8.ObservableWWW 优雅的网络请求操作

以往我们不管使用 WWW 还是 UnityWebRequest 都要使用 Coroutine 去驱动。

但是使用协程写出来的代码，需要一堆判断，导致代码非常混乱。

而 UniRx 则是以以往一样简练的风格提供了对网络请求的支持。

代码如下:

```
ObservableWWW.Get("http://sikiedu.com")
             .Subscribe(_ =>
             {
                 // todo some thing
             }).AddTo(this);
```

非常简单。

当然，ObservableWWW 同样支持 WhenAll 操作符。

代码如下:

```
var aStream = ObservableWWW.Get("http://sikiedu.com");
var bStream = ObservableWWW.Get("http://qframework.io");

ObservableWWW.WhenAll(aStream, bStream)
             .Subscribe(_ =>
             {
                 // do something
             }).AddTo(this);
```

除了 Get 也支持了 Post，还有 GetWWW 和 PostWWW 这种的辅助封装，还有 GetAndGetBytes 和 PostAndGetBytes。

列出 QFramework 中一段下载文件的代码:

```
// http://liangxiegame.com/media/QFramework_v0.0.9.unitypackage
protected override void OnBegin()
{
    ...

    var progressListener = new ScheduledNotifier<float>();

    ObservableWWW.GetAndGetBytes(mRequestPackageData.DownloadUrl, null,
progressListener)
            .Subscribe(bytes =>
            {
                ...
            });

    progressListener.Subscribe(OnProgressChanged);
}

private void OnProgressChanged(float progress)
{
    EditorUtility.DisplayProgressBar("插件更新",
        "插件下载中 {0:P2}".FillFormat(progress), progress);
}
```

ObservableWWW 的 API 都可以传进去一个 ScheduledNotifier<T>() ，用来监听下载进度的。

Subscribe 之后传回来的值则是，当前的进度。

而且 ObservableWWW 的 Get 和 Post 请求都可以自己传对应的 header 和 WWWForm。

除了常用的 Get 和 Post 请求，也对 AssetBundle 的加载也做了简单的封装。

提供了诸如 ObservableWWW.LoadFromCacheOrDownload 这样的 API。

如果想深入了解，可以参考 ObservableWWW.cs

总之对 WWW 提供的 API 非常简练，也足够使用。

今天的内容就这些。