Services are a great way to share information among classes that ***do not know each other*.**When we are developing an Angular app, we will most likely run into a scenario in which we need to use the same code across multiple components. In that case, **Services**will help us to get rid of this problem.

[2 Step 1: Install Angular via CLI.](https://appdividend.com/2018/01/20/angular-services-tutorial-example-scratch/#Step_1_Install_Angular_via_CLI)

* [3 Step 2. Create a new project.](https://appdividend.com/2018/01/20/angular-services-tutorial-example-scratch/#Step_2_Create_a_new_project)
* [4 Step 3: Serve the application.](https://appdividend.com/2018/01/20/angular-services-tutorial-example-scratch/#Step_3_Serve_the_application)
* [5 Step 4: Create Service class.](https://appdividend.com/2018/01/20/angular-services-tutorial-example-scratch/#Step_4_Create_Service_class)
* [6 Step 5: Use services into the component.](https://appdividend.com/2018/01/20/angular-services-tutorial-example-scratch/#Step_5_Use_services_into_the_component)

**Step 1: Install Angular via CLI.**

You need to set up a dev environment before you can do anything.

Install [**Node.js® and npm**](https://nodejs.org/en/download/) if they are not already on your machine.

 Then **install the**[**Angular CLI**](https://github.com/angular/angular-cli) globally.

npm install -g @angular/cli

**Step 2. Create a new project.**

Type the following command.

ng new ngservices

**Step 3: Serve the application.**

Go to the project directory and launch the server.

cd my-app

ng serve --open

**Step 4: Create Service class.**

To create a service, at the console, type the following command in your root of the folder.

ng g service crypto

It will create the following files.

1. **crypto.service.ts**
2. **crypto.service.spec.ts**

Now, import the service file into the **app.module.ts**file.

import { CryptoService } from './crypto.service';

@NgModule({

providers: [CryptoService],

})

Now, we need to add some code to the **crypto.service.ts** file. This file contains the data that we need to consume. So this is service file. In the Live project, this file will hit the GET or POST request to the server and insert or fetch the data. So this file provides services from frontend to backend.

*// items.service.ts*

import *{ Injectable }* from '@angular/core';

@Injectable()

export class CryptoService *{*

*coins= [*

*{id: 1, name: 'BTC'}*,

*{id: 2, name: 'XRP'}*

];

constructor() *{ }*

getMyItems()

*{*

*return this.coins;*

*}*

}

**Step 5: Use services into the component.**

So your **app.component.ts**file looks like this.

*// app.component.ts*

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

import { CryptoService } from './crypto.service';

@Component({

selector: 'app-root',

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

styleUrls: ['./app.component.css']

})

export class AppComponent {

coins = [];

constructor(private cryptoservice: CryptoService)

{

this.coins = cryptoservice.getMyItems();

}

}

Also, we need to update the HTML as well.

<table>

<tr>

<th>ID</th>

<th>Name</th>

</tr>

<tr \*ngFor="let coin of coins">

<td>{{coin.id}}</td>

<td>{{coin.name}}</td>

</tr>

</table>

Finally, you can see in the browser: [http://localhost:4200](http://localhost:4200/)

You can see the data is displaying the table format.

In a real-world scenario, data is fetching from the Backend API. In this case, we have taken simple static Array.