What is the exact meaning of export keyword in Angular 2\TypeScript?

2

1

I am pretty new in **Angular 2.** I am studying how to create modules into an Angular app and I have the following doubt related a tutorial that I am following.

My doubt is related to the routing.

So in my example there is defined this **AuthModule** module:

```
import { NgModule } from '@angular/core';
import { FormsModule } from '@angular/forms';

import { SigninComponent } from './signin/signin.component';
import { SignupComponent } from './signup/signup.component';
import { AuthRoutingModule } from './auth-routing.module';

@NgModule({
  // Components and directives used by the module:
  declarations: [
    SigninComponent,
    SignupComponent
  ],
  // Import modules used by this features module:
  imports: [
    FormsModule,
    AuthRoutingModule
  ]
})
export class AuthModule {}
```

and I have the related rotues configuration class defined:

```
import { NgModule } from '@angular/core';
import { Routes, RouterModule } from '@angular/router';
```

```
@NgModule({
  imports: [RouterModule.forRoot(appRoutes)],
  exports: [RouterModule]
})
export class AppRoutingModule {

}
```

So I think that the **export** keyword means that the content related to this class can be exported and used somewhere else (in this case I think into the **imports** array of the **AuthModule** class).

Is it? Or am I missing something? What it the exact meaning of the **export** statment?

I am not understanding if it is something related to Angular or more generally to TypeScript (because here I found https://www.typescriptlang.org/docs/handbook/modules.html). So it seems to me that this module concept is not directly bounded to Angular 2 framework but is a TypeScript concept to subdivide our code in a smart way (then Angular 2 can use this kind of feature of the language).

Is it or am I missing something?

javascript angular typescript javascript-framework

asked Oct 22 '17 at 15:24

AndreaNobili
14.1k 64 187 357

read the article Avoiding common confusions with modules in Angular – Max Koretskyi aka Wizard Oct 22 '17 at 18:02

2 Answers

Angular imports/exports and TypeScript imports/exports are two different concepts.

So, if you use `FormsModule` there can't be any ambiguity, what `FormsModule` is meant. If there is more than one `FormsModule` in your code or any of your dependencies, then you need to make it clear with imports which one is meant. You can't import 2 `FormsModule` from different locations without disambiguation (for example using `as foo` in the import and then reference it using `foo.FormsModule`).

This way you can use code from arbitrary 3rd-party libraries and avoid name collisions.

Angular imports/exports are used to make the content of one module available to be used in another module.

Your

imports: [FormsModule, AuthRoutingModule]

Allows you to use the directives from `FormsModule` and `AuthRoutingModule` in `AuthModule` and registers the services provided by these modules in the `AppModule` scope or the closed lazy-loaded root scope.

If you reference any of Angulars directives or services in TypeScript code, you also need to add TypeScript imports. Above `FormsModule` and `AuthRoutingModule` need to be imported with TypeScript imports, to make the Angular `imports: [...]` work.

For example like

```
<form #f="ngForm">
  <input type="text">
</form>
```

works only if `FormsModule` is listed in `imports: [ ... ]` of your current module.

There is no TypeScript import required, because there is no TypeScript code.

answered Oct 22 '17 at 15:39

Günter Zöchbauer
352k 81 1103 1012

Yes you are right by using export keyword before your typescript class you can use that class somewhere else .. in your project