaps then animating gibs. Low end systems like some mobiles will thank you.

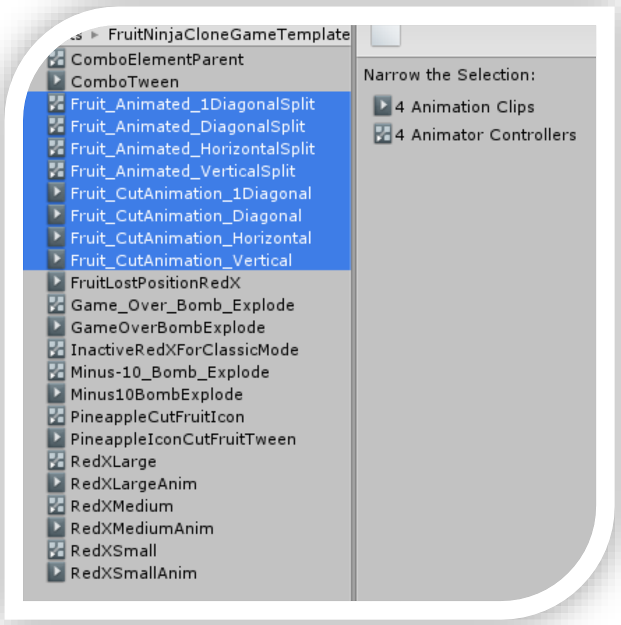
I personally loved the Mesh Deformation Version that cut the mesh, added faces to cover the whole, and then had a predetermined set of coordinates on a texture sheet that had the "Fruit Core" where it unwrapped the UV's to. It was just too heavy for my mobile, and had basically a 2-4 second. Delay before finishing the job, and it made use of rigidbodies for the “new” piece. One Rb for the original deformed piece, and another for the discarded “half”. I am sure it can be optimized and I will be looking to do that to make this kit more “Proper”, but for now, and for an easy to understand implementation. This technique will do. The Fixed Time Step has been boosted in this project (to produce more collision and cast checks), and since there isn’t much going on in the game it runs just fine with the boosted physics time… as far as I can tell.

The FNCTouchSlicer mainly triggers our “CutFruit” method, and provides the current direction vector to a few other components… I.e. the direction of the fruit splatter that is plastered on the dojo walls. It gets the direction from our FNCTouchSlicer, so it knows how far out to instantiate the splatter sprite renderer, and what rotation to make the splatter face. Etc. etc.…

There are plenty of other Notes/Comments in the FNCTouchSlicer.cs / DoxyGen. If you need any further explanation have a look at them, and if you still don’t understand it then reach out, and I’ll make some more material.

The Animated Fruit Cut Types

There are 4 Unity Animations that were made on Empty GameObjects, which had 2 Empty Children. This was done so that we had a generic base to use for the Fruit Halves. When making these animation I added the fruit halves as children of the Empty Children. That way I could visualize the necessary motion of the different cut types. After they were close enough to "done", I removed the fruit halves I had used temporarily, lined up the necessary fruit half combinations, and started duplicating the different Empty Cut Type GameObjects for the number of fruit I had. One by one I brought the Animated parent I.e. "VerticalSliceAnimatedParent01" to the world cords 0,0,0 and then I brought "AppleHalvedVertically_Piece1", and "AppleHalvedVertically_Piece2" to world position 0,0,0 and I parented each half to the empty children. Went into the animation tab, and hit play. Then I made sure that the animation threw one gib to the left, and the other to the right, and called it "done". Then I labelled that instance of "VerticalSliceAnimatedParent01" as "Apple_Animated_VerticalSplit" and dragged it into the project side to create a prefab. Then I brought over a horizontal cut animated empty, and repeated the whole process. That was kind of tedious but it only took an hour (not counting the initial 30mins to create the 4 animations with the empties).

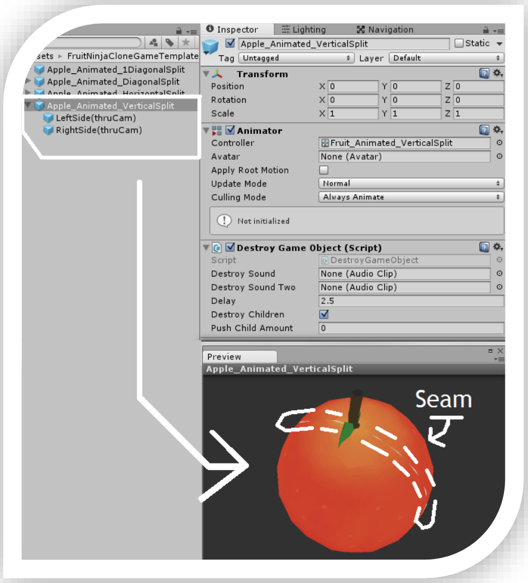


FruitNinjaCloneGameTemplate
ComboElementParent
ComboTween
Fruit_Animated_1DiagonalSplit
Fruit_Animated_DiagonalSplit
Fruit_Animated_HorizontalSplit
Fruit_Animated_VerticalSplit
Fruit_CutAnimation_1Diagonal
Fruit_CutAnimation_Diagonal
Fruit_CutAnimation_Horizontal
Fruit_CutAnimation_Vertical
FruitLostPositionRedX
Game_Over_Bomb_Explode
GameOverBombExplode
InactiveRedXForClassicMode
Minus-10_Bomb_Explode
Minus10BombExplode
PineappleCutFruitIcon
PineappleIconCutFruitTween
RedXLarge
RedXLargeAnim
RedXMedium
RedXMediumAnim
RedXSmall
RedXSmallAnim

Narrow the Selection:
4 Animation Clips
4 Animator Controllers

Figure 14 – Unity Animations and Animator Controllers

Later I made a particle system with stretched bill-boarded directional “tear drop” shaped sprites and a “round drop” shaped (non-stretched) particle system… but as usual all the transparency and particle systems started to kill me.. Draw calls were up above 50, and the transparency was the bulk of the rendering hit when profiling on mobile (this was after decimating the meshes in 4 passes to bring down the total number of batches). The fruit destroy “particle blast” was reduced from a pretty “3 system setup”, to a “1 system setup”, and the “2 system slice Particles” went down to a “1 particle system” setup. Lots of things were reduced until the materials and drawcalls were under control. That effectively describes the “Fruit Halves”. The same 4 animations are used for all fruit. There are fruit juice splash particle systems, a slice particle system (elongated yellow diamond that is instantiated the direction of swipe), and then the splatter meshes which fade in and out (as described at the end of the TouchSlicer section above. They orient to the direction of swipe as well).



Inspector
Lighting
Navigation
sets ▸ FruitNinjaCloneGameTemplate
Apple_Animated_1DiagonalSplit
Apple_Animated_DiagonalSplit
Apple_Animated_HorizontalSplit
Apple_Animated_VerticalSplit
LeftSide(thruCam)
RightSide(thruCam)
Preview
Apple_Animated_VerticalSplit
Seam

Apple_Animated_VerticalSplit — Static: ☐
Tag: Untagged — Layer: Default

Transform
Position X 0 Y 0 Z 0
Rotation X 0 Y 0 Z 0
Scale X 1 Y 1 Z 1

Animator
Controller: Fruit_Animated_VerticalSplit
Avatar: None (Avatar)
Apply Root Motion: ☐
Update Mode: Normal
Culling Mode: Always Animate
Not initialized

Destroy Game Object (Script)
Script: DestroyGameObject
Destroy Sound: None (Audio Clip)
Destroy Sound Two: None (Audio Clip)
Delay: 2.5
Destroy Children: ✓
Push Child Amount: 0

Figure 15 – An Animated Apple Half Selected (Vertical Split)

The only other thing these “DestroyedFruitPrefabs” have is a Destroy GameObject Script, which will destroy it 2.5 seconds after we spawn it. That can be seen in As an example… to get the vertical sliced apple shown above, we would need to slice an apple (the whole untouched fruit), and be swiping at an upward, or downward angle… that would call:

```
CutFruit(0); (Zero for a Vertical Split)
```

The Particles Used for the Fruit “Cut/Destroy”

There are 3 particle systems used when a fruit is sliced. As mentioned in the above text. The first particle system is the “FruitExplosionParticleSystem”. It is the “juice” from inside the fruit. *See Figure 17.* The next is the “Slice” particle. It plays very quickly and is not that noticeable, but it adds a fair amount. It used to have a nice sparkle, as well, but there were just too many particle systems for low end hardware. Now it is just the diamond shape quickly growing to full size, and then zips back to nothing. Based on the slice direction the matching “slice diamond” is spawned. There are four versions of it as well. 0 = Vertical, 1 = Horizontal, 2 = 1Diagonal, and 3 = Diagonal, just like with the “Fruit Cut Type”. Note that in the “FruitDestroy.cs” Inspector above that the slices are in the exact same order as the gibs. This is important. They have to be in that order.

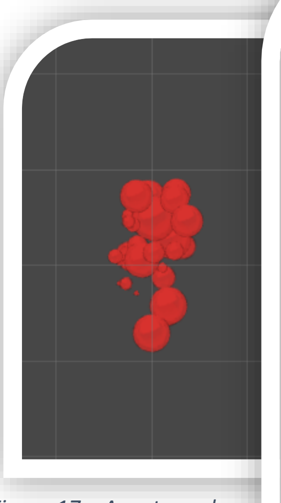
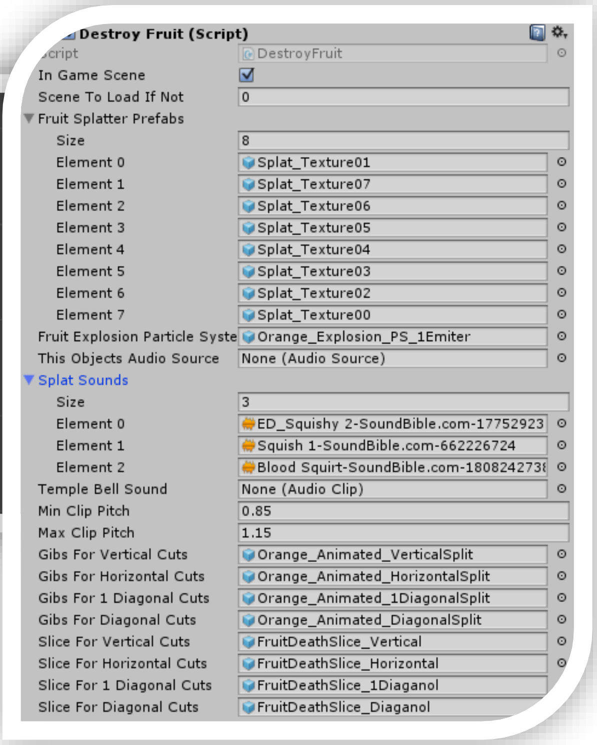


Figure 17 – A watermelon explosion particle



Destroy Fruit (Script)

Script	DestroyFruit
In Game Scene	✓
Scene To Load If Not	0
Fruit Splatter Prefabs	
Size	8
Element 0	Splat_Texture01
Element 1	Splat_Texture07
Element 2	Splat_Texture06
Element 3	Splat_Texture05
Element 4	Splat_Texture04
Element 5	Splat_Texture03
Element 6	Splat_Texture02
Element 7	Splat_Texture00
Fruit Explosion Particle Syste	Orange_Explosion_PS_1Emiter
This Objects Audio Source	None (Audio Source)
Splat Sounds	
Size	3
Element 0	ED_Squishy 2-SoundBible.com-17752923
Element 1	Squish 1-SoundBible.com-662226724
Element 2	Blood Squirt-SoundBible.com-180824273
Temple Bell Sound	None (Audio Clip)
Min Clip Pitch	0.85
Max Clip Pitch	1.15
Gibs For Vertical Cuts	Orange_Animated_VerticalSplit
Gibs For Horizontal Cuts	Orange_Animated_HorizontalSplit
Gibs For 1 Diagonal Cuts	Orange_Animated_1DiagonalSplit
Gibs For Diagonal Cuts	Orange_Animated_DiagonalSplit
Slice For Vertical Cuts	FruitDeathSlice_Vertical
Slice For Horizontal Cuts	FruitDeathSlice_Horizontal
Slice For 1 Diagonal Cuts	FruitDeathSlice_1Diaganol
Slice For Diagonal Cuts	FruitDeathSlice_Diaganol

Figure 16 – The Expanded View of Destroy Fruit. (Figure 5 is the collapsed view)

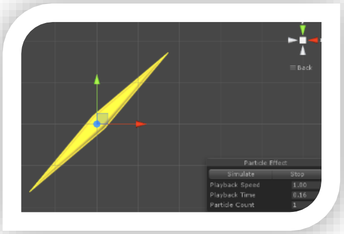
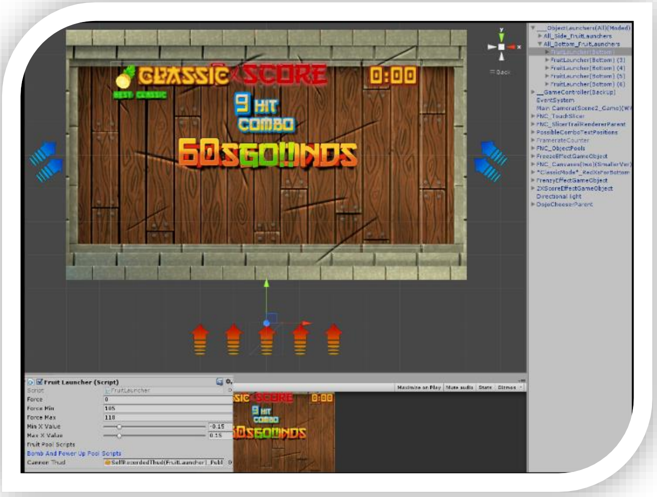


Figure 18 - Slice Particle (2) “Diagonal Up-Right”

The Fruit Launchers… not the Launcher Controller

The FruitLauncher is on several scene objects. See the Red and Blue arrows in the SceneView. *See Figure 19.* The red arrows are the bottom launchers in the scene, there are 5 of them in each game scene. The LauncherController gets a reference to each of these. The FruitLauncher is the class that actually fires the fruit,bombs, or powerups(per the name), but the LauncherController is what requests they fire. That way the LauncherController can loop thru the "salvoAmt" and call "Launch" for every loop iteration. If you look closely at *Figure 19.* You will see that there are not many options on the FruitLauncher.cs. The "force" is the force that the object is fired at. We use "ForceMin" and "ForceMax" to generate a random value and we assign it to "force". "force" is always between "forceMin", and "forceMax". The last variable is a "CannonThud" sound clip. This is the sound that plays anytime a bottom launcher fires. Finally, just above the "CannonThud" There are two Arrays… "FruitPoolScripts", and "BombAndPowerUpPoolScripts".



```
Fruit Launcher (Script)
Script               FruitLauncher
Force                0
Force Min            105
Force Max            118
Min X Value          -0.15
Max X Value          0.15
Fruit Pool Scripts
Bomb And Power Up Pool Scripts
Cannon Thud          SelfRecordedThud(FruitLauncher|_Publ
```

Figure 19 - The Arrows in the Scene are Fruit Launchers

The Arrays will actually contain all of the object pools in the scene, once the game is running. There are two private gameObject variables named "fruitPoolsGameObject" and "bombAndPowerUpPoolsGameObject". The "fruitPoolsGameObject" is tagged with "FruitPools", and "bombAndPowerUpPoolsGameObject" is tagged with "OtherPools". Each of these gameObjects have several children. There is a child for every type of fruit we launch in the game. The same can be said for the "bombAndPowerUp PoolsGameObject" but it has children for every type of bomb and power-up instead. So we will assign the "FruitPoolScripts" array the returned values of "fruitPoolsGameObject.GetComponentsInChildren<ObjectPoolScript>()". That way “FruitPoolScripts” has a reference to all of the fruit pools. We will do the same thing for the “BombAndPowerUpPoolScripts” array. So we do “bombAndPowerUpPoolsGameObject.GetComponentsInChildren<ObjectPoolScript>() that way “BombAndPowerUpPoolScripts” has a reference to all of the different bomb and power up pools. *See Figure 20a.*

So, when the LauncherController needs to launch 4 fruit, It will for loop through the number of fruit requested(4), And each time it will start a coroutine (still in the LauncherController) that will call the method “LoadAndFireRandomFruit()” on the FruitLauncher.

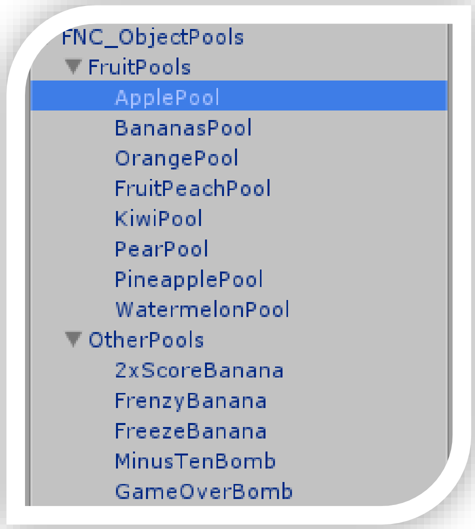
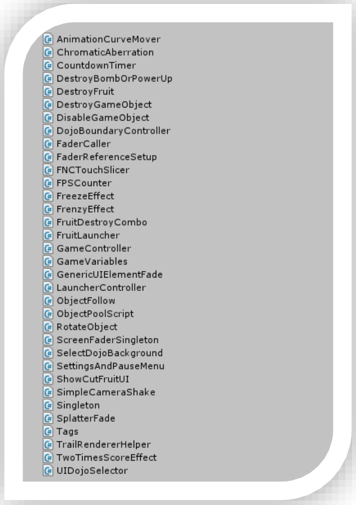

```
FNC_ObjectPools
  ▼ FruitPools
      ApplePool
      BananasPool
      OrangePool
      FruitPeachPool
      KiwiPool
      PearPool
      PineapplePool
      WatermelonPool
  ▼ OtherPools
      2xScoreBanana
      FrenzyBanana
      FreezeBanana
      MinusTenBomb
      GameOverBomb
```

Figure 20a - The 2 GameObject that are Tagged “FruitPools” and “OtherPools”... and their children. The Fruit Children each have an ObjectPoolScript (fruitType), and The PowerUp/Bomb Children each have an ObjectPoolScript for the (powerUp/BombType). We use FruitPools.GetComponentsInChildren<ObjectPoolsScript>(), and OtherPools.GetComponentsInChildren<ObjectPoolsScript>() to fill the two arrays discussed above.

3.) Scripts *See Figure 20b.*

The scripts we didn't cover… The Inspector, and a brief description.

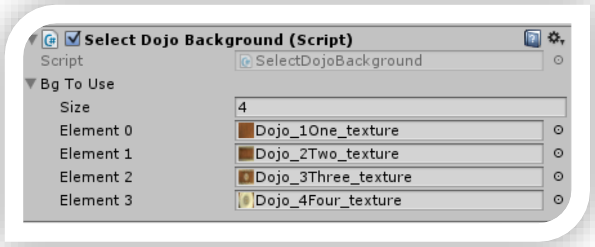


AnimationCurveMover
ChromaticAberration
CountdownTimer
DestroyBombOrPowerUp
DestroyFruit
DestroyGameObject
DisableGameObject
DojoBoundaryController
FaderCaller
FaderReferenceSetup
FNCTouchSlicer
FPSCounter
FreezeEffect
FrenzyEffect
FruitDestroyCombo
FruitLauncher
GameController
GameVariables
GenericUIElementFade
LauncherController
ObjectFollow
ObjectPoolScript
RotateObject
ScreenFaderSingleton
SelectDojoBackground
SettingsAndPauseMenu
ShowCutFruitUI
SimpleCameraShake
Singleton
SplatterFade
Tags
TrailRendererHelper
TwoTimesScoreEffect
UIDojoSelector

Figure 20b - List of all Scripts in FNC

I will try to keep this part short. The info in the Setup and Things to Know sections really covers some of the most important points. Again, there is a lot of information in the scripts themselves, and if you happen to not be in the IDE, you can always access the DoxyGen Docs Page… for the Class/Method Descriptions and the members list.

Select Dojo Background *See Figure 21.*



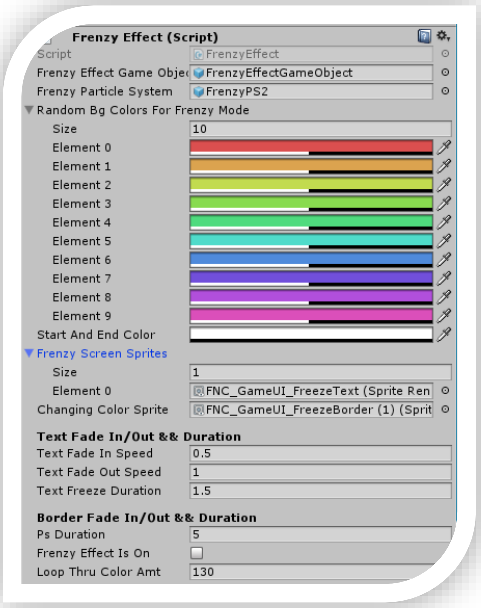
Select Dojo Background (Script)

Script	SelectDojoBackground
Bg To Use	
Size	4
Element 0	Dojo_1One_texture
Element 1	Dojo_2Two_texture
Element 2	Dojo_3Three_texture
Element 3	Dojo_4Four_texture

Figure 21 - DojoBackgroundSelector.cs - Inspector

There is a gameObject in all of the scenes that have a Dojo Background named "DojoChooserParent", and it has a child gameObject named "FNC_ChooseDojo". This script has an array, a few simple variables, and 2 methods that cycle forward, and backwards through the array of "Backgrounds", and then change the materials texture to one of the background textures. The value that holds where we are at in the cycle is a Static Int in the GameVariables class. When its value is updated we save its value to PlayerPrefs. That way all of the scenes, and even when the game is closed you can come back to the same dojo. I was originally going to make them unlock based on the Players Experience/Level, but never got to it. The project took me a little longer than I intended to spend on it, with all of the performance issues and refactoring.

Frenzy Effect *See Figure 22.*

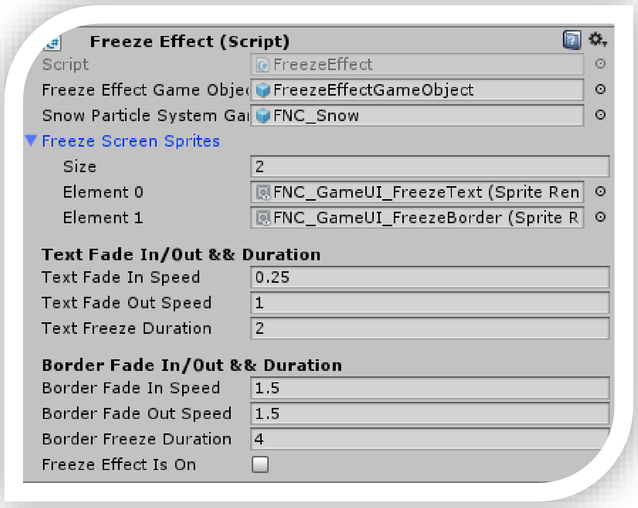


```
Frenzy Effect (Script)
Script                       FrenzyEffect
Frenzy Effect Game Objec     FrenzyEffectGameObject
Frenzy Particle System       FrenzyPS2
▼ Random Bg Colors For Frenzy Mode
    Size                     10
    Element 0                🟥
    Element 1                🟧
    Element 2                🟨
    Element 3                🟩
    Element 4                🟩
    Element 5                🟦
    Element 6                🟦
    Element 7                🟪
    Element 8                🟪
    Element 9                🟪
Start And End Color          ⬜
▼ Frenzy Screen Sprites
    Size                     1
    Element 0                FNC_GameUI_FreezeText (Sprite Ren
Changing Color Sprite        FNC_GameUI_FreezeBorder (1) (Sprit

Text Fade In/Out && Duration
Text Fade In Speed           0.5
Text Fade Out Speed          1
Text Freeze Duration         1.5

Border Fade In/Out && Duration
Ps Duration                  5
Frenzy Effect Is On          ☐
Loop Thru Color Amt          130
```

Figure 22 - FrenzyEffect.cs – Inspector (Expanded)

This is Class is attached to a gameObject in the scene. Actually, all of the "power-up" effects have their own gameObject in the scene. The typically hold sprite renderer with text/graphics, and a particle system. All of the effects also have a Boolean variable that is only "true" when the effect is active (to make sure we do not try to activate it when the effect is already active). The Frenzy Effect has a sprite renderer the same size as the Dojo Background, and when the effect is active, we make it partially transparent. Then we strobe through the colors in the list to the left, so that it gives off a bit of a disco effect. The fist variable you see is a reference to the FrenzyEffect object itself. The next variable is the particle system that we activate while the effects active. It shoots stars out of the area where the side fruit launchers, launch fruit. The single element array has a sprite from the UI atlas that says "Frenzy!"… Then we have some values at the bottom that determine when certain sprites are faded in, out, and how long they stay visible.

FreezeEffect *See Figure 23.*



```
Freeze Effect (Script)
Script                       FreezeEffect
Freeze Effect Game Objec     FreezeEffectGameObject
Snow Particle System Gai     FNC_Snow
▼ Freeze Screen Sprites
    Size                     2
    Element 0                FNC_GameUI_FreezeText (Sprite Ren
    Element 1                FNC_GameUI_FreezeBorder (Sprite R

Text Fade In/Out && Duration
Text Fade In Speed           0.25
Text Fade Out Speed          1
Text Freeze Duration         2

Border Fade In/Out && Duration
Border Fade In Speed         1.5
Border Fade Out Speed        1.5
Border Freeze Duration       4
Freeze Effect Is On          ☐
```

Figure 23 - FreezeEffect.cs = Inspector (Expanded)

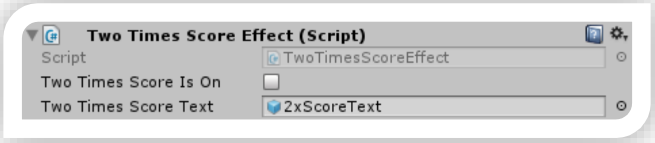
The "FreezeEffect.cs" is also on a scene gameObject. This effect has 2 sprites... (1.) The "Freeze!!" Text on the UI atlas AND (2.) The Ice Border that goes around the scene. There is also a particle system that creates snow-flakes. When the "FreezeEffect" is active, Time.timeScale is set to 0.6f. Note:

these effects would be made better if they had some sound effects. An ice style sound would be good for when the ice fades onto the scene, or some sort of disco tune when the "Frenzy Effect" is on. Then we have some values at the bottom that determine when certain sprites are faded in, out, and how long they stay visible.

<u>2 x Score Effect</u> *See Figure 24.*

"TwoTimesScoreEffect.cs", the final "power-up". These effects really need their own gibs, destruction particles, and destroy sounds to make them "pop". They look a little weak just "vanishing", and them all spawning the same star blast and star falling particle system. I was kind of running out of time on dressing this project... so I had to cut a corner there. I figured I could leave that to the user, if they felt like using it for anything more than learning/tinkering.

The final effects inspector is pretty simple… We just activate a text Component that resides at the top of the scene while the 2 x Score effect is active, and that is it. The GameController knows to increase the score by two, if 1.) The round type is Arcade, and 2.) The effect is on (Boolean below – "twoTimesScoreIsOn"). This effect could use the above mentioned upgrades, but also maybe something like a "+2" pop up that fires right next to any fruit cut during 2xScoreEffect… would have to check out how many draw calls that would be added under load (like when frenzy is on simultaneously…). It would need to be tested, but it would greatly increase the quality of the "power-up" side of things. I will make that the first thing I work on for the 1.01 update. ☺

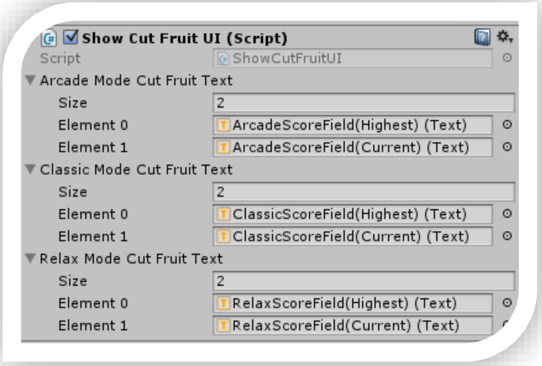

Two Times Score Effect (Script)

Script	TwoTimesScoreEffect
Two Times Score Is On	☐
Two Times Score Text	2xScoreText

Figure 24 - TwoTimesScoreEffect.cs - Inspector

<u>Show Cut Fruit UI</u>

ShowCutFruitUI.cs has a very simple job. The only thing it does is update WHICH current score/best score is displayed at the top left corner of the game. It disables all of the text elements in the 3 arrays on start, and then based on the current game mode it enables only those text elements. The game controller controls the values, and this class controls whether or not they are enabled, or disabled. In hindsight, I should have just put this functionality on the game controller itself. I don't think I really know that I did that until writing this… lol ;) How silly… I will restructure and have the game controller take over that responsibility at a later time.

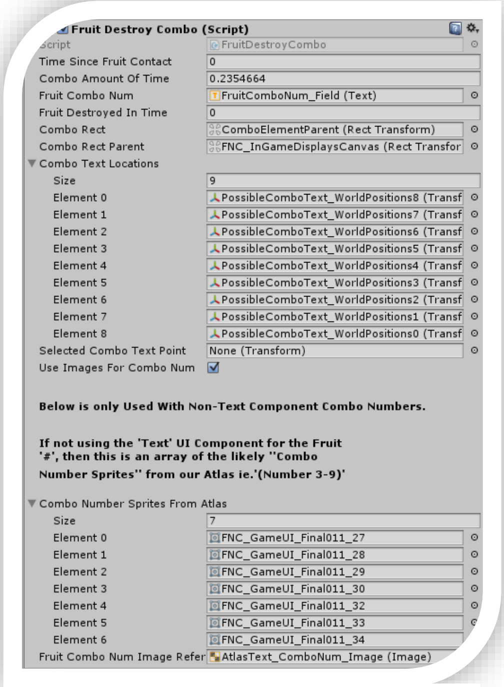


```
Show Cut Fruit UI (Script)
Script                      ShowCutFruitUI
▼ Arcade Mode Cut Fruit Text
    Size                    2
    Element 0               ArcadeScoreField(Highest) (Text)
    Element 1               ArcadeScoreField(Current) (Text)
▼ Classic Mode Cut Fruit Text
    Size                    2
    Element 0               ClassicScoreField(Highest) (Text)
    Element 1               ClassicScoreField(Current) (Text)
▼ Relax Mode Cut Fruit Text
    Size                    2
    Element 0               RelaxScoreField(Highest) (Text)
    Element 1               RelaxScoreField(Current) (Text)
```

Figure 25 - ShowCutFruitUI.cs - Inspector

Fruit Destroy Combo

FruitDestroyCombo.cs takes care of the fruit cut combo pop-ups. The system has an approximate 0.5 second count in which it will record how many fruit are destroyed. As soon as that time runs out it starts it again. So basically if you swipe through 6 fruit in a half a second, it will say… "ok, the player reached the minimum required "fruitDestroyedInTime" within a half second countdown. Enable the combo text at the anchor position nearest to the last fruit that was cut, and select the appropriate sprite for the number of fruit they cut. By unchecking the Boolean "useImagesForComboNum" you can have a text component fed the combo amount. Which works fine… I just so happened to have numbers 3 – 9 on the UI atlas in the same style as the combo text so I use them. It looks okay with the regular text, but mine had a subtle gradient and I wanted it to match.



```
Fruit Destroy Combo (Script)
Script                        FruitDestroyCombo
Time Since Fruit Contact      0
Combo Amount Of Time          0.2354664
Fruit Combo Num               FruitComboNum_Field (Text)
Fruit Destroyed In Time       0
Combo Rect                    ComboElementParent (Rect Transform)
Combo Rect Parent             FNC_InGameDisplaysCanvas (Rect Transfor
▼ Combo Text Locations
    Size                      9
    Element 0                 PossibleComboText_WorldPositions8 (Transf
    Element 1                 PossibleComboText_WorldPositions7 (Transf
    Element 2                 PossibleComboText_WorldPositions6 (Transf
    Element 3                 PossibleComboText_WorldPositions5 (Transf
    Element 4                 PossibleComboText_WorldPositions4 (Transf
    Element 5                 PossibleComboText_WorldPositions3 (Transf
    Element 6                 PossibleComboText_WorldPositions2 (Transf
    Element 7                 PossibleComboText_WorldPositions1 (Transf
    Element 8                 PossibleComboText_WorldPositions0 (Transf
Selected Combo Text Point     None (Transform)
Use Images For Combo Num      ✓

Below is only Used With Non-Text Component Combo Numbers.

If not using the 'Text' UI Component for the Fruit
'#', then this is an array of the likely ''Combo
Number Sprites'' from our Atlas ie.'(Number 3-9)'

▼ Combo Number Sprites From Atlas
    Size                      7
    Element 0                 FNC_GameUI_Final011_27
    Element 1                 FNC_GameUI_Final011_28
    Element 2                 FNC_GameUI_Final011_29
    Element 3                 FNC_GameUI_Final011_30
    Element 4                 FNC_GameUI_Final011_32
    Element 5                 FNC_GameUI_Final011_33
    Element 6                 FNC_GameUI_Final011_34
Fruit Combo Num Image Refer   AtlasText_ComboNum_Image (Image)
```

Figure 26 - FruitDestroyCombo.cs - Inspector

The “Text” component is in place (just disabled), so you can try that out too. This class has some detailed comments in it, I would recommend checking them out, and the inspecting the canvas in the scene that is labelled “FNC_InGameDisplaysCanvas” if you have any additional questions.

ChromaticAberration & SimpleCameraShake

Both of these scripts are very similar to the versions in CSC. I did more complete commenting in them, and I made several fields that were formerly public, private. Keeps the clutter down in the inspector. If you have seen these before you will be able to tell that the inspector is much cleaner… I am just going to recommend you go look at how these are setup.

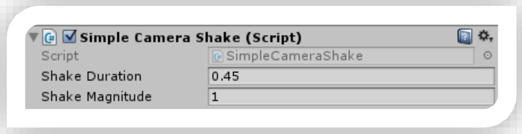

Simple Camera Shake (Script)
Script SimpleCameraShake
Shake Duration 0.45
Shake Magnitude 1

Figure 27 - SimpleCameraShake.cs - Inspector

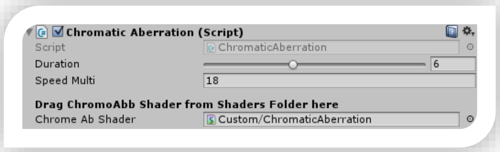

Chromatic Aberration (Script)
Script ChromaticAberration
Duration 6
Speed Multi 18
Drag ChromoAbb Shader from Shaders Folder here
Chrome Ab Shader Custom/ChromaticAberration

Figure 28 - ChromaticAberration.cs - Inspector

As you can see from the inspector of the “SimpleCameraShake” the variables are pretty self-explanatory… once those variables are set, and that script is attached to your main camera… you can call StartShake(); and If you look at the Chromatic Aberration.cs’s Inspector… All you have to do is drag the ChromaticAberration Shader from the project directory onto the field, set the duration, and speed (both pretty self-explanatory), and then call StartAberration(); if you want to really see what the chromatic aberration does then set the duration high, and the speed low, that way you can watch the color channels expand and contract.

Settings and Pause Menu

SettingsAndPauseMenu.cs is responsible for displaying the menu in the game. There is only one menu and it is not a traditional “Pause Menu”, or a traditional “Settings Screen”. This settings and pause menu can be accessed from any scene (it is a child of the fader canvas). When the user hits “Esc”, or “Back” on android the menu slides up onto the screen.

From it you can exit the game, reload the current level, go to the main menu, toggle music, see the player’s experience/level, and if you are in a scene with a dojo background you can cycle forward, and backward through the available dojos. For this class you will need to drag the “fullSizeBG” onto the “tintedColorBG” variable, and then drag the “menuWindow”panel onto the “pauseMenu” variable. Check “ignoreGameTime”(so that it will still slide up onto the screen if called, and the Time.timeScale is set to 0f). Check “useScreenSizeCalculations” to make the class measure the screen size on game start, and then find the center. It will make it so that no matter the screen size, the menu will always slide from off screen below the viewable area, and it will always slide to the center of the screen.

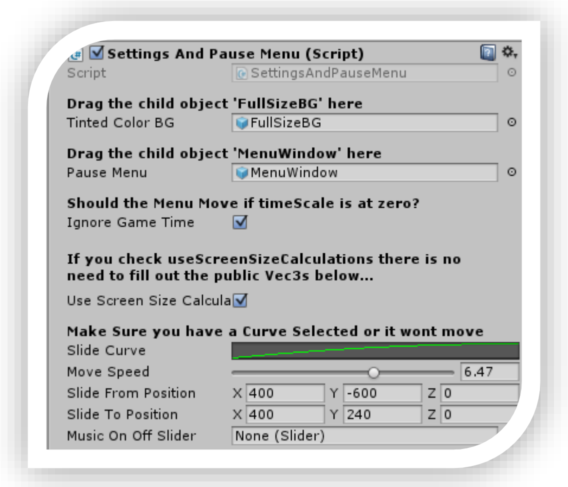

Settings And Pause Menu (Script)
Script: SettingsAndPauseMenu
Drag the child object 'FullSizeBG' here
Tinted Color BG: FullSizeBG
Drag the child object 'MenuWindow' here
Pause Menu: MenuWindow
Should the Menu Move if timeScale is at zero?
Ignore Game Time ✓
If you check useScreenSizeCalculations there is no need to fill out the public Vec3s below...
Use Screen Size Calcula ✓
Make Sure you have a Curve Selected or it wont move
Slide Curve
Move Speed: 6.47
Slide From Position: X 400 Y -600 Z 0
Slide To Position: X 400 Y 240 Z 0
Music On Off Slider: None (Slider)

Figure 29 - SettingsAndPauseMenu.cs

Note: if you change the screen size after the game has started, or later you make the game work in multiple orientations, then will have to set a way to call the “calculate screen size methods” again, otherwise the menu will use coordinates that are not accurate anymore.

For now as long as the game is locked to landscape, in production it will be fine, and in the editor as long as you don’t resize the game windows “mid-play” it will be fine.

The rest of the variables are fairly straight forward. The curve needs to be created (this is how it will slide (the “easing”)). Set the move speed to your liking. The “slideFromPosition’ and “slideToPosition” only have to be filled out IF you did **NOT** check useScreenSizeCalculations. Finally the slider is a reference the the slider underneath the Music On/Off text, that shows us which is selected. It cannot be moved by tapping or sliding, it is just for visual reference. To Mute or UnMute the music you hit the red button with the speaker on it in the middle of the settings and pause menu… *See Figure 7.* Refer to the SettingsAndPauseMenu.cs Comments or DoxyGen Doc Pages for any more information about the class.

FaderCaller & UIDojoSelector

FaderCaller.cs and UIDojoSelector.cs or both very short and simple. The FaderCaller has returned from the CSC complete project template… It only does one thing. It instantiates our ScreenFaderSingleton. Way back when it did a little more, but it only has one responsibility now. I will probably start handing this off to some other manager class, but up until now… it has worked well. "If it isn't broke, don't fix it". So FaderCaller is all of the cameras in the game, and that is one of a handful of reasons that the game can be started from scenes other than Scene0 without there being several errors/reference exceptions. You should be able to fire up the game from any scene while in the editor, BUT the intended use is start from Scene0 (I know I have said that a few times, just keep an eye out for any issues if you start the game elsewhere).

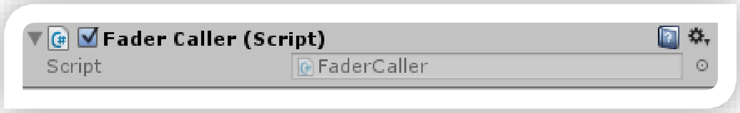

Fader Caller (Script)
Script: FaderCaller

Figure 30 - FaderCaller.cs has returned from CSC. It calls the ScreenFaderSingleton into existence from the splash scene. That is it.

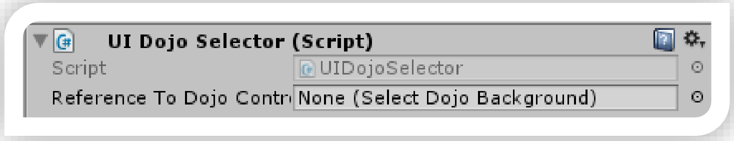

UI Dojo Selector (Script)
Script: UIDojoSelector
Reference To Dojo Contr: None (Select Dojo Background)

Figure 31 - The UIDojoSelector.cs is a short script that we attach to the SettingAndPauseMenu gameObject and it makes the "Forward" / "Backward" arrows that change the "Current Dojo" call Methods in the class, and those methods increment/write to disk the Static var that determines what DojoBG we use.

The UIDojoSelector.cs is a pretty small class as well. It uses a static "instance" reference so we can access the SelectDojoBackground class that switches the backgrounds. The only methods this class contains are RunChangeNext(), and RunChangePrevious(). The SettingsAndPauseMenu UI holds two buttons (a back arrow, and a forward arrow) under the "Dojo". Those arrows call these two methods. *See Figure 7, again.* That is pretty much everything… I know some things have been missed, but most the time how these things work can be deduced by jumping in, and looking around. I am sorry this ended up being so long. I try really hard to make sure I include lots of comments, and documentation, but often I feel it is too much. Let me know what you guys (and gals) are interested in, in the way of documentation.

4.) UI, Menu, and the Canvas

UI… stuffs!

The UI…

Okay, I think that the UI/Canvas Scaling portion of this complete project template will go much better than with version 1.0 of CSC. This project was built for landscape mode only, and I did add canvas scalers to the canvas objects. The canvas scalers are all set to scale with “width”, and the reference resolution has been set to my device resolution, which is 800x480 in landscape. Most android devices will be 15:9, 16:9, or 16:10, in my experiences, and this game should not have any problems with those. So, if for some reason you run into any inappropriate scaling, stretching, or misalignment of UI resources, then you have some work to do!! Lol. Try inputting your own devices reference resolution if you have problems, and make sure that the Canvas Scalers are on.

I will leave it up to you guys to let me know if you have any specific "fit" issues on your various devices. Just let me know, and make sure to tell me your device type, version of unity, and try to capture a screenshot. This kit is not necessarily a "ready to release" template, but rather a fairly complete "vertical slice". It still needs some upgrades, a few additional assets, and some good social/score integration work, but it is close. These kits are normally intended to serve as a starting point, and then the user expands, adapts, and dresses it to their liking. You can do whatever you would like with it, besides the audio files that I have cited sources for, everything else is mine and I am telling you to use it however you wish. Whether you use it for learning, or as a mini-game in a larger experience, or if you strip it apart, and only use a small chunk… you can do anything you want, with one exception. You cannot repack it and sell it as a complete project template… It's a free template for a reason.

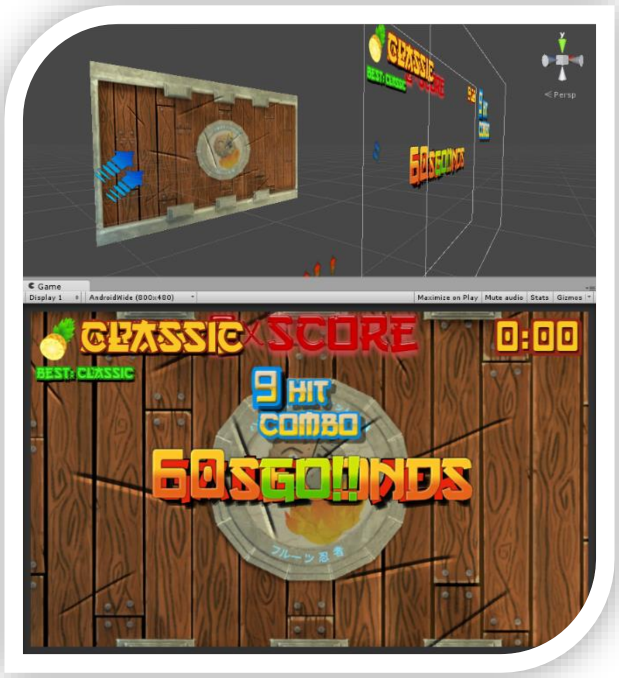


Figure 32 - Scene and Game View

*** Note ***

The Top and Bottom of the “AboutAndReadMe” Document are the same for *“Most”* every project, unless I specifically tag and date some new text (like I did after …”c’est la vie” on the second page), and like I am doing here (***Note***).

This is because the “about me / intro” paragraphs at the top of the page are still true, and very relevant. I may want to add a little message, topic, or point each time. If someone is a first time user and they have downloaded a kit, I want them to have as much information as someone who has downloaded all of the kits. If it seems a little cheap/redundant that is not my intention. I will also put each prior kits Store Listing at the bottom of this document. I don’t do very well with the social media thing ;-) lol. Maybe one of these docs might help me bootstrap another kit or two... who knows.

Best Regards

-Brian

If you haven’t done so already check out Color Switch Clone on the Asset Store…

https://www.assetstore.unity3d.com/en/#!/content/59187As always, I appreciate reviews, everyone has been great about that. It really helps when working through a project. When you can see people that appreciate having some stuff to look through/learn from.

I felt like there could be a few extra free complete project templates on the asset store. Unity keeps pushing out new features and releases, and more and more people are diving in. There are some really good kits, and

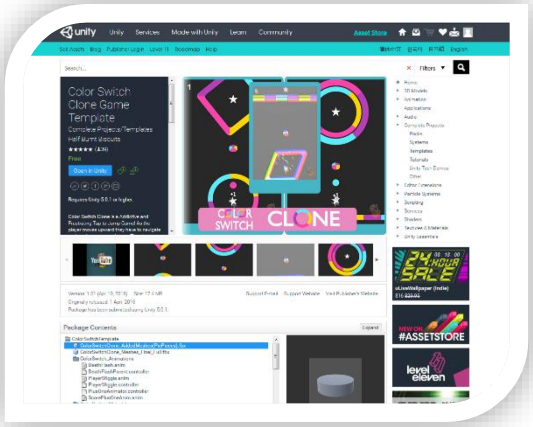


Figure 33 - ColorSwitchClone store page screenshot

lots of good resources on the asset store. Fruit Slicing Game Template on the Asset Store

The Fruit Slicing Game Template's store page…

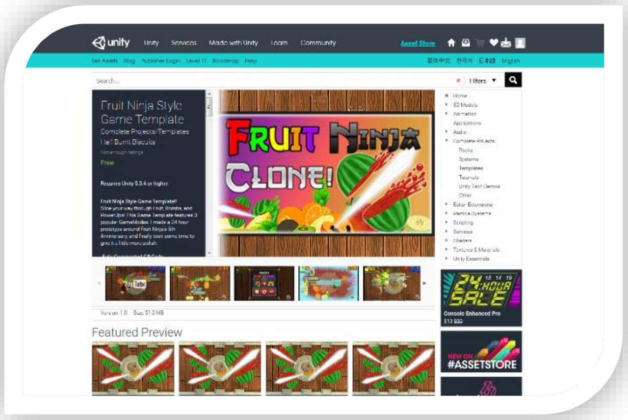
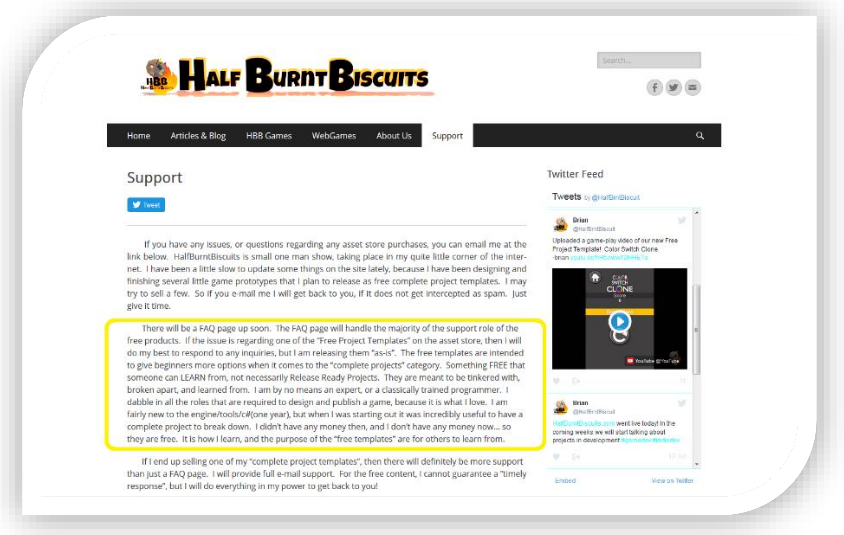


Figure 34 – Fruit Slicing Game Template Store Page

*** End of Note ***

If anyone has any comments, questions, or suggestions you can e-mail me at HalfBurntBiscuits@gmail.com. I will respond as soon as I can. I will eventually get a FAQ page setup, but as a solo developer it takes me time to take care of game design, making the game, and updating the website. The "free" Complete Project Templates are released "as-is", and the majority of the "support" will be provided by the "FAQ".

I will also be selling some of the Complete Project Templates I publish, but they will be super cheap, and e-mail support will be provided. I guess what I am saying is if you send me an e-mail, be patient. As long as the spam filter does not grab the e-mail, I will get back to you. It may however take a few days.



Support

If you have any issues, or questions regarding any asset store purchases, you can email me at the link below. HalfBurntBiscuits is small one man show, taking place in my quite little corner of the internet. I have been a little slow to update some things on the site lately, because I have been designing and finishing several little game prototypes that I plan to release as free complete project templates. I may try to sell a few. So if you e-mail me I will get back to you, if it does not get intercepted as spam. Just give it time.

There will be a FAQ page up soon. The FAQ page will handle the majority of the support role of the free products. If the issue is regarding one of the "Free Project Templates" on the asset store, then I will do my best to respond to any inquiries, but I am releasing them "as-is". The free templates are intended to give beginners more options when it comes to the "complete projects" category. Something FREE that someone can LEARN from, not necessarily Release Ready Projects. They are meant to be tinkered with, broken apart, and learned from. I am by no means an expert, or a classically trained programmer. I dabble in all the roles that are required to design and publish a game, because it is what I love. I am fairly new to the engine/tools/c#(one year), but when I was starting out it was incredibly useful to have a complete project to break down. I didn't have any money then, and I don't have any money now... so they are free. It is how I learn, and the purpose of the "free templates" are for others to learn from.

If I end up selling one of my "complete project templates", then there will definitely be more support than just a FAQ page. I will provide full e-mail support. For the free content, I cannot guarantee a "timely response", but I will do everything in my power to get back to you!

Brian @HalfBrntBiscuit
Uploaded a game-play video of our new Free Project Template! Color Switch Clone -brian youtu.be/[illegible]

Brian @HalfBrntBiscuit
HalfBurntBiscuits.com went live today! In the coming weeks we will start talking about projects in development #gamedev #indiedev

Best Regards

-Brian🚀👤

Contact E-mail:

Brian

HalfBurntBiscuits@gmail.com: Comments and Suggestions are Welcome. This is the best way to contact me, with any issues.

YouTube Video of Fruit Slicing Game in action…

https://youtu.be/KtDrY_213Os

A link to a public Google Drive folder that currently has the copy of the signed android version. I will put some of the asset design documents up soon too. Like the .blend file, some of the other "Mid" progress files. They might be worthwhile/helpful to a few. I want to start putting some of the assets for the kits on the google drive (stuff you don't squeeze into a .unitypackage). I will start at least zipping .blend's, .psd's, and .xcf's. I usually end up being extra cautious when it comes to saving things… Fruit Slicing Game comes from a folder dedicated just to it, and its 8GB/1,000 non-unity files. Just for FA… I.e. The normal/ao maps from the low poly katana were generated from a high poly mesh (2million triangles). The wood, and a few other things were sculpted the same way. When I export to MeshLab / XNormal I use .obj's (can be big files). So, it was all relevant Dev stuff, but now there is not a good place for them, but maybe somebody can tell me if they have any interest in any earlier source files. Just let me know. Just the Apk for now though…

APK:

https://drive.google.com/file/d/0B2k4ZlzalPfReEltSzMxcVhWZ2s/view?usp=sharing

WebGL:

http://www.halfburntbiscuits.com/fruitslicinggame/

DoxyGen Docs(there is also a copy in a zip file within the project):

http://www.halfburntbiscuits.com/DoxyGen_Pages/FruitNinjaStyleGameTemplate/html/index.html

HBB Homepage:

http://www.halfburntbiscuits.com/

**Website: Working on getting some content uploaded. Site has a WebGL Demo of Color Switch Clone, and the Fruit Slicing Game. **

Twitter: New Account. Once some more content is uploaded to the site, I will use this more.

@HalfBrntBiscuit

*The Change-logs are now in the same folder that the rest of the Documentation is in.

- None yet for FSG "Fruit Slicing Game Template"…

****AudioClips for Fruit Slicing Game - Cites and Sources:****

- Game Background Music is "SHAMISEN Slow Mood In D-Minor" ny Jens Egbert. You can check out the loop at : http://www.looperman.com/loops/detail/96855/shamisen-slow-mood-in-d-minor-by-jensmuse-free-80bpm-ethnic-harp-loop OR check out his YouTube Page at https://www.youtube.com/user/jensdab. He has lots of nice work.

- "Sword_cutting_through_air" was used for the "swipe/slash" sounds. The original File contained about 20 seconds of slashes. I cut 11 individual sounds out of it. Check their website at : http://www.freesfx.co.ukAbout The track is half way down the page at listed as "sword_cutting_through_air" at : http://www.freesfx.co.uk/soundeffects/swords-knives/

- Tracks [BloodSquirt, ED-Squishy 2, Squish-1, Metal Gong, Japanese temple bell] are from soundbible.com at: http://soundbible.com/ The original author / Recorded by Mike Koenig. All of these tracks from SoundBible.com is Mike Koenig, and all sounds are under Attribution 3.0 Unported (CC BY 3.0). Full License file is in this files "parent directory". I would like to give special thanks to Mike Koenig... he has a lot of great sounds! Thanks http://soundbible.com/royalty-free-sounds-1.html

- "FreeSfx Pyro – Fuse Burning", by VoxHouse Studio: [Film * Music * Design]}. Source file found at https://www.youtube.com/watch?v=hhl8TE0xfc. All credit goes to the original author, and/or recorder... There is a graphic inside this font's folder that declares this sound available to use for free. Check out the YouTube link, and their website at http://www.VoxHouse.net Thanks, Great fuse burning sound!!