Unity講座 ~ブロック崩し②~

3年 海苔 威

1. 初めに

1-1. オブジェクトをまとめる

- いきなり作業もアレなんで
- ▶ Hierarcyをきれいにしましょう
- ▶ Unityを起動 → 前回のプロジェクトを開く
- Hierarchyを見る
- いろいろ増えて見づらい

🚝 Hierarchy	â -=
Create * Q*All	
Main Camera	
Directional Light	
Wall_left	
Wall_right	
Wall_top	
Wall_bottom	
Ball	
Bar	
GameRoot	

- ► $\lceil Game Object \rfloor \rightarrow \lceil Create Empty \rfloor$
- ▶ 名前は「Wall」
- ▶ 壁を作るオブジェクトを1つにまとめよう
- 作った「Wall」に「Wall_right」「Wall_left」「Wall_top」「Wall_bottom」を入れる

ø

B

E Hierarchy Create ▼ QTAII

Main Camera

Divertienal Liebi

File Edit Assets

+ S 1

▶ Hierarchyがすっきり

T Hierarchy Create ▼ (Q*All	_
Main Camera Directional Light Ball Bar GameRoot	
▶ Wall	

▶ Wallを開けば4つのオブジェクトが見える



GameObject Component Window Help

Create Empty

3D Object

2D Object

Create Empty Child

Unit

Ctrl+Shift+N

Alt+Shift+N

こんな風にまとめられるものはまとめると良い!

2. スコア処理

2-1. タグを使う

- というわけで本題へ
- 前回ブロックに「Block01」などの名前を付けましたが、これ以外にもブロックを特定する方法があります
- ▶ Hierarchyの「Ball」を選択し、Inspectorを見る





- ▶ Untaggedを選び、「Add Tag...」をクリック
- ▶ ↓みたいな画面が出る

0 Inspector	ii •≡
Iags & Layers	10 Q.
▼ Tags	
List is Empty	
	+ -
Sorting Layers	
and the second se	



▶ +を3回押す

Inspector) Lavora		
I ags o	Layers		M Mry
▼ Tags			
Tag 0	New Tag		
Tag 1	New Tag		
Tag 2	New Tag		
		+	
れぞれ	「block1」	[block2]	「block3」と書く



- ▶ タグが出来たのでAssetsにあるブロックで設定する
- 「Block01」のInspectorのTagを「block1」にする
 同様に「Block02」は「block2」
 「Block03」は「block3」とタグ付けする
- ▶ タグ付けの確認をする
- ▶ 「SO2_Block」を開く



プログラムを書く Switch(条件) case(条件1)① case(条件2)② これで条件が条件1の場合①を 条件が条件2の場合②を実行する

▶ つまり衝突した時の物体のタグが

「block1」のときはコンソールに「ブロック1です」と表示する

using UnityEngine;

using System.Collections;

public class S02_Block : MonoBehaviour {

case "block1":

break; case "block2":

break; case "block3":

break;

void OnCollisionEnter(Collision other)

Debug.Log("ブロック1です");

Debug.Log("ブロック2です");

Debug.Log("ブロック3です");

GameObject.Destroy(this.gameObject);

switch (this.gameObject.tag)

▶ ゲームを実行





- ► コンソールが無い人は「Window」→「Console」で開ける
- ▶ ここまで出来たらタグ付け完了

2-2. スコア表示部分

- ▶ スコア加算は置いといて、まず表示部分を作ります
- ► $\lceil Assets \rfloor \rightarrow \lceil Create \rfloor \rightarrow \lceil C\# Script \rfloor$
- ▶ 名前は「SO2_Score」
- ▶ 「GameRoot」にドラッグ&ドロップ
- ▶ ↓のプログラムを書く

```
using UnityEngine;
using System.Collections;
public class S02_Score : MonoBehaviour {
    private int score = 0;
    public GUIStyle gui_score;
    void OnGUI() {
        GUI.Label(new Rect(0, 0, screen.width, 30), "SCORE : " + score, gui_score);
    }
}
```

- ▶ GUIをいじる「GUIStyle」という変数を定義
- 「SCORE: (実際の数値)」という文を表示させる
- ▶ この時、gui_scoreを渡すことでguiをいじれるようになる
- ▶ GameRootのInspectorを見てみよう
- ▶ Inspector内に「GUI_score」という項目が増えている
- ▶ 赤枠内を変更
- ▶ ゲームを実行
- 上の右端にスコアが表示される

6:10		Maximize on Play Mute audio Stats Gizmos
	(10.0, 10.0, 0.0)	SCORE : 0

Normalの中で色を変えれるので好みでどうぞ

Inspector	Hack	<u></u>
Element 0	Blockor	
Element 1	BlockU2	
Element 2	WBIOCK03	0
🖲 🗹 S02_Score ((Script)	💽 🌣,
Script	₢ S02_Score	0
Gui_score		
Name		
▶ Normal		
▶Hover		
▶ Active		
▶ Focused		
▶ On Normal		
▶ On Hover		
▶ On Active		
▶ On Focused		
▶ Border		
▶ Margin		
▶ Padding		
▶ Overflow		
Font	None (Font)	0
Font Size	24	
Font Style	Italic	\$
Alignment	Upper Right	\$
Word Wrap		
Rich Text		
Text Clipping	Overflow	\$
Image Position	Image Left	\$
Content Offset	X 0 Y 0	
Fixed Width	0	
Fixed Height	0	
Stretch Width		
Stretch Height		
A	dd Component	

2-3. スコア加算関数

▶ 衝突時にスコアを加算するための関数を用意

▶ 「SO2_Score」をいじる



- ▶ Additional_scoreはvalueという整数値を受け取り、socreに加算する
- しかし、まだ関数を作っただけで呼ばれない
- どこから呼ぶのか?
- ▶ →衝突時に呼び出す!!
- ▶ というわけで「SO2_Block」を開く
- ▶ 赤枠を変更
- ▶ Additional_scoreを使うために
- ▶ 変数s02_scoreを用意
- ▶ Start関数でs02_scoreを定義
- それぞれのブロックに衝突したとき ブロック1は1、ブロック2は2、 ブロック3は3を引き渡す





こんな感じになるはず



▶ 上画面だとブロック1が4つとブロック2が2つでスコアは10点みたいな

▶ スコア処理はおけ

3. ブロックの詳細設定

3-1. ブロックに体力をつける

- ▶ よくある何度かぶつけて壊れるブロックを作る
- ▶ 「SO2_Block」を開く
- ▶ まずはStart関数を書き換え
- ▶ Int でhpを定義
- Start関数でブロック2だけhpを2に設定 ブロック1とブロック3はhpが1



次はOnCollisionEnterの中身



▶ 衝突時にhp-1をしてhpが0の場合に処理を行う

これでゲームを実行すると真ん中の列のブロックだけ2回衝突しないと壊れない

- ▶ 2回衝突しないと壊れないが、今が何回目かわからない
- →衝突ごとに色を変えよう!
- ▶ 衝突ごとに色が薄くなるようにする
- ▶ 「SO2_Block」を開いて
- ▶ ifにelse文を追加して記述

```
}
GameObject.Destroy(this.gameObject);
}
else
{
GetComponent<Renderer>().material.color *= new Color(1.0f, 1.0f, 1.0f, 0.66f);
}
```

▶ もし衝突時に-1して、hpが0でなければ色の(r,g,b)はそのままで濃さを0.66倍する

▶ hpが2だとつまらないので3にしとこう

void Start()

s02_score = GameObject.Find("GameRoot").GetComponent<S02_Score>();

switch (tag)
{
 case "block2":
 hp = 3;

break:

- ▶ プログラムは以上で保存
- ▶ Block02の設定をいじる
- AssetsのBlock02のInspectorを見る
- ▶ M_Blockの「Shader」をクリック
- ▶ $\lceil \text{Regacy Shaders} \rfloor \rightarrow \lceil \text{Transparent} \rfloor$
 - →「Diffuse」を選択

Inspector Block02	_		Stati	с ч
Tag block2	‡ Lay	er Default	-	-
▼→ Transform	-10	11	[•
Position	X 0	Y 0	Z 0	
Rotation	X 0	Y O	Z 0	
Scale	X 4	Y 1	Z 1	
🖲 Cube (Mesh	Filter)		(
Mesh	Ube 📲			
🔻 🤪 🗹 Box Collider	r.		ĺ,	1
Is Trigger				
Material	None (Phy	sic Material)	
Center	X 0	YO	Z 0	
Size	X 1	Y 1	Z 1	
🔻 🛃 Mesh Rende	rer		(1
Cast Shadows	On			
Receive Shadows				
▶ Materials				
Use Light Probes				
Reflection Probes	Blend Prob	es		
Anchor Override	None (Tra	nsform)		
▼ 🕞 🗹 S02_Block (Script)		(
Script	S02_Blo	ck	.02	
M Block02			6	10
- Shader Sta	andard			
Rendering Mode	Opaque			1

	•= O Inspector		ŵ -
	Materials Use Light Probe	es Bland Drobas	+
-	Anchor Overrid	e None (Transform)	0
		ck (Script)	۵.
2	Script	S02_Block	0
	Bumped Diffuse	p2 🚺	¢,
	Bumped Specular	Standard	•.
	Decal	Standard	
	Diffuse	Standard (Specular setup)	
	Diffuse Datail	FX	
		GUI	,
i -	Diffuse Fast	Mobile	,
	Lightmapped	Nature	
	Parallax Diffuse	Darticles	
	Parallax Specular	Palucies	1
	Reflective +	Skybox	1
	Self-Illumin	Sprites	
	Specular	UI	*
	Transparent +	Bumped Diffuse	*
	VertexLit	Bumped Specular	•
	o Detail Alb	Cutout	
	🗌 🗌 🖂 🛛 Normal M	Diffuse	
	Tilina Block02	Parallax Diffuse	
	BIOCK02	Parallax Specular	
		Specular	
		VertexLit	





- ▶ 真ん中のブロックだけ衝突の度に薄くなって3回目の衝突で破壊
- ▶ スコアも破壊時のみ加算される

▶ できました?

3-2. ブロックを差し替える

▶ ブロックが破壊されたら違うブロックがもう1回出てくる

▶ →面白くない?

- ▶ 「SO2_Block」を開く
- ▶ 赤枠2か所追加
- Block03の破壊時、prefabを同じ場所に配置する
 プログラムは以上
- AssetsのBlock03のInspectorを開く

}public class S02_Block : MonoBehaviour {
 private S02_Score s02_score;
 private int hp;
 public GameObject prefab;

- ▶ M_Block03に「prefab」という項目が追加されている
- ここにAssetsにある「Block01」を追加する
- これでBlock03破壊時のprefab = Block01となる







▶ こんな感じで紫が破壊されると緑が出てくる

4. 効果音を付ける

4-1. アセットストア

- アセットストアは最強
- ▶ Unityで使用できる色々なデータが無料、有料で置いてある
- ▶ $\prec = \neg \square$ [Window] \rightarrow [Asset Store]
- ▶ 画面がある人は直接

▶ 右の画面が出る



n	dow	Help	
	Next Window Previous Window Layouts		Ctrl+Tab
			Ctrl+Shift+Tab
			•
	Serv	vices	Ctrl+0
	Scene Game Inspector Hierarchy		Ctrl+1
			Ctrl+2
			Ctrl+3
			Ctrl+4
	Project		Ctrl+5
	Anin	nation	Ctrl+6
	Profi	iler	Ctrl+7
	Audi	o Mixer	Ctrl+8
	Asse	et Store	Ctrl+9
_			

- ▶ 検索欄に「retro games」と入力し検索
- 「Retro Games Sound FX」をクリック

インポートをクリック



▶ ダウンロード終了を待つ



- ▶ ダウンロードが終わると→の画面が出る
- ここでチェックが入っているものをインポートする
- 今回はこのままで「Import」をクリック

▶ 待つ

▶ 終わるとAssetsにフォルダが追加されている



ここにインポートしたファイルが入っている

▶ これで音のデータは手に入ったので実際に鳴らす

Import Unity Package	×	
Retro Games Sound FX		
🔻 🗹 🚞 Audio	A	
▼ 🗹 🚞 RetroGamesSoundFX		
V 🗹 🚔 Alert	NEW	
Alert01.wav	NEW	
Alert03.wav	NEW	
Alert04.wav	NEW	
🗹 🌞 Alert05.wav	NEW	
🗹 렞 Alert06.wav	NEW	
Alert07.wav	NEW	
Alert08.wav	NEW	
Alert09.wav	NÉW	
Alert10.wav	NEW	
Alerti 2		
All None	Import	

4-2. 衝突時に音を鳴らす

- Main CameraのInspectorを見ると一番下に「Audio Listener」
 というコンポーネントがある
- こいつがシーン中で発生した音を拾う
- 「Audio Listener」は1シーンにつき1つしか置けない(初期状態 ではMain Cameraの中)ため、他のオブジェクトに着けたい場合 main Cameraにある「Audio Listener」は削除する
- 音を鳴らすためのプログラム書きます
- ▶ 「SO2_Ball」を開く

	Inspector		â +=
	👕 🗹 Main Camera	3	🗌 Static 👻
	Tag MainCamera	‡ Layer Default	\$
	▼人 Transform		ې 🕼
	Position	X 0 Y 14	Z -25
	Rotation	X 0 Y 0	Z 0
	Scale	X 1 Y 1	Z 1
	🔻 🖶 🗹 Camera		ې 🗊
	Clear Flags	Skybox	÷
	Background	_	<i>1</i> /
	Culling Mask	Everything	\$
	Projection	Perspective	\$
	Field of View		60
	Clipping Planes	Near 0.3	
		Far 1000	
	Viewport Rect	X 0 Y 0	
		W 1 H 1	
	Depth	-1	
	Rendering Path	Use Player Settings	÷
	Target Texture	None (Render Texture)	c
	Occlusion Culling		
	HDR		
	🖵 🗹 GUI Layer		ې 💭
/	💣 🗹 Flare Layer		🔯 🌣
	🙆 🗹 Audio Listen	ier	ې 💭

Add Component

▶ 以下のように付け足す



- ▶ SudioSourceは音のデータとループするかの情報を持つ
- ▶ AudioClipは使用する音をInspectorで設定するために使う
- OnCollisionEnterにより衝突時に音が鳴る

上書き保存

- ▶ BallのInspectorを開く
- ▶ 右のSoundという項目が追加されているはず
- $\blacktriangleright \quad [Assets] \rightarrow [Audio] \rightarrow [RetroGamesSoundFX]$
 - \rightarrow [Hit] \rightarrow [Hit4]



▶ の「Hit4」をBallの「Sound」までドラッグ&ドロップ



- ▶ ゲーム実行
- 音が出てますか?

4-3. BGMを付ける

- ▶ BGMは衝突時に鳴る!みたいな条件が無いのでやり方を変えます
- ► $\lceil GameObject \rfloor \rightarrow \lceil Audio \rfloor \rightarrow \lceil Audio Source \rfloor$
- Audio SourceのInspectorを見る
- 「Audio Clip」に使いたいBGMをドラッグ&ドロップし
 「Loop」にチェックを入れる



Ga	meObject Component	Window Help	
	Create Empty Create Empty Child 3D Object 2D Object	Ctrl+Shift+N Alt+Shift+N	
	Light	•	
	Audio	•	Audio Source
	UI	•	Audio Reverb Zone

- ▶ ゲーム実行
- 設定したBGMが聞こえればおけ
- 音量が大きすぎる場合はInspectorの
 Volumeを下げよう
- 他にも値をいじれるので気になるところは 修正すればいいんじゃないかな?

BGMを付けたい人は付けてください別につけなくてもいいです

	udio Source						Static	•
Tag 🛛	Jntagged		÷	Layer	Defa	ult		\$
🙏 ті	ansform	114		H.:	<i></i>	- 15 /		\$,
Position		X	D	Y	14	Z	-25	U.
Rotation	š.	X	2	Y	0	Z	0	
Scale		X	1	Y	1	Z	1	
🛋 🗹 A	udio Source	e						\$,
AudioCl	ip I	-	Adve	nture	Them	e-Looj	p	0
Output		No	ne (A	Audio	Mixer	Group)	0
Mute								-
Bypass	Effects							
Bypass	Listener Effe							
Bypass	Reverb Zone							
Play On	Awake							
Loop								
Priority				-0		Low	128	
Volume		-0	<u> </u>			_	0.07	4
Pitch		-			0		1	- 0
	Pan	_	~	-0		Right	0	IJ
Stereo I	en.	Lef	t					
Stereo I Spatial I	Blend	Lef	t			20	0	
Stereo I Spatial I Reverb	Blend Zone Mix	Lef 2D	t.			3D	0	

5. ゲームクリアの処理

5-1. ゲームクリアの条件設定

> ブロック崩しのゲームクリアは?
 →全てのブロックを壊す

▶ 判定方法は色々...

今回は、ブロックが1つ破壊されたときシーン上にブロックがあるか調べる

5-2. ブロックの有無を調べる

- 前回設定したタグを使って調べる
- ▶ 「SO2_Ball」を開く

```
void OnCollisionEnter()
{
    audioSource.Play();
    checkAllBlocks();
}
```

```
bool gameClear;
void checkAllBlocks()
{
    if ((GameObject.FindWithTag("block1") == null) && (GameObject.FindWithTag("block2") == null) && (GameObject.FindWithTag("block3") == null))
    {
        gameClear = true;
        transform.GetComponent<Rigidbody>().velocity = Vector3.zero;
    }
}
```

```
public GUIStyle gui_gameClear;
void OnGUI()
{
    GUI.Label(new Rect(0,0,200,20),""+transform.GetComponent<Rigidbody>().velocity);
    if (gameClear)
    {
        GUI.Label(new Rect(0, 0, Screen.width, Screen.height), "GameClear", gui_gameClear);
    }
}
```

- ▶ 衝突時にcheckAllBlocksという関数を呼ぶ
- ▶ checkAllBlocksはFindWithTag("タグ名")という関数を使う。

これはゲームオブジェクトに"タグ名"というタグがある場合trueを、ない場合 falseを返す。

つまり衝突時に「block1」「block2」「block3」のタグを持つオブジェクトが1 つも存在しない場合ifの条件を満たす。

▶ ifの中身 … ゲームクリアのフラグgameClearをtrueにする。 ボールの速度を0にする。(停止させる)

- ▶ OnGUI関数ではgameClearがtrueの場合のみ「GameClear」と画面に表示する
- つまり、通常はボールの速度を左上に表示するだけだがクリア時には 「gameClear」を表示するようになる。
- ▶ 新たにGUIのスタイルを定義したのでボールのInspectorを見る

🔻 健 🗹 S02_Ball (Script)		0
Script	≥ S02_Ball	e
Sound	Hit4	0
▶ Gui_game Clea		

- S02_Ballの中のGui_game Clearを見る
- ▶ 赤枠を変更
- テキストカラーは自由でいいかも
- これが出来たらゲーム実行といきたいが ちゃんと動くか確かめるために全てのブロックが 破壊されるのを待つのはキツイ!

Inspector Freeze Position X Y ✓ Freeze Rotation ✓ Y ✓ ✓ Soz_Ball (Script) ✓ Script Sound ✓ Hatta Outgame Clear Name ✓ Normal Background None (Texture 2D) ○ Text Color ✓ Hover Active Focused On Normal On Active On Focused Border Margin Padding Overflow Font None (Font) Font Size 50 Font Size 50 Font Size So Font Size <th></th> <th></th> <th></th>			
Freeze Position X Y ✓Z Freeze Rotation ✓X ✓Y ✓Z Image Rotation ✓X ✓Y ✓Z Image Sold Sold Image Normal Image Normal Background None (Texture 2D) Image Normal Background None (Texture 2D) Image Normal Background None (Texture 2D) Image Normal Image Normal Image Normal Image Normal Image Normal Image Normal Image Normal Image Normal Image Normal Image Normal Image Position Image Left Image Normal Image Rostion Image Normal Image Normal Image Rostion Image No	Inspector	<u>a</u>	-
Freeze Rotation Image Normal Sound Image Hit4 Gui_game Clear Name Normal Background None (Texture 2D) Image Clear Name Normal Background None (Texture 2D) Image Clear Normal Background None (Texture 2D) Image Clear Name Image Clear Name Image Position Image Position Image Left Image Position Image Position Image Position Image Left Image Position Image Position Image Left Image Position	Freeze Position	□x □y v z	
Image: Sole of the second	Freeze Rotation	✓X ✓Y ✓Z	
Script © S02_Ball Sound Hit4 Gui_game Clear Name Name Normal Background None (Texture 2D) O Text Color Hover Active Focused On Normal On Hover On Active On Focused Border Margin Padding Overflow Font Size Font Size S0 Font Style Normal # Alignment Middle Center ‡ Mord Wrap Rich Text ✓ Text Clipping Overflow ‡ Image Position Image Left ‡ Content Offset × Ø Y Fixed Height O Stretch Width	🕞 🗹 S02_Ball (Sc	ript) 🔯 🖏	
Sound Hit4 ○ Gui_game Clear Name Name Image Clear Active Image Clear Active Image Clear Normal Image Clear Normal Image Clear On Normal Image Clear On Normal Image Clear On Active Image Clear On Focused Image Clear Border Image Clear Margin Image Clear Padding Image Clear Overflow Image Clear Font Style Image Clear Nore (Font) Image Clear Image Position Image Left Image Position Image Left Fixed Width Image Clear Fixed Height Image Clear Stretch Width Image Clear Stretch Height Image Clear	Script	€ S02_Ball ©	
Gui_game Clear Name Normal Background None (Texture 2D) ○ Text Color Hover Active Focused On Normal On Hover On Active On Focused Border Margin Padding Overflow Font None (Font) ○ Font Size 50 Font Style Normal ‡ Alignment Middle Center ‡ Word Wrap Rich Text ✓ Text Clipping Overflow ‡ Image Position Image Left ‡ Content Offset × 0 Y 0 Fixed Height 0 Stretch Width ✓ Stretch Height □	Sound	₩Hit4 ©	
Name Normal Background None (Texture 2D) Text Color Hover Active Focused On Normal On Hover On Active On Focused Border Margin Padding Overflow Font Normal Font Size 50 Font Style Normal # Alignment Middle Center # Word Wrap Rich Text Vort Orgen Fixed Width 0 Fixed Width 0 Fixed Height Stretch Width	Gui_game Clear		
Normal Background None (Texture 2D) ▼ Text Color Hover Active Focused On Normal On Hover On Active On Focused Border Margin Padding Overflow Font Normal Font Size S0 Font Size S0 Font Style Normal ↓ Alignment Middle Center ↓ Word Wrap Rich Text ✓ Text Clipping Overflow ↓ Image Position Image Left ↓ Fixed Width 0 Fixed Height Stretch Width	Name		$ \rangle$
Background None (Texture 2D) Text Color Hover Active Focused On Normal On Hover On Active On Focused Border Margin Padding Overflow Font None (Font) Font Size 50 Font Style Normal + Alignment Middle Center + Word Wrap Rich Text ✓ Text Clipping Overflow + Image Position Image Left + Content Offset X Y Fixed Width O Fixed Height Stretch Width	▼ Normal		
Text Color Hover Active Focused On Normal On Hover On Active On Focused Border Margin Padding Overflow Font None (Font) Font Size 50 Font Size Stretch Uidth 0 Y Fixed Height Stretch Height	Background	None (Texture 2D) O	
 Hover Active Focused On Normal On Hover On Active On Focused Border Margin Padding Overflow Font None (Font) Font Size S0 Font Style Normal Alignment Middle Center Word Wrap Rich Text M Text Clipping Overflow Fixed Width O Fixed Width Stretch Width Stretch Height 	Text Color	P	
 Active Focused On Normal On Hover On Active On Focused Border Margin Padding Overflow Font None (Font) Font Size S0 Font Style Normal Normal Alignment Middle Center Word Wrap Rich Text Text Clipping Overflow Image Position Image Left Image Left Fixed Width Stretch Width 	▶Hover		
 Focused On Normal On Hover On Active On Focused Border Margin Padding Overflow Font None (Font) ○ Font Size 50 Font Style Normal + Alignment Middle Center + Word Wrap Rich Text ✓ Text Clipping Overflow + Image Position Image Left + Content Offset × 0 Y 0 Fixed Width 0 Fixed Height 0 Stretch Width ✓ 	▶ Active		
 On Normal On Hover On Active On Focused Border Margin Padding Overflow Font None (Font) o Font Size 50 Font Style Normal ‡ Alignment Middle Center ‡ Word Wrap Rich Text of the style overflow ‡ Image Position Image Left ‡ Content Offset × 0 Y 0 Fixed Width 0 Fixed Height 0 Stretch Width of 	▶ Focused		
 On Hover On Active On Focused Border Margin Padding Overflow Font None (Font) ○ Font Size 50 Font Size 50 Font Style Normal ‡ Alignment Middle Center ‡ Word Wrap Rich Text ☑ Text Clipping Overflow ‡ Image Position Image Left ‡ Content Offset X 0 Y 0 Fixed Width 0 Fixed Height 0 Stretch Width ☑ 	▶ On Normal		
 > On Active > On Focused > Border > Margin > Padding > Overflow > Font None (Font) > Font Size > 50 > Font Style > Normal + Alignment > Middle Center + > Word Wrap > Rich Text > M > Text Clipping > Overflow + > Image Position > Image Left + > Fixed Width > Stretch Width > Stretch Height > Stretch Height 	▶ On Hover		
 On Focused Border Margin Padding Overflow Font None (Font) ○ Font Size 50 Font Style Normal ‡ Alignment Middle Center ‡ Word Wrap Rich Text ✓ Text Clipping Overflow ‡ Image Position Image Left ‡ Content Offset × 0 Y 0 Fixed Width 0 Fixed Height 0 Stretch Width ✓ 	▶ On Active		
 Border Margin Padding Overflow Font Size 50 Font Size 50 Font Style Normal ‡ Alignment Middle Center ‡ Word Wrap Rich Text ✓ Text Clipping Overflow ‡ Image Position Image Left ‡ Content Offset X0 Y0 Fixed Width 0 Fixed Height 0 Stretch Width ✓ 	▶ On Focused		
 Margin Padding Overflow Font None (Font) ○ Font Size 50 Font Style Normal + Alignment Middle Center + Word Wrap Rich Text ✓ Text Clipping Overflow + Image Position Image Left + Content Offset × 0 Y 0 Fixed Width 0 Fixed Height 0 Stretch Width ✓ 	▶ Border		
 ▶ Padding ▶ Overflow Font Size 50 Font Style Normal + Alignment Middle Center + Word Wrap Rich Text ✓ Text Clipping Overflow + Image Position Image Left + Content Offset × 0 Y 0 Fixed Width 0 Fixed Height 0 Stretch Width ✓ 	▶ Margin		
► Overflow Font None (Font) Font Size 50 Font Style Normal Font Style Normal Alignment Middle Center Word Wrap Rich Text Rich Text ✓ Text Clipping Overflow Image Position Image Left Fixed Width 0 Fixed Height 0 Stretch Width ✓	▶ Padding		
Font None (Font) Font Size 50 Font Style Normal Font Style Normal Alignment Middle Center Word Wrap Image Center Rich Text Image Content Offset Text Clipping Overflow Image Position Image Left Fixed Width O Fixed Height O Stretch Width Image Stretch Height	▶ Overflow		
Font Size 50 Font Style Normal Alignment Middle Center Word Wrap	Font	None (Font) O	
Font Style Normal ‡ Alignment Middle Center ‡ Word Wrap	Font Size	50	
Alignment Middle Center + Word Wrap	Font Style	Normal ‡	
Word Wrap Rich Text Text Clipping Overflow Image Position Image Left + Content Offset × 0 Fixed Width 0 Fixed Height 0 Stretch Width ✓	Alignment	Middle Center +	
Rich Text Image Position Text Clipping Overflow Image Position Image Left Content Offset X 0 Y 0 Fixed Width Fixed Height Stretch Width Stretch Height	Word Wrap		
Text Clipping Overflow ‡ Image Position Image Left ‡ Content Offset X 0 Y 0 Fixed Width 0 § Fixed Height 0 § Stretch Width ✓ § Stretch Height □ §	Rich Text		
Image Position Image Left ≠ Content Offset X 0 Y 0 Fixed Width 0 Fixed Height 0 Stretch Width ✓ Stretch Height □	Text Clipping	Overflow \$	
Content Offset X 0 Y 0 Fixed Width 0 Fixed Height 0 Stretch Width ✓ Stretch Height □	Image Position	Image Left +	
Fixed Width 0 Fixed Height 0 Stretch Width Image: Comparison of the stretch Height	Content Offset	X 0 Y 0	
Fixed Height 0 Stretch Width ☑ Stretch Height □	Fixed Width	0	
Stretch Width 🗹 Stretch Height 🗌	Fixed Height	0	
Stretch Height	Stretch Width		
	Stretch Height		

5-3. ラクにデバッグ?

- Hierarchyの「GameRoot」のInspectorを見る
- ▶ 「SO2_Stage」というスクリプトがある
- こいつがステージのブロックを生成する
 - →「S02_Stage」を読まなければブロック生成されない
 - →ゲーム実行後1回目の衝突でクリア判定になる!!
- どうすればいいか?
- ▶ GameRootのInspector「SO2_Stage」のチェックボックスを外す
- ▶ これでゲーム実行時に読み込まれなくなる







こんな感じになりましたか?

- ・最初にブロックが1つも生成されない
- ・1回目の衝突時に「GameClear」が表示される
- ・ボールが衝突した場所で停止する

▶ ちゃんと動いていたらGameRootの「SO2_Stage」にチェックを入れ直しましょう

6. ステージ追加

6-1. クリア処理の移動

- ▶ ステージを追加する前に…
- ゲームクリア処理がボールで行われるのはおかしい!
- ▶ クリア処理のプログラムを「SO2_Stage」に移す
- ▶ 「S02_Stage」と「S02_Ball」両方書き直す。まずは「S02_Stage」から

▶ 赤枠変更(次ページに続く)

□ pub	<pre>lic class S02_Stage : MonoBehavi public GameObject[] prefab; private bool gameClear = false; public GUIStyle gui_gameClear;</pre>	lour	{
	<pre>void Start() { blockSetting(); }</pre>		

```
デバッグ用にコメントアウトしておく
```

```
private void blockSetting()
    GameObject block right1 = GameObject.Instantiate(prefab[0]) as GameObject;
                                                                                18.0f にしてください
    block right1.transform.position = new Vector3(3.0f + (0), 19.0f, 0.0f);
   /* for (int i = 0; i < 4; i++)
        GameObject block right1 = GameObject.Instantiate(prefab[0]) as GameObject;
        GameObject block right2 = GameObject.Instantiate(prefab[1]) as GameObject;
        GameObject block_right3 = GameObject.Instantiate(prefab[2]) as GameObject;
        GameObject block left1 = GameObject.Instantiate(prefab[0]) as GameObject;
        GameObject block left2 = GameObject.Instantiate(prefab[1]) as GameObject;
        GameObject block_left3 = GameObject.Instantiate(prefab[2]) as GameObject;
        block right1.transform.position = new Vector3(3.0f + (5 * i), 19.0f, 0.0f);
        block_right2.transform.position = new Vector3(3.0f + (5 * i), 21.0f, 0.0f);
        block right3.transform.position = new Vector3(3.0f + (5 * i), 23.0f, 0.0f);
        block left1.transform.position = new Vector3(-3.0f - (5 * i), 19.0f, 0.0f);
        block_left2.transform.position = new Vector3(-3.0f - (5 * i), 21.0f, 0.0f);
        block left3.transform.position = new Vector3(-3.0f - (5 * i), 23.0f, 0.0f);
    (*/
public bool checkAllBlocks()
   if ((GameObject.FindWithTag("block1") == null) && (GameObject.FindWithTag("block2") == null) && (GameObject.FindWithTag("block3") == null))
        gameClear = true;
   return gameClear;
void OnGUI()
   if (gameClear)
       GUI.Label(new Rect(0, 0, Screen.width, Screen.height), "GameClear", gui gameClear);
```

▶ 次は「SO2_Ball」

```
using UnityEngine;
 using System.Collections;
□ public class S02_Ball : MonoBehaviour {
     private AudioSource audioSource;
     public AudioClip sound;
     private S02 Stage s02 Stage;
     void Start () {
         audioSource = gameObject.AddComponent<AudioSource>();
         audioSource.clip = sound;
         audioSource.loop = false;
         s02_Stage = GameObject.Find("GameRoot").GetComponent<S02_Stage>();
         transform.GetComponent<Rigidbody>().velocity = new Vector3(10.0f, 10.0f, 0.0f);
     void OnCollisionEnter()
         audioSource.Play();
         if (s02_Stage.checkAllBlocks())
             transform.GetComponent<Rigidbody>().velocity = Vector3.zero;
     void OnGUI()
         GUI.Label(new Rect(0, 0, 200,20),""+transform.GetComponent<Rigidbody>().velocity);
```

- ▶ GUIをいじる部分がBallからGameRootに移るので再設定
- ▶ GameRootのInspectorを開き赤枠をいじる
- ▶ 5でチェックボックスが空のままの人は入れ直す



- ▶ ゲーム実行
- 真ん中にあるブロックが破壊されたらクリアになる

Lionione L	
▼Gui_game Clear	
Name	
▼ Normal	
Background	None (Texture 2D) 💿
Text Color	¥
►Hover	
▶ Active	
▶ Focused	
▶ On Normal	
▶ On Hover	
▶ On Active	
▶ On Focused	
▶ Border	
🕨 Margin	
▶ Padding	
▶ Overflow	
Font	None (Font) O
Font Size	50
Font Style	Normal +
Alignment	Middle Center 🗧
Word Wrap	
Rich Text	
Text Clipping	Overflow +
Image Position	Image Left +
Content Offset	X 0 Y 0
Fixed Width	0
Fixed Height	0
Churchele Wildele	

6-2. ステージ2の準備

- 1からステージ2を作るのはめんどくさい
- 今のステージをコピペしよう!
- ► $[File] \rightarrow [Save Scene as...]$
- ▶ 名前は「stage2」
- シーン遷移のためにビルドセッティングをする
- ▶ 「File」→「Build&Setting」を開き、Scenes In Buildに「stage2」を追加
- FBuild&Setting」を閉じる

Scenes In Build	
🗹 title	0
✓ stage	1
☑ stage2	2



6-3. スコア引継ぎ

シーンごとに変数を管理するためステージ1と2では別のスコアとなる

Iのスコアに2のスコアを足していくにはどうするか?

→ゲーム起動時に作成され終了まで保持され続ける変数を使う

- ▶ 「SO2_Score」を開く
- ▶ 赤枠追加



- ▶ staticという1単語追加するだけ
- ※staticは多用しない

6-4. ステージ2の作成

- ▶ 土台はステージ1のコピペにより完成している
- あとはプログラムをちょっと書き直すだけ
- ▶ 「SO2_Stage」を開く
- ▶ 赤枠変更
- 何ステージ目かを保持する変数stage_noを用意
 stage_noによってブロックの配置を変更(後で設定)
 stage_noが用意したステージの数を超える場合 遷移しないようにする



▶ ゲーム実行

▶ 1度目のゲームクリア時に左クリック → ステージ2に進む

▶ 2度目のゲームクリア時に左クリック → 何も起きない

▶ 以上が確認できれば大丈夫

ここまで出来たらステージ2のブロック生成は置いといて
 先にエンディングを作ります

- ▶ $[File] \rightarrow [Save Scene as...]$
- ▶ 名前は「ending」
- ▶ 「ending」シーンを開く

- ▶ 開いたらHierarchyから「Ball」「Bar」「Wall」を削除
- ▶ 「GameRoot」のInspectorを開き「SO2_Stage」を削除





- エンディング用のプログラムを書く
- ► $\lceil Assets \rfloor \rightarrow \lceil Create \rfloor \rightarrow \lceil C\# Script \rfloor$
- ▶ 名前は「SO3_ending」
- できたらHierarchyの「GameRoot」にドラッグ&ドロップ

プログラムを書く

Eusing UnityEngine; using System.Collections; using UnityEngine.SceneManagement; Public class S03_ending : MonoBehaviour { private float step_timer = 0.0f; // Update is called once per frame void Update () { step_timer += Time.deltaTime; if (step_timer > 5.0f) { SceneManager.LoadScene(0); } } }

- ▶ 秒数をカウントする変数を用意
- ▶ 5秒経ったら0番目のシーン(title)に遷移
- 出来たら保存
- ▶ Unityの画面に戻り、エンディング用のテキストを書く
- ▶ $\lceil GameObject \rfloor \rightarrow \lceil UI \rfloor \rightarrow \lceil Text \rfloor$
- ▶ Inspectorの赤枠変更

• Inspector			 →	
📬 🗹 Text			🗌 Static 🔻	
Tag Untagged	‡ Laye	er UI	\$	
Rect Transfor	rm		a .	
center	Pos X	Pos Y	Pos Z	
	0	0	0	
	Width	Height		
	300	50		
▼ Anchors			_	
Min	X 0.5	Y 0.5	-	
Piwet		Y 0.5	-	
Pivot	V 0.5	1 0.5		
Rotation	X 0	Y 0	Z 0	
Scale	X 1	Υ 1	Z 1	
O Canvas Rend	erer		💽 🌣,	
🔻 🕇 🗹 Text (Script)		_	💽 🔅,	
Text				
All Stage Clear!				
Character				
Font	🛯 Arial		0	
Font Style	Normal		\$	
Font Size	30			
Line Spacing	1			
Rich Text 🗹				
Paragraph				
Alignment	EEE			
Align By Geometr				

- ▶ 「File」→「Build&Setting」でScenes In Buildに「ending」を追加
- ▶ endingへの遷移をプログラムに追加
- ▶ 「SO2_Stage」を開く



- ここまでできたらtitleシーンを開いてからゲーム実行
- ▶ ※stageのInspector「SO2_Stgae」にチェックが入っているか確認



前ページの動作がちゃんとできた人はステージ2のブロック生成をやる。

▶ 「SO2_Stage」を開く

ι

void Start()

switch (stage no)

case 2:

break;

break;

blockSetting1();

blockSetting2();

case

```
private void blockSetting1()
```

GameObject block right1 = GameObject.Instantiate(prefab[0]) as GameObject; block right1.transform.position = new Vector3(18.0f , 19.0f, 0.0f); /*for (int i = 0; i < 4; i++)</pre> GameObject block right1 = GameObject.Instantiate(prefab[0]) as GameObject; GameObject block right2 = GameObject.Instantiate(prefab[1]) as GameObject; GameObject block right3 = GameObject.Instantiate(prefab[2]) as GameObject; GameObject block left1 = GameObject.Instantiate(prefab[0]) as GameObject; GameObject block left2 = GameObject.Instantiate(prefab[1]) as GameObject; GameObject block left3 = GameObject.Instantiate(prefab[2]) as GameObject; block right1.transform.position = new Vector3(3.0f + (5 * i), 19.0f, 0.0f); block right2.transform.position = new Vector3(3.0f + (5 * i), 21.0f, 0.0f); block right3.transform.position = new Vector3(3.0f + (5 * i), 23.0f, 0.0f); block left1.transform.position = new Vector3(-3.0f - (5 * i), 19.0f, 0.0f); block left2.transform.position = new Vector3(-3.0f - (5 * i), 21.0f, 0.0f); block left3.transform.position = new Vector3(-3.0f - (5 * i), 23.0f, 0.0f); }*/

private void blockSetting2()

```
for (int i = 0; i < 3; i++){
    for(int j = 0; j < 4; j++)</pre>
```

GameObject block_r = GameObject.Instantiate(prefab[i]) as GameObject; GameObject block_l = GameObject.Instantiate(prefab[i]) as GameObject;

block_r.transform.position = new Vector3(3.0f + (5 * i) + (2 * j), 17.0f + (2 * j), 0.0f); block_l.transform.position = new Vector3(-3.0f - (5 * i) - (2 * j), 17.0f + (2 * j), 0.0f);

- blockSetting1のコメントアウトはそのままでtitleシーンから実行
- ステージ2に進んだ時にブロックの配置がこんなんならおっけー



 ここまで出来た人はblockSetting1のコメントアウトを戻して 上の2行を削除





- ▶ 遊んでる人は気づいたと思いますがボールが左右にしか動かなくなることがある →初速と物理法則のみで動くため変な計算されると終わる yoid OnCollisionEnter()
- ▶ 衝突ごとに速度を設定してあげればいい
- ▶ 「SO2_Ball」に赤枠を追加
- これで衝突時に速度を再設定するので 常に10か-10の速度を持つ

audioSource.Play();

```
Vector3 v = transform.GetComponent<Rigidbody>().velocity;
if (v.x < 0.0f)
   v.x = -10.0f;
else
   v.x = 10.0f;
if (v.y < 0.0f)
    v.y = -10.0f;
else
   v.y = 10.0f;
transform.GetComponent<Rigidbody>().velocity = v;
if (s02_Stage.checkAllBlocks())
   transform.GetComponent<Rigidbody>().velocity = Vector3.zero;
```



次回は

- ・クリア時の初期化
- ・挙動改善
- ・アイテム追加?
- etc...
- よりゲームっぽいことを付け足したいと思います
- ▶ 質問がある人は連絡ください

お疲れ様でした