Exposing Features Using the Exports Array in Angular

## **Introduction**

In Angular, you can separate features with the help of feature modules. This helps you easily maintain applications. Feature modules also help you to track errors quickly. You can use a feature module in many other modules, but these feature modules will not able to access other modules unless they are exposed. You must understand how to expose a feature so it they can be imported into other modules.

## **Predefined Feature Modules in Angular**

Angular comes with many predefined feature modules. These modules expose the components, pipes, or directives that help you build apps faster without implementing every single thing.

A couple of them are listed below:

1. ReactiveForms Module : This module exports the required directive, like formControlName or formGroup, that helps you create reactive forms.
2. RouteModule : This module exports the required directive and providers, like routerLink or routerLinkActive, that help you create client-side routing.

## **Create a Feature Module**

In this feature module, you'll make a pipe that will use a date format. You'll pass a date in yyyy-mm-dd format, and it will return the date in dd-mm-yyyy format.

Create a project using the following command.

1

$ ng new features-export-demo

console

Create a feature module using the following command.

1

$ ng g m date-format

console

Create a pipe using the following command.

1

$ ng g pipe /date-format/date-format

console

The environment is created, now you'll write the code for Pipe.

**date-format.pipe.ts**

1

2

3

4

5

6

7

8

9

10

import { Pipe, PipeTransform } from "@angular/core";

@Pipe({

name: "dateFormat",

})

export class DateFormatPipe implements PipeTransform {

transform(value: any, ...args: any[]): any {

return value.split("-").reverse().join("-");

}

}

ts

**date-format.module.ts**

1

2

3

4

5

6

7

8

9

10

import { NgModule } from "@angular/core";

import { CommonModule } from "@angular/common";

import { DateFormatPipe } from "./date-format.pipe";

@NgModule({

declarations: [DateFormatPipe],

imports: [CommonModule],

exports: [DateFormatPipe],

})

export class DateFormatModule {}

ts

In DateFormatModule, we're exposing a DateFormatPipe so that other modules can get the advantage of this pipe.

DateFormat pipe is ready to use. To you can use it in AppComponent you need to import the DateFormatModule in app.module.ts.

**app.module.ts**

1

2

3

4

5

6

7

8

9

10

11

12

13

14

import { BrowserModule } from "@angular/platform-browser";

import { NgModule } from "@angular/core";

import { AppRoutingModule } from "./app-routing.module";

import { AppComponent } from "./app.component";

import { DateFormatModule } from "./date-format/date-format.module";

@NgModule({

declarations: [AppComponent],

imports: [BrowserModule, AppRoutingModule, DateFormatModule],

providers: [],

bootstrap: [AppComponent],

})

export class AppModule {}

ts

DateFormatModule has been imported with the help of the imports array of NgModule. You can use the features of the DateFormatModule within the AppComponent.

Create a variable in app.component.ts that contains a date in yyyy-mm-dd format.

**app.component.ts**

1

2

3

4

5

6

7

8

9

10

11

12

13

14

import { Component } from "@angular/core";

@Component({

selector: "app-root",

templateUrl: "./app.component.html",

styleUrls: ["./app.component.scss"],

})

export class AppComponent {

testDate: string;

constructor() {

this.testDate = "2019-05-06";

}

}

ts

You have defined the variable testDate in yyyy-mm-dd format and assigned a value in a constructor. You can use interpolation to bind the value of testDate with the HTML template and use the pipe to transform the date format.

**app.component.html**

1

{{testDate| dateFormat}}

html

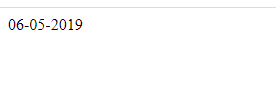
The application is now ready. Run the project using the following command.

1

$ npm start

console

Check the output; it will convert the date to dd-mm-yyyy format as shown below.



You have successfully made a pipe in a feature module and exposed it using the exports array of the NgModule.

## **Conclusion**

In this guide, we discussed how to expose the features of a feature module so you can use them in other modules. We listed some predefined feature modules, but Angular provides even more feature modules that you can read about [here](https://angular.io/). In this guide, we implemented a simple example of a date format, but I highly recommend that you explore further and add more features to your application.