3.UniRx 的基本语法格式

在之前的两节课,我们学习了 UniRx 的 Timer 和 Update 这两个 API,但是没有介绍过代码。今天就简单给大家介绍一下,UniRx 的基本语法格式。

搬出第一堂课 Delay 实现的代码:

```
/**********************************
* http://sikiedu.com liangxie
*******************************
using System;
using UniRx;
using UnityEngine;
namespace UniRxLesson
{
   public class DelayExample : MonoBehaviour
      private void Start()
         Observable.Timer(TimeSpan.FromSeconds(2.0f)).Subscribe(_ =>
         {
            Debug.Log("延时两秒");
         }).AddTo(this);
      }
   }
}
```

Observable.XXX().Subscribe() 是非常典型的 UniRx 格式。

只要理解什么意思就可以看懂大部分的 UniRx 的用法了。

首先解决词汇问题:

Observable: 可观察的,形容词,形容后边的词(Timer) 是可观察的,我们可以粗暴地把 Observable 后边的理解成发布者。

Timer: 定时器, 名词, 被 Observable 描述, 所以是发布者, 是事件的发送方。

Subscribe: 订阅,动词,订阅谁呢? 当然是前边的 Timer, 这里可以理解成订阅者,也就是事件的接收方。

AddTo: 暂不用理解。

连起来则是:可被观察(监听)的.Timer().订阅()

顺下来应该是:订阅可被观察的定时器。

其概念关系很容易理解。

- Timer 是可观察的。
- 可观察的才能被订阅。

Observable.XXX().Subscribe();

可被观察(监听)的 XX, 注册。

以上笔者从发布者和订阅者这个角度来进行的介绍,以便大家理解。

但是 UniRx 的侧重点,不是发布者和订阅者这两个概念如何使用,而是事件从发布者到订阅者之间的过程如何处理。

所以两个点不重要,重要的是两点之间的线,也就是事件的传递过程。

这里先不说得太深入,在入门之后,会用很大的篇幅去进行讲解。

UniRx 的基本语法格式 就介绍到这里。