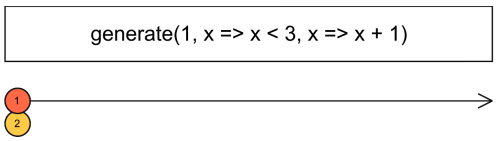
generate

Genera un Observable ejecutando un bucle impulsado por el estado que emite un elemento en cada iteración

▶ Signatura

Descripción

Se utiliza en lugar de hacer llamadas a `next` dentro de un bucle for.



generate permite crear un flujo de valores generador con un bucle muy similar a un bucle for tradicional.

> El primer argumento de generate es el valor inicial.
>
> El segundo valor es una función que acepta este valor y comprueba si una condición se sigue o no cumpliendo. En caso afirmativo, el bucle continúa. Si no, el bucle se para.
>
> El tercer valor es una función que recibe el valor definido anteriormente y lo modifica en cada iteración.

Estos tres parámetros son equivalentes a las tres expresiones de un bucle for tradicional: la primera expresión inicializa un estado (como por ejemplo un índice numérico), la segunda comprueba si el bucle puede o no hacer la siguiente iteración (como por ejemplo si el índice es menor que 10) y la tercera indica cómo el valor definido se modifica en cada iteración (como por ejemplo, incrementar dicho valor en 1.)

El vaor retornado del operador generate es un Observable que emite un valor en cada iteración del bucle. Primero, se ejecuta la función de condición. Si la función retorna *true*, el Observable emite el valor almacenado (el valor inicial en la primera iteración) y después actualiza dicho valor con la función de iteración. Si en algún momento la función de condición retorna *false*, el Observable se completa.

Opcionalmente, se le puede proporcionar un cuarto parámetro a generate - una función de selección de resultado.

Si se encuentra que las tres funciones anónimas en la llamada a generate son difíciles de leer, se le puede proporcionar un solo objeto en su lugar. Dicho objeto tiene las siguientes propiedades:

initialState

condition

iterate

resultSelector

La propiedad condition es opcional en este objeto. Si se omite dicha propiedad, la condición siempre se cumplirá, por lo que el Observable de salida nunca llegará a completarse. El valor de la propiedad resultSelector sigue siendo opcional.

Ambas formas de generate reciben un planificador de forma opcional. En el caso de la llamada multiparámetro, el planificador se proporciona como último argumento (independientemente de que haya una función resultSelector o no.) En el caso de la llamada monoparámetro, se puede proporcionar como propiedad *scheduler* en el objeto proporcionado al operador. En ambos casos el planificador decide el momento en el que ocurre la siguiente iteración del bucle, y por tanto, cuándo se emite la siguiente notificación next. Por ejemplo, para asegurar que cada valor se emite en una tarea distinta del bucle de eventos, se puede utilizar el Planificador async. Por defecto los valores se emiten de forma síncrona.

Ejemplos

Emitir los números del 1 al 10

StackBlitz

```
import { generate } from "rxjs";

const number$ = generate(
  1,
```

```
  (x) => x < 10,
  (x) => x + 1
);

number$.subscribe(console.log);
// Salida: 1, 2, 3, 4, 5, 6, 7, 8, 9, 10
```

Emitir los números pares del 2 al 10, utilizando un objeto como parámetro

StackBlitz