Reactive Programming

Reaktif Programlama Nedir?

- Günümüzde geliştirilen birçok SPA(Single Page Applications) uygulaması events(olaylar) üzerine işlevsellik göstermektedir.
    - Herhangi bir buton'a tıklarken,
    - Login olurken,
    - Bir istekte bulunurken
- Haliyle bu uygulamalarda olay yönetimi oldukça önemlidir.

Bu Durumu Örneklendirelim

- Bir buton'a tıklandığında tetiklenecek olayın bir kereliğine mahsus olmasını istiyorsak eğer;

JavaScript

```javascript
const button = document.getElementsByTagName("button")[0];

button.addEventListener("click", (e) => {
  e.currentTarget?.removeEventListener(e.type, () => { });
  console.log("Merhaba");
});
```

Görüldüğü üzere reaktif programlama ile bu gibi olayları daha basit, okunaklı ve dinamik geliştirebilmekteyiz.

RxJS

```javascript
import { fromEvent } from "rxjs";
import { first } from "rxjs/operators";

const button = document.getElementsByTagName("button")[0];

fromEvent(button, "click").pipe(first())
  .subscribe(x => console.log("Merhaba"));
```

Peki nedir bu Reaktif Programlama

Günümüzde çoğu programlama dili prosedürel programlama temelinde çalışmaktadır.

Bu yaklaşım ile, problemin çözüm aşamalarını yazılımcı adım adım kodlar, mkaine ise adım adım bu aşamaları çalıştırarak veriler işlenir ve sonuç üretilir.

Oluşturulan kod yapısı ardışıl olarak oluşturulmuş aşamalara ve sürecin sonuna odaklanmıştır ve sarsılmaz bir şekilde bu süreç işlemektedir.

Eğer ki, bu ardışıl yapıdaki sürece farklı bir değer/parametre/yan etki(side effect) katılmaya çalışırsa yapı buna müsaade etmeyecek, sıralama ya baştan yapılması gerekecek ya da tüm çalışmalar geri alınarak(callback) hesap tekrardan uygulanacaktır.

İşte burada sürece katılmaya çalışan farklı değerleri/parametreleri/yan etkileri kodun akış sürecindeki herhangi bir 'T' noktasında işleme alabilecek şekilde yapılan tasarımlara sahip uygulamalara ya da kod parçaççıklarına Reaktif Programlama denmektedir.

Reaktif programlama, bir akış içerisi nde bir datanın çeşitli evrelerden g eçerek sonuca ulaşmasıdır.

Her evrede data bir dönüşüme ya d a bir başka deyişle yan etkiye(side effect) uğrar ya da uğramaz.

OOP İle Reaktif Programlama Arasındaki Fark Nedir?

- OOP ile yazılımlar, gerçek hayattaki olguları nesneler ile modelleyerek tasarlamaktadır.
- Reaktif programlamada ise gerçek hayattaki olaylar baz alınarak tasarım gerçekleştirilmektedir.

X = Y + Z Denklemini Ele Alırsak!

- Prosedürel programlama mantığında hareket edersek, denklemden sonraki süreçte Y veya Z değişkenlerinden birinin değeri değişirse bu değişiklik X değişkenine yansımayacaktır.
- Çünkü ardışıl işlem sıralamasında X değişkeninin değeri çoktan belirlenmiştir ve artık bu değer Y ve Z'den bağımsızdır.
- Reaktif programlamada ise Y ve Z değişkenlerinin değeri değiştiğinde X değişkeninin değeride bu değişime göre şekillenecektir.
- Çünkü ardışıl işlem sıralamasından öte olaylar baz alınacağıdan dolayı sistem hangi noktada olursa olsun x değişkeninin değerini Y ve Z değişkenlerinin toplam sonucuna bağlamaktadır ve bu değişkenlerin değerleri değişse dahi bu olay üzerine X değişkenide tepkisini yeni değişikler üzerinden kendi değerini güncelleyerek verecektir.

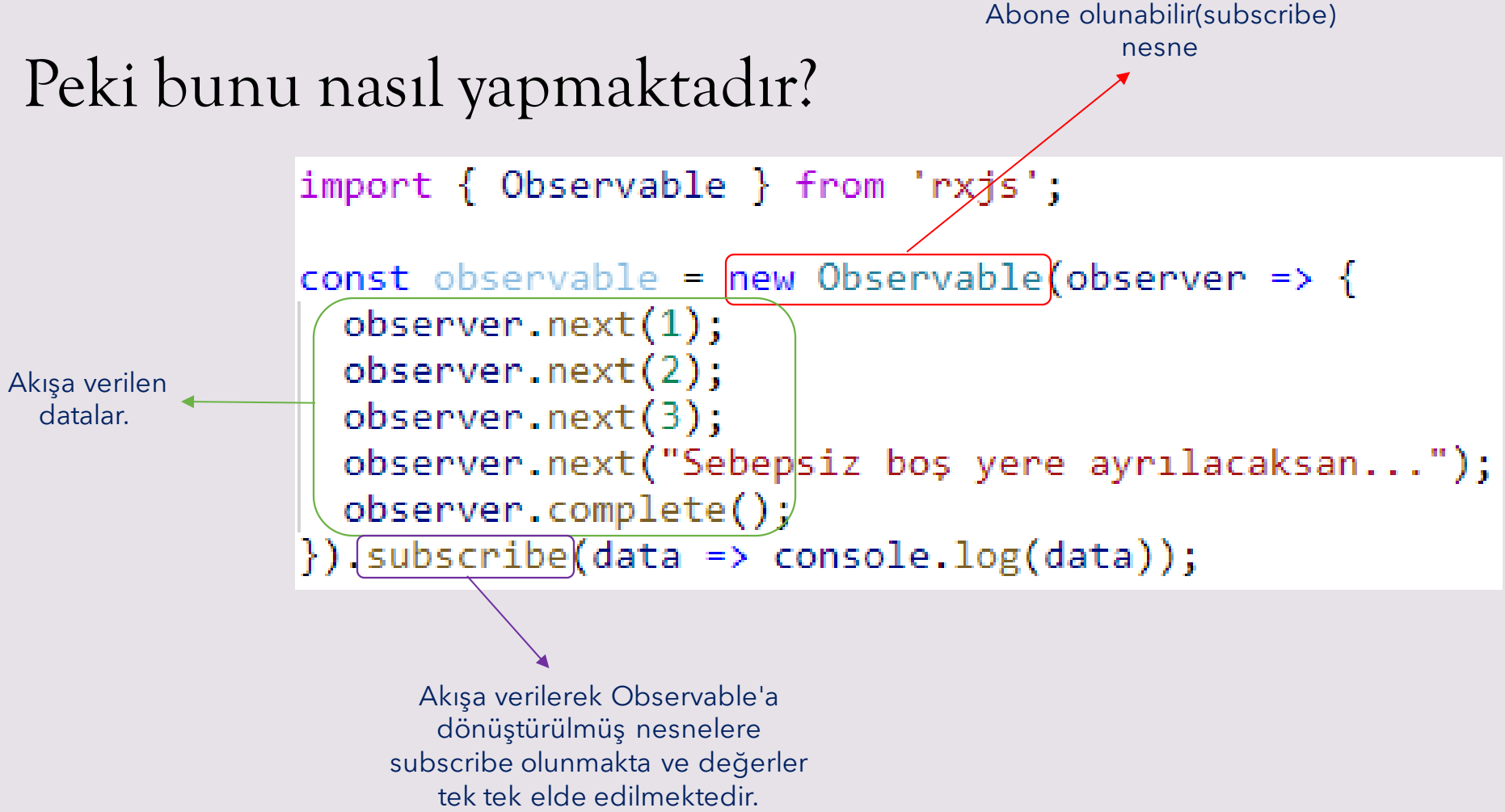
RxJS

RxJS Nedir?

RxJS

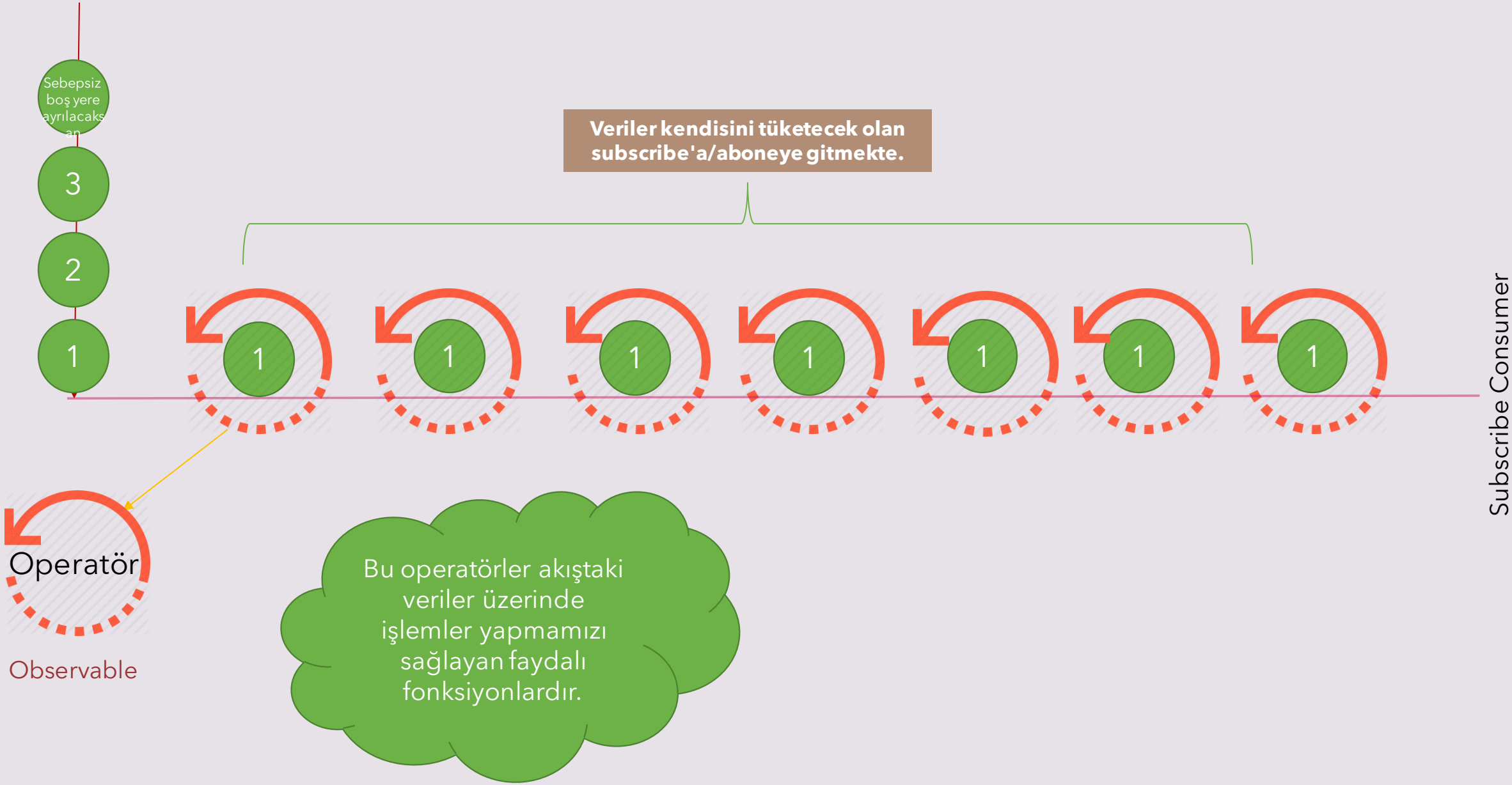
RxJS, olay ve veri kaynaklarını abone olunabilir(subscribable) nesnelere(observable) dönüştürüp, bunlar üzerinde operatörler vasıtasıyla dönüşümler/etkiler gerçekleştirilebilmesini ve sonucu elde edebilmemizi/tüketebilmemizi sağlayan JavaScript ile yazılmış bir Reaktif Programlama kütüphanesidir.

Peki bunu nasıl yapmaktadır?




```
import { Observable } from 'rxjs';

const observable = new Observable(observer => {
  observer.next(1);
  observer.next(2);
  observer.next(3);
  observer.next("Sebepsiz boş yere ayrılacaksan...");
  observer.complete();
}).subscribe(data => console.log(data));
```

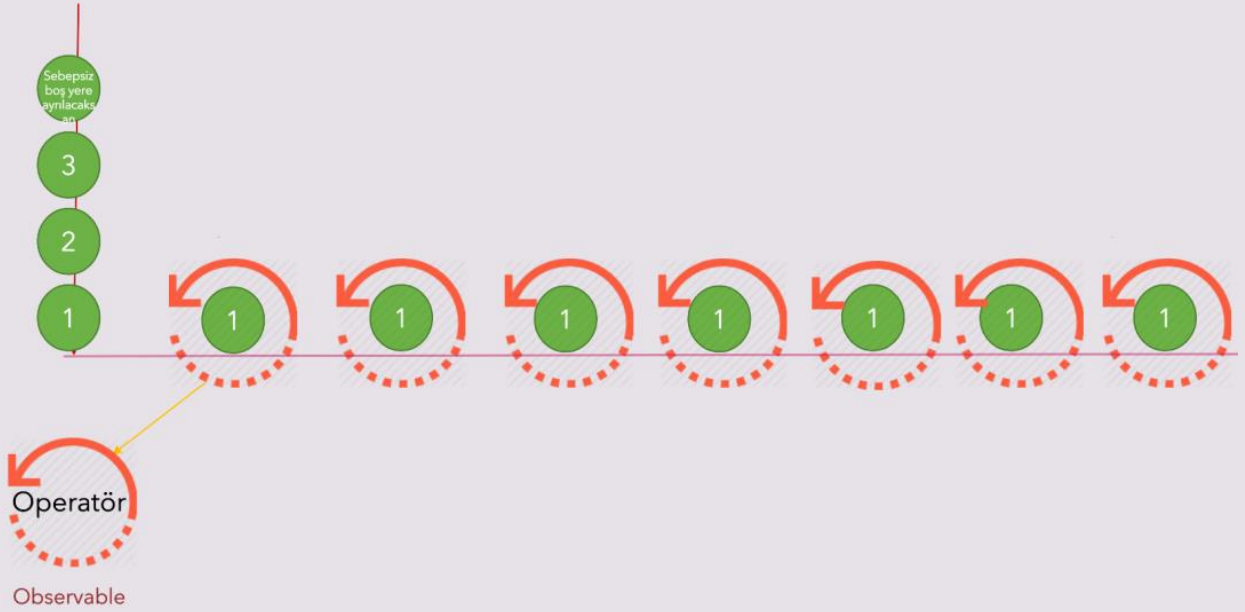

Sebepsiz boş yere ayrılacaks an
3
2
1
Veriler kendisini tüketecek olan subscribe'a/aboneye gitmekte.
1
1
1
1
1
1
1
Subscribe Consumer
Operatör
Observable
Bu operatörler akıştaki veriler üzerinde işlemler yapmamızı sağlayan faydalı fonksiyonlardır.

RxJS

Observable Nedir?

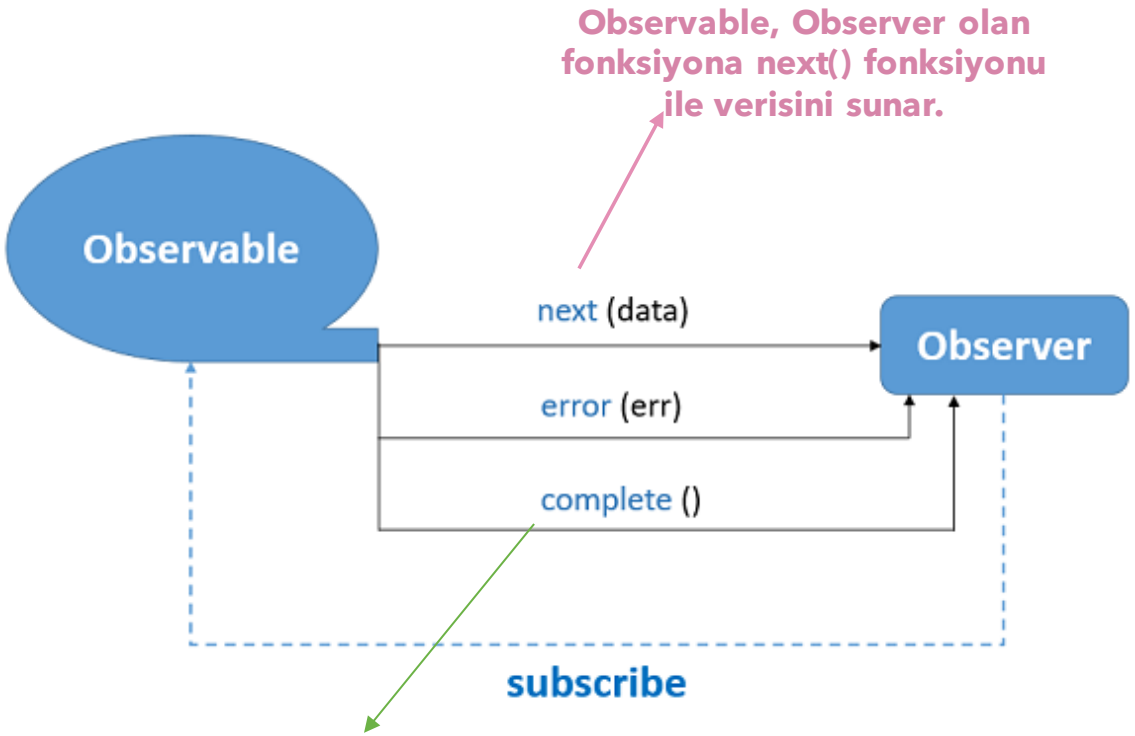
Observable

- RxJS, Reaktif Programlamayı Observable nesnesi sayesinde gerçekleştirmektedir.
- Süreçte dönüşüme uğratılacak dataların akışını sağlayan nesnedir.
- Yani oluşturulan veri akışıdır.



RxJS

Observer Nedir?

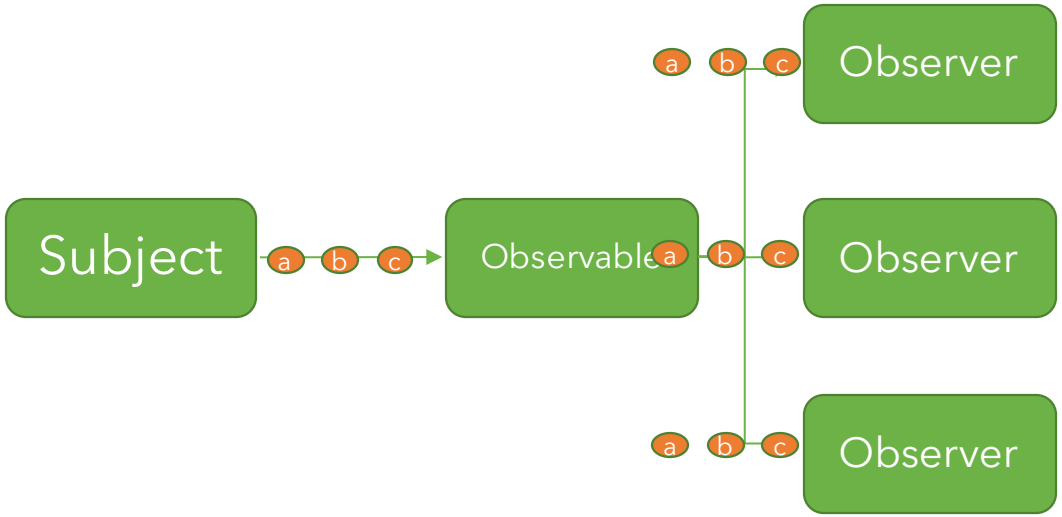


Observer

- Observable nesnesindeki akışı izleyen/tüketen fonksiyondur.
- Akışı izleme olayına subsciption(abonelik) denir.

RxJS

Subject Nedir?



Subject

- Birden fazla Observer'ın bir Observable'a abone olması durumudur.
- Özelleştirilmiş Observable gibi de düşünülebilir.

Subject Türleri

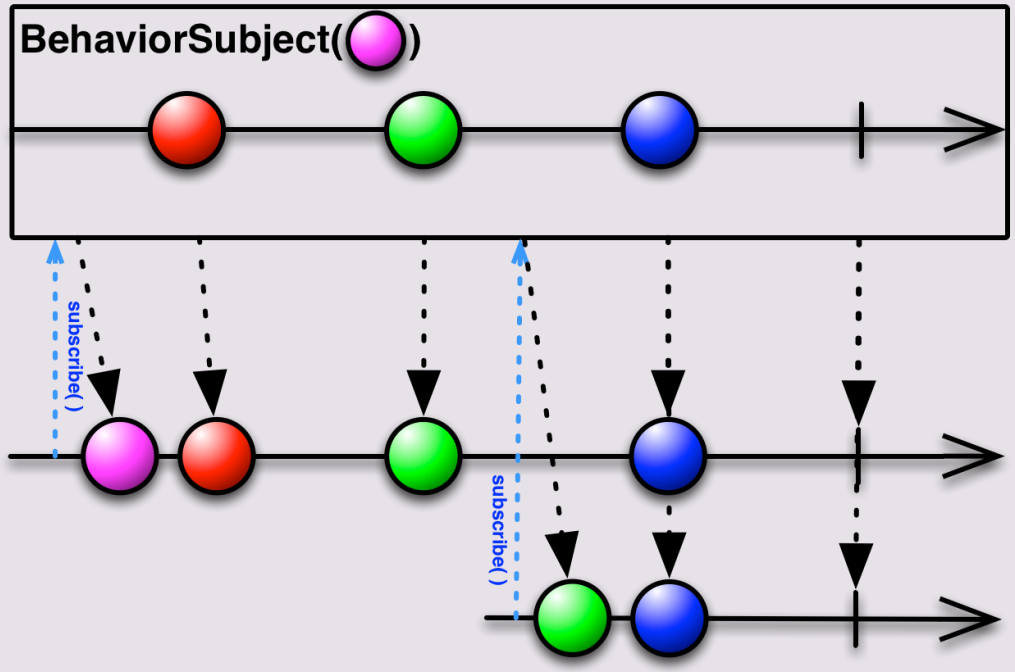
Subject;

- BehaviorSubject
- ReplaySubject
- AsyncSubject

olmak üzere üç çeşittir.

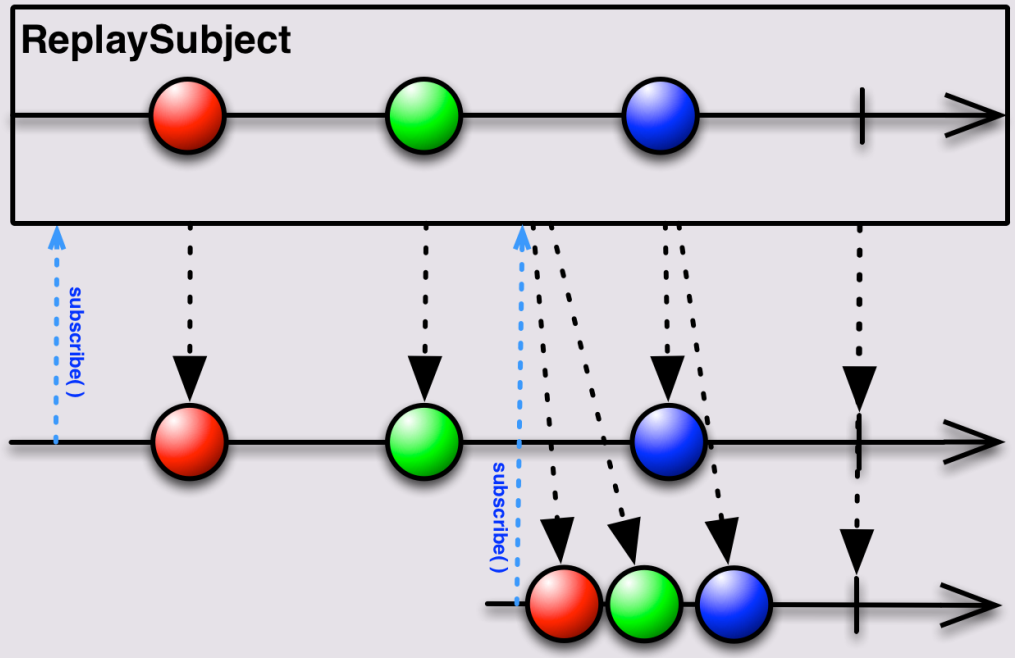
BehaviorSubject

- Akışa abone olan Observer'ın, akıştaki bir önceki veriden başlayarak gelen verileri almasını sağlar.



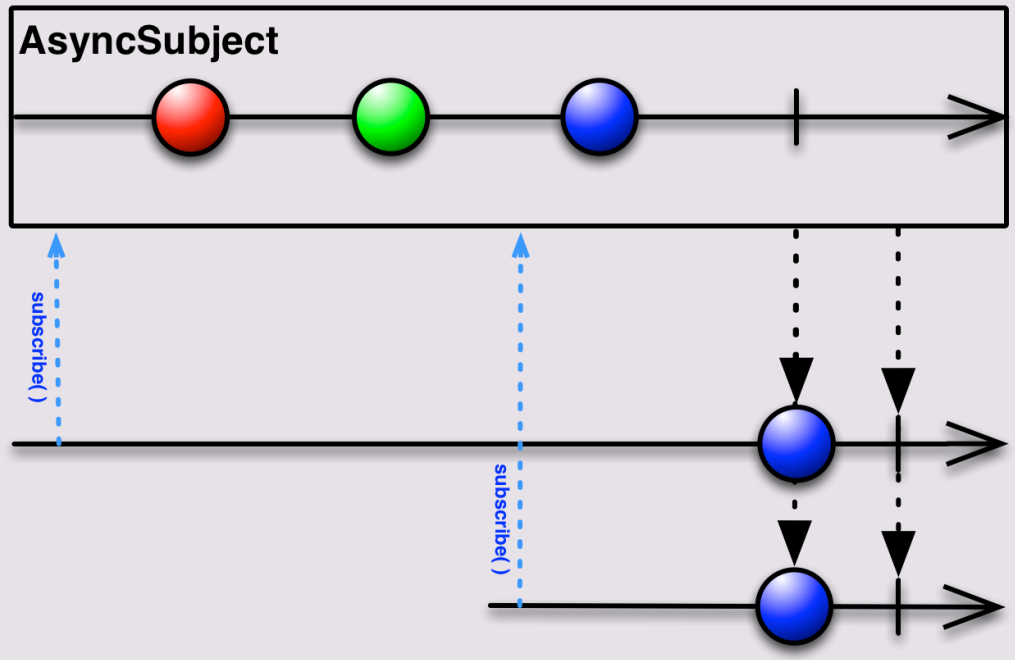
ReplaySubject

- BehaviorSubject'te Observar , girdiği akışta bir önceki veriyi alarak devam ediyordu. ReplaySubject ise istenildiği kadar öncedeki veriyi alabilmektedir.



AsyncSubject

- Akıştaki son değerin alınabilmesi için kullanılan Subject türüdür.
- Akışta sonuncu veriyi anlayabilmek için **complete()** fonksiyonunun tetiklenmesini bekler.

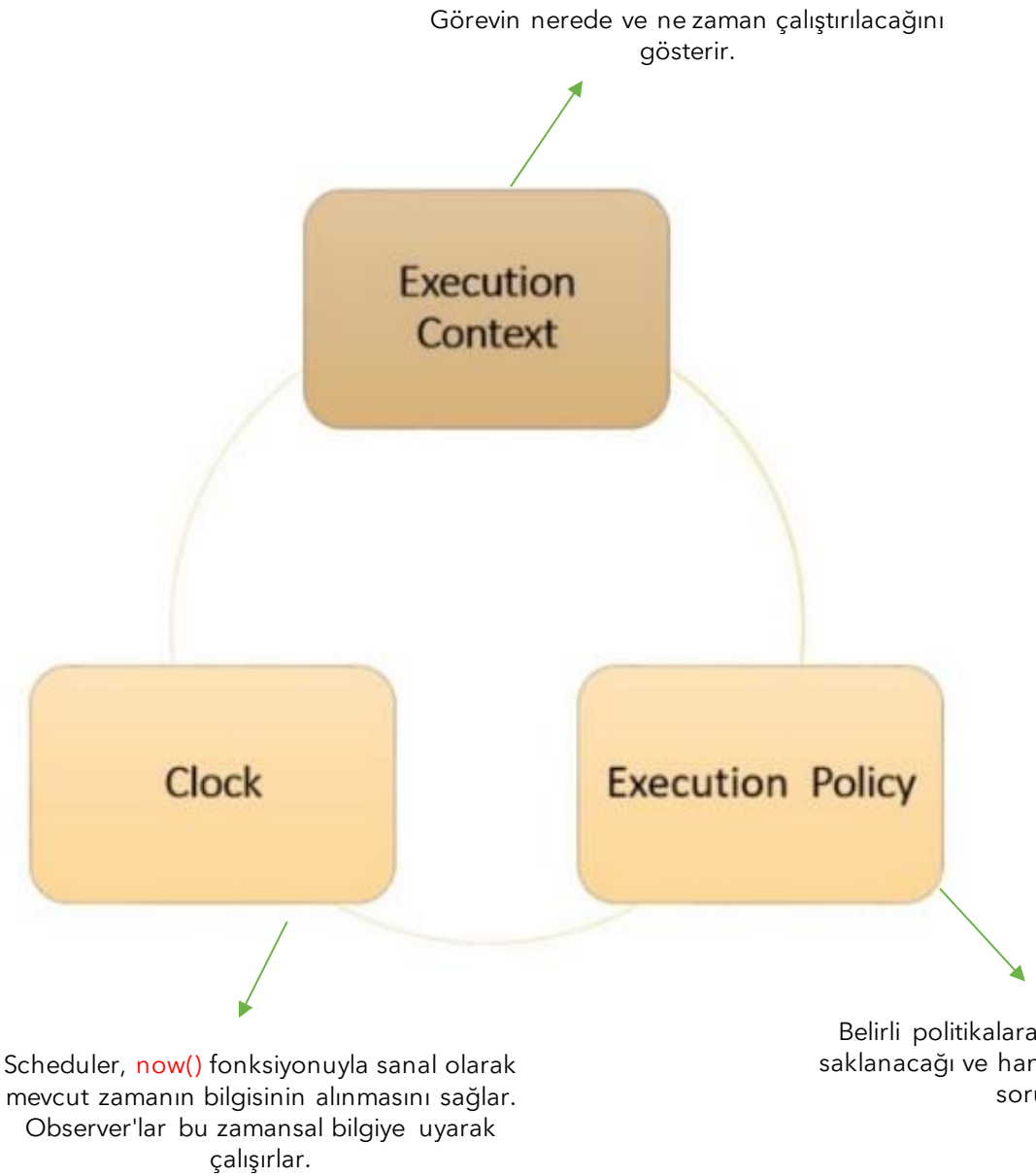


RxJS

Scheduler Nedir?

Scheduler

- Bir aboneliğin ne zaman başlayacağını ve verinin ne zaman observer'lara iletileceğini kontrol eden nesnedir.



- Scheduler, üç bileşenden oluşur.

Scheduler Türleri

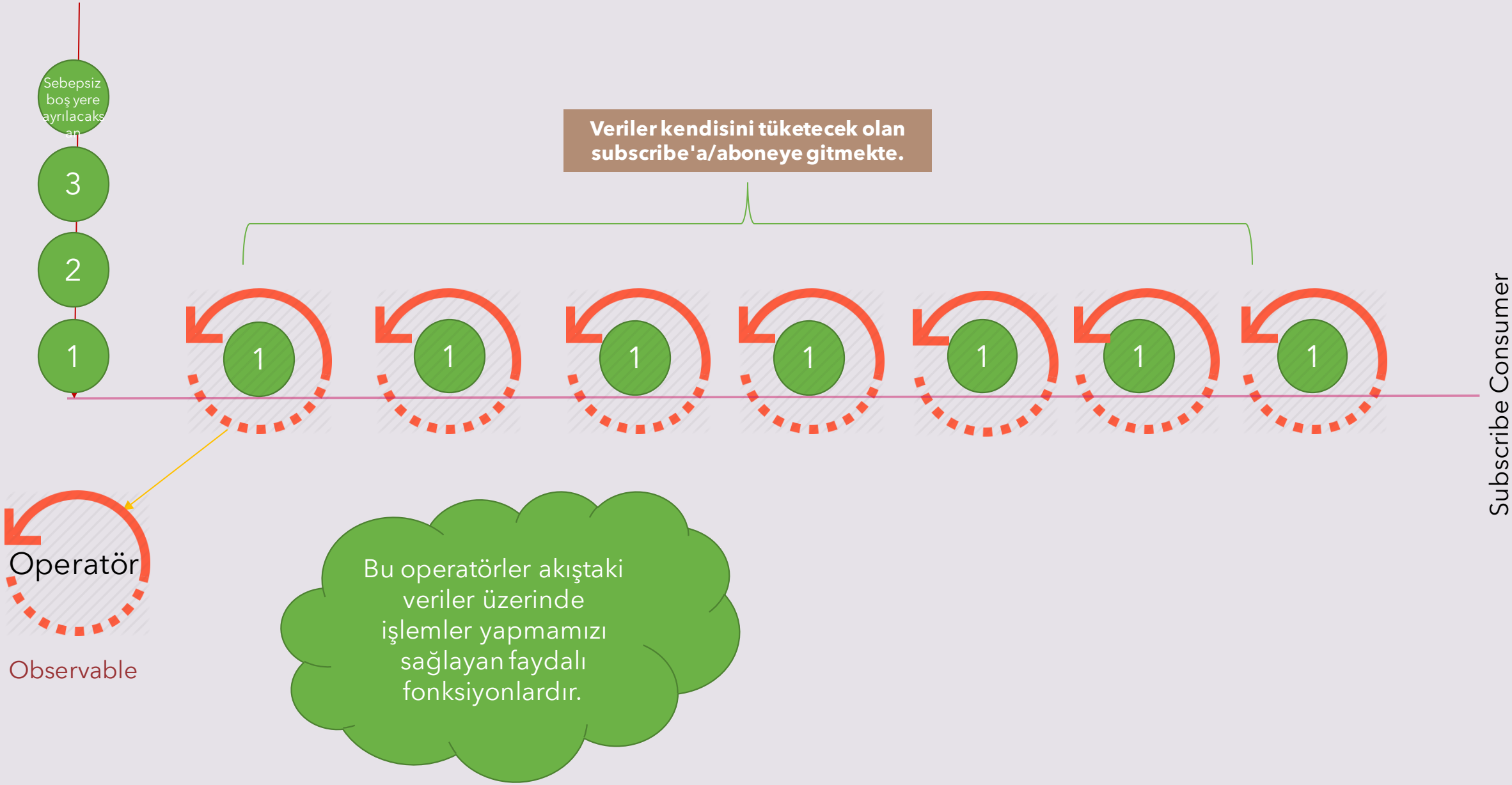
SCHEDULER	KULLANIM AMACI
queueScheduler	Yapılacak işi kuyruğa alır. Belli bir sıra ile arka arkaya çalıştırılacak işler için kullanılır.
asapScheduler	Micro tasklar için kullanılır. Mevcut işten sonra ve bir sonraki işten önce çalışır. Asenkron dönüşümler için kullanılır.
asyncScheduler	Zaman tabanlı işlemlerde kullanılır. SetTimeout fonksiyonu ile çalışır haliyle veriler observer'a asenkron iletilir.
animationFrameScheduler	Browserın, bir sonraki içeriği yeniden boyamadan hemen öncesinde kullanılır. Akıcı animasyonlar üretmek için kullanılır.

RxJS

Operators Nedir?

RxJS

RxJS, olay ve veri kaynaklarını abone olunabilir(subscribable) nesnelere(observable) dönüştürüp, bunlar üzerinde operatörler vasıtasıyla dönüşümler/etkiler gerçekleştirilebilmesini ve sonucu elde edebilmemizi/tüketebilmemizi sağlayan JavaScript ile yazılmış bir Reaktif Programlama kütüphanesidir.


Sebepsiz boş yere ayrılacaks an
3
2
1
Veriler kendisini tüketecek olan subscribe'a/aboneye gitmekte.
1
1
1
1
1
1
1
Subscribe Consumer
Operatör
Observable
Bu operatörler akıştaki veriler üzerinde işlemler yapmamızı sağlayan faydalı fonksiyonlardır.