

ESEMPIO: HttpClient, Angular

`http.get(...).subscribe();`

- Sappiamo che esegue una chiamata Http
- Sappiamo che la chiamata viene effettuata solo al subscribe
- Sappiamo che emette al massimo un valore, e poi completa

Quindi, presumiamo che:

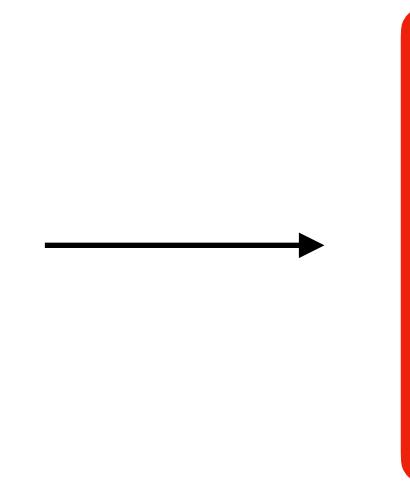
- Sia un observable cold
- Due subscribe allo stesso observable generino **due** chiamate distinte
- Dobbiamo evitare di usare una async pipe direttamente sull'observable, perché ogni ciclo di Change Detection farebbe ripartire la chiamata
- Se vogliamo condividere il risultato, dobbiamo usare 1 subscribe e salvare il risultato, oppure dobbiamo renderlo tiepido.

Multicasting HttpClient

```
http.get(...).pipe(  
  share()  
)
```



```
http.get(...).pipe(  
  multicast(() => new Subject()),  
  refCount()  
)
```



Di due subscribe, se la seconda arriva dopo l'emissione, la chiamata viene ripetuta

```
http.get(...).pipe(  
  shareReplay(1)  
)
```



```
http.get(...).pipe(  
  multicast(() => new ReplaySubject(1)),  
  refCount()  
)
```



**Anche se la seconda subscribe arriva a chiamata terminata, viene ri-emesso, solo per lei, l'ultimo valore.
Ok!**

Attenzione: utilizzatelo solo localmente e non a livello di servizi!