fromFetch

Utiliza la API Fetch para hacer una petición HTTP

▶ Signatura

Descripción

ADVERTENCIA: Partes de la API de `fetch` siguen siendo experimentales. `AbortController` es imprescindible para que esta implementación funcione y para llevar a cabo la cancelación apropiadamente.

`fromFetch` automáticamente genera un `AbortController` interno para eliminar el `fetch` interno cuando se cancele la suscripción.

Si se proporciona una señal vía el argumento `init`, esta se comportará igual que con `fetch`. Si la señal proporcionada aborta, el error que `fetch` normalmente lanza se emitirá como un error del Observable.

Ejemplos

Realizar una petición `fetch` y emitir el objeto `Response` al completo

StackBlitz

```
import { fromFetch } from "rxjs/fetch";

const ghibliFilmResponse$ = fromFetch("https://ghibliapi.herokuapp.com/films

ghibliFilmResponse$.subscribe(console.log);
// Salida: Response {...}
```

Ejemplos de la documentación oficial

```
import { of } from "rxjs";
import { fromFetch } from "rxjs/fetch";
import { switchMap, catchError } from "rxjs/operators";

const data$ = fromFetch("https://api.github.com/users?per_page=5").pipe(
  switchMap((response) => {
    if (response.ok) {
      // OK devolver los datos
      return response.json();
    } else {
      // El servidor retorna un status requiriendo que el cliente intente ot
      return of({ error: true, message: `Error ${response.status}` });
    }
  }),
  catchError((err) => {
    // Gestionando cualquier tipo de error
    console.error(err);
    return of({ error: true, message: err.message });
  })
);

data$.subscribe({
  next: (result) => console.log(result),
  complete: () => console.log("done"),
});
```

Uso con codificación de transferencia fragmentada

Con las respuestas HTTP que utilicen codificación de transferencia fragmentada, la promesa retornada por fetch se resolverá en cuanto se reciban las cabeceras de la respuesta.

Esto implica que el Observable fromFetch emitirá una respuesta - y se completará - antes de recibir el cuerpo de la petición. Cuando uno de los métodos de la respuesta - como text() o json() - sea llamado, la promesa retornada no se resolverá hasta que el cuerpo completo se

haya recibido. Cancelar la suscripción a cualquier Observable que use la promesa como entrada no abortará la petición.

Para facilitar el aborto de la recuperación de respuestas que utilicen codificación de transferencia fragmentada, se puede especificar un selector vía el parámetro init: