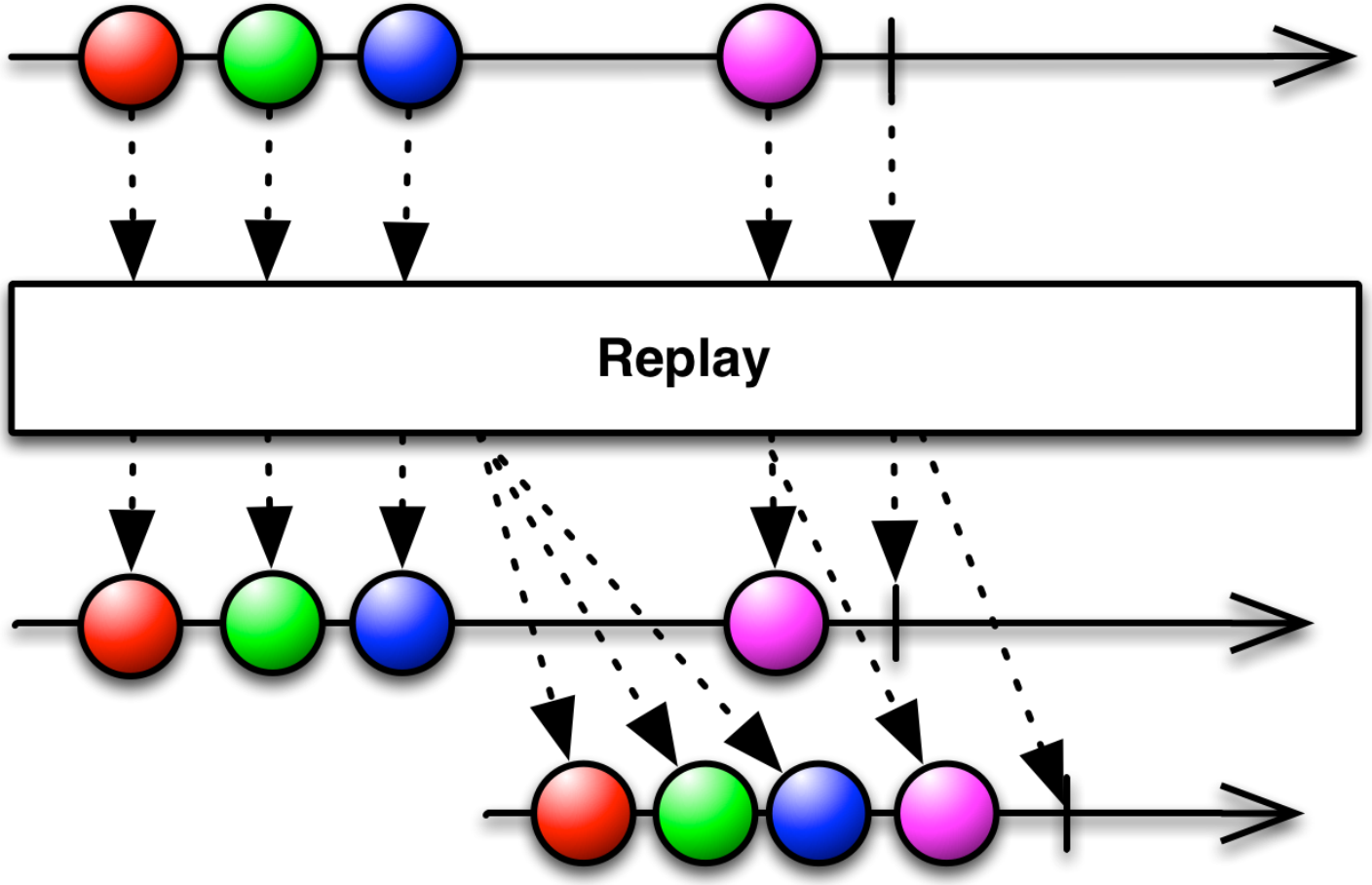
31.Replay

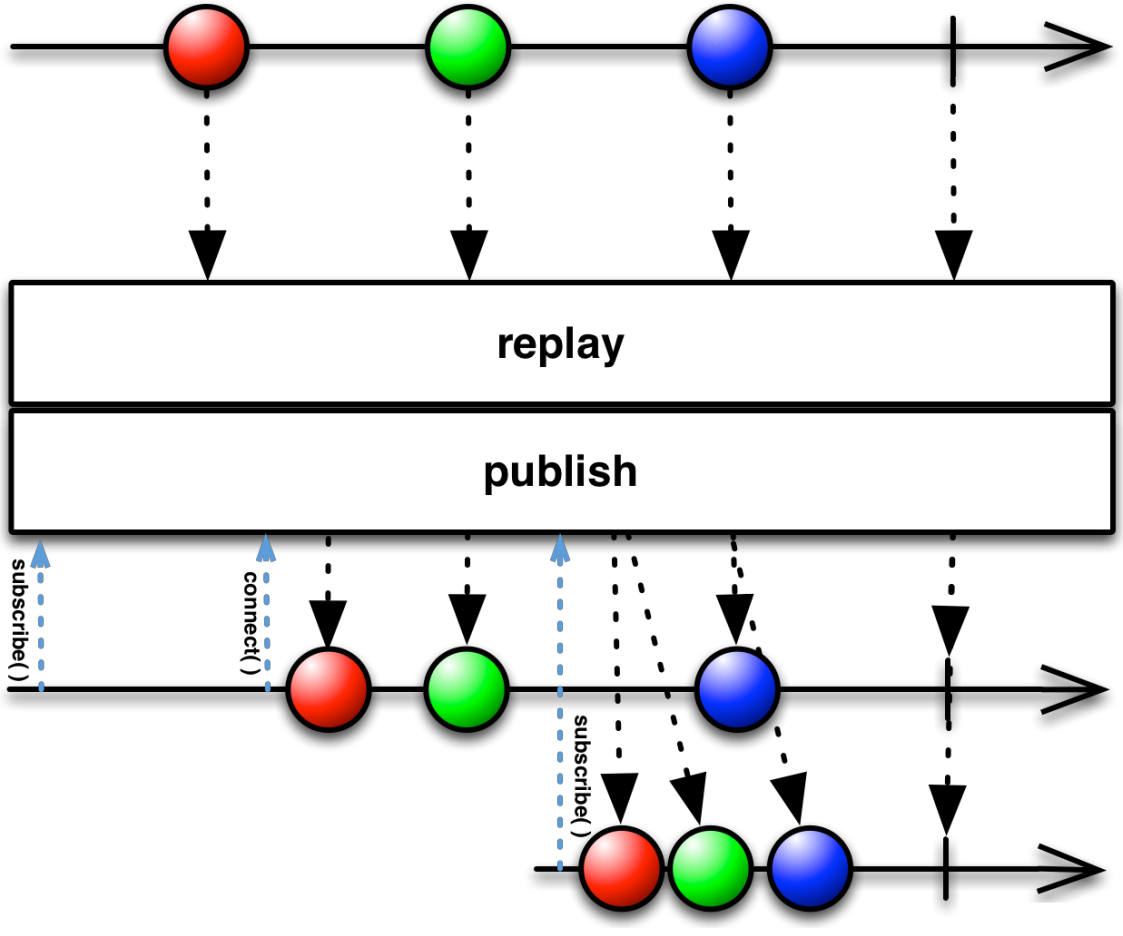
Replay 示意图

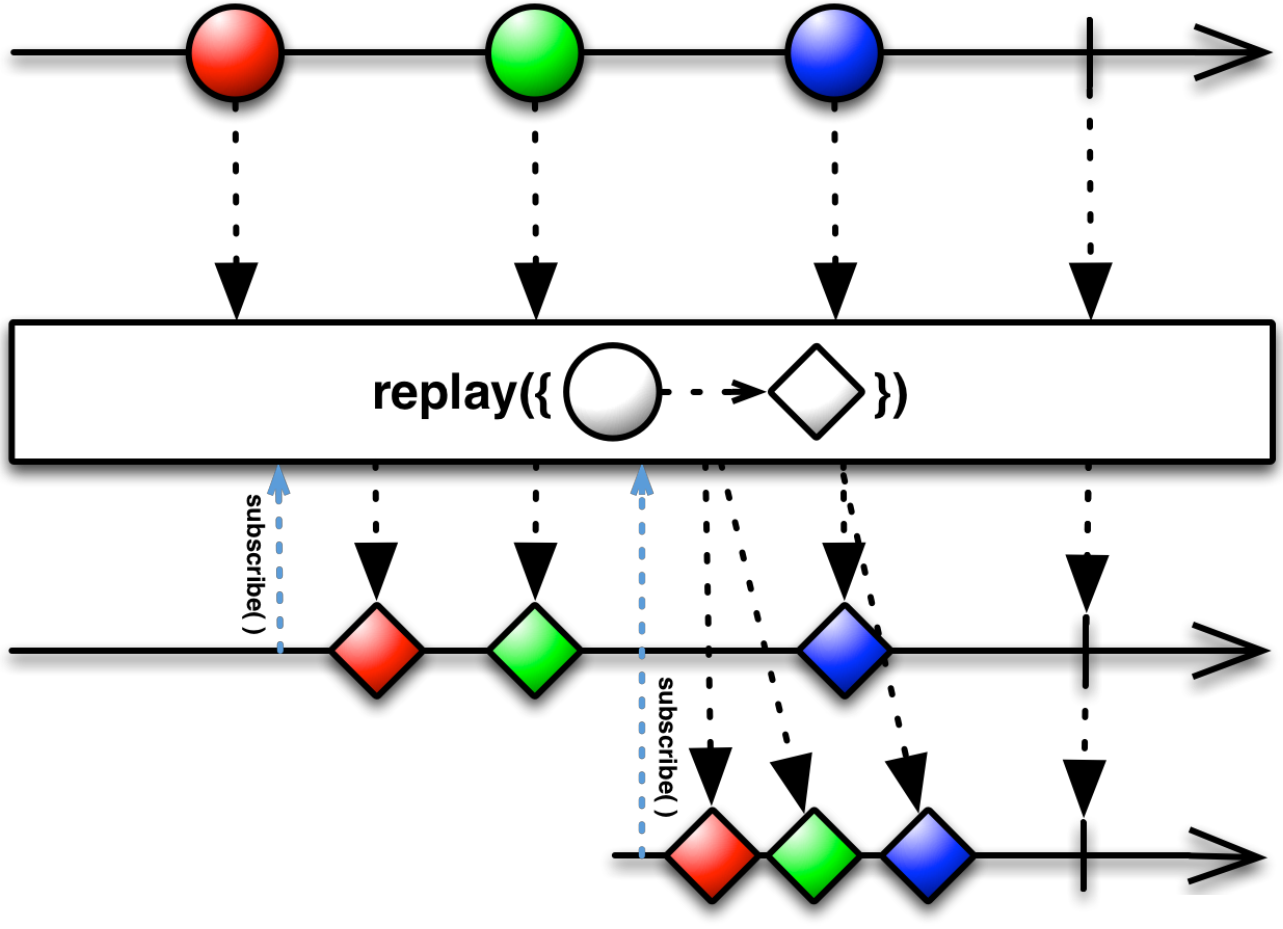
保证所有的观察者收到相同的数据序列，即使它们在Observable开始发射数据之后才订阅


Replay

可连接的 Observable (*connectable Observable*)与普通的 Observable 差不多，不过它并不会在被订阅时开始发射数据，而是直到使用了`Connect`操作符时才会开始。用这种方法，你可以在任何时候让一个 Observable 开始发射数据。

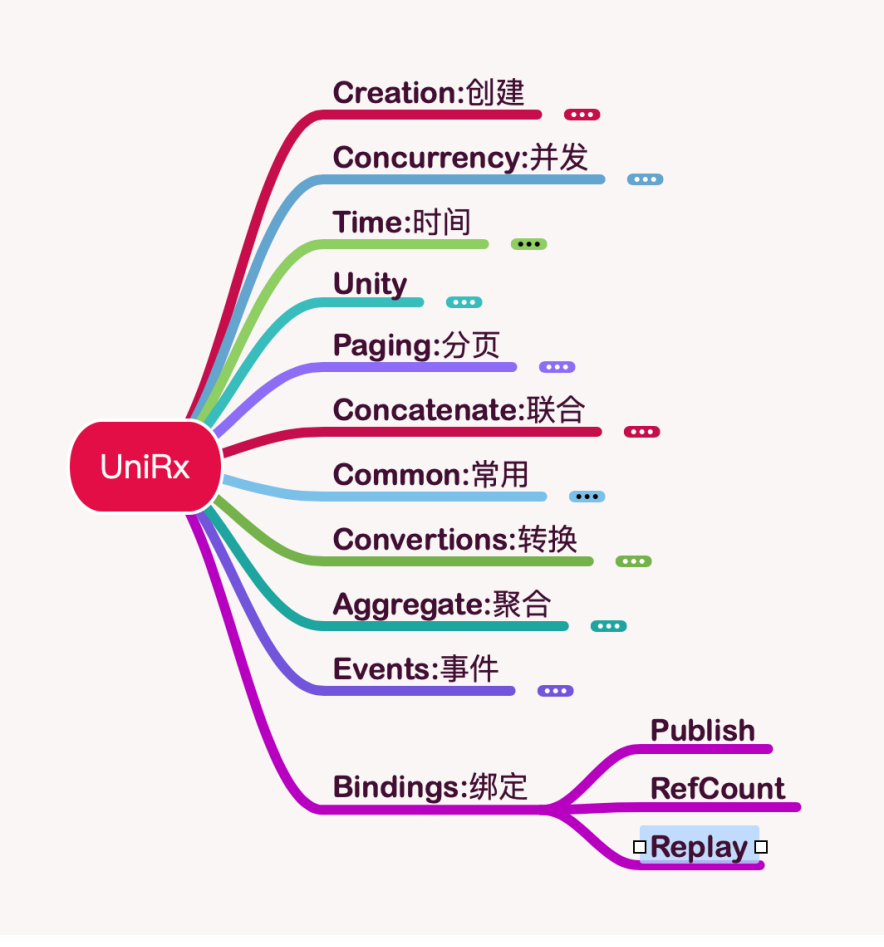
如果在将一个Observable转换为可连接的Observable之前对它使用`Replay`操作符，产生的这个可连接 Observable 将总是发射完整的数据序列给任何未来的观察者，即使那些观察者在这个 Observable 开始给其它观察者发射数据之后才订阅。


replay
publish
subscribe( )
connect( )
subscribe( )


replay({ - - })
subscribe( )
subscribe( )

有一种 `replay`返回一个普通的Observable。它可以接受一个变换函数为参数，这个函数接受原始Observable发射的数据项为参数，返回结果Observable要发射的一项数据。因此，这个操作符其实是`replay`变换之后的数据项。

Replay 所在知识地图中的位置


UniRx
Creation:创建
Concurrency:并发
Time:时间
Unity
Paging:分页
Concatenate:联合
Common:常用
Convertions:转换
Aggregate:聚合
Events:事件
Bindings:绑定
Publish
RefCount
Replay

Replay 示例代码

```
/*********************************************************************************
 * http://sikiedu.com liangxie
 *********************************************************************************/

using System;
using UniRx;
using UnityEngine;


namespace UniRxLesson
{
      public class UniRxReplayExample : MonoBehaviour
      {
            void Start()
            {
                  var period = TimeSpan.FromSeconds(1);
```

```
                var hot = Observable.Interval(period)
                .Take(3)
                .Publish();
                hot.Connect();

                Observable.Timer(period)
                              .Select(_ => hot.Replay())
                              .Do(observable =>
                              {
                                    observable.Connect();
                                    observable.Subscribe(i => Debug.LogFormat("first
subscription : {0}", i));
                              })
                              .Delay(period)
                              .Do(observable => observable.Subscribe(i =>
Debug.LogFormat("second subscription : {0}", i)))
                              .Select(observable => Observable.EveryUpdate()
                                          .Where(_ => Input.GetMouseButtonDown(0))
                                          .Select(_ => observable)
```

```
                                        .First())
                                .Switch()
                                .Do(observable => observable.Subscribe(i =>
Debug.LogFormat("third subscription : {0}", i)))
                                .Subscribe();
            }
        }
}
```

输出结果为

```
first subscription : 1
second subscription : 1
first subscription : 2
second subscription : 2
third subscription : 1
third subscription : 2
```