# 5.WhenAll:Coroutine 的并行操作

WhenAll 意思是，当所有的。

当所有的什么呢?

就是当所有的事件流都结束，就会触发 Subscribe 注册的回调。

使用 WhenAll 可以实现 Coroutine 的并行操作。

```
public class WhenAllCoroutineTest: MonoBehaviour
{
    IEnumerator A()
    {
        yield return new WaitForSeconds(1.0f);
        Debug.Log("A");
    }

    IEnumerator B()
    {
        yield return new WaitForSeconds(2.0f);
        Debug.Log("B");
    }

    void Start()
    {
        var aStream = Observable.FromCoroutine(_ => A());
        var bStream = Observable.FromCoroutine(_ => B());

        Observable.WhenAll(aStream,bStream)
                .Subscribe(_ =>
                {

                }).AddTo(this);
    }
```

```
}
```

一秒后输出结果为:

```
A
```

两秒后输出结果为:

```
A
B
```

WhenAll 和 Merge 是同类型的，是处理多个流的操作符。

理解起来非常简单。

除了并行实现 Coroutine 之外，还可以实现，当所有的按钮都点击过一次的逻辑。

```
public class ButtonAllClickedOnce : MonoBehaviour
{
    [SerializeField] Button mButtonA;
    [SerializeField] Button mButtonB;
    [SerializeField] Button mButtonC;

    void Start()
    {
        var aStream = mButtonA.OnClickAsObervable().First();
        var bStream = mButtonB.OnClickAsObervable().First();
        var cStream = mButtonC.OnClickAsObervable().First();

        Observable.WhenAll(
                    aStream,
                    bStream,
                    cStream)
                .Subscribe(_ =>
                {
                    Debug.Log("clicked");
```

```
                }).AddTo(this);
    }
}
```

当点击完，A、B、C 按钮之后就会输出:

```
clicked
```

WhenAll 可以配合非常多的操作符使用。理解也非常简单。

今天的内容就这些。