名前空間

C#スクリプト（初期画面）

```
1  using System.Collections;
2  using System.Collections.Generic;
3  using UnityEngine;
4
5  public class NewBehaviourScript : MonoBehaviour
6  {
7      // Start is called before the first frame update
8      void Start()
9      {
10
11     }
12
13     // Update is called once per frame
14     void Update()
15     {
16
17     }
18 }
```

C#の機能を使うよという初期設定

Unityの機能を使用

usingの右側

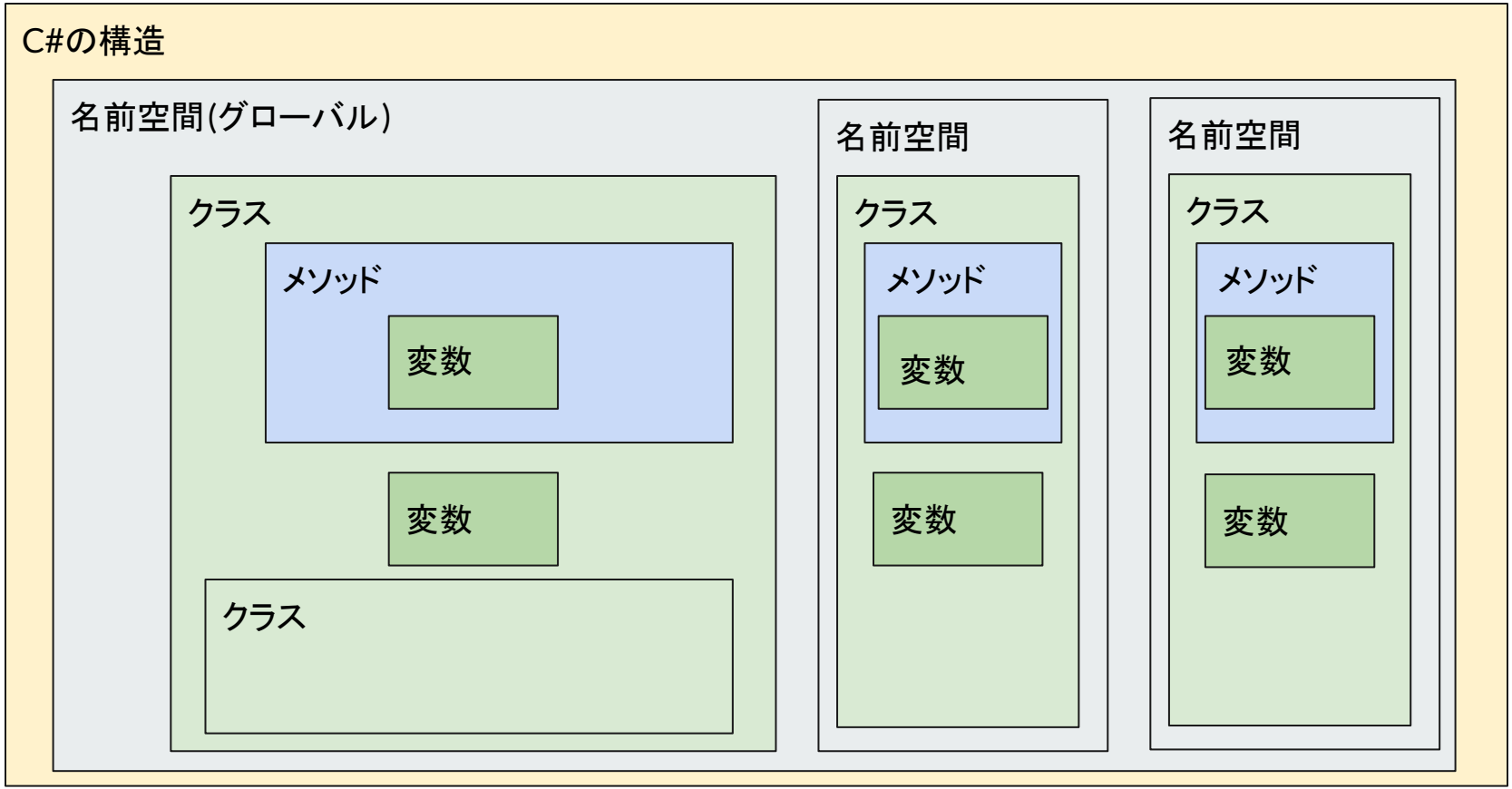
→**「名前空間」の名前**

名前空間

名前空間が活躍するときの例：

誰かと一緒にプログラム作成するときなどに関数名が被っても
問題なく使えるようになる

C#の構造


C#の構造
名前空間(グローバル)
クラス
メソッド
変数
変数
クラス
名前空間
クラス
メソッド
変数
変数
名前空間
クラス
メソッド
変数
変数

名前空間(test.cs) 表現の仕方

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

namespace World
{
  public class test : MonoBehaviour
  {
    // Start is called before the first frame update
    void Start(){}
    // Update is called once per frame
    void Update(){}
  }
}
```

何も書いてない時は「グローバル名前空間」になっている

名前空間(test.cs) 表現の仕方

```
namespace World
{
  public class test : MonoBehaviour
  {
    void Start(){}
    void Update(){}
  }
  class String { }
  namespace Human
  {
    class List
    {
      public string str = "現実世界　人間";
    }
  }
}
```

1つのファイルに複数の名前空間があっていいし、
名前空間内に名前空間があっていい

名前空間(test.cs) 表現の仕方

```
namespace World
{
  public class test : MonoBehaviour
  {
    void Start(){}
    void Update(){}
  }
  class String { }
}
namespace World.Human
  {
    class List
    {
      public string str = "現実世界　人間";
    }
  }
```

同じ意味

```
namespace World
{
  public class test : MonoBehaviour
  {
    void Start(){}
    void Update(){}
  }
  class String { }
  namespace Human
  {
    class List
    {
      public string str = "現実世界　人間";
    }
  }
}
```

名前空間(test.cs) 利用の仕方

```
namespace World{
  public class test : MonoBehaviour
  {
    void Start()
    {
      Human.List humanList = new Human.List();
      // World.Human.List humanList = new World.Human.List();
      Debug.Log(humanList.str);
    }
  }
  namespace Human
  {
    class List
    {
      public string str = "現実世界　人間";
    }
  }
}
```

別のファイル・別のフォルダの名前空間

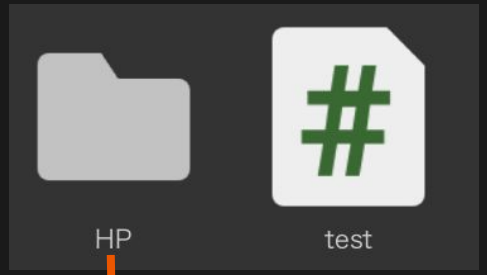
test0.cs

```
namespace World2.Animal
{
  class List
  {
    public string str = "並行世界　動物";
  }
}
```

test.cs

```
void Start()
{
  World2.Animal.List world2animal = new World2.Animal.List();
  Debug.Log(world2animal.str);
}
```

別のファイル・別のフォルダの名前空間




```
void Start()
    {
        Physical.List physical = new Physical.List();
        Debug.Log(physical.str);
    }
```

```
namespace Physical
{
  /// <summary>
  /// 並行世界 人間 体力
  /// </summary>
  class List
  {
     public string str = "並行世界　人間 体力";
  }
}
```

コメントアウト説明　例

```
/// <summary>
/// 並行世界 人間 体力
/// </summary>

void Start()
{
    Physical.List physical = new Physical.List();
    Debug.Log(physical.str);
}
```

```
class Physical.List
並行世界 人間 体力
```

別のファイル・別のフォルダの名前空間

test0.cs

```
namespace World2.Animal
{
  class List
  {
    public string str = "並行世界　動物";
  }
}
```

test.cs

```
using World2.Animal;

void Start()
{
  List world2human = new List();
  Debug.Log(world2human.str);
}
```

名前空間(test.cs) 表現の仕方

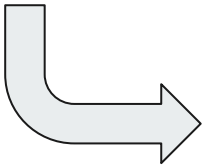
```
namespace World
{
  public class test : MonoBehaviour
  {
    void Start(){}
    void Update(){}
  }
  class String { }
  namespace Human
  {
    class List
    {
      public string str = "現実世界　人間";
    }
  }
}
```

```
using World;
```

```
using World.Human;
```

usingの注意点

例えば、別々の名前空間名を持つ2つの名前空間があり、
2つの名前空間内に同じ同じメソッド名があり、
2つの名前空間に対して「using」を使用した場合：



エラーになったり、優先順位をつけることもできるが、
基本は整理しやすくするために「名前空間」を使用する。
なので、よっぽど頻繁に使用する機会がある名前空間にだけ
「using」を使用し、プログラムを省略できるようにする。

基本はusingさせずに使用する方がいいと思う

C#の構造