Creating Sequences of Observables

Edin Kapić

@ekapic www.edinkapic.com

How Sequences Are Generated?

Returning simple results

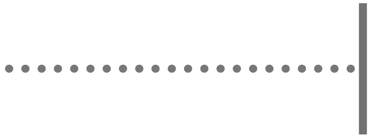
Wrapping an existing result

Writing a generation function

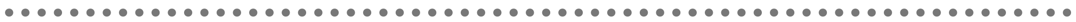
Simple Observables

Useful when the result is trivial enough

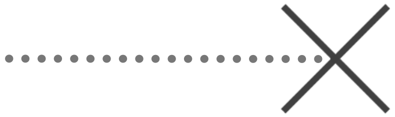
```
Observable.Empty;
```



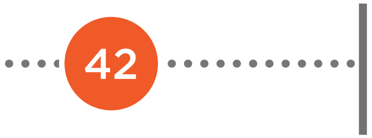
```
Observable.Never;
```



```
Observable.Throw(exception);
```



```
Observable.Return("42");
```



Observable Wrappers

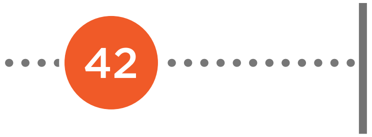
Action

Task

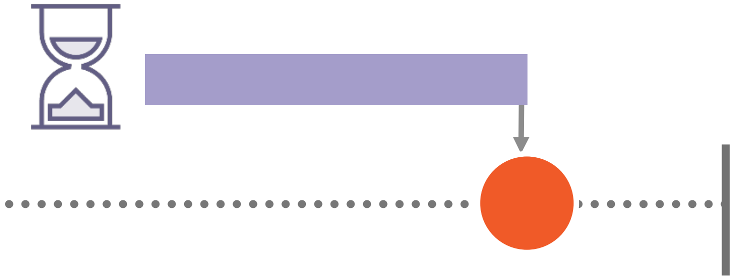
IEnumerable

Events

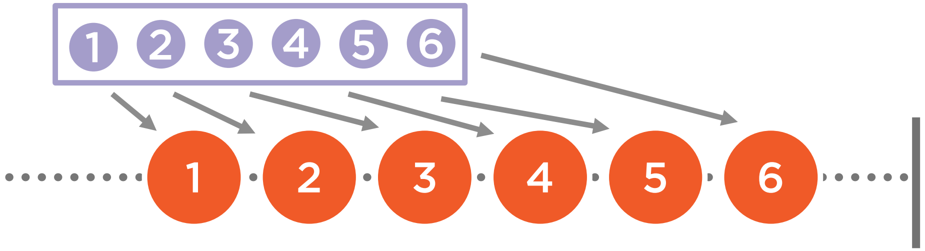
```
Observable.Start(() => 42);
```



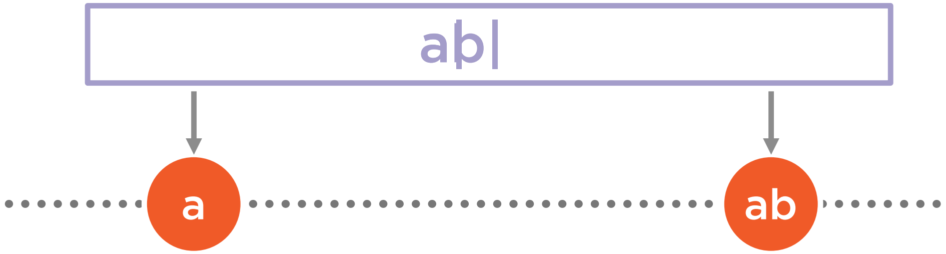
```
Task.ToObservable();
```



```
IEnumerable numbers = ...
numbers.ToObservable();
```



```
Observable.FromEventPattern
(searchTextbox, "TextChanged");
```



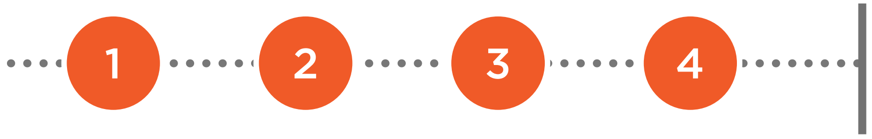
Generation Functions

Range

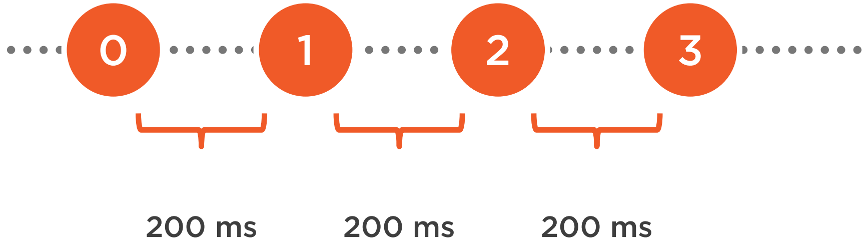
Interval, Timer

Create, Generate

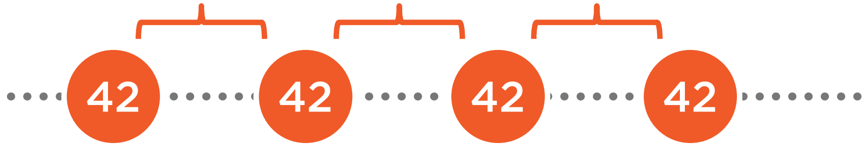
```
Observable.Range(1,4);
```



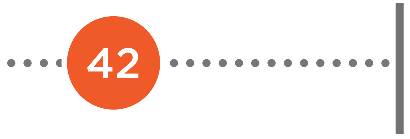
```
// timespan = 200ms
Observable.Interval(timespan);
```



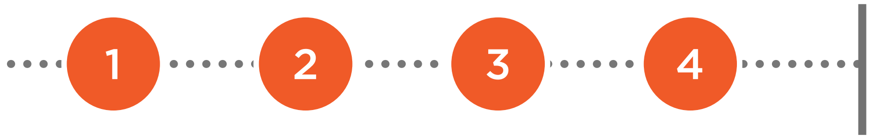
```
Observable.Timer(timespan, ()=>
42);
```



```
Observable.Create<int>(o =>
{
  o.OnNext(42);
  o.OnCompleted();
  return Disposable.Empty;
});
```



```
Observable.Generate(1,
value => value < 5,
value => value + 1,
value => value);
```



Demo

Simple observables

Observable wrappers

Generation functions

Subjects and Subscriptions

```
var subject = new
Subject<string>();
```

- A *subject* is an observable and observer at the same time

```
subject.Subscribe( t =>
Console.WriteLine(t) );
```

- Here the subject acts as an observer

```
subject.OnNext("a");
subject.OnNext("b");
subject.OnNext("c");
```

- Here the subject pushes new values to the observers

```
var subject2 = new
ReplaySubject<string>();
```

- There are specialized subjects that cache or replay values for new subscribers

Hot and Cold Observables

Hot Observables

Emit new values even if there is no observer

UI events

Subjects

Critical information

Cold Observables

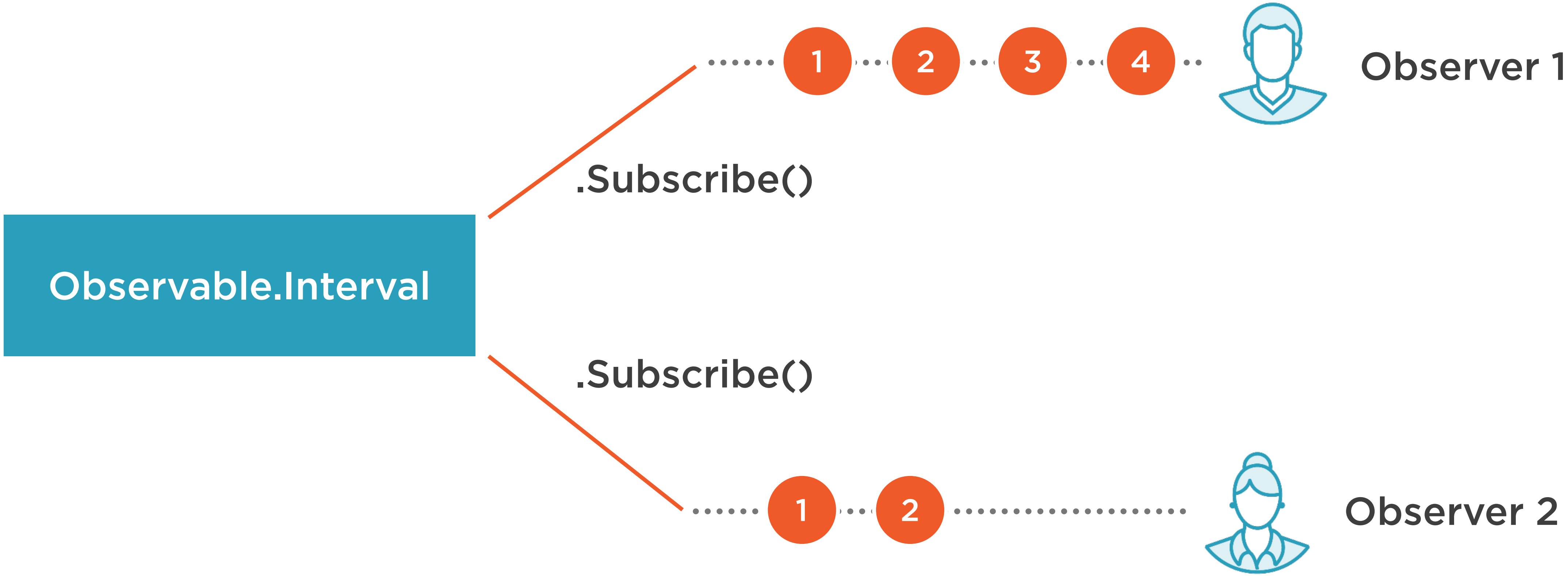
Emit “lazily” only if there is somebody observing

Async wrappers

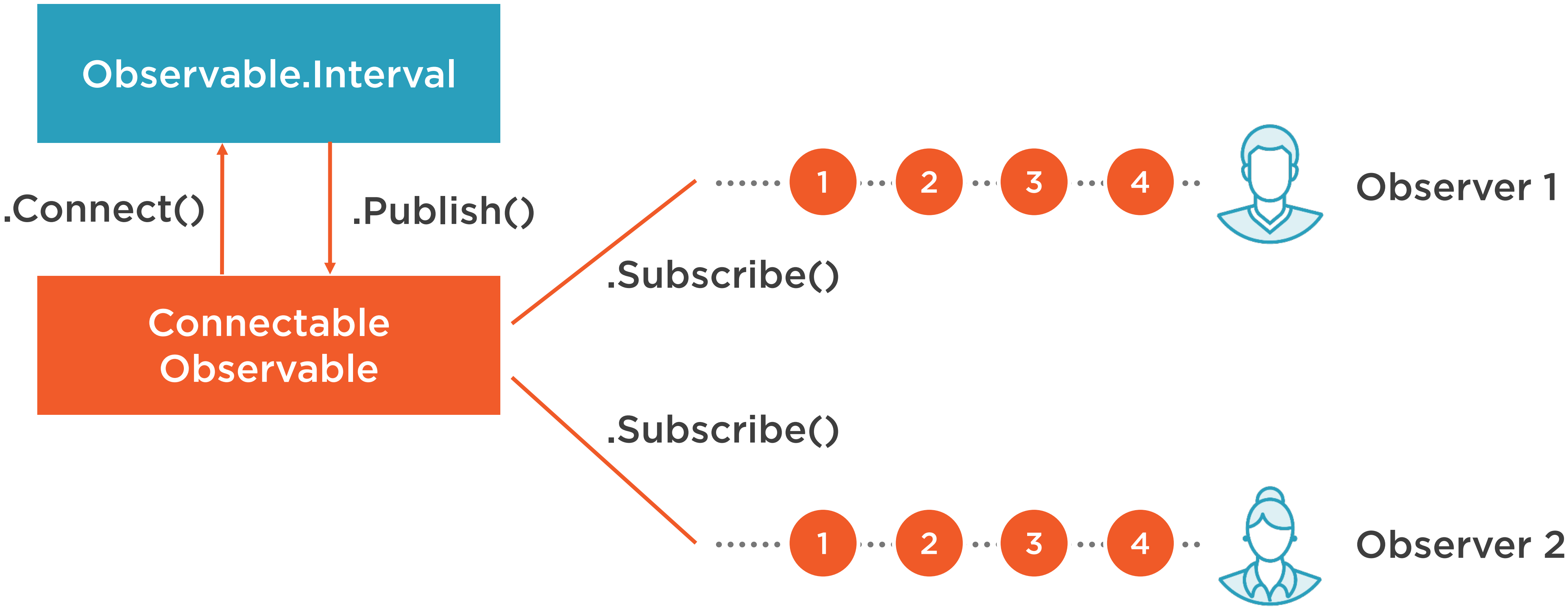
Queues

On-demand sequences

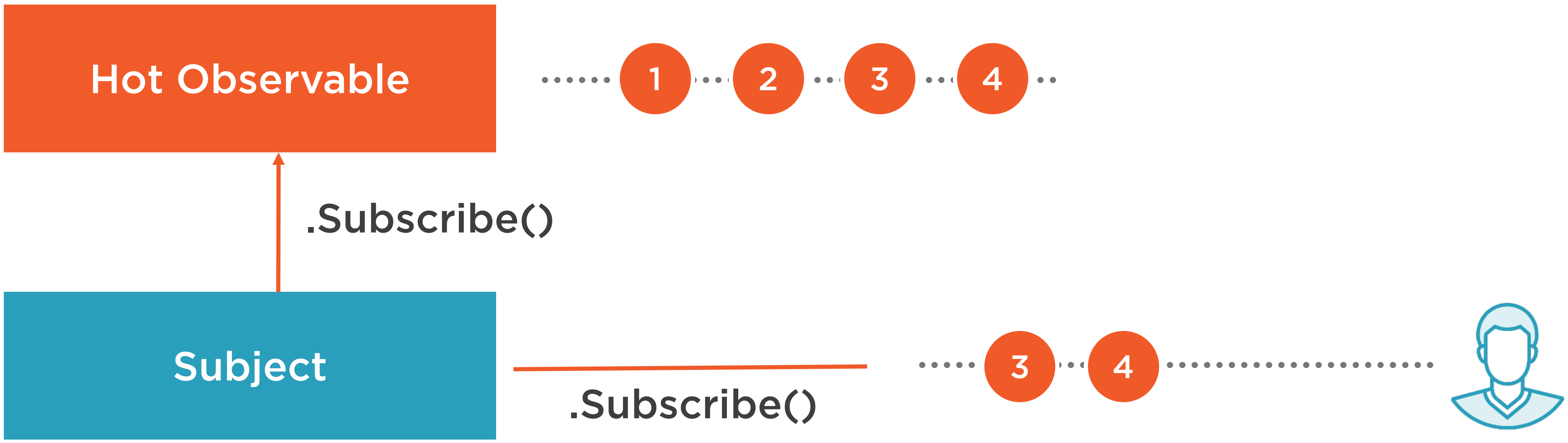
Cold Observables Have Unique Subscriptions



“Warming” Cold Observables



“Cooling” Hot Observables



Demo

Enhancing “Reactive Tickets”

- Adding a command queue
- Adding logging to commands
- Wrapping async methods

Summary

Observables can be made out of

- Simple results
- Other result types
- Generation functions

Subjects are both observables and observers

Hot and cold observables

- Publish a cold observable
- Wrap a hot observable