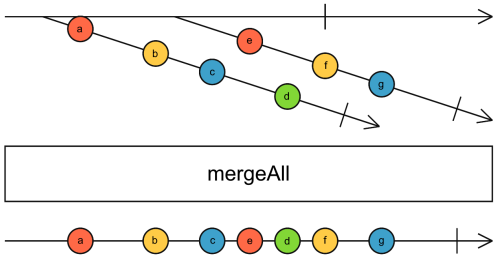
mergeAll

Convierte un Observable de orden superior en uno de primer orden que emite las emisiones de los Observables internos de forma concurrente

▶ Signatura

Descripción

Convierte un Observable de orden superior en uno de primer orden.



`mergeAll` se suscribe a un Observable que emite Observables, también conocido como Observable de orden superior. Cada vez que observa la emisión de uno de los Observables internos, se suscribe a él y emite todos los valores del Observable interno en el Observable resultante. El Observable resultante se completa cuando todos los Observables internos se hayan completado.

Cualquier error que se produzca en uno de los Observables internos se emite de forma inmediata en el Observable resultante.

Ejemplos

Realizar todas las peticiones AJAX de forma concurrente (en paralelo)

StackBlitz

```
import { mergeAll, map, delay } from "rxjs/operators";
import { ajax } from "rxjs/ajax";
import { of } from "rxjs";

const pokemonId$ = of(1, 5, 6);

function getPokemonName(id: number) {
  return ajax.getJSON(`https://pokeapi.co/api/v2/pokemon/${id}`).pipe(
    map(({ name }) => name),
    delay(2000)
  );
}

pokemonId$
  .pipe(
    map((id) => getPokemonName(id)),
    mergeAll()
  )
  .subscribe(console.log);
// Salida: (2s) bulbasaur, charmeleon, charizard
```

Realizar como mucho dos peticiones AJAX de forma concurrente

StackBlitz

```
import { mergeAll, map, delay } from "rxjs/operators";
import { ajax } from "rxjs/ajax";
import { of } from "rxjs";

const pokemonId$ = of(1, 5, 6);

function getPokemonName(id: number) {
  return ajax.getJSON(`https://pokeapi.co/api/v2/pokemon/${id}`).pipe(
    map(({ name }) => name),
    delay(2000)
  );
}
```

```
const maxConcurrent = 2;

pokemonId$
  .pipe(
    map((id) => getPokemonName(id)),
    mergeAll(maxConcurrent)
  )
  .subscribe(console.log);
// Salida: (2s) bulbasaur, charmeleon (2s) charizard
```

Ejemplos de la documentación oficial

Generar un Observable intervalo por cada evento click, y unir sus emisiones en un solo Observable

```
import { fromEvent, interval } from "rxjs";
import { map, mergeAll } from "rxjs/operators";

const clicks = fromEvent(document, "click");
const higherOrder = clicks.pipe(map((ev) => interval(1000)));
const firstOrder = higherOrder.pipe(mergeAll());
firstOrder.subscribe((x) => console.log(x));
```

Copy

Emitir los números del 0 al 9 a intervalos de un segundo por cada click, permitiendo únicamente 2 temporizadores concurrentes