Unity講座③

2018/7/2(月)

ゲームっぽい要素

こんな感じの要素を追加

- バーの加速
- 貫通玉
- 玉の複製

バー加速

バー加速

スペースキーを押してる時のみ
バーの速度を上げる
Barスクリプトを開き
右のようにする

```
public class Bar : MonoBehaviour {
    float[] limit_area = new float[2];
    float bar_width;
    float default_speed = 1.0f;//通常時の速さ
    float acceleration_speed = 2.0f;//加速時の速さ


// Update is called once per frame
void Update () {
        float speed;
        if (Input.GetKey(KeyCode.Space))
        {
            speed = acceleration_speed;
        }
        else {
            speed = default_speed;
        }

        if (Input.GetKey(KeyCode.D) || Input.GetKey(KeyCode.RightArrow)) {
            transform.position += new Vector3(10, 0, 0) * speed * Time.deltaTime;
        }
        else if (Input.GetKey(KeyCode.A) || Input.GetKey(KeyCode.LeftArrow))
        {
            transform.position += new Vector3(-10, 0, 0) * speed * Time.deltaTime;
        }
        limit_area_over();
    }
}
```

貫通玉

アイテムオブジェクトを作る

1. Cubeオブジェクトを作り、名前を「item1」にする
2. 座標、大きさを図1のようにする(適当に色も付けとく)
3. BoxColliderのIsTriggerにチェックを入れる(図2)
4. Item1にRigidbodyを追加し図3のようにする

図1

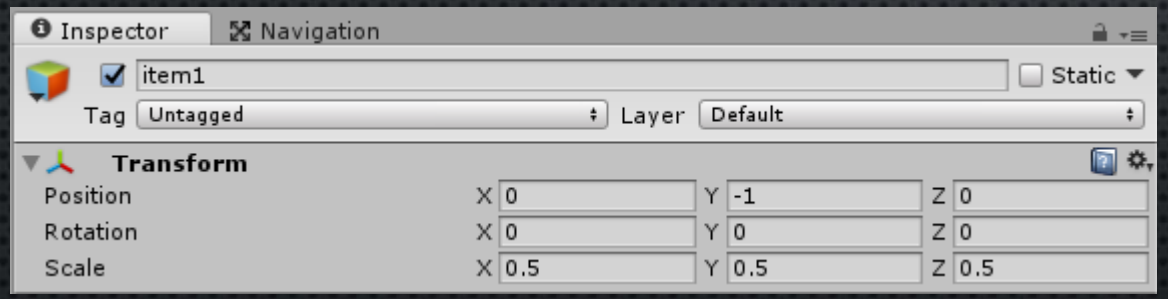


図2

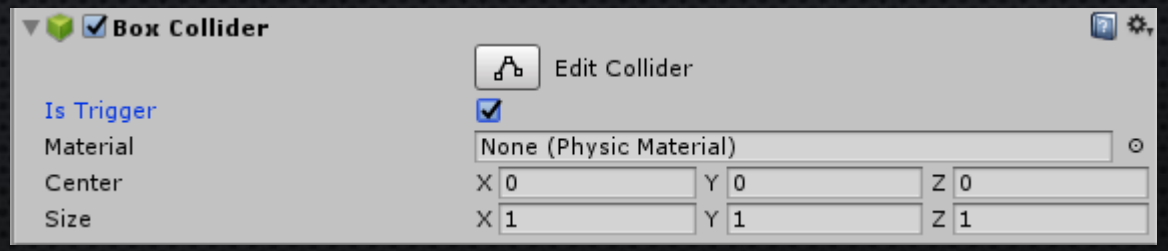
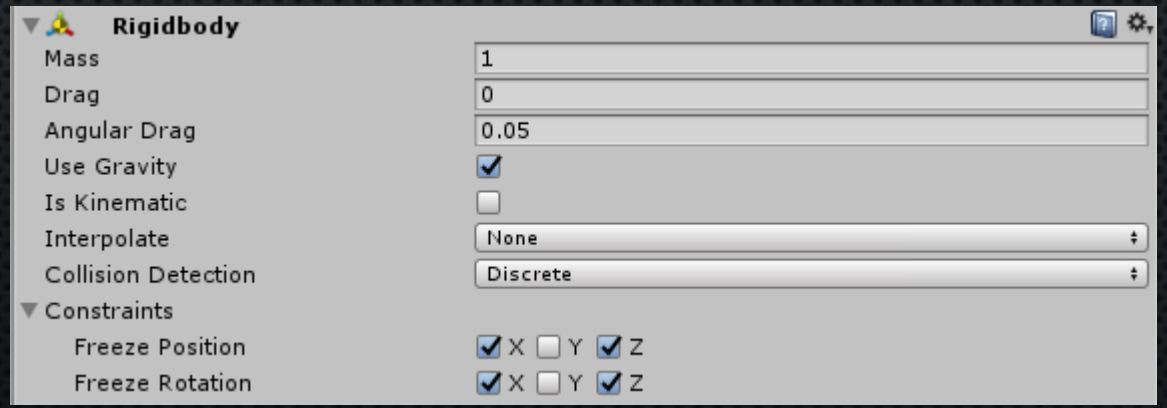
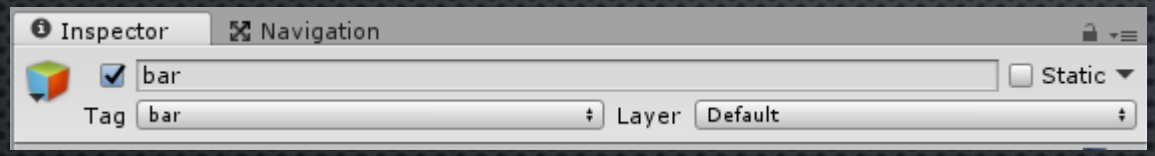


図3



barタグを生成

1. 前回教えた方法で「bar」という名前のタグを追加し、
   barオブジェクトのタグを変更する



C#スクリプト Itemを作る
Item1にアタッチ
コードを以下のようにする(後々いろいろ追加する予定)

transform.Rotate(Vector3)
現在のオブジェクトのRotationに引数の値を足している
(つまりRotationが引数の値になるわけではない！)
今回は1秒間1回転するようにしている
(Time.deltaTimeの機能を思い出そう！)

void OnTriggerEnter(Collider)
void OnCollisionEnter()のIsTriggerバージョン
自身または当たったオブジェクトのIsTriggerがtrueの場合
呼び出される
OnCollisionの引数はCollisionだが
OnTriggerの引数はColliderなので注意
(間違えるとうまく起動しません！)

```
public class Item : MonoBehaviour {

    void OnTriggerEnter(Collider other) {
        if (other.gameObject.tag == "bar")
        {
            if (this.gameObject.name == "item1")
            {
                Destroy(this.gameObject);
            }
        }
        else if (other.gameObject.name == "wall_bottom") {
            Destroy(this.gameObject);
        }
    }

// Use this for initialization
void Start () {

}

// Update is called once per frame
void Update () {
        transform.Rotate(new Vector3(0,360,0) * Time.deltaTime);

}
}
```

最後にプレハブ化させてアイテム完成！！

Hierarchyにあるitem1を消すのを忘れずに

ブロックからアイテムを出すようにする

- ブロックが破壊された際、アイテムが出るようにする

1. Blockスクリプトを開き次のように変更
2. 各block_1,2,3のBlockスクリプトのitemsにitem1をドラッグ＆ドロップ

```
public class Block : MonoBehaviour {
    private int hp;
    public GameObject[] items;
```

```
void OnCollisionEnter(Collision other)
    {
        hp -= 1;
        if (hp == 0)
        {
            int rnd = Random.Range(0, 100);
            if (rnd < 30) {
                GameObject item = Instantiate(items[0], transform.position, Quaternion.identity);
                item.name = items[0].name;
            }
            Destroy(this.gameObject);
        }
        ～省略～
}

void OnTriggerEnter(Collider other)
    {
        hp -= 1;
        if (hp == 0)
        {
            int rnd = Random.Range(0, 100);
            if (rnd < 30)
            {
                GameObject item = Instantiate(items[0], transform.position, Quaternion.identity);
                item.name = items[0].name;
            }
            Destroy(this.gameObject);
        }
        ～省略～

}
```

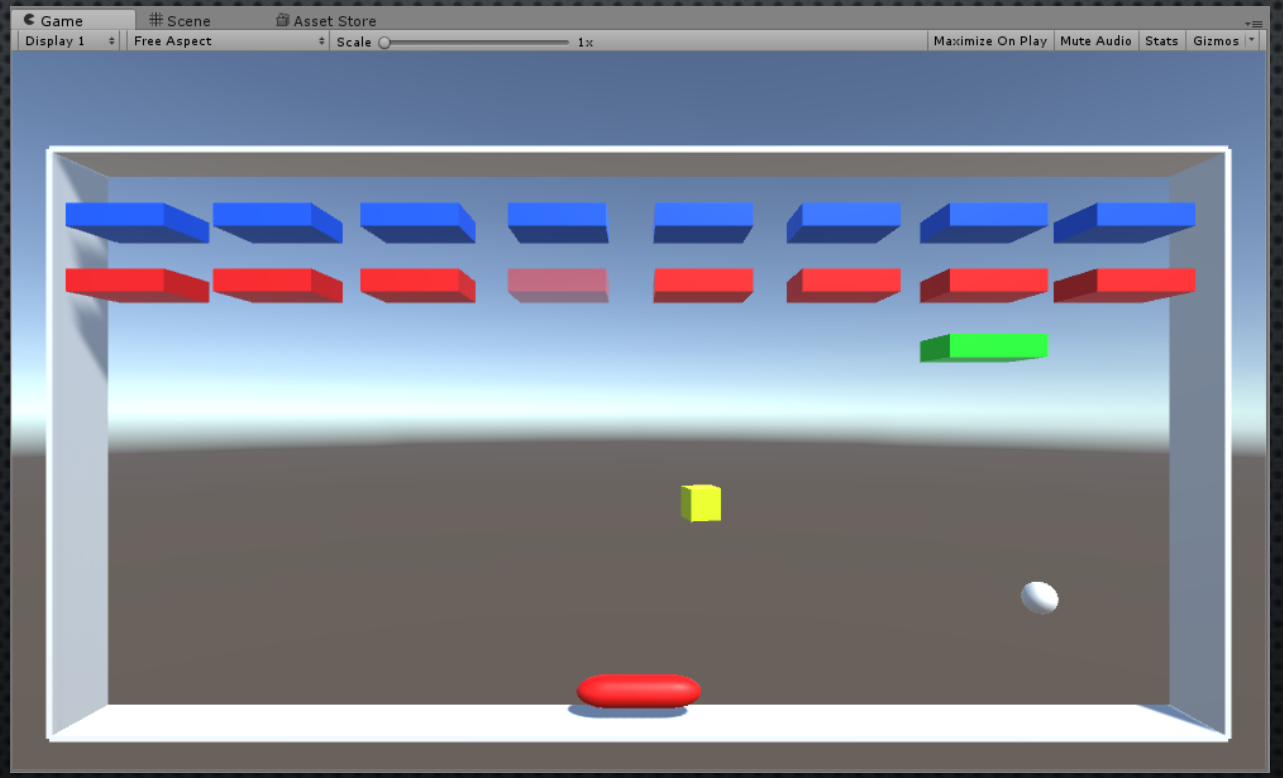
`Random.Range(a,b);`
aからbの間の値をランダムに返す

今回は「rnd<30」と
30%の確立でアイテムが
発生するようになっているので
デバッグの際はこの値を101にしとけば確実に発生する

`item.name = items[0].name;`
Instantiate()でオブジェクトを
作った場合
作った名前の後ろに「(Clone)」がついてしまいので修正している。

ブロックが壊れるとたまにアイテムを吐き出すようになる

出てきたアイテムがバーに当たるか床に当たった時に消えればOK!



アイテムを取ったら貫通玉になるようにする

C#スクリプト Penetrating_Ballを作る

Penetrating_Ballのコードを以下のようにする

GameObject.FindGameObjectsWithTag()
()の中に入れた文字列と同じタグを持っているオブジェクトを配列で返す

GetComponent<BoxCollider>().isTrigger
BoxCollider内のIsTriggerをプログラム内から変更する

```
Penetrating_Ball p_ball =
this.gameObject.GetComponent<Penetrating_Ball>();
Destroy(p_ball);
```
プログラム内から特定のスクリプトを削除している

コールチン
今回は説明を省きますがコード内でやってることは10秒後に全てのブロックのIsTriggerをfalseにし、このスクリプトを削除している
詳しく聞きたい人がいたら言ってください

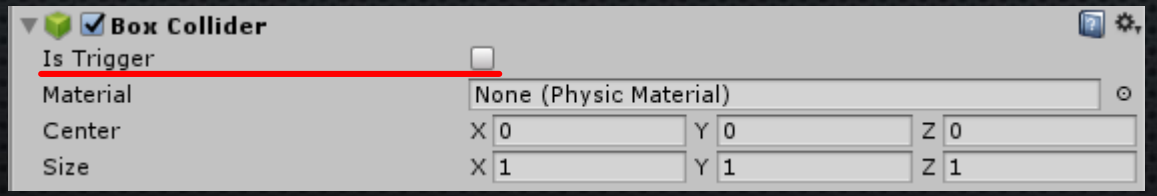
```
public class Penetrating_Ball : MonoBehaviour {

    void set_trigger(string name,bool check) {
        GameObject[] blocks = GameObject.FindGameObjectsWithTag(name);
        for (int i = 0; i < blocks.Length; i++)
        {
            blocks[i].GetComponent<BoxCollider>().isTrigger = check;
        }
    }

    IEnumerator destroy_component() {
        yield return new WaitForSeconds(10f);
        set_trigger("block1", false);
        set_trigger("block2", false);
        set_trigger("block3", false);
        Penetrating_Ball p_ball = this.gameObject.GetComponent<Penetrating_Ball>();
        Destroy(p_ball);

    }

// Use this for initialization
void Start () {
        set_trigger("block1", true);
        set_trigger("block2", true);
        set_trigger("block3", true);
        StartCoroutine(destroy_component());
    }
```

IsTriggerって何…？

以下Unityのサイトから
Rigidbody の衝突判定を行わなくなります。ですが、Rigidbody がトリガーの範囲内に「入り」、「出る」ときにコールバックとして OnTriggerEnter、OnTriggerExit、OnTriggerStayが呼び出されます。

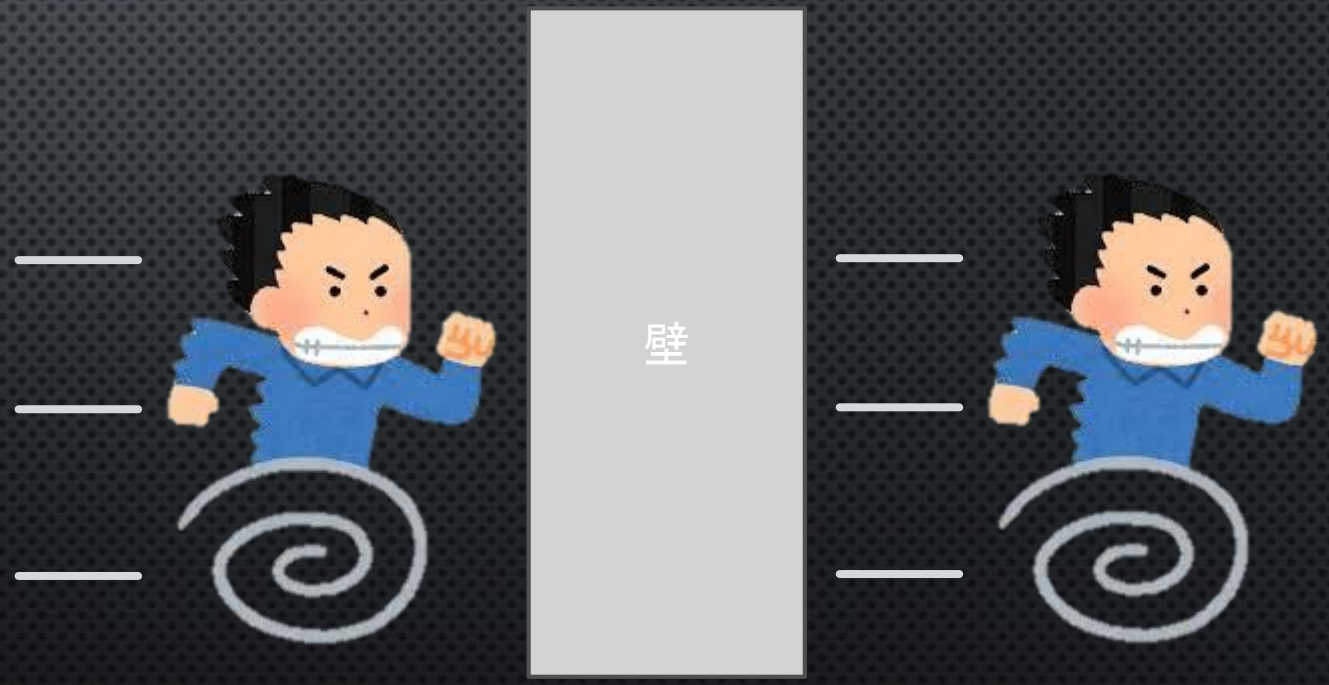
つまり…
Istriggerにチェックを入れる：
当たってもそのまま進み続ける
IsTriggerのチェックを外す：
当たったらその場で止まったり跳ね返ったりする

どちらも「衝突した」という判定は出る



IsTrigger:False

IsTrigger:True



すり抜ける！

主な使い方

ゴールテープ

ゴールラインに当たり判定を設けて
ゴールしたかを判定したい

もしIsTriggerをFalseにしていたら…

ゴール出来ねぇ…

ゴールラインのColliderのIsTriggerをTrueにしてすり抜けられるようにしよう!!

アイテムがバーに当たった際、 バーにPenetrating_Ballを追加するようにする

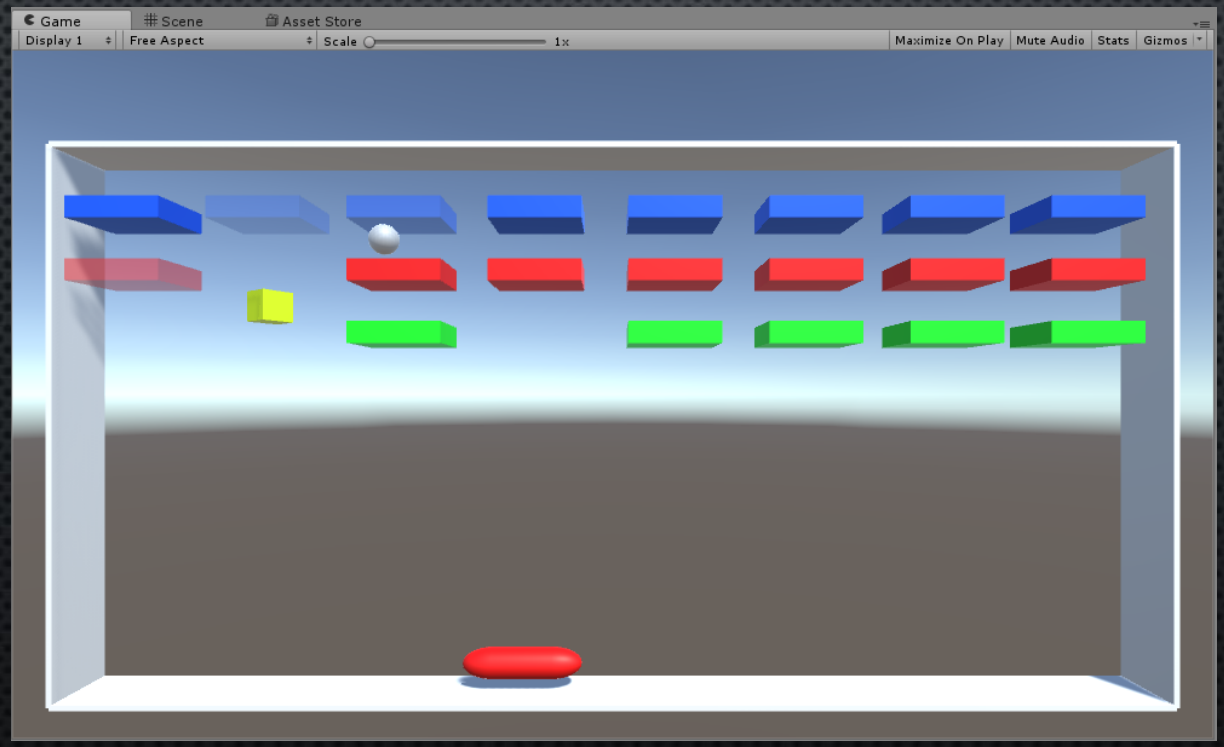
Itemスクリプトを開き以下のように変更

```
void OnTriggerEnter(Collider other) {
        if (other.gameObject.tag == "bar")
        {
            if (this.gameObject.name == "item1")
            {
                if (other.GetComponent<Penetrating_Ball>() == null)
                {
                    other.gameObject.AddComponent<Penetrating_Ball>();
                }
                Destroy(this.gameObject);
            }
        }
        else if (other.gameObject.name == "wall_bottom") {
            Destroy(this.gameObject);
        }
    }
```

`other.GetComponent<Penetrating_Ball>() == null`
<>内のスクリプトを指定したオブジェクトからGetComponeして持っていなかった場合、nullが返される
(つまり、そのオブジェクトが指定したスクリプトを持っているかの判断ができる)

`AddComponent<>()`
<>の中に入れたスクリプトを
対象のオブジェクトに追加する

アイテムを取ると10秒間だけブロックを貫通するようになる！！



玉の複製

玉を複製するアイテムを作る

1. 貫通玉の時と同じようにitemオブジェクトを作り、名前を「item2」にする
2. item2をプレハブ化し、Hierarchyのitem2を削除
3. 各block_1,2,3のBlockスクリプトのitemsにitem2をドラッグ＆ドロップ
4. 「ball」という名前のタグを作り、ballオブジェクトのタグにする
5. C#スクリプト Add_Ballを作る

玉を複製するプログラム

Add_Ballスクリプトを開き以下のようにする

List<GameObject>
C#でのリストの作り方
(配列とはまた違うものだから注意！)

今回は
ボールを探す→配列に格納→
各ボールからボールを2個ずつ生成→
生成したボールをcreated_ballsに格納→
5秒後、
created_ballsから生成したボールを削除し
このスクリプトも削除する

```
public class Add_Ball : MonoBehaviour {
    List<GameObject> created_balls = new List<GameObject>();

    IEnumerator delete_balls()
    {
        yield return new WaitForSeconds(5f);
        for (int i = 0; i < created_balls.Count; i++) {
            Destroy(created_balls[i]);
        }
        Add_Ball add_ball = this.gameObject.GetComponent<Add_Ball>();
        Destroy(add_ball);
    }

// Use this for initialization
void Start () {
        GameObject[] balls = GameObject.FindGameObjectsWithTag("ball");
        for (int i = 0; i < balls.Length; i++) {
            for (int j = 0; j < 2; j++) {
                GameObject new_ball = Instantiate(balls[i], balls[i].transform.position, Quaternion.identity);
                new_ball.name = balls[i].name;
                created_balls.Add(new_ball);
            }
        }
        StartCoroutine(delete_balls());
}
```

Itemスクリプトから Add_Ballを追加できるようにする

Itemスクリプトを開き以下のように変更する

自身のオブジェクトの名前が
「item2」にだった場合、
「Penetrating_Ball」ではなく、
「Add_Ball」を追加するように変更

```
void OnTriggerEnter(Collider other) {
        if (other.gameObject.tag == "bar")
        {
            if (this.gameObject.name == "item1")
            {
                if (other.GetComponent<Penetrating_Ball>() == null)
                {
                    other.gameObject.AddComponent<Penetrating_Ball>();
                }
            }
            else if (this.gameObject.name == "item2") {
                other.gameObject.AddComponent<Add_Ball>();
            }
            Destroy(this.gameObject);
        }
        else if (other.gameObject.name == "wall_bottom") {
            Destroy(this.gameObject);
        }
    }
```

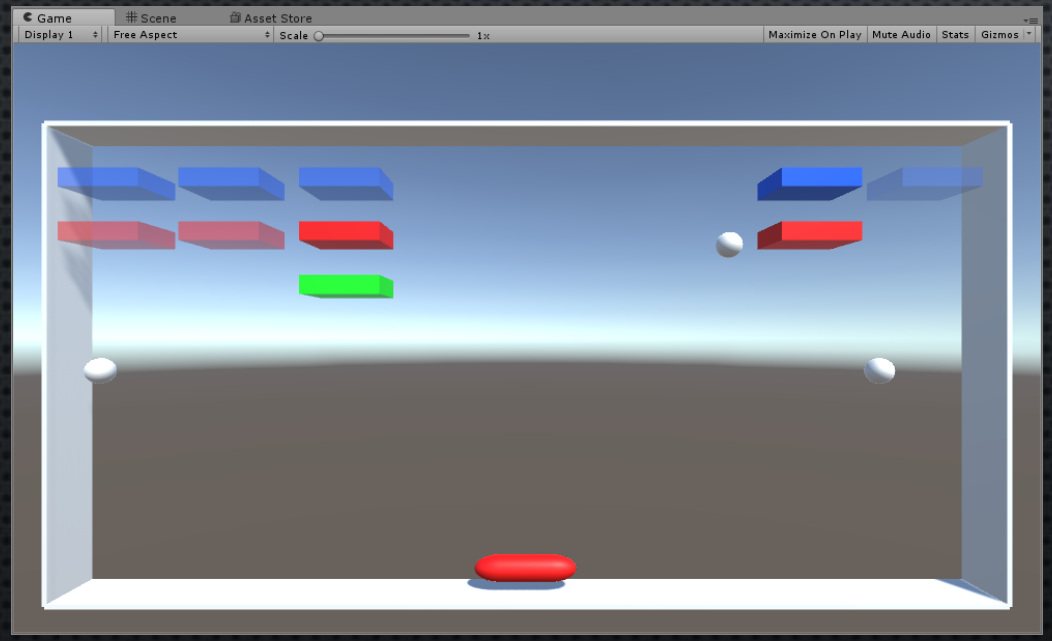
Blockからitem2オブジェクトも出るようにする

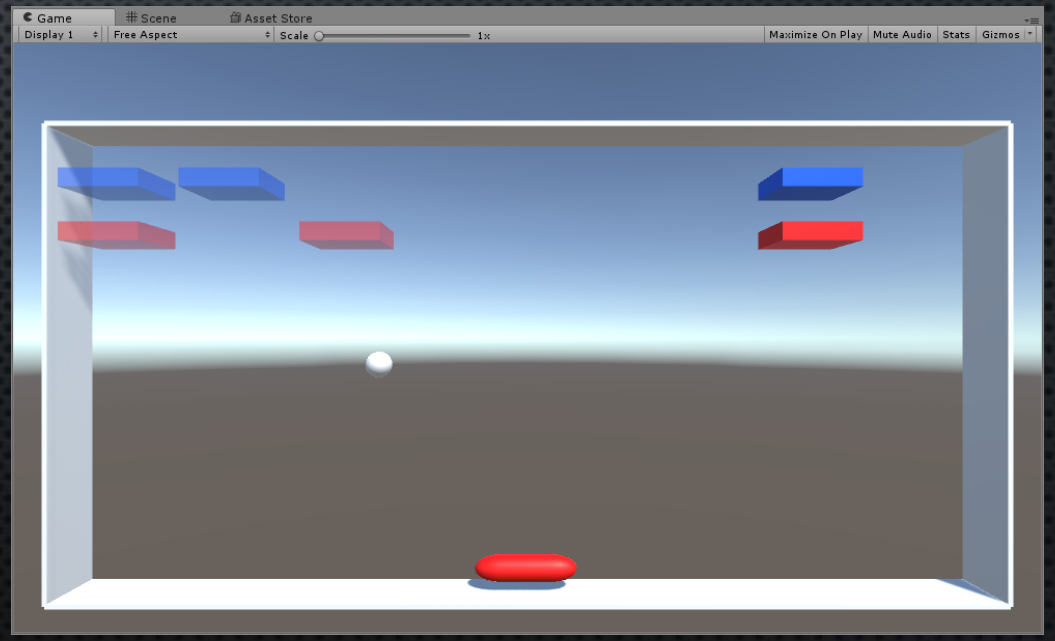
Blockスクリプトを開き以下のように変更する

Random.Range(0, 2);
ではなく
Random.Range(0, items.Length);
にすることで後でアイテムを
いろいろ追加しても問題なく動作する

```
void OnCollisionEnter(Collision other)
    {
        hp -= 1;
        if (hp == 0)
        {
            int rnd = Random.Range(0, 100);
            if (rnd < 30) {
                int item_rnd = Random.Range(0, items.Length);
                GameObject item = Instantiate(items[item_rnd], transform.position, Quaternion.identity);
                item.name = items[item_rnd].name;
            }
            Destroy(this.gameObject);
        }
        ～省略～
}
    void OnTriggerEnter(Collider other)
    {
        hp -= 1;
        if (hp == 0)
        {
            int rnd = Random.Range(0, 100);
            if (rnd < 30)
            {
                int item_rnd = Random.Range(0, items.Length);
                GameObject item = Instantiate(items[item_rnd], transform.position, Quaternion.identity);
                item.name = items[item_rnd].name;
            }
            Destroy(this.gameObject);
        }
        ～省略～
}
```

アイテムを取ると5秒間だけ玉が3つに増える！！





質問コーナー

unity内でのプログラムと授業のものとの違い

学校のプログラムの授業:
大体プログラムファイル一つでものを完成させている。

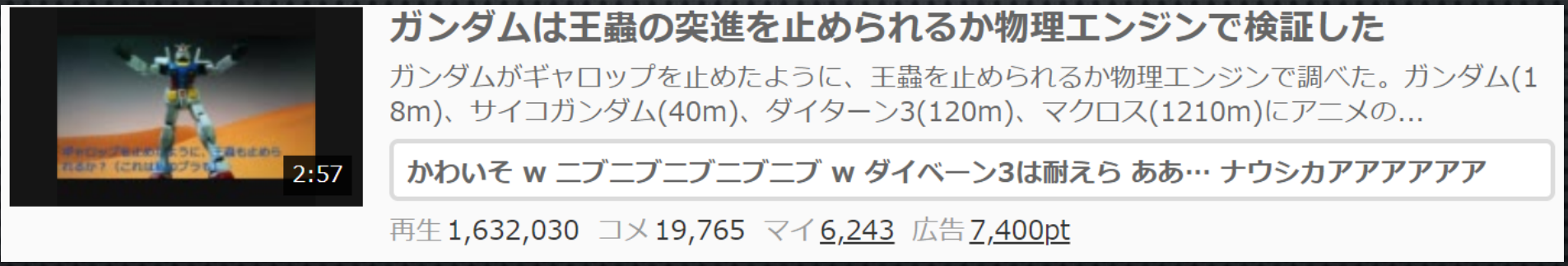
Unity :

いくつかのプログラムファイル(スクリプト)をつなぎ合わせて一つのものを完成させる

オブジェクト指向(？)

ゲーム以外の(きちんとした)使い方

ニコニコ動画とかでたまに見る「物理エンジンを用いた検証」とかに使えたりする



↑例なだけであってUnity使ってるかは不明

C#って何？

以下サイトから

C#とは、マイクロソフトが開発しているプログラミング言語です。

C++やJavaと同じオブジェクト指向と呼ばれるプログラミング言語で、文法はJavaに似ています。

そのため、C#を使用した経験があれば少しの学習でJavaも同じように使うことができ、応用が利きます。

サイトURL:https://www.sejuku.net/blog/26073#C

僕はとある教授から「C#はJavaをC(言語)でかけるようにした感じかな」みたいな
ふわふわした説明を受けました。

言語は何を使うのか

UnityはC#,JavaScript,Booで開発できたが

現在は、

C#,JavaScriptのみ対応している

JavaScriptもそのうち消えるとの噂

ネットに載ってる情報の大体がC#で書かれているのでC#を使うことがオススメ

よく使うメゾットの説明

- OnCollision～やOnTrigger～は衝突判定させるには必須なのでこれは覚えておきましょう
  衝突判定オススメサイト：https://qiita.com/yando/items/0cd2daaf1314c0674bbe
- MathやVector2,3などはめんどくさい計算をやってくれる関数が多いので一度見ておこう！
  (線形代数で学んでるようなことを全て自動でやってくれるのでほんとオススメ)
  Vector3(Unityサイト):https://docs.unity3d.com/ja/current/ScriptReference/Vector3.html
- Update関数、Start関数は必ずと言っていいほど使うのでどういうものか知っておこう！
- Debug.Log()は***神***
  「あれ？ちゃんとここ読み込んでる？」、「あれ？値どうなってる？」などゲームの裏の動きは普通見ることができなくて、何かバグって原因を調べる時、
  Debug.Log()を使いこなせるかで作業効率が大きく変わるのでどんどん使っていこう

広告の入れ方（真面目）

ググれ

Unity講座はこれにて終了！

- 今回はUnityの基礎的な部分を説明したのでここからいろいろと発展できると思います
(オブジェクトの置き方、作り方、当たり判定等々...)

- 夏休みに今回やったこと活用して開発してみましょう！
(そのまま文化祭で出展してみよう)

- 皆さんお疲れさまでした！粗末な説明でごめんなさい！m(__)m

- もしわからないことや気になることがあればSlackなり何なりで気軽に僕に聞いてください