## Similitudes

Ambas nos sirven para trabajar procesos asíncronos: no necesariamente se trata exclusivamente de peticiones HTTP, **se puede utilizar para cualquier tarea que queramos ejecutar y que la respuesta no se obtiene de forma instantánea.**

También se pueden utilizar para estar pendientes de cambios en objetos y ser notificados cuando algo ocurra. Por ejemplo, contadores, timer, etc.

## Diferencias

### Promesas

* Trabaja con un único flujo de datos
* Se usan con una única data asíncrona de respuesta
* No son simples de cancelar

### Observables

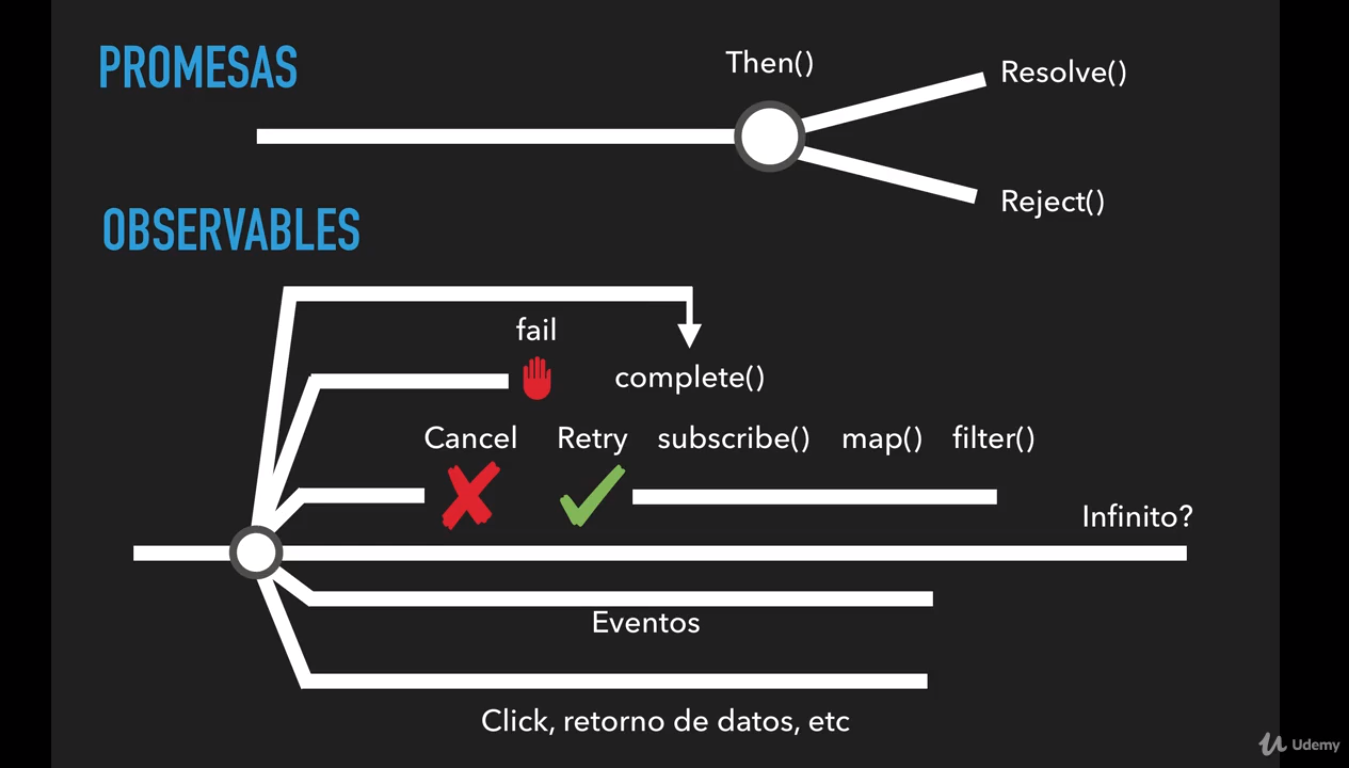
* Trabajan con un flujo continuo de datos
* Al fallar pueden ejecutar comandos para reintentar
* Se pueden encadenar operadores adicionales como: *map, foreach, reduce, filter,* etc.
* Se pueden crear desde diversas fuentes, como, por ejemplo, eventos
* Nos podemos suscribir desde múltiples lugares en simultaneo.

## Funcionamiento

### Promesas

Las promesas son más simples: una vez que el proceso asíncrono termino (el *then ()*) tenemos dos posibles caminos, que se haya ejecutado correctamente, entonces se va ejecutar el código que dejemos en Resolve () o que haya ocurrido un error, entonces se va ejecutar lo que hayamos especificado en Reject ()

En cambio, los observables nos permiten un esquema más complejo: trabajamos con un flujo de datos que puede ser infinito (por ejemplo, un listado de twits). Podemos suscribirnos, cancelar la suscripción, ejecutar algo cuando se haya terminado el flujo (complete ()), etc.



En este ejemplo, tenemos una promesa (batman) que nos va avisar cuando aparezca un Capitan America



Con las promesas tenemos el problema que cuando nuestro código ejecuto una tarea asincrónica y por alguna razón no queremos más la devolución de dicha tarea (por ejemplo, el usuario cerro el navegador) es bastante complejo cancelar esa tarea.

La devolución de la promesa ocurre para ya no hay nadie que la reciba)



Observables

Tenemos un flujo de datos (el observable) y tenemos un Observador que va estar observando el flujo de datos. Cuando el observador cuando algo relevante que le indiquemos nos va a notificar

