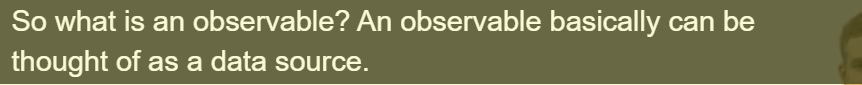
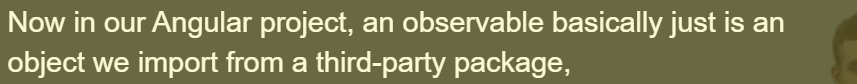
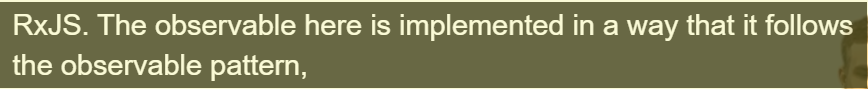
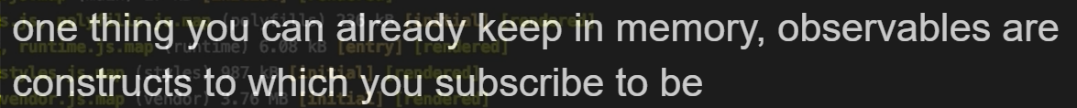
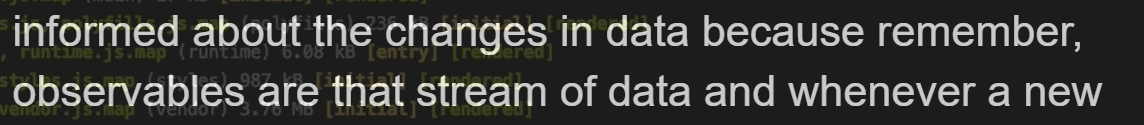
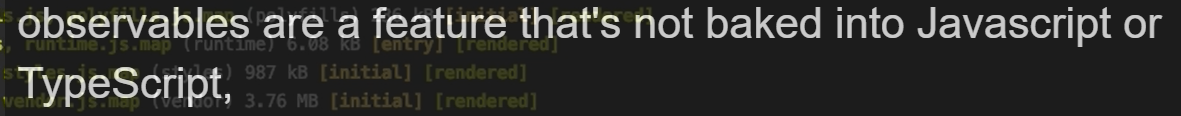
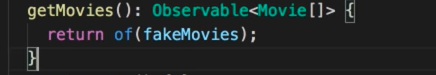
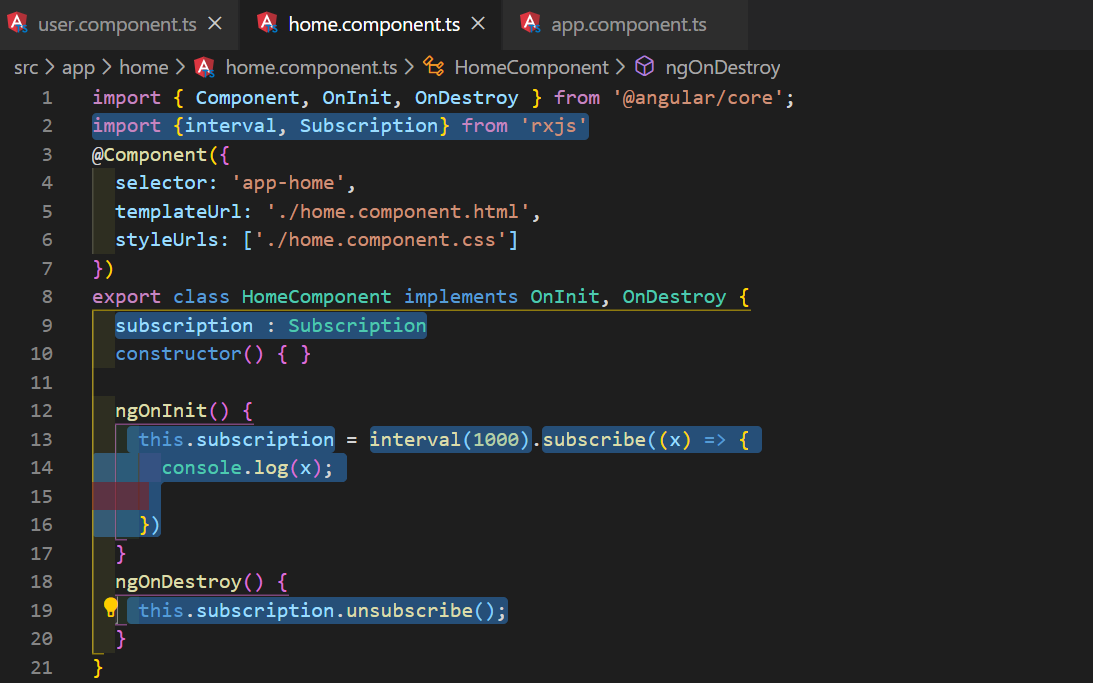
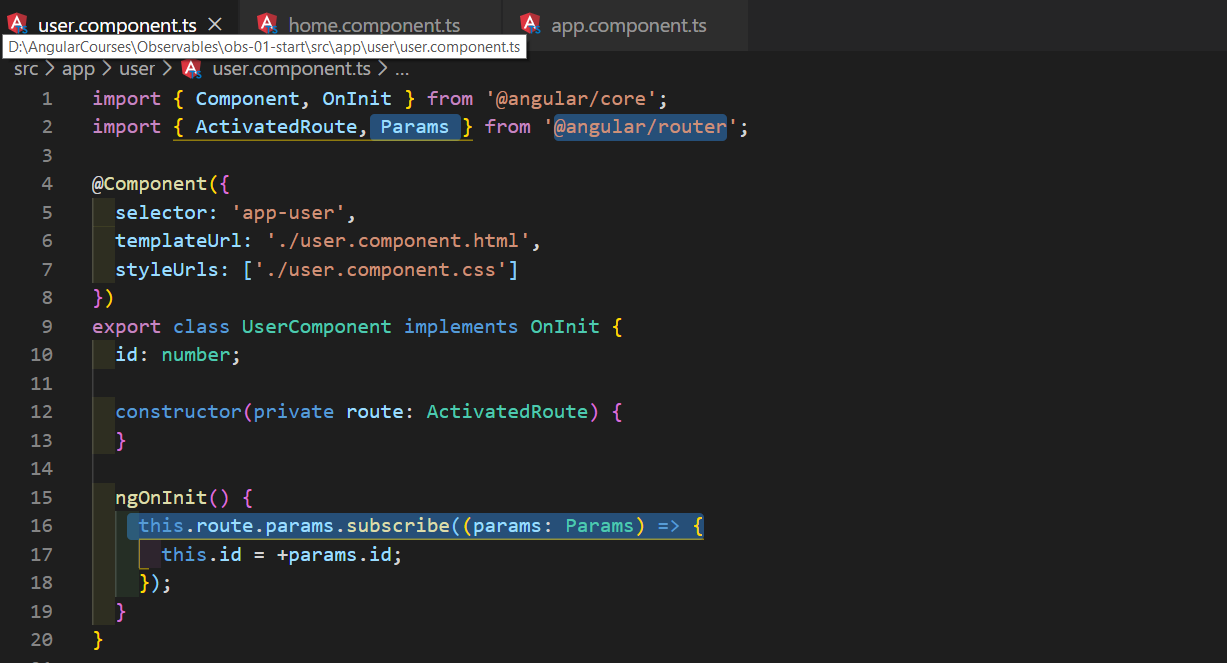
SECTION 13: **OBSERVABLES**

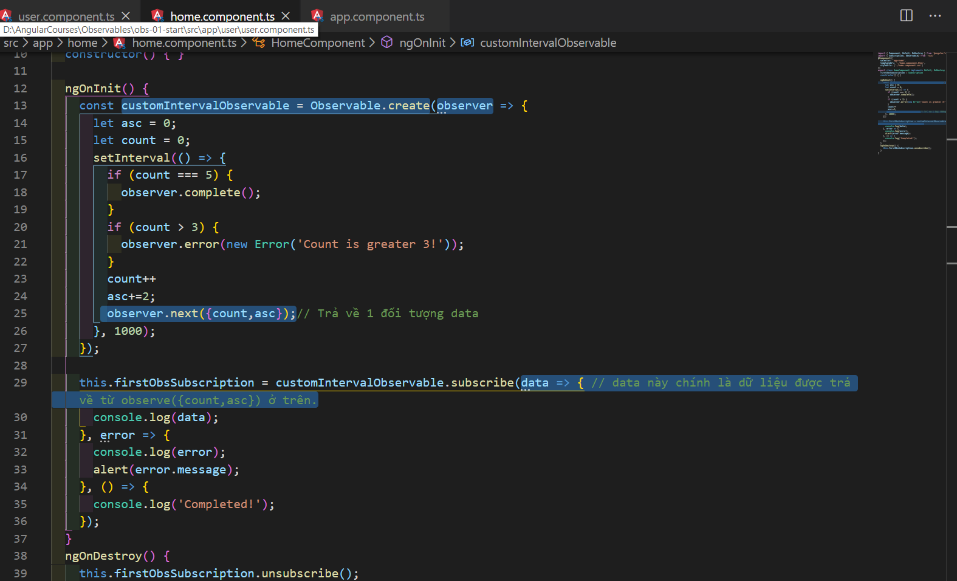
* 1. What ?
* 
* 
* 
* 
* 
* 
* 
* 
  1. **Getting Closer to the Core of Observables :** 
     1. **Observable object là gì?** Là 1 object mà nó bị thay đổi, nghĩa là khi nào nó có sự thay đổi thì nó sẽ nhảy vào trong 1 cái hàm, mà từ cái hàm ấy ta cập nhật lại giao diện.
        1. -Hàm getMovies() là 1 object Observable, trả về 1 mảng các Movie.
        2. - Ta theo dõi hàm getMovies(), nghĩa là mỗi khi mảng Movie[] do hàm getMovies() trả về có sự Thay Đổi, nó sẽ chạy hàm Subscribe() để cập nhật lại giao diện.
        3.  -Gọi nó trong hàm onInit() nghĩa là, vừa tạo component => ta gọi hàm subscribe() để theo dõi biến Observable ‘getMovies’ => Như vậy là nó sẽ chạy suốt, cứ thay đổi là chạy hàm Subscribe(), chứ không phải subscribe() 1 lần rồi kết thúc.
     2. **Tạo thử 1 observables**:

-Hàm Interval() là 1 observable, cứ sau 1s là nó bắn ra 1 sự kiện và được subscribe().

-Hàm Subcribe() trả về 1 Subscription, và ta phải unsubscribe() biến Observable đó nếu không nó sẽ chạy mãi dù ta có chuyển qua trang khác cũng vậy, khiến rò rỉ vùng nhớ.

-Những Observables được import từ @Angular sẽ được tự động unSubscribe().

* + 1. **Hiểu rõ hơn về 1 Observable:**



GIẢI THÍCH HÌNH TRÊN:

-Line 13-27:

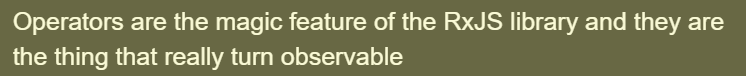
+ là 1 Observable và các hành vi của Observable đó.

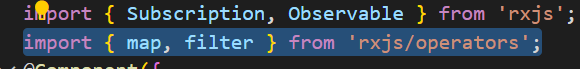
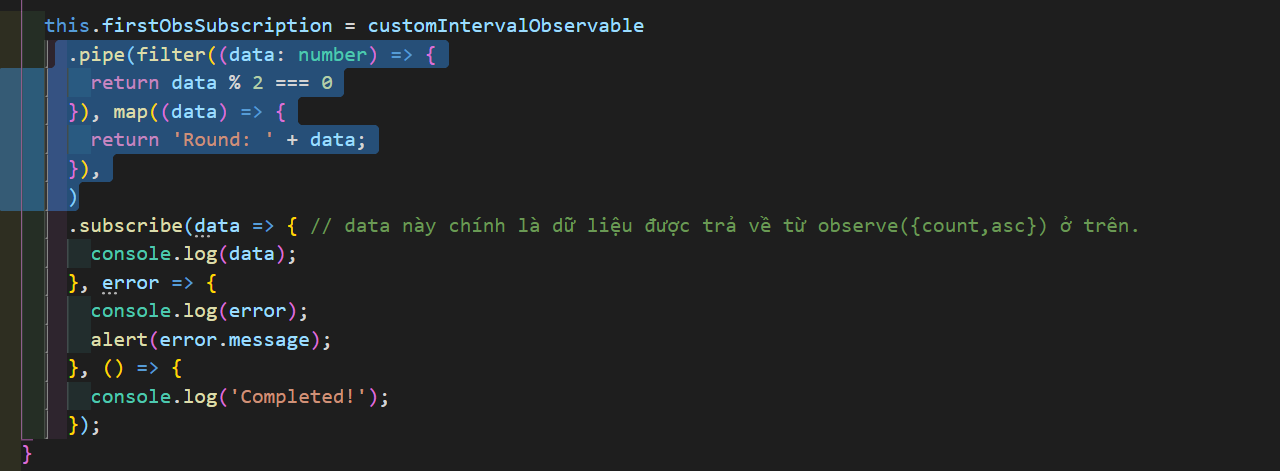
+ mọi dữ liệu(*asc, count*) được merge lại thành 1 đối tượng, sau đó sẽ được gán cho **observer** (line 25), sau đó observer sẽ trả về cho Observable.

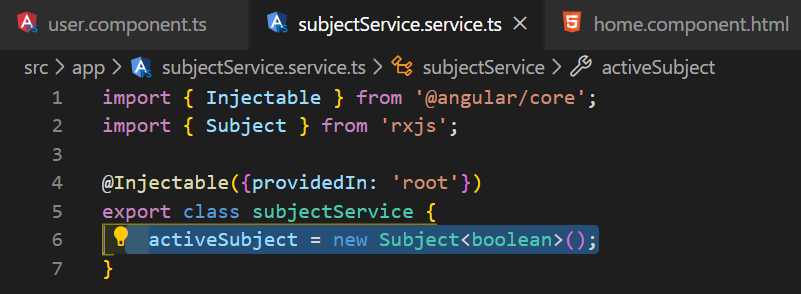
+ **Observer** cũng có thể qui định khi nào thì bị lỗi(error), khi nào thì thành công(complete) và sẽ thông báo cho Observable.

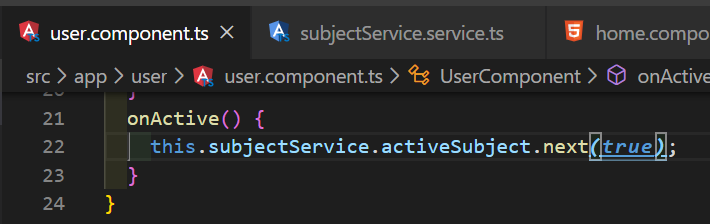
+Khi subcribe Observale ấy, ta có thể giải quyết logic khi observable thông báo Error, Complete như line 29-36.Những cái đó gọi là handler Function.

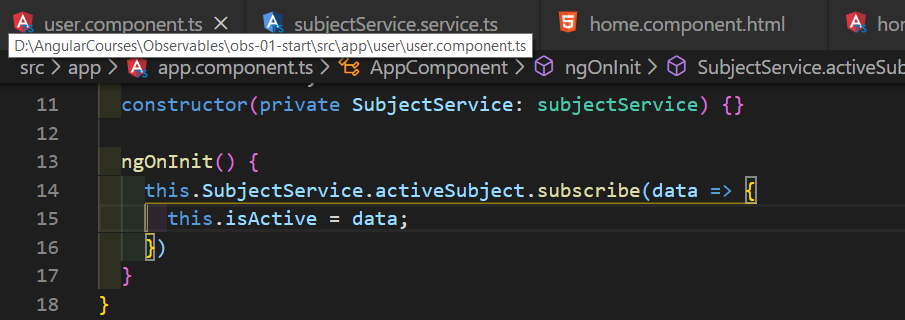
* 1. **Tìm hiểu về các Operators trong RxJS:** Tìm hiểu thêm về RxJS: <https://rxjs-dev.firebaseapp.com/guide/overview>
     1. **What ?:** Nghĩa là ta có thể thay đổi Format, thay đổi cấu trúc, thay đổi giá trị của biến Observable bằng .pipe()(build-in RxJs) và hàng tấn Operators khác như map(), fitler()…..<https://rxjs-dev.firebaseapp.com/guide/operators>



* + 1. **How?:** 
       1. -Import các Operators trong thư viện RxJS/Operators.
       2. - Có thể truyền không giới hạn số lượng Operators. (**giải thích hình trên:** chỉ nhận các phần tử số chẵn từ mảng(hàm fitler), sau đó mỗi Phần tử sẽ chèn thêm chuỗi ‘round’ trước data.) 2 cái biến ‘data’ của hàm pipe() và hàm subscribe() trên là 1.
  1. **Subject:** 
     1. **WHAT?** is a special type of Observable <https://rxjs-dev.firebaseapp.com/guide/subject>. Dùng y như 1 biến EventEmitter<>().
     2. **Lưu ý:** Nên sử dụng hoàn toàn **Subject** thay cho **EventEmiter**<>() (**recommnened**), bay giờ chỉ sử dụng EventEmitter cho việc @Ouput().
     3. **HOW?**

 -Khai báo 1 Subject trong Service, Nhận vào 1 arg Boolean

- Khi click vào 1 button, nó sẽ bắn, phát ra 1 tính hiệu, trả về cho biến Subject(true) ở Service, như vậy biến activeSubject sẽ bị thay đổi.

-khi biến activeSubject ở Service thay đổi, hàm subcribe() sẽ nhận ra và thực thi các handle function để cập nhật output cho người dùng.