



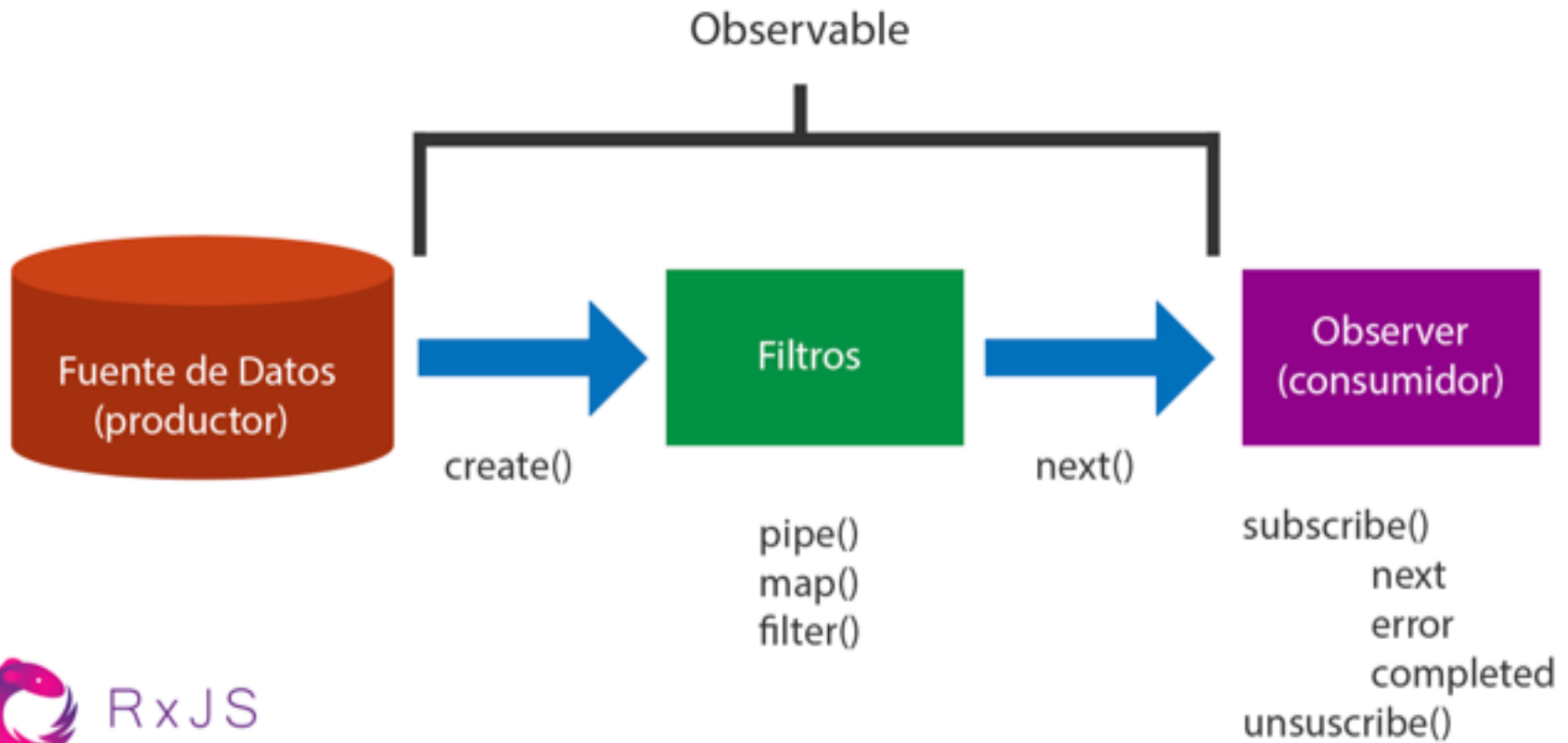
```
function(b, c) {  
  var d = a(c.form.querySelectorAll('input[type=checkbox][name="' + b.el.name + '"']));  
  if (0 === d.index(b.el)) {  
    var e = d.filter(":checked").length;  
    return e >= b.arg || g.minChecked.replace("{count}", b.arg)  
  }  
},  
maxSelected: function(a) {  
  return null !== a.val ? a.val  
},  
minSelected: function(a) {  
  return null !== a.val && a.val  
},  
radio: function(b) {  
  var c = a(this.form.querySelector  
  return 1 === c  
},  
custom: function(a, b) {  
  var c = b.options.custom[a.arg],  
  d = new RegExp(c.pattern);  
  return d.test(a.val) || c.errorMessage  
},  
remote: function(a) {  
  a.remote = a.arg  
}
```

Formación Desarrollo Aplicaciones Web con Angular 5

17.- Trabajando con API REST y Observables

Angular | Trabajando con API Rest y Observables

- Observables



Angular | Trabajando con API Rest y Observables

- Rxjs

Producción

```
import { Observable, Observer } from 'rxjs';
```

```
...
```

```
Observable.create((observer: Observer<any>) => {  
  observer.next(datos);  
});
```

```
...
```

Angular | Trabajando con API Rest y Observables

- Rxjs

Consumo

...

```
.subscribe(  
  (data: any) => {  
    //código tratamiento datos  
  },  
  (error) => {  
    //código manejo errores  
  },  
  () => {  
    //código manejo proceso completado  
  }  
)
```

...

Angular | Trabajando con API Rest y Observables

- HttpClient

En el módulo

```
...  
import { HttpClientModule } from '@angular/common/http';  
...  
imports: [  
..., ReactiveFormsModule, ...  
],
```

Angular | Trabajando con API Rest y Observables

- HttpClient

En el servicio

```
...
import { HttpClient } from '@angular/common/http';
import { map } from 'rxjs/operators';
...
constructor(private http: HttpClient) { }
...
metodo(parametros) {
    return this.http.get | post | put | delete(url, opciones)
        .pipe(
            map((resp: any) => { return resp })
        );
}
```

Angular | Trabajando con API Rest y Observables

- HttpClient

En la clase del componente

```
...
import { ClaseService } from '../..';
...
constructor(private claseService: ClaseService) { }
...
this.claseService.metodo(parametros)
    .subscribe(
        (resp:any) => { // código tratamiento datos},
        (error) => { // código manejo errores }
    )
...
```