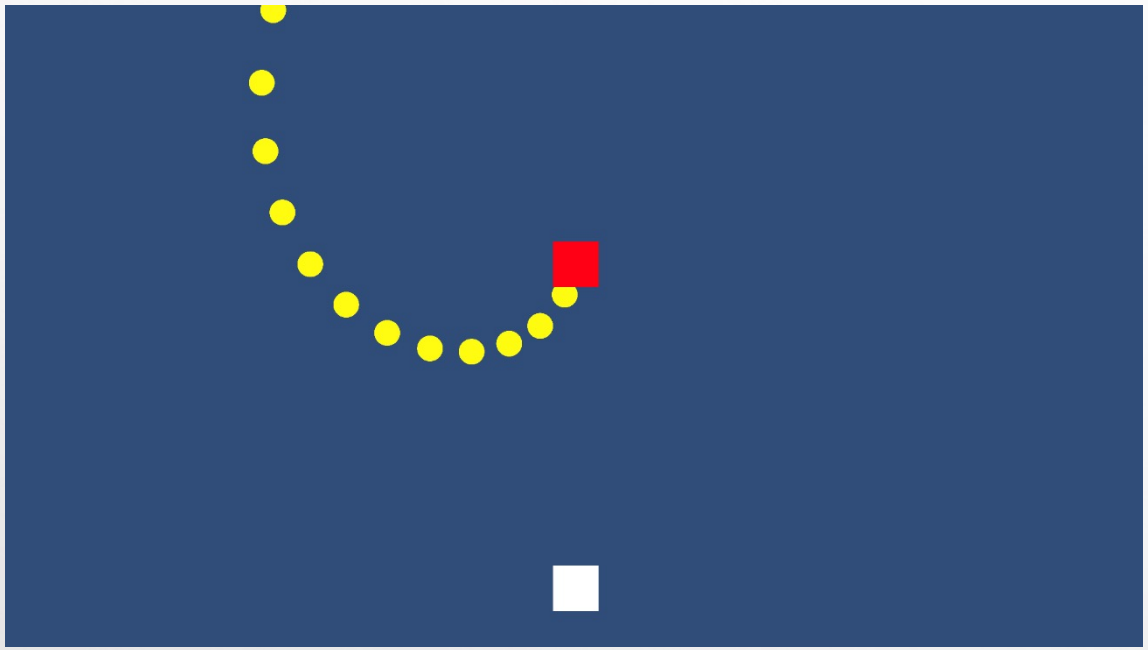
三角関数とラジアン

ゲーム数学

目次

１．UnityPackageを使用しての準備

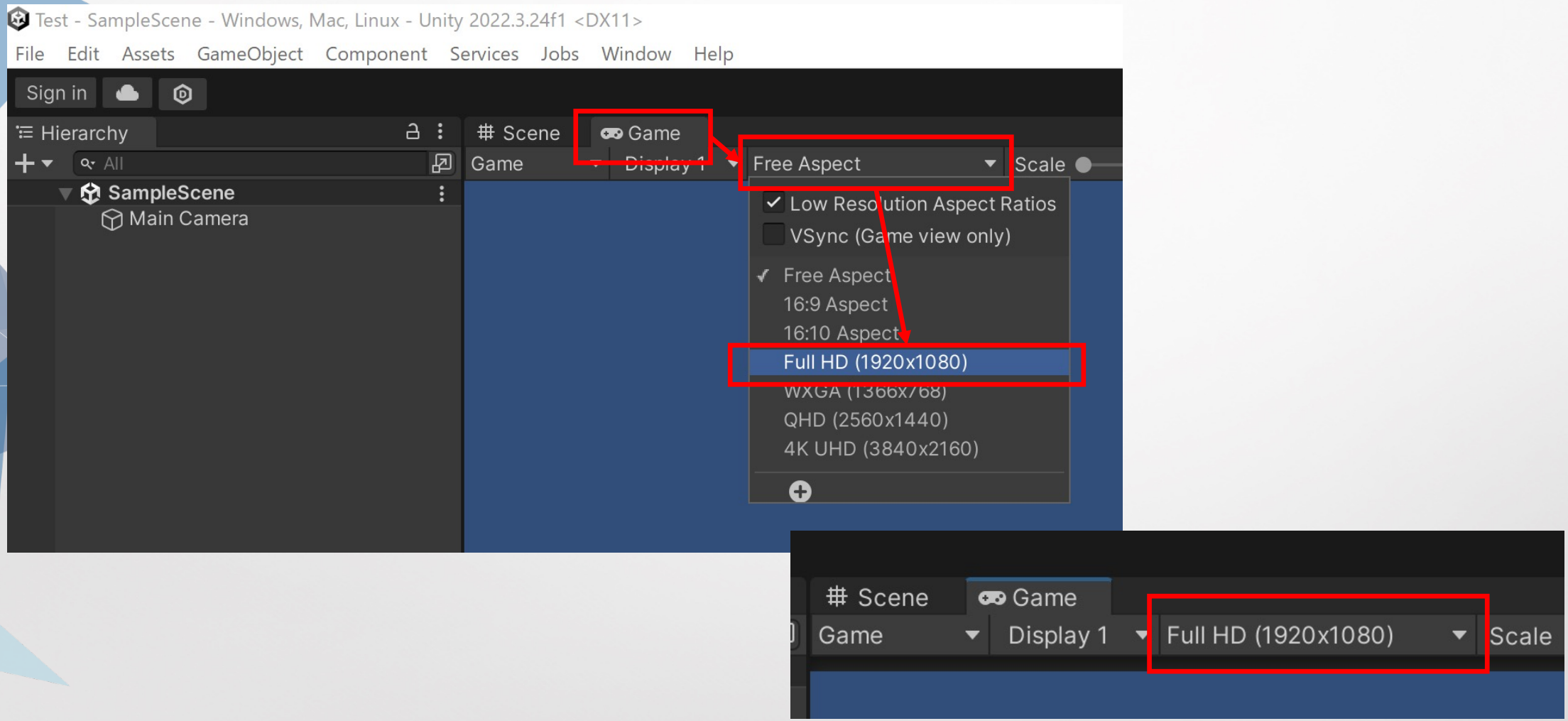
いわゆる弾幕ゲーと呼ばれる弾の発射について、どのような計算によって
行われているか実際に試してみましょう！



それでは、まずはゲームのプロジェクトの設定をしていきたいので、
UnityHubを開いて2Dでプロジェクトを作成してください。
また、別途unitypackageのファイルを自分のPCにコピーしてください。

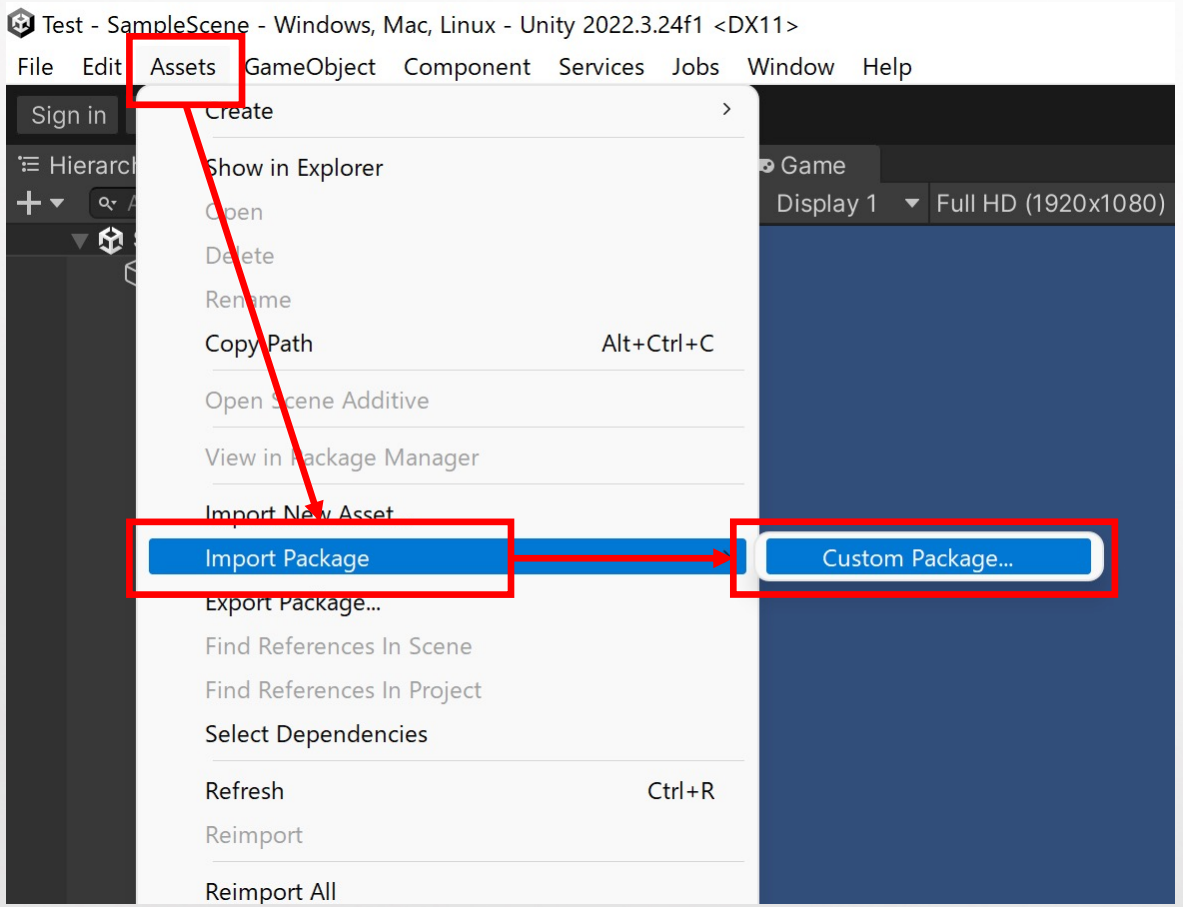
1．UnityPackageを使用しての準備

プロジェクト開いた人は、以下の操作をお願いします。(画面の解像度の設定)



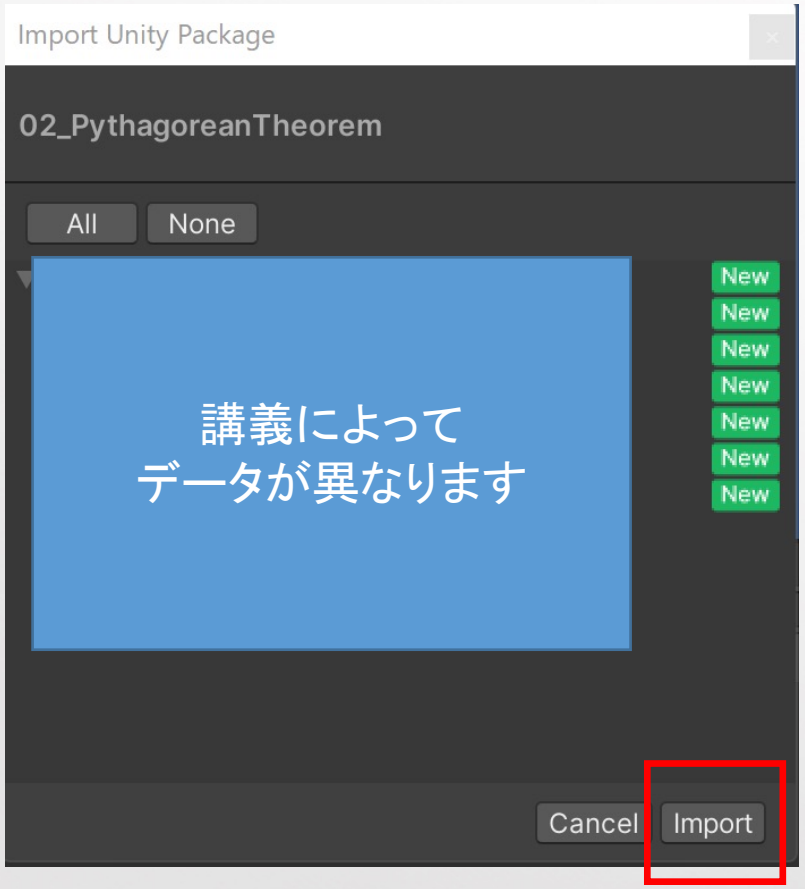
１．UnityPackageを使用しての準備

次に、unitypackageというパッケージファイル(データが詰め込まれたファイル)を読み込みます。



１．UnityPackageを使用しての準備

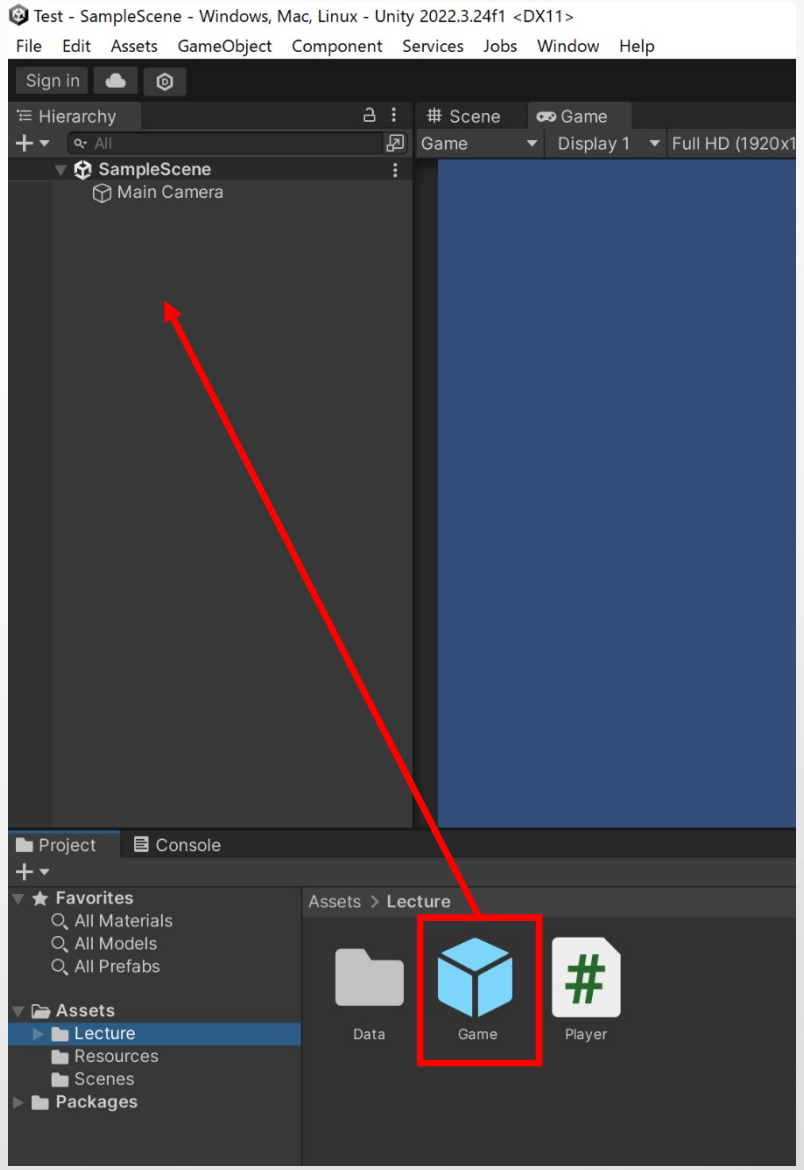
ファイルを指定し、以下の画面が出たらImportを押すと、ファイルが展開されます。



１．UnityPackageを使用しての準備

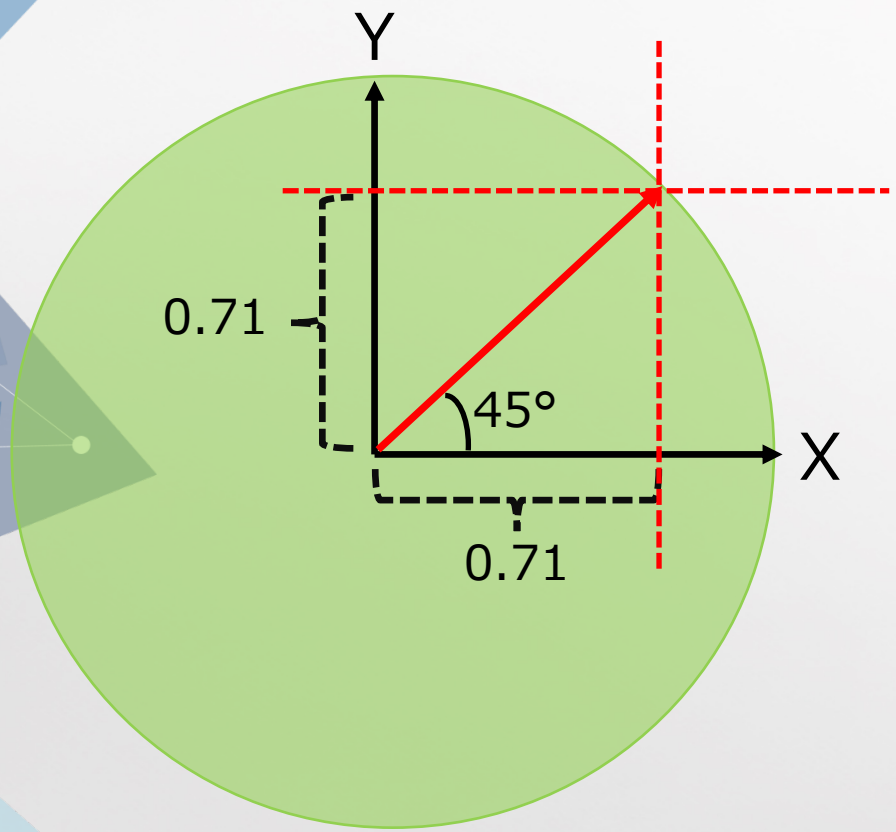
「Game」のファイルを、ヒエラルキー上に
ドラッグしてみてください。(右記参照)

以下のような画面になっていればOKです！



２．なぜ三角関数を使うのか

そもそもなぜ三角関数(sin/cos/tan)を使うのか、、を考えてみましょう！
以前、キャラクターの斜め移動で、以下のような動きを確認したと思います。

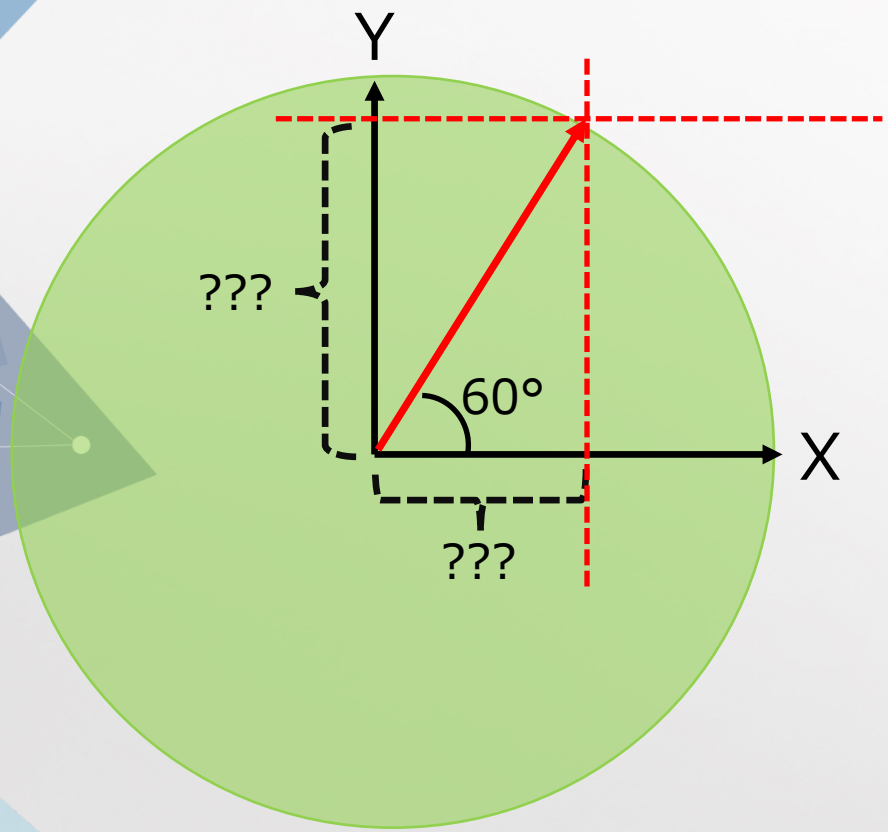


XとYの入力があった場合、0.71を乗算して
斜めの移動量を1にしていましたが、その際、
ちょうど45度の方向に移動していたと思います！

XとYの移動量が同じ0.71なので、ちょうど90度
の半分の45度になるかと予想がつくかと思います。

２．なぜ三角関数を使うのか

では、今度は以下のように60度の方向にしたい場合、XとYにはどのような入力にすれば良いでしょうか。。。



分かっていることは、以下のことだと思います。

・斜辺は1

・角度は60度

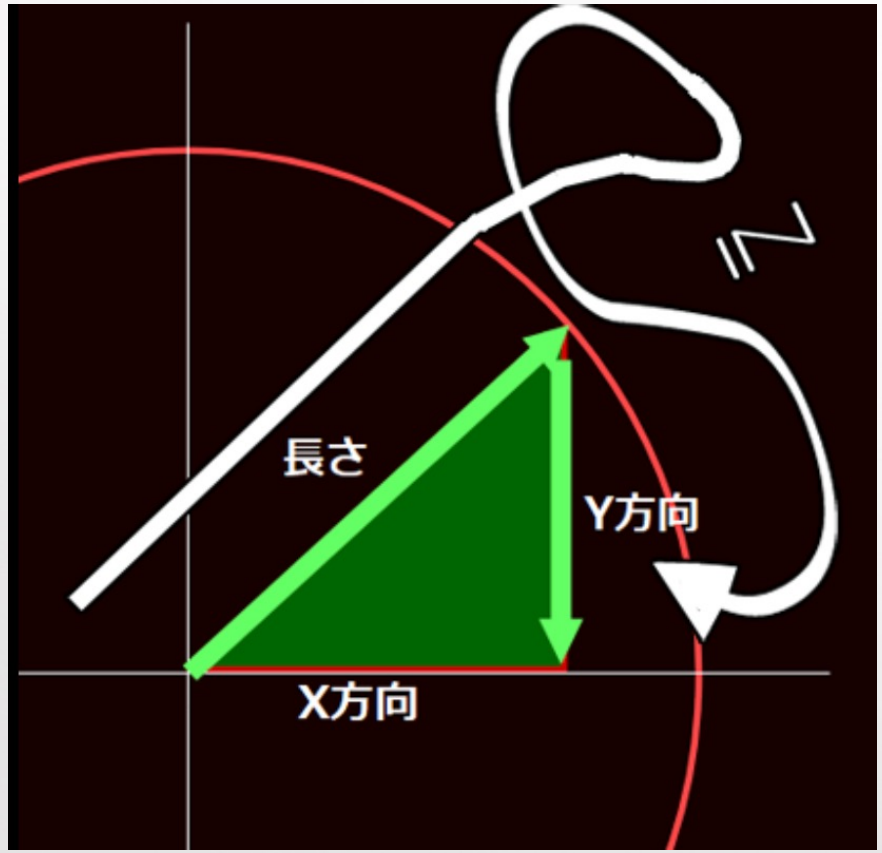
三角関数を使うとここからXとYの値を算出することが出来ます！！

２．なぜ三角関数を使うのか

まずは、Yの値を求めていきたいと思います！ここではsinを使っていきます！
θは角度になります、ただsinθなので実際の値は一覧表を検索してみてください。

一覧表例

θ(deg)	θ(rad)	sinθ	tanθ	cosθ
0	0	0	0	1
1	0.0175	0.0175	0.0175	0.9998
2	0.0349	0.0349	0.0349	0.9994
3	0.0524	0.0523	0.0524	0.9986
4	0.0698	0.0698	0.0699	0.9976
5	0.0873	0.0872	0.0875	0.9962
6	0.1047	0.1045	0.1051	0.9945
7	0.1222	0.1219	0.1228	0.9925
8	0.1396	0.1392	0.1405	0.9903
9	0.1571	0.1564	0.1584	0.9877
10	0.1745	0.1736	0.1763	0.9848
11	0.192	0.1908	0.1944	0.9816
12	0.2094	0.2079	0.2126	0.9781
13	0.2269	0.2250	0.2309	0.9744
14	0.2443	0.2419	0.2493	0.9703
15	0.2618	0.2588	0.2679	0.9659
16	0.2793	0.2756	0.2867	0.9613
17	0.2967	0.2924	0.3057	0.9563
18	0.3142	0.3090	0.3249	0.9511
19	0.3316	0.3256	0.3443	0.9455
20	0.3491	0.3420	0.3640	0.9397



公式としては

sinθ ＝ Y方向 / 長さ

なので、

Y方向 ＝ sin θ ＊ 長さ

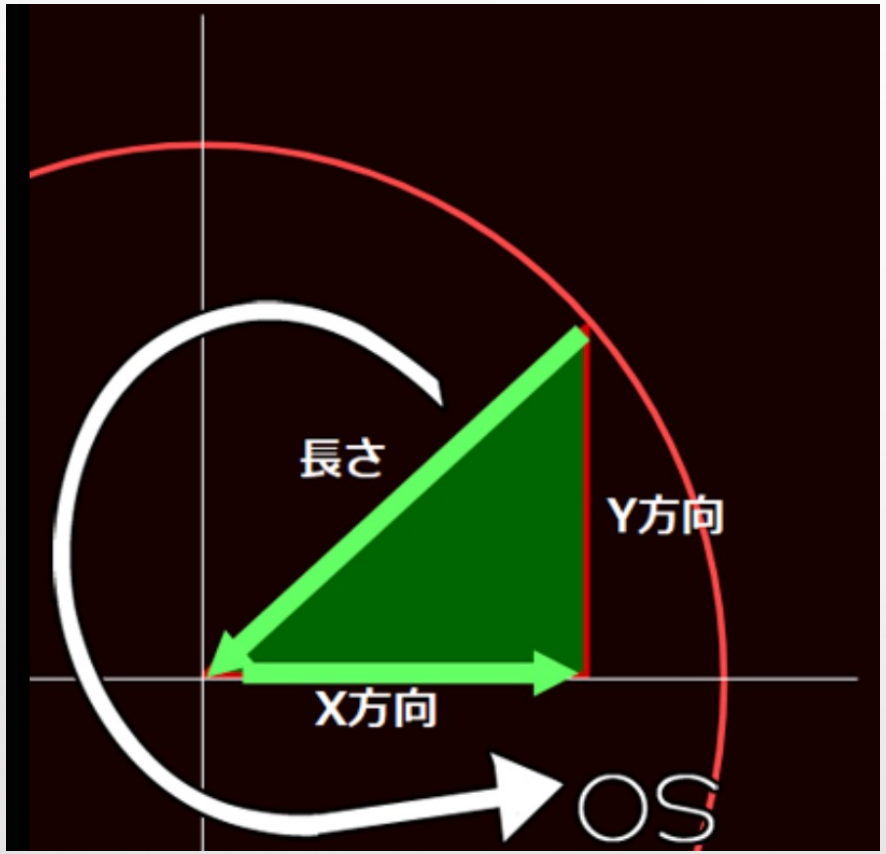
に変更出来ますよね！！

2．なぜ三角関数を使うのか

次にXの値を求めていきたいと思います！ここではcosを使っていきます！
cosθの実際の値についても一覧表を検索してみてください。

一覧表例

θ(deg)	θ(rad)	sinθ	tanθ	cosθ
0	0	0	0	1
1	0.0175	0.0175	0.0175	0.9998
2	0.0349	0.0349	0.0349	0.9994
3	0.0524	0.0523	0.0524	0.9986
4	0.0698	0.0698	0.0699	0.9976
5	0.0873	0.0872	0.0875	0.9962
6	0.1047	0.1045	0.1051	0.9945
7	0.1222	0.1219	0.1228	0.9925
8	0.1396	0.1392	0.1405	0.9903
9	0.1571	0.1564	0.1584	0.9877
10	0.1745	0.1736	0.1763	0.9848
11	0.192	0.1908	0.1944	0.9816
12	0.2094	0.2079	0.2126	0.9781
13	0.2269	0.2250	0.2309	0.9744
14	0.2443	0.2419	0.2493	0.9703
15	0.2618	0.2588	0.2679	0.9659
16	0.2793	0.2756	0.2867	0.9613
17	0.2967	0.2924	0.3057	0.9563
18	0.3142	0.3090	0.3249	0.9511
19	0.3316	0.3256	0.3443	0.9455
20	0.3491	0.3420	0.3640	0.9397



公式としては

cosθ ＝ X方向 / 長さ

なので、

X方向 ＝ cos θ * 長さ

に変更出来ますよね！！

3．三角関数の使い方

少し数学的な話が出てきましたが、Unityでは先ほどのような計算をしなくても、以下のような便利な関数(メソッド)が用意されているので、以下のものを使えばXとYの値を出すことが可能です！

よく使うので、ぜひ覚えておきましょう！

3．三角関数の使い方

使用例：

```
float cosX = Mathf.Cos(10.0f);
float cosY = Mathf.Sin (10.0f);
```

というか、引数で指定するラジアン角とは何者？？

MathF.Cos(Single) メソッド

リファレンス

定義

名前空間: System
アセンブリ: System.Runtime.dll
パッケージ: Microsoft.Bcl.Numerics v9.0.0-preview.3.24172.9
ソース: MathF.cs

指定された角度のコサインを返します。

C#

```
public static float Cos (float x);
```

パラメーター

x Single
ラジアンで表した角度。

MathF.Sin(Single) メソッド

リファレンス

定義

名前空間: System
アセンブリ: System.Runtime.dll
パッケージ: Microsoft.Bcl.Numerics v9.0.0-preview.3.24172.9
ソース: MathF.cs

指定された角度のサインを返します。

C#

```
public static float Sin (float x);
```

パラメーター

x Single
ラジアンで表した角度。

４．度数法と弧度法

UnityやDirectXなどでは、通常みなさんが使っている度数法ではなく、
弧度法という角度の指定が使われています。

＜度数法＞

　円一周は何度でしょうか？？

＜弧度法＞

　円一周は何度でしょうか？？

４．度数法と弧度法

UnityやDirectXなどでは、通常みなさんが使っている度数法ではなく、
弧度法という角度の指定が使われています。

＜度数法＞

円一周は何度でしょうか？？　⇨ 360度

＜弧度法＞

円一周は何度でしょうか？？　⇨ 2π(単位はrad)

なぜ、円一周が2πになるのか知りたい人は、検索してみてください。。

４．度数法と弧度法

こちらもUnityで便利な関数が用意されているので、使用していきたいと思います。

Mathf.Deg2Rad

public static float **Deg2Rad** ;

説明

度からラジアンに変換する定数（読み取り専用）

これは *(PI * 2) / 360* と等しいです

関連項目: Rad2Deg 定数

```
using UnityEngine;
using System.Collections;

public class ExampleClass : MonoBehaviour
{
    public float deg = 30.0F;

    void Start()
    {
        float rad = deg * Mathf.Deg2Rad;
        Debug.Log(deg + " degrees are equal to " + rad + " radians.");
    }
}
```

右記の例では、度数法の30度をラジアン角に変更しています。

５．演習問題

今まで覚えてきた知識を元に、以下の問題を解いてみてください！

Bullet.csのInit関数内で、60度の角度で弾を発射してみてください。
vec2.xにXの移動量、vec2.yにYの移動量を代入することで発射出来る仕組みとなっています。

```
public void Init(float angle)
{
    //弾を飛ばす角度を設定する
    //Xの移動量はvec2.xに格納してください
    //Yの移動量はvec2.yに格納してください
    vec2.x = 0;
    vec2.y = 0;
}
```

5．演習問題

＜解答例＞

```
public void Init(float angle)
{
    //弾を飛ばす角度を設定する
    //Xの移動量はvec2.xに格納してください
    //Yの移動量はvec2.yに格納してください
    vec2.x = Mathf.Cos(60.0f * Mathf.Deg2Rad);
    vec2.y = Mathf.Sin(60.0f * Mathf.Deg2Rad);
}
```

出来た人は、引数で取っているangleを度数法の角度として指定すると
どうなるか確認してみてください。
また、分かるようであればなぜそのような動きになるかについても確認してください。
出来た人はMathf.Deg2Rad使わずに計算してみてください！！

5．演習問題

＜解答例＞

・度数法から弧度法に変換する式