Angular Module

In angular, a module is a mechanism to group components, directives, pipes and services that are related, in such a way that can be combined with other modules to create an application. An angular application can thought of as a puzzle where each piece is a module. There are two types of modules, root module and feature module. Root module imports BrowserModule. Feature module imports CommonModule.

Routing

Routing is optional for angular application. But routing can make application linkable and allow user to share link with others. It helps modularize the application.

Feature module

In feature module, it encapsulates its own routing configure, components, pipes and services. Service is the layer to communicate with backend services. Services can be injected into component through the private parameter in constructor (also needs to be declared in providers list). In onInit method of component, the service method gets called to populate the properties of the component which are bound to UI. When creating feature module, it has more challenge to deal with singleton instance of services etc. by default, when a module loaded, services, components, directives and pipes will be initiated in the module level. Therefore, the service might have multiple instances. The services should not be created through providers array in ngModule annotation. A static method must be defined in feature module to do so.

*credit-card/credit-card.module.ts --- feature module*

import { NgModule, ModuleWithProviders } from '@angular/core';

/\* ...other imports... \*/

@NgModule({

imports: [CommonModule],

declarations: [

CreditCardMaskPipe,

CreditCardComponent

],

exports: [CreditCardComponent]

})

export class CreditCardModule {

static forRoot(): ModuleWithProviders {

return {

ngModule: CreditCardModule,

providers: [CreditCardService]

}

}

}

In root module,

app/app.module.ts

/\* ...imports... \*/

@NgModule({

imports: [

BrowserModule,

CreditCardModule.forRoot()

],

declarations: [AppComponent],

bootstrap: [AppComponent]

})

export class AppModule { }

As a rule of thumb, **always use the forRoot syntax when exporting services from feature modules**, unless you have a very special need that requires multiple instances at different levels of the dependency injection tree.

**The road you've travelled**

Here's what you achieved in this page:

* You created a service class that can be shared by many components.
* You used the ngOnInit lifecycle hook to get the hero data when the AppComponentactivates.
* You defined the HeroService as a provider for the AppComponent.
* You created mock hero data and imported them into the service.
* You designed the service to return a Promise and the component to get the data from the Promise.

. integrate bootstrap into angular

<https://medium.com/codingthesmartway-com-blog/using-bootstrap-with-angular-c83c3cee3f4a>

* 1. Npm install bootstrap@next –save or npm install –save @ng-bootstrap/ng-bootstrap
  2. Modify angular-cli.json

Ng-bootstrap + angular

Ngx-bootstrap widgets

<http://valor-software.com/ngx-bootstrap/#/>

install bootstrap 3 for css

npm install bootstrap@3 –save

change angular-cli.json style:[]

. style: <http://getbootstrap.com/>

Define multiple routing modules

<https://johnpapa.net/introducing-angular-modules-feature-modules-2/>

Steps build dtp ui:

1. Create dtp parent folder
2. Ng new calculators
3. Npm install ngx-bootstrap –save
4. Npm install bootstrap@3 –save
5. Setup navbar from <http://getbootstrap.com/>

Tool

Vsc adds auto-import plugin

<https://marketplace.visualstudio.com/items?itemName=steoates.autoimport>

Sep 14,2017

Module

2 types of modules: root module/feature module

Decorator @NgModule, properties to describe the module

* declarations - the view classes that belong to this module. Angular has three kinds of view classes: [components](https://angular.io/guide/architecture#components), [directives](https://angular.io/guide/architecture#directives), and [pipes](https://angular.io/guide/pipes).
* exports - the subset of declarations that should be visible and usable in the component [templates](https://angular.io/guide/architecture#templates) of other modules.
* imports - other modules whose exported classes are needed by component templates declared in this module.
* providers - creators of [services](https://angular.io/guide/architecture#services) that this module contributes to the global collection of services; they become accessible in all parts of the app.
* bootstrap - the main application view, called the root component, that hosts all other app views. Only the root module should set this bootstrap property.