4. UI Trigger

在本章的第二课，介绍了 UI 的所有的 Observable 增强。

不过还不够，因为我们上堂课学习了 Trigger。

Trigger 也有支持 UI 的部分。

在上堂课的结尾说过，Trigger 除了 MonoBehaviour 还支持其他的类型，比如 Transform、RectTransform、还有 UIBehaviour。

那么 这个 UIBehaviour 就是 本文要讲解的重点。

为什么?

因为 UIBehaivour 是 UGUI 所有控件的基类。

只要支持 UIBehaivour，就支持所有的 UGUI 控件等会继承 UIBehaviour 的支持。

那么从哪方面支持呢?

是从各种事件开始支持的。

比如所有的 Graphic 类型都支持 OnPointerDownAsObservable、OnPointerEnterAsObservable、OnPointerEnterAsObservable 等 Trigger。

Graphic 简单介绍下，所有的在 Inspector 上显示，Raycast Target 选定框的都是 Graphic 类型，包括 Image、Text 等全部都是。

也就是说 Image、Text 全部支持 OnPointerDownAsObservable、OnPointerEnterAsObservable 等 Trigger。

我们知道，如果想自己去接收一个 OnPointerDown 事件，需要实现一个 IPointerDownHandler 接口，而 UniRx 则把 所有的 IXXXHandler 接口都做成 Trigger了。

这样再也不用需要网上到处流传的 UIEventListener.Get(gameObejct).onClick 这种方式了。

因为这种方式问题很多，比如，由于它集成了 EventTriggers，实现了所有的事件接口，他就会吞噬掉 OnScroll 等事件。

而 UniRx 的实现非常细，也就是 一个 IXXXHandler 就是一个 Trigger（本来老师的 QFramework 也想全部都实现了）。

需要一个全部实现并且吞并事件的版本也没关系，UniRx 也实现了一个 ObservableEventTrigger。和 UIEventListener 一样的。

老师在项目中用的比较多的几个 Trigger:

```
mImage.OnBeginDragAsObservable().Subscribe(dragEvent => {});
mGraphic.OnDragAsObservable().Subscribe(dragEvent => {});
mText.OnEndDragAsObservable().Subscribe(dragEvent => {});
mImage.OnPointerClickAsObservable().Subscribe(clickEvent => {});
```

非常方便，导致 QFramework 的一些脚本都弃用了，哈哈哈。

除了常用的几个 Trigger 之外 还有非常多的实用的 Trigger。

比如: OnSubmitAsObservable、OnDropAsObservable 等等。

具体可以参考 ObservableTriggerExtensions.Component.cs

只要能想到的 基本上 UniRx 都支持。

忘了说一点， 要使用 各种 Trigger 类型，就要导入命名空间:

```
using UniRx.Triggers;
```

本节课的内容就这些。