¿Cuándo usar RX?

1. Eventos en la interfaz de usuario.
2. Cuando es necesario notificar sobre cambios en un objeto(s).
3. Comunicaciones por sockets
4. Cuando necesitamos trabajar con flujos de información (streams).

Piezas fundamentales de la programación Reactiva

Observables

* Son la fuente de la información
* Pueden emitir múltiples valores, solo uno o ninguno.
* Pueden emitir errores.
* Pueden ser infinitos, finitos (completarse).
* Pueden ser síncronos o asíncronos.

Subscribers

* Se subscriben a un observable, es decir, están pendiente de lo que realiza el observable.
* Consumen / observan la data del observable.
* Desconocen todo lo que se encuentra detrás del observable.

Operators

* Usados para transformar oservables (map, group, scan…).
* Usados para filtrar observables (filter, distinct, skip, bebounce).
* Usados para combinar observables.
* Usados para crear nuevos observables.

Beneficios de la programación reactiva

1. Evitar el “callback hell”.
2. Trabajar de forma simple tareas sincrónicas y asíncronas.
3. Uso de operadores para reducir y simplificar el trabajo.
4. Es fácil transformar los flujos (streams) de información
5. Código más limpio y fácil de leer.
6. Facil de implementar.

¿Qué es ReactiveX?

Es una Api para programación asíncrona usando observables. Implementa el patrón observer, la programación funcional y el patron iterator.

Patron observer: Es un patrón de diseño de software que define una dependencia del tipo uno a muchos entre objetos, de manera que cuando uno de los objetos cambia su estado, notifica este cambio a todos los dependientes.

Patron iterador: Define una interfaz que declara los métodos necesarios para acceder secuencialmente a un grupo de objetos de una colección.

Patron funcional: Crear un conjunto de funciones que tengan un objetivo específico.

Lectura de diagrama de canicas