

How to share service between two modules - @NgModule in angular not between to components?



34



13

In my application, I have two different bootstrap module (`@NgModule`) running independently in one application. There are not one angular app bit separate bootstrap module and now I want they should communicate to each other and share data.

I know through `@Injectable` service as the provider in the module, I can share data in all component under `@NgModule` but how I will share data between two different module (not component inside module).

Is there a way one service object can be accessed in another module? Is there a way I can access the object of Service available in browser memory and use it in my other angular module?

 angular `angular2-services` `angular2-modules`

edited Sep 28 '17 at 20:28

asked Oct 17 '16 at 14:40





Aniruddha Das

6,801 9 54 67

What do you mean by module? An `@NgModule()` . Services provided in `@NgModule()` are shared with the whole application and therefore also with all modules in your application (except when the module where you provide the service is lazy loaded). – [Günter Zöchbauer](#) Oct 17 '16 at 14:43

I mean two different angular2 module. Yes I know service can share data all across my module. I want to share data out side my module and in another module created by other team – [Aniruddha Das](#) Oct 17 '16 at 14:46

- 1 If you've provided a service in an `@NgModule`, all non-lazy-loaded modules below may use that service. If you lazy-load, each lazy-loaded module must provide those services. If you need to use it in a different application, either offer an API endpoint or send them the code. – [gelliott181](#) Oct 17 '16 at 14:46 

What do you mean with "Angular2 module" and what do you mean by "running independently". It's still not clear to me. Do you mean two Angular2 applications where two different modules are bootstrapped? – [Günter Zöchbauer](#) Oct 17 '16 at 14:47 

- 1 Possible duplicate of [What is the best way to share services across Modules in angular2](#) – [Martin Schneider](#) Mar 31 '18 at 16:04

5 Answers

You can instantiate a service outside Angular and provide a value:

8

```
class SharedService {
  ...
}
```



```
window.sharedService = new SharedService();

@NgModule({
  providers: [{provide: SharedService, useValue: window.sharedService}],
  ...
})
class AppModule1 {}

@NgModule({
  providers: [{provide: SharedService, useValue: window.sharedService}],
  ...
})
class AppModule2 {}
```

If one application change the state in `SharedService` or calls a method that causes an `Observable` to emit a value and the subscriber is in a different application than the emitter, the code in the subscriber is executed in the `NgZone` of the emitter.

Therefore when subscribing to an observable in `SharedService` use

```
class MyComponent {
  constructor(private zone:NgZone, private sharedService:SharedService) {
    sharedService.someObservable.subscribe(data => this.zone.run(() => {
      // event handler code here
    }));
  }
}
```

See also [How to dynamically create bootstrap modals as Angular2 components?](#)

edited May 23 '17 at 12:34



Community ♦

1 1

answered Oct 17 '16 at 15:01



Günter Zöchbauer

356k 82 1123 1028

@Gunter, I am getting `Duplicate identifier` error doing the same thing as you've done. – [Ahmad Baktash Hayeri](#) Feb 21 '17 at 11:03

- 1 Sorry, but that is not enough information. Can you provide a Plunker? I'd suggest you create a new question that shows your code, a Plunker, and the full error message. – [Günter Zöchbauer](#) Feb 21 '17 at 11:08
- 1 @GünterZöchbauer, I managed to bring a communication logic between two separately bootstrapped app modules using a `Subject` (which I suppose inherits from `Observable` class) instance in the service without ever calling the `zone.run()`. Can you tell me if I still need the `ngZone` here? – [Ahmad Baktash Hayeri](#) Feb 23 '17 at 7:13
- 2 Also, isn't there a syntax error in doing `... .subscribe(data => this.zone.run() => { // event handler code here })`, as I don't understand how it works? – [Ahmad Baktash Hayeri](#) Feb 23 '17 at 7:18
- 3 This could work but it's not an elegant solution. the right way of doing this is implementing a static method `forRoot()` and returning a unique instance of your services. Have a look here: stackoverflow.com/a/46243224/1683040 – [LeonardoX](#) Sep 15 '17 at 15:53



47

As per the final version of Angular 2, services provided by a module are available to every other module that imports it. The [Official Style Guide](#) advice that application-wide services (singletons) that are to be reused anywhere in the application should be provided by some `Core Module`, that is to be imported in the main `App Module` so it would be injectable everywhere.

If you do not use a structure that involves a `Core Module` with shared singletons, and you are independently developing two `NgModules`, and you want a service in one of them to be used in the other, then the only solution is to import the provider into the other :

Here's the provider module:

```
/// some.module.ts
import { NgModule } from '@angular/core';

import { SomeComponent } from './some.component';

@NgModule({
  imports: [],
  exports: [],
  declarations: [SomeComponent],
  providers: [ MyService ], // <===== PROVIDE THE SERVICE
})
export class SomeModule { }
```

Here's the other module, that wants to use `MyService`

```
/// some-other.module.ts
import { NgModule } from '@angular/core';

import { SomeModule } from 'path/to/some.module'; // <=== IMPORT THE JSMODULE
```

```
import { SomeOtherComponent } from './some.other.component';

@NgModule({
  imports: [ SomeModule ], // <===== IMPORT THE NG MODULE
  exports: [],
  declarations: [SomeOtherComponent],
  providers: [],
})
export class SomeOtherModule { }
```

This way, the service should be injectable in any component `SomeOtherModule` declares, and in `SomeModule` itself - just ask for it in the constructor:

```
/// some-other.module.ts

import { MyService } from 'path/to/some.module/my-service';

/* ...
   rest of the module
*/

export class SomeOtherModule {
  constructor( private _myService: MyService ) { <===== INJECT THE SERVICE
    this._myService.dosmth();
  }
}
```

If this doesn't answer your question, I invite you to re-formulate it.

edited Nov 15 '16 at 8:32

answered Nov 13 '16 at 17:27



FacelessPanda

611 5 10

9 This is the right answer – Dre Mar 23 '17 at 16:07

I have tried this line from your code: `import { MyService } from 'path/to/some.module/my-service';` But I am getting an error stating that `'path/to/some.module/my-service'` is not found – Vishal Apr 19 '17 at 17:45

1 @Vishal this generally means that your transpiler can't find your file at that path. Check that your service file's path (relative to your project source) and your import statement match, (you don't need the extension in the import path). But this is a compilation/transpilation issue, not Angular stuff, so I won't further this problem here. – FacelessPanda Apr 20 '17 at 15:19

3 Does this answer really work for separately bootstrapped modules? I tried, but it does not work for me. It works only if the shared service is used in the components within a module. – Dipendu Paul Jul 7 '17 at 12:35

- 1 this doesn't work. It creates two instances of each service, so you're not really sharing a service... Have a look here: stackoverflow.com/a/46243224/1683040 – LeonardoX Sep 15 '17 at 15:56
-

1. create shared module

13

```
@NgModule({})
export class SharedModule {
  static forRoot(): ModuleWithProviders {
    return {
      ngModule: SharedModule,
      providers: [SingletonService]
    };
  }
}
```

2. in the app module are your parent app module import the shared module like that

```
SharedModule.forRoot()
```

3. any the children modules if you need to import the shared module import it without the for root

```
SharedModule
```

<https://angular-2-training-book.rangle.io/handout/modules/shared-modules-di.html>

edited Mar 14 at 12:01



Ferie

458 8 21

answered Jan 15 '18 at 11:30



Hager Aly

820 8 19

One of the best answer. Ionic does the same thing. – Kanchan Jun 27 '18 at 7:53

The answer that Günter Zöchbauer gave is great, however, one issue you may have is that

1

```
window.sharedService
```



will cause an error in TypeScript. You can get around this by replacing all instances of

```
window.sharedService
```

with

```
(<any>window).sharedService
```

answered Oct 20 '16 at 16:45



[Ryan Anderson](#)

41 8

1 Thanks [@Ryan](#) for the addition. Little busy in other part of the application. will come to that part in few days. – [Aniruddha Das](#) Oct 20 '16 at 17:50

You could also extend the window interface: interface Window { sharedService: any } and avoid multiple repetiton of that change with <any>. – [Matthias Max](#) Jul 14 '17 at 9:43

and we are back to placing things in the global scope... – [Ray Suelzer](#) Nov 2 '17 at 5:16



1

I attempted to use the ideas in this answer to share data between multiple bootstrapped modules in my angular application. The data I was interested in was coming from an `http.get()` call. It was an expensive call and I wanted to only make the call once and share its results between my 2 bootstrapped modules.



Initially, I stored the service as a property on my global window object as suggested, but in the end, I decided to use static variables and call `publishReplay()` on my observable to create a `ReplaySubject` to replay my emissions to future subscribers. This allowed multiple subscribers to share data and only make the REST call once.

One caveat here... The original question asked how to share one service... This answer does not share one service... Multiple services are created, but the data is shared as the observable is stored in a static variable... That means the observable sticks around as long as your application and because of the call to `publishReplay()` anyone who subscribes to the observable replays the data previously returned.

Here is my code:

```
import { Injectable } from '@angular/core';
import { Observable, ReplaySubject } from 'rxjs/Rx';
import { Http } from '@angular/http';
```

```
@Injectable()
export class InitService {

    static observable: Observable<number> = null;

    constructor(private http: Http) {}

    getInitData(): Observable<number> {
        if (InitService.observable == null) {
            InitService.observable = this.http.get('api/Init')
                .map(this.extractData)
                .publishReplay()
                .refCount()
                .catch(this.handleError);
        }
        return InitService.observable;
    }

    extractData(res: Response) {
        let body: any = res.json();
        return body || {};
    }

}
```

Hope this helps.

answered Jun 11 '17 at 21:00



[birwin](#)

1,403 2 9 27

