**Observable:**

* A sequence of data which is emitted asynchronously (most of the times)over period of time.
* Angular makes use of observables as an interface to handle a variety of common asynchronous operations.
* They are used frequently in Angular and are the recommended technique for event Handling,asynchronous programming,and handling multiple values.
* An observable can deliver multiple values of any type-literals,messages,or events,depending on the context.

**What is an observer?**

* When we create an observable,we keep track on the observable using Observer.
* Observer is continuously listening to observable
* We can control when we start and end listening to the observable using observer
* Observer has 3 methods that we can use

1. Next()
2. Error()
3. Complete()

**What is subscription (or) subscribe?**

* We have to always “subscriber” to an observable in order to process the data
* We can have multiple subscribers for any given observable
* We can also unsubscribe from an subscriber

**Observable:**

* observable is part of RXjs library.
* Import observable into our component where we want to make use of it.
* Observable is a sequence of data that is emitted over period of time
* This data can be of any type-string,events,etc
* Angular uses observables very frequently in most async operations

1. Http
2. Routing
3. Event Handling

* In order to listen and crack the changes of observable-we need observer
* Observer will continuously track the changes in observable
* Observer has many methods

1. next()
2. error()
3. complete()

* observable as it-is useless unless we subscribe to it
* By subscribe we mean that we are processing the data /values sent by observables over a period of time
* We can have multiple subscribers to our observable
* We can also unsubscribe from an subscriber

-----------------------------------------------------------------------------------------------------------------

Observables are used to perform asynchronous operations and asynchronous data.

Java script is a single threaded programming language

Synchronize code will block the data so they are moving into asynchronous data like observables

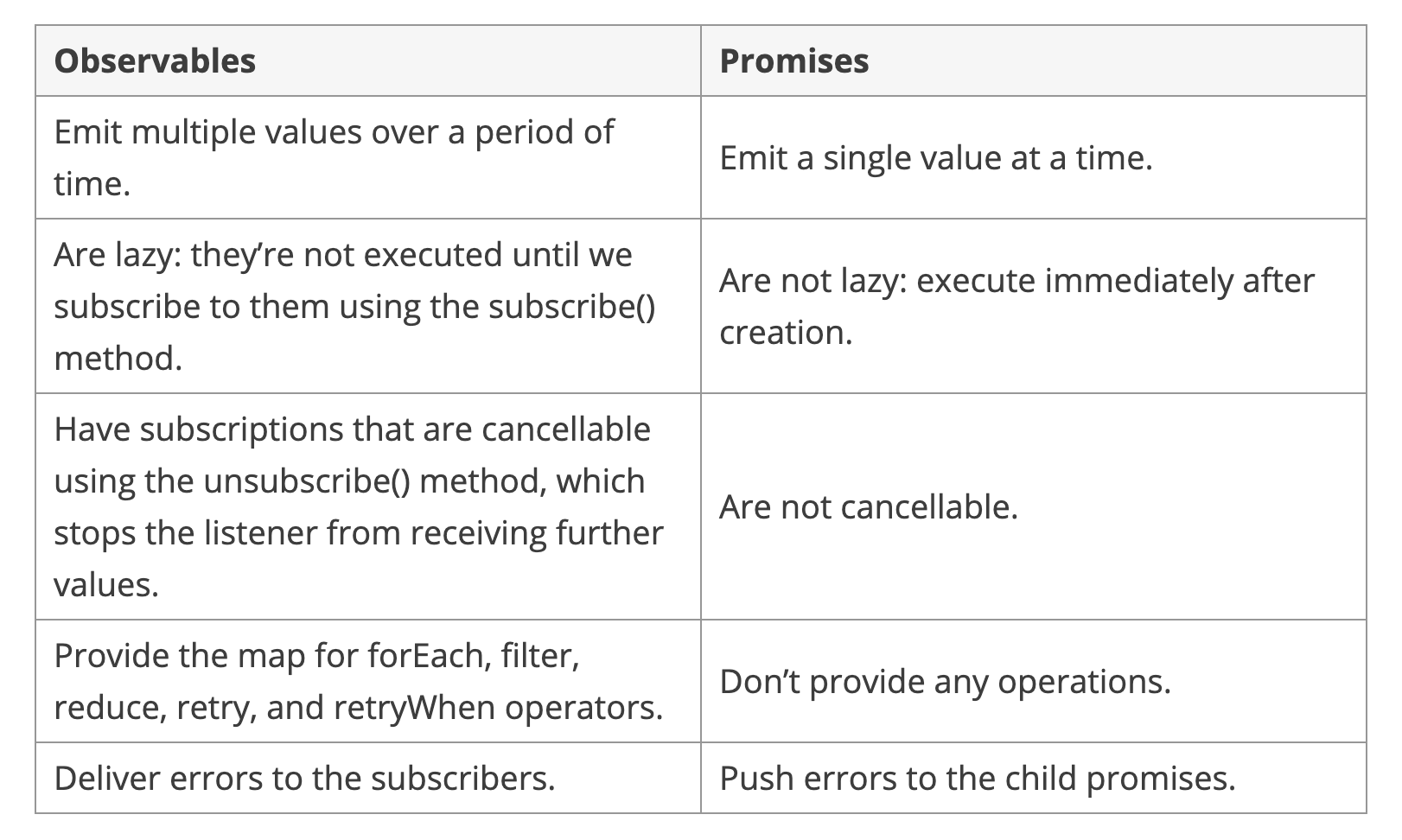
Observable will emit the data when there is a subscriber if there is no subscriber then it will not emit the data

Next,error,complete is a callback functions

**Promises:**

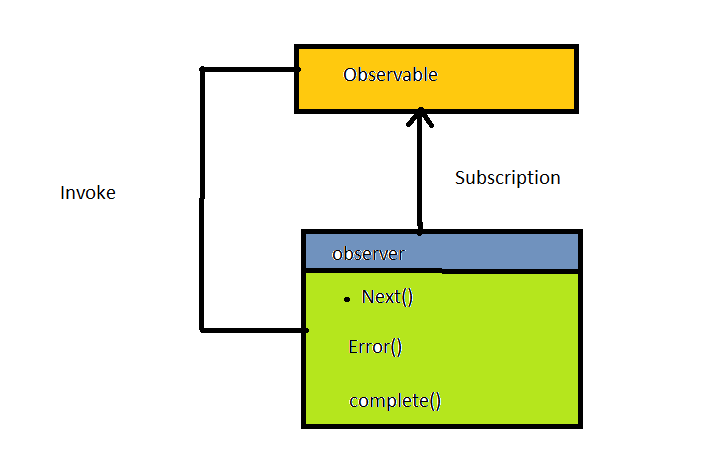
In javascript, a Promise is an object which ensures to produce a single value in the future (when required). Promise in javascript is used for managing and tackling asynchronous operations.

1. **Resolve:** When the promise is executed successfully, the resolve argument is invoked, which provides the result.
2. **Reject:** When the promise is rejected, the reject argument is invoked, which results in an error.



|  |  |  |
| --- | --- | --- |
| **Operations** | **Observables** | **Promises** |
| Creation | const obs = new Observable((observer) => {  observer.next(10);  }) ; | const promise = new Promise(() => {  resolve(10);  }); |
| Transform | Obs.pipe(map(value) => value \* 2); | promise.then((value) => value \* 2); |
| Subscribe | const sub = obs.subscribe((value) => {  console.log(value)  }); | promise.then((value) => {  console.log(value)  }); |
| Unsubscribe | sub.unsubscribe(); | Can’t unsubscribe |

Promise is a part of java script but observables is not a part of javascript but in rxjs package so need to import the observables from rxjs



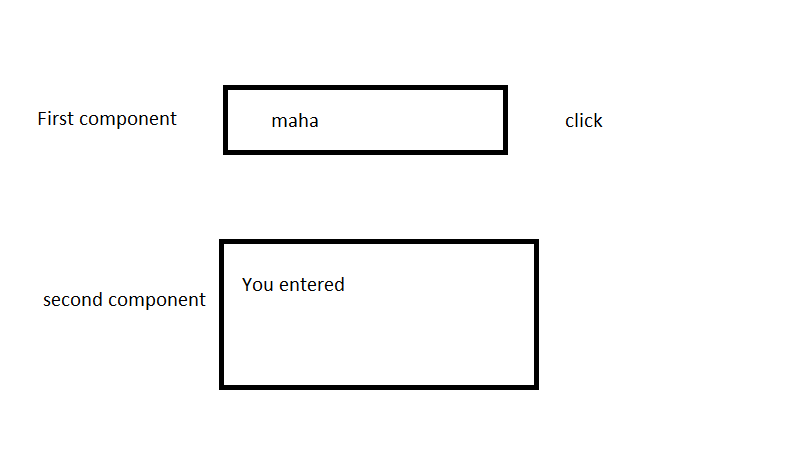
|  |  |
| --- | --- |
| promise | Observable |
| Emits a single value | Emits multiple values over a period of time |
|  |  |
|  |  |
|  |  |
|  |  |

**Subjects:**

Subject is used for component communication.

A subject is a special type of observable that allows values to be multicasted to many observers.subjects are liked EventEmitters

In an web page we have two componenets first component have a text box and a click button.second component have a h3 element .both the componenet are not related to each other and does not have a parent and child relationship.if entering some value in the first component means I want to get the same value in second component try to achieve this things by service and event emitters



A subject is like an observable ,but can multicast to many observers.

Subjects:

Subjects are multicast

Observables:

Observables are unicast.

An observable by default is unicast

Unicasting means that each subscribed observer owns an independent execution of the observable

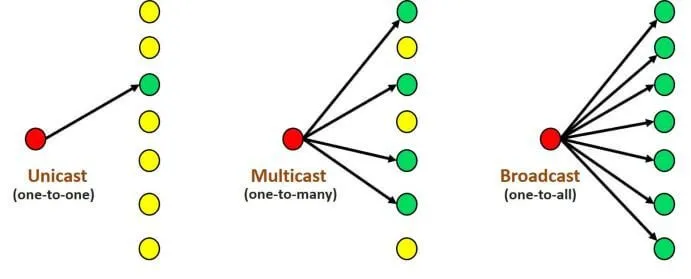
To demonstrate this:

While observables are unicast by design,this can be pretty annoying if you expect that each subscriber receives the same values.

Subjects are multicast:

Multicasting basically means that one observable execution is shared among multiple subscribers.

Subjects are like EventEmitters,they maintain a registry of many listeners.



Using subjects we can able to create a custom observable

**Subject:**

Subject is a special type of observable in RXJS Library in which we can send our data to other components and other services.A subject is like an observable but can multicast to many observers which means subject is at the same time an observable and an observer.

**Usage of angular subject?**

Rxjs is responsible for the Reactivity in angular.A subject is a particular type of observable in RXJS library.

Basically the usage of Angular subject is to share the data between components or throughout the application.

**Types of subjects:**

1.Behaviour subject

2.Replay subject

3.Async subject

**Behaviour subject:**

Behaviour subject is a type of subject, a subject is a special type of observable so you can subscribe to messages like any other observable.The unique feature of behavior subject is:it needs an initial value as it must always return a value on subscription even if it hasn’t received a next().

It has a notion of the current value that it stores and emits to all new subscriptions.this current value is either the item most recently emitted by the source observable or a seed/default value if none has yet been emitted.

**Replay subject:**

A variant of the subject that “replays” or emits old value to new subscribers.It buffers a set numbers of values and will emit those values immediately to any new subscribers in addition to emitting new values to existing subscribers