

Angular Advanced short recap – day #1



Peter Kassenaar –
info@kassenaar.com

Day 1:

- Some Angular CLI tips & tricks
 - `--flags`, options
- NG applications with multiple modules
 - Using multiple modules in your app
 - Routing and Lazy loading modules
 - Custom Loading strategies
 - Child Routes, `RouterModule.forChild(...)`
 - Configure Router with `ExtraOptions`

```
let routerConfig: ExtraOptions = {  
  enableTracing: true  
};  
  
@NgModule({  
  imports: [RouterModule.forRoot(routes, routerConfig)],  
  exports: [RouterModule]  
})
```

Routing events are actually Observables.

Which means we can subscribe! And do something like:

```
this.router.events
    .subscribe(event =>{
        console.log(' router event: ', event);
    })
```

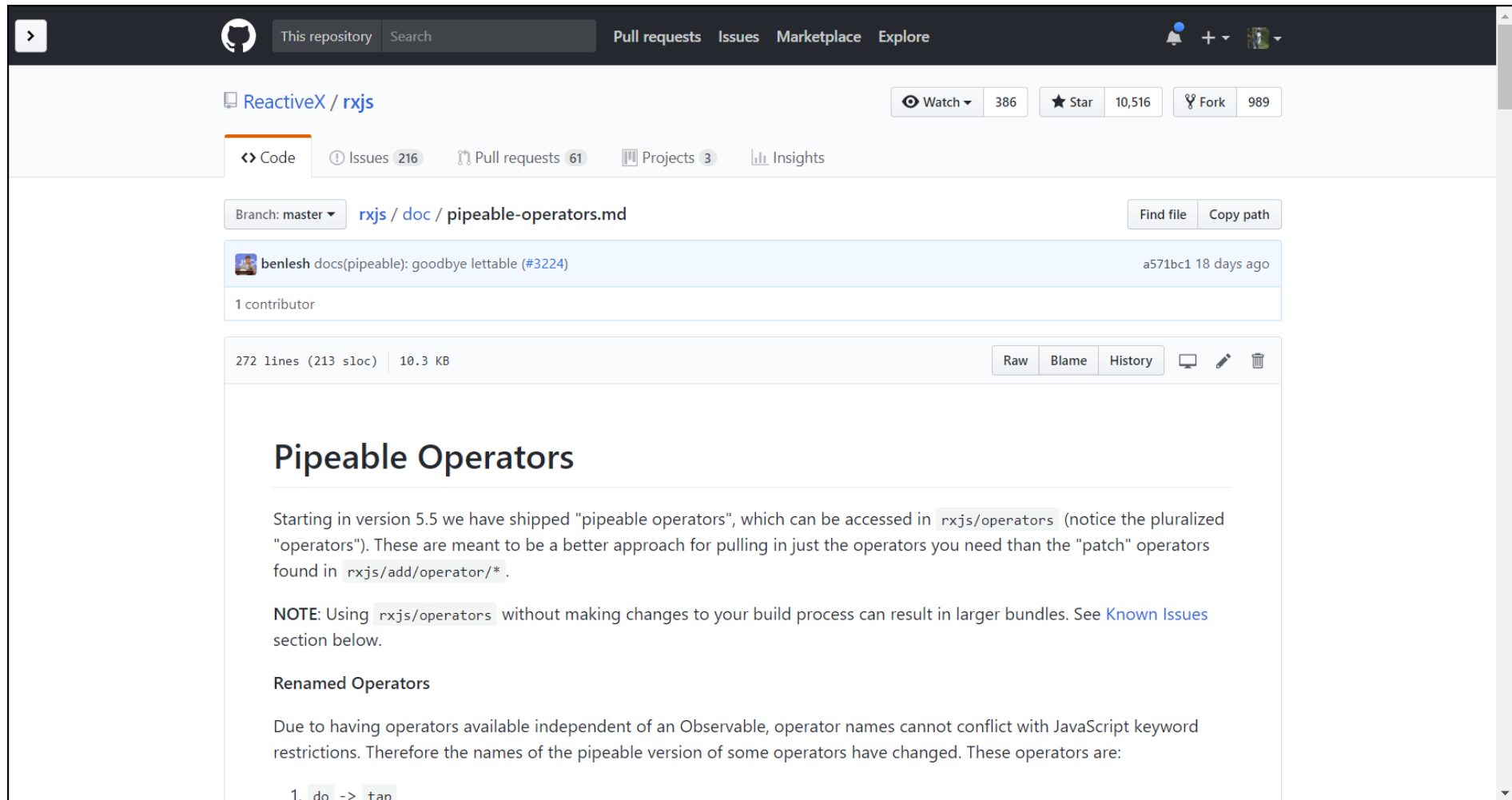
(Don't forget to inject router:

```
constructor(private router: Router) {  })
```

Or, in a more reactive way of programming:

```
this.router.events
  .filter(event => event instanceof NavigationEnd)
  .map(...)
  . ...
  .subscribe(event =>{
    console.log(' router event: ', event);
  })
```

RxJS 5.5+: Pipeable operators en `.pipe()`



The screenshot displays the GitHub interface for the `ReactiveX/rxjs` repository. The file `rxjs/doc/pipeable-operators.md` is selected, showing its commit history and content. The commit by `benlesh` is highlighted, with the message `docs(pipeable): goodbye lettable (#3224)`. The file content is displayed below, starting with the title `Pipeable Operators`.

Pipeable Operators

Starting in version 5.5 we have shipped "pipeable operators", which can be accessed in `rxjs/operators` (notice the pluralized "operators"). These are meant to be a better approach for pulling in just the operators you need than the "patch" operators found in `rxjs/add/operator/*`.

NOTE: Using `rxjs/operators` without making changes to your build process can result in larger bundles. See [Known Issues](#) section below.

Renamed Operators

Due to having operators available independent of an Observable, operator names cannot conflict with JavaScript keyword restrictions. Therefore the names of the pipeable version of some operators have changed. These operators are:

1. `do` -> `tap`

<https://github.com/ReactiveX/rxjs/blob/master/doc/pipeable-operators.md>

```
// new: rxjs 5.5 lettable operators with .pipe()  
return  
this.http.get<City[]>('assets/data/cities.json')  
  .pipe(  
    tap(res => console.log(res)),  
    catchError(err => {  
      console.log(err);  
      return Observable.of([])  
    })  
  )
```

Today

- Content Projection
 - reuse of content inside components
- State Management w/ `@ngrx/store`
 - Introduction & terminology
 - Simple store, Abstractions
 - Old & New way of using Stores
- Pro's & cons of a single store