免費電子書

學習

Angular

Free unaffiliated eBook created from
Stack Overflow contributors.

#angular

You can share this PDF with anyone you feel could benefit from it, downloaded the latest version from: angular

It is an unofficial and free Angular ebook created for educational purposes. All the content is extracted from Stack Overflow Documentation, which is written by many hardworking individuals at Stack Overflow. It is neither affiliated with Stack Overflow nor official Angular.

Use the content presented in this book at your own risk; it is not guaranteed to be correct nor accurate, please send your feedback and corrections to info@zzzprojects.com

1: Angular

Angular “ **Angular 2+** ”“ **Angular 2** ”TypeScriptWebGoogleAngular。 Web。 AngularAngularJS。[1]

@ angular / core @ angular / animations 。

AngularizedHTML Angular。

。 Angular。

Angular 。 HTML。

SRC // app.component.ts

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

@Component({
    selector: 'my-app',
    template: `<h1>Hello {{name}}</h1>`
})
export class AppComponent {
    name = 'Angular';
}
```

`@Component`。 HTML。

`selector`Angular*index.html*`<my-app>`。

index.html`body`

```
<my-app>Loading AppComponent content here ...</my-app>
```

template`<h1>`。 “Hello”`{{name}}` Angular 。 Angular`{{name}}`name。 Angular。 `name'Angular''World'`。

TypeScriptJavaScript。 AngularTypeScript。 **TypeScript JavaScript** 。 JavaScriptAngular; 。

Angular

5.0.0-beta.1	2017727
4.3.2	2017726
5.0.0-beta.0	2017719
4.3.1	2017719
4.3.0	2017714

4.2.6	201778
4.2.5	201769
4.2.4	2017621
4.2.3	2017616
4.2.2	2017612
4.2.1	201769
4.2.0	201768
4.2.0-rc.2	201761
4.2.0-RC.1	2017526
4.2.0-rc.0	2017519
4.1.3	2017517
4.1.2	2017510
4.1.1	201754
4.1.0	2017426
4.1.0-rc.0	2017421
4.0.3	2017421
4.0.2	2017411
4.0.1	2017329
4.0.0	2017323
4.0.0-rc.6	2017323
4.0.0-rc.5	2017317
4.0.0-rc.4	2017317
2.4.10	2017317
4.0.0-rc.3	2017310
2.4.9	201732
4.0.0-rc.2	201732

4.0.0-RC.1	2017224
2.4.8	2017218
2.4.7	201729
2.4.6	201723
2.4.5	2017125
2.4.4	2017119
2.4.3	2017111
2.4.2	201716
2.4.1	20161221
2.4.0	20161220
2.3.1	○
2.3.0	2016127
2.3.0-rc.0	
2.2.4	
2.2.3	20161123
2.2.2	20161122
2.2.1	20161117
2.2.0	20161114
2.2.0-rc.0	2016112
2.1.2	20161027
2.1.1	○○
2.1.0	
2.1.0-rc.0	2016105
2.0.2	2016105
2.0.1	2016923
2.0.0	2016914

2.0.0 rc.7	2016913
2.0.0 rc.6	2016831
2.0.0 rc.5	201689
2.0.0 rc.4	2016630
2.0.0 rc.3	2016621
2.0.0 rc.2	2016615
2.0.0 RC.1	201653
2.0.0 rc.0	201652

Examples

angular-cli

Angular。

- Node.js 6.9.0 。
- npm v3 。
- v1。

```
npm install -g typingsyarn global add typings

npm install -g @angular/cliyarn global add @angular/cli
```

`typings`PATH。 **@ angular / cli** `ng`PATH。

。

```
ng new PROJECT_NAME
cd PROJECT_NAME
ng serve
```

Angular。 。

。

```
ng init
```

。 。

。

```
ng serve
```

。

```
http://localhost:4200
```

htmltypescriptcss。

`ng generate <scaffold-type> <name>` `ng g <scaffold-type> <name>` Angular

```
# The command below will generate a component in the folder you are currently at
ng generate component my-generated-component
# Using the alias (same outcome as above)
ng g component my-generated-component
# You can add --flat if you don't want to create new folder for a component
ng g component my-generated-component --flat
# You can add --spec false if you don't want a test file to be generated (my-generated-
component.spec.ts)
ng g component my-generated-component --spec false
```

angular-cli

	ng g module my-new-module
	ng g component my-new-component
	ng g directive my-new-directive
	ng g pipe my-new-pipe
	ng g service my-new-service
	ng g class my-new-class
	ng g interface my-new-interface
	ng g enum my-new-enum

。

`ng gm my-new-moduleng gc my-new-component`。

I

Angular WebApache TomcatWebbuild。 。

```
ng build
```

```
ng build --prod
```

/dist。

Ahead-of-Time

```
ng build --prod --aot
```

Angularangular-cli。 jasmineKarma。

```
ng test
```

。

angular-cli github

“Hello World”

。

- Node.jsnpm 。

  /node -vnpm -v6.9.xnpm 3.xx。 。

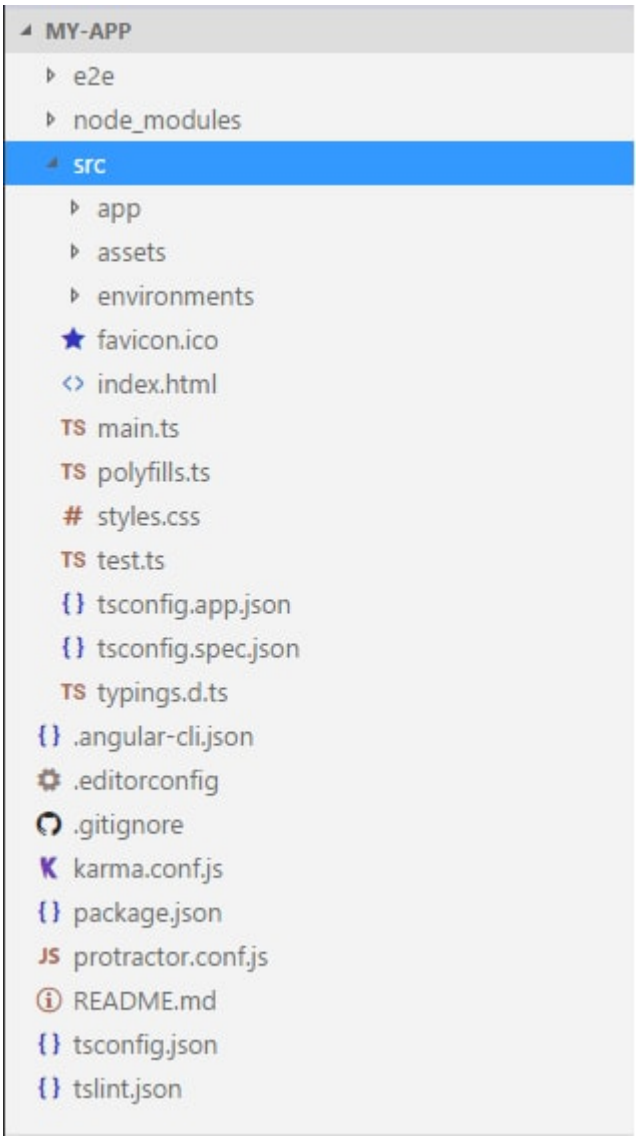
- `npm install -g @angular/cli`Angular CLI 。

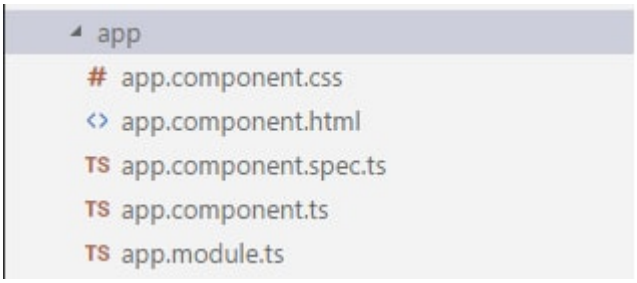
1

WindowsNode.js。

```
ng new my-app
```

`ng`Angular。 。





。 。

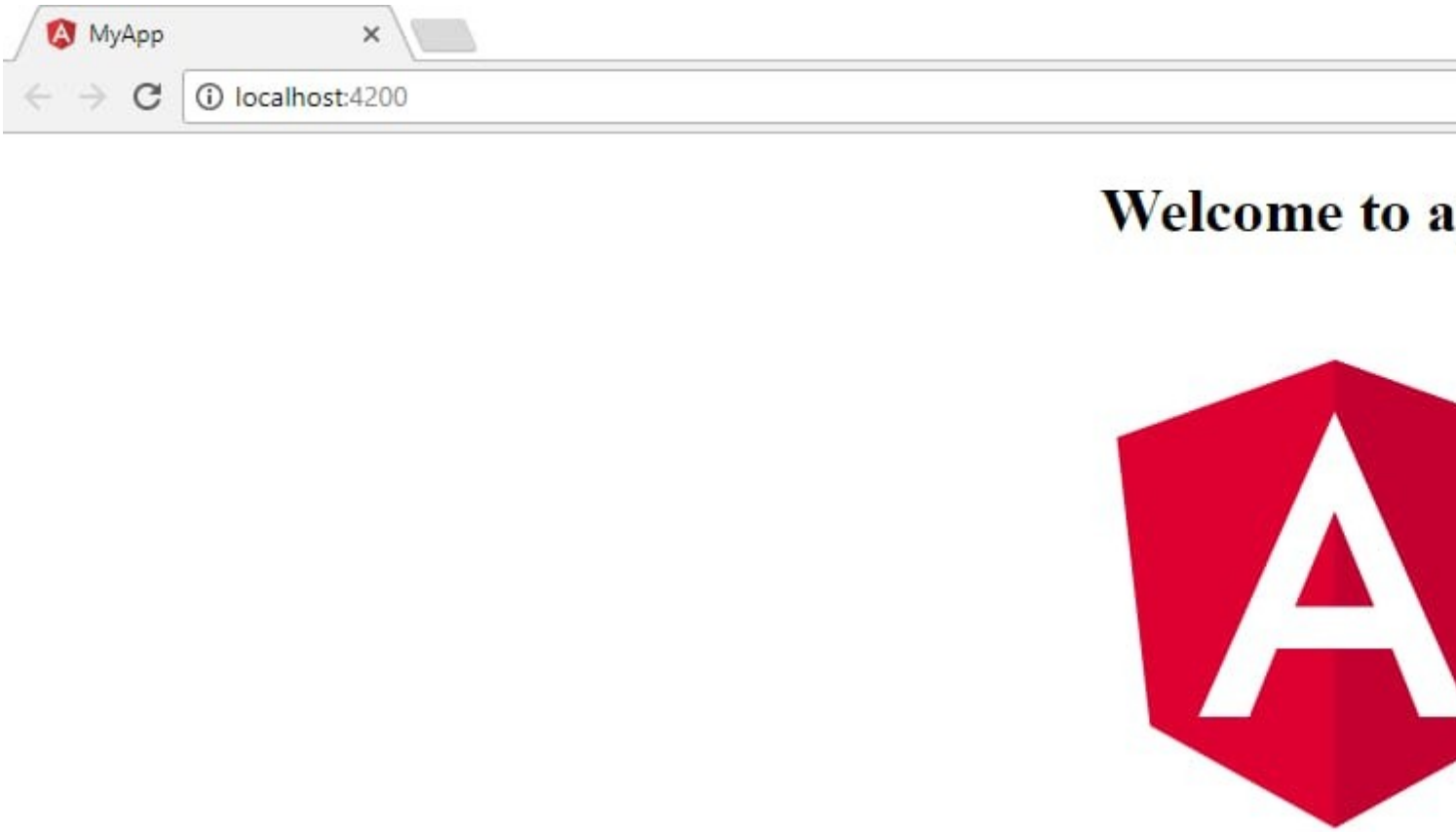
2

```
ng serve
```

-open -o http://localhost:4200/

```
ng serve --open
```

http://localhost:4200/ 。



3Angular

CLIAngular。 `app-root` 。 `./src/app/app.component.ts`。

```
Welcome to app!!Welcome to app!!Hello World 。 。
```

```
title = 'app';
```

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

  Angular CLI, 20 minutes ago | 1 author (Angular CLI)
  @Component({
    selector: 'app-root',
    templateUrl: './app.component.html',
    styleUrls: ['./app.component.css']
  })
  export class AppComponent {
    title = 'app';
  }
```

`title`。

```
import { Component } from '@angular/core';
@Component({
  selector: 'app-root',
  templateUrl: './app.component.html',
  styleUrls: ['./app.component.css']
})
export class AppComponent {
  title = 'Hello World';
}
```

◦ / ./src/app/app.component.html◦

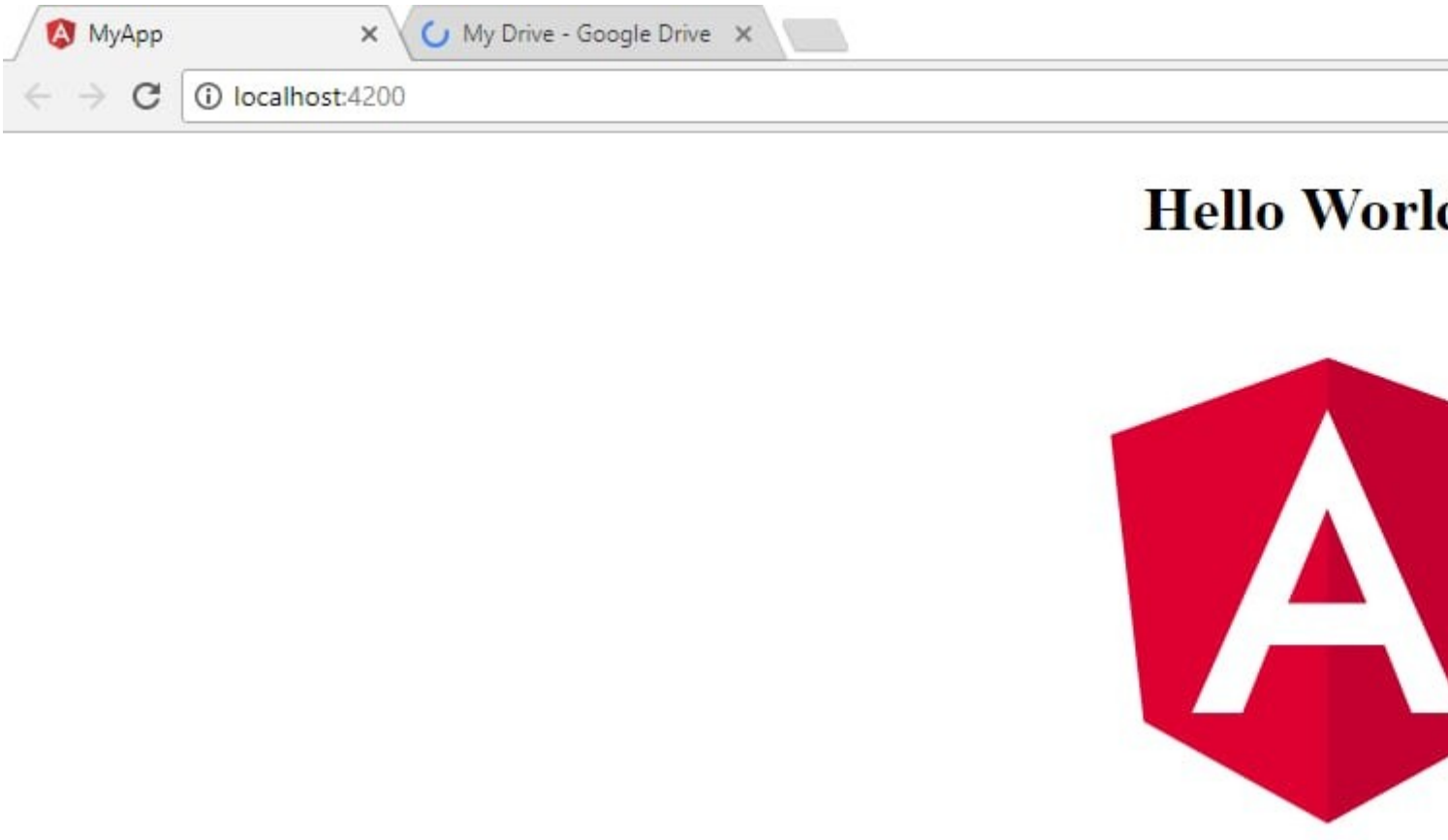
HTML

```
<div style="text-align:center">
  <h1>
    Welcome to {{title}}!!
  </h1>
  <img width="300" src="
```

HTML

```
<!--The content below is only a placeholder and can be replaced.-->
<div style="text-align:center">
  <h1>
    {{title}}!!
  </h1>
  <img width="300" src="
```

./src/app/app.component.tstitle◦ ◦ ◦



。

Angular https://riptutorial.com/zh-TW/angular/topic/9754/angular

2: RXJSObservables

Examples

◦ forkJoin◦

forkJoinObservables◦ Observable.subscribe◦ .subscribe.forkJoin ◦ posts tags ◦

```
loadData() : void {
    Observable.forkJoin(
        this.blogApi.getPosts(),
        this.blogApi.getTags()
    ).subscribe((([posts, tags]: [Post[], Tag[]]) => {
        this.posts = posts;
        this.tags = tags;
    }));
}
```

HTTP GET◦ http.get()subscribeObservable ◦ posts◦

```
var posts = []

getPosts(http: Http): {
    this.http.get(`https://jsonplaceholder.typicode.com/posts`)
        .subscribe(response => {
            posts.push(response.json());
        });
}
```

RXJSObservables https://riptutorial.com/zh-TW/angular/topic/9829/rxjsobservables

3:

Examples

-

```
import {EventEmitter} from 'angular2/core';
export class NavService {
    navchange: EventEmitter<number> = new EventEmitter();
    constructor() {}
    emitNavChangeEvent(number) {
        this.navchange.emit(number);
    }
    getNavChangeEmitter() {
        return this.navchange;
    }
}
```

-

```
import {Component} from 'angular2/core';
import {NavService} from '../services/NavService';

@Component({
    selector: 'obs-comp',
    template: `obs component, item: {{item}}`
    })
    export class ObservingComponent {
    item: number = 0;
    subscription: any;
    constructor(private navService:NavService) {}
    ngOnInit() {
        this.subscription = this.navService.getNavChangeEmitter()
        .subscribe(item => this.selectedNavItem(item));
    }
    selectedNavItem(item: number) {
        this.item = item;
    }
    ngOnDestroy() {
        this.subscription.unsubscribe();
    }
}

@Component({
    selector: 'my-nav',
    template:`
        <div class="nav-item" (click)="selectedNavItem(1)">nav 1 (click me)</div>
        <div class="nav-item" (click)="selectedNavItem(2)">nav 2 (click me)</div>
    `,
})
export class Navigation {
    item = 1;
    constructor(private navService:NavService) {}
    selectedNavItem(item: number) {
        console.log('selected nav item ' + item);
        this.navService.emitNavChangeEvent(item);
```

```
    }
}
```

https://riptutorial.com/zh-TW/angular/topic/9828/

4:

。

。 。

Examples

service.ts

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

@Injectable()
export class AppState {

  public mylist = [];

}
```

parent.component.ts

```
import {Component} from '@angular/core';
import { AppState } from './shared.service';

@Component({
  selector: 'parent-example',
  templateUrl: 'parent.component.html',
})
export class ParentComponent {
  mylistFromParent = [];

  constructor(private appState: AppState){
    this.appState.mylist;
  }

  add() {
    this.appState.mylist.push({"itemName":"Something"});
  }

}
```

parent.component.html

```
<p> Parent </p>
  <button (click)="add()">Add</button>
<div>
  <child-component></child-component>
</div>
```

child.component.ts

```
import {Component, Input } from '@angular/core';
```

```
import { AppState } from './shared.service';

@Component({
  selector: 'child-component',
  template: `
    <h3>Child powered by shared service</h3>
        {{mylist | json}}
  `,
})
export class ChildComponent {
  mylist: any;

  constructor(private appState: AppState){
    this.mylist = this.appState.mylist;
  }

}
```

@Input

parent.component.ts

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

@Component({
  selector: 'parent-example',
  templateUrl: 'parent.component.html',
})

export class ParentComponent {
  mylistFromParent = [];

  add() {
    this.mylistFromParent.push({"itemName":"Something"});
  }

}
```

parent.component.html

```
<p> Parent </p>
  <button (click)="add()">Add</button>

<div>
  <child-component [mylistFromParent]="mylistFromParent"></child-component>
</div>
```

child.component.ts

```
import {Component, Input } from '@angular/core';

@Component({
  selector: 'child-component',
  template: `
    <h3>Child powered by parent</h3>
        {{mylistFromParent | json}}
```

```
  `,
})

export class ChildComponent {
  @Input() mylistFromParent = [];
}
```

@Output

emitter.component.ts

```
import { Component, OnInit, EventEmitter, Output } from '@angular/core';

@Component({
  selector: 'event-emitting-child-component',
  template: `<div *ngFor="let item of data">
               <div (click)="select(item)">
                 {{item.id}} = {{ item.name}}
               </div>
             </div>
            `
})

export class EventEmitterChildComponent implements OnInit{

  data;

  @Output()
  selected: EventEmitter<string> = new EventEmitter<string>();

  ngOnInit(){
    this.data = [ { "id": 1, "name": "Guy Fawkes", "rate": 25 },
                  { "id": 2, "name": "Jeremy Corbyn", "rate": 20 },
                  { "id": 3, "name": "Jamie James", "rate": 12 },
                  { "id": 4, "name": "Phillip Wilson", "rate": 13 },
                  { "id": 5, "name": "Andrew Wilson", "rate": 30 },
                  { "id": 6, "name": "Adrian Bowles", "rate": 21 },
                  { "id": 7, "name": "Martha Paul", "rate": 19 },
                  { "id": 8, "name": "Lydia James", "rate": 14 },
                  { "id": 9, "name": "Amy Pond", "rate": 22 },
                  { "id": 10, "name": "Anthony Wade", "rate": 22 } ]
  }

  select(item) {
      this.selected.emit(item);

  }

}
```

receiver.component.ts

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

@Component({
    selector: 'event-receiver-parent-component',
    template: `<event-emitting-child-component (selected)="itemSelected($event)">
                </event-emitting-child-component>
```

```
                <p *ngIf="val">Value selected</p>
                <p style="background: skyblue">{{ val | json}}</p>`
})

export class EventReceiverParentComponent{
  val;

  itemSelected(e){
    this.val = e;
  }
}
```

ObservableSubject

shared.service.ts

```
import { Injectable }    from '@angular/core';
import { Headers, Http } from '@angular/http';

import 'rxjs/add/operator/toPromise';

import { Observable } from 'rxjs/Observable';
import { Observable } from 'rxjs/Rx';
import {Subject} from 'rxjs/Subject';


@Injectable()
export class AppState {

  private headers = new Headers({'Content-Type': 'application/json'});
  private apiUrl = 'api/data';

  // Observable string source
  private dataStringSource = new Subject<string>();

  // Observable string stream
  dataString$ = this.dataStringSource.asObservable();

  constructor(private http: Http) { }

  public setData(value) {
    this.dataStringSource.next(value);
  }

  fetchFilterFields() {
    console.log(this.apiUrl);
    return this.http.get(this.apiUrl)
               .delay(2000)
               .toPromise()
               .then(response => response.json().data)
               .catch(this.handleError);
  }

  private handleError(error: any): Promise<any> {
    console.error('An error occurred', error); // for demo purposes only
    return Promise.reject(error.message || error);
  }

}
```

parent.component.ts

```
import {Component, OnInit} from '@angular/core';
import 'rxjs/add/operator/toPromise';
import { AppState } from './shared.service';

@Component({
  selector: 'parent-component',
  template: `
              <h2> Parent </h2>
              <h4>{{promiseMarker}}</h4>

              <div>
                <child-component></child-component>
              </div>
            `
})
export class ParentComponent implements OnInit {

  promiseMarker = "";

  constructor(private appState: AppState){ }

  ngOnInit(){
    this.getData();
  }

  getData(): void {
    this.appState
        .fetchFilterFields()
        .then(data => {
          // console.log(data)
          this.appState.setData(data);
          this.promiseMarker = "Promise has sent Data!";
        });
  }

}
```

child.component.ts

```
import {Component, Input } from '@angular/core';
import { AppState } from './shared.service';


@Component({
  selector: 'child-component',
  template: `
    <h3>Child powered by shared service</h3>
        {{fields | json}}
  `,
})
export class ChildComponent {
  fields: any;

  constructor(private appState: AppState){
    // this.mylist = this.appState.get('mylist');

    this.appState.dataString$.subscribe(
```

```
      data => {
        // console.log("Subs to child" + data);
        this.fields = data;
      });

  }

}
```

https://riptutorial.com/zh-TW/angular/topic/10836/

5:

Examples

NgFor -

NgFor。iterable。

NgFor**trackBy**。 **trackBy**indexitem。**trackBy** Angular。

```
<li *ngFor="let item of items; let i = index; trackBy: trackByFn">
    {{i}} - {{item.name}}
</li>
```

NgFor

- **index**。
- **first**。
- **last**。
- **even**。
- **odd**。

https://riptutorial.com/zh-TW/angular/topic/9826/

6:

Examples

app.module.ts

app.module.ts

```
import { NgModule } from '@angular/core';
import { BrowserModule } from '@angular/platform-browser';
import { FormsModule, ReactiveFormsModule } from '@angular/forms';
import { AppComponent } from './app.component';

@NgModule({
    imports: [
        BrowserModule,
        FormsModule,
        ReactiveFormsModule,
    ],
    declarations: [ AppComponent ]
    providers: [],
    bootstrap: [ AppComponent ]
})
export class AppModule {}
```

app.component.ts

```
import { Component,OnInit } from '@angular/core';
import template from './app.component.html';
import { FormGroup,FormBuilder,Validators } from '@angular/forms';
import { matchingPasswords } from './validators';

@Component({
    selector: 'app',
    template
})
export class AppComponent implements OnInit {
    addForm: FormGroup;

    constructor(private formBuilder: FormBuilder) {
    }

    ngOnInit() {
        this.addForm = this.formBuilder.group({
            username: ['', Validators.required],
            email: ['', Validators.required],
            role: ['', Validators.required],
            password: ['', Validators.required],
            password2: ['', Validators.required]
        }, { validator: matchingPasswords('password', 'password2') });
    };

    addUser() {
```

```
        if (this.addForm.valid) {
            var adduser = {
                username: this.addForm.controls['username'].value,
                email: this.addForm.controls['email'].value,
                password: this.addForm.controls['password'].value,
                profile: {
                    role: this.addForm.controls['role'].value,
                    name: this.addForm.controls['username'].value,
                    email: this.addForm.controls['email'].value
                }
            };

            console.log(adduser);// adduser var contains all our form values. store it where
you want
            this.addForm.reset();// this will reset our form values to null
        }
    }

}
```

app.component.html

```
<div>
    <form [formGroup]="addForm">
        <input
            type="text"
            placeholder="Enter username"
            formControlName="username" />

        <input
            type="text"
            placeholder="Enter Email Address"
            formControlName="email"/>

        <input
            type="password"
            placeholder="Enter Password"
            formControlName="password" />

        <input
            type="password"
            placeholder="Confirm Password"
            name="password2"
            formControlName="password2" />

        <div class='error' *ngIf="addForm.controls.password2.touched">
            <div
                class="alert-danger errormessageadduser"
                *ngIf="addForm.hasError('mismatchedPasswords')">
                    Passwords do not match
            </div>
        </div>
        <select name="Role" formControlName="role">
            <option value="admin" >Admin</option>
            <option value="Accounts">Accounts</option>
            <option value="guest">Guest</option>
        </select>
        <br/>
        <br/>
```

```
        <button type="submit" (click)="addUser()">
            <span>
                <i class="fa fa-user-plus" aria-hidden="true"></i>
            </span>
            Add User
        </button>
    </form>
</div>
```

validators.ts

```
export function matchingPasswords(passwordKey: string, confirmPasswordKey: string) {
    return (group: ControlGroup): {
        [key: string]: any
    } => {
        let password = group.controls[passwordKey];
        let confirmPassword = group.controls[confirmPasswordKey];

        if (password.value !== confirmPassword.value) {
            return {
                mismatchedPasswords: true
            };
        }
    }
}
```

- **signup.component.html**

```
<form #signUpForm="ngForm" (ngSubmit)="onSubmit()">

  <div class="title">
    Sign Up
  </div>

  <div class="input-field">
    <label for="username">username</label>
    <input
      type="text"
      pattern="\w{4,20}"
      name="username"
      required="required"
      [(ngModel)]="signUpRequest.username" />
  </div>

  <div class="input-field">
    <label for="email">email</label>
    <input
      type="email"
      pattern="^\S+@\S+$"
      name="email"
      required="required"
      [(ngModel)]="signUpRequest.email" />
  </div>

  <div class="input-field">
    <label for="password">password</label>
```

```
    <input
      type="password"
      pattern=".{6,30}"
      required="required"
      name="password"
      [(ngModel)]="signUpRequest.password" />
  </div>

  <div class="status">
    {{ status }}
  </div>

  <button [disabled]="!signUpForm.form.valid" type="submit">
    <span>Sign Up</span>
  </button>

</form>
```

- **signup.component.ts**

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

import { SignUpRequest } from './signup-request.model';

@Component({
  selector: 'app-signup',
  templateUrl: './signup.component.html',
  styleUrls: ['./signup.component.css']
})
export class SignupComponent {

  status: string;
  signUpRequest: SignUpRequest;

  constructor() {
    this.signUpRequest = new SignUpRequest();
  }

  onSubmit(value, valid) {
    this.status = `User ${this.signUpRequest.username} has successfully signed up`;
  }

}
```

- **signup-request.model.ts**

```
export class SignUpRequest {

  constructor(
    public username: string="",
    public email: string="",
    public password: string=""
  ) {}

}
```

- **app.module.ts**

```
import { BrowserModule } from '@angular/platform-browser';
import { NgModule } from '@angular/core';
import { FormsModule } from '@angular/forms';

import { AppComponent } from './app.component';
import { SignupComponent } from './signup/signup.component';

@NgModule({
  declarations: [
    AppComponent,
    SignupComponent
  ],
  imports: [
    BrowserModule,
    FormsModule
  ],
  bootstrap: [AppComponent]
})
export class AppModule { }
```

- **app.component.html**

```
<app-signup></app-signup>
```

https://riptutorial.com/zh-TW/angular/topic/9825/

7:

AngularJS。 | Angular。

Examples

my.pipe.ts

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

@Pipe({name: 'myPipe'})
export class MyPipe implements PipeTransform {

  transform(value:any, args?: any):string {
    let transformedValue = value; // implement your transformation logic here
    return transformedValue;
  }

}
```

my.component.ts

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

@Component({
  selector: 'my-component',
  template: `{{ value | myPipe }}`
})
export class MyComponent {

    public value:any;

}
```

my.module.ts

```
import { NgModule } from '@angular/core';
import { BrowserModule } from '@angular/platform-browser';

import { MyComponent } from './my.component';
import { MyPipe } from './my.pipe';

@NgModule({
  imports: [
    BrowserModule,
  ],
  declarations: [
    MyComponent,
    MyPipe
  ],
})
export class MyModule { }
```

◦ ◦

Module◦

breaklines.ts

```
import { Pipe } from '@angular/core';
/**
 * pipe to convert the \r\n into <br />
 */
@Pipe({ name: 'br' })
export class BreakLine {
    transform(value: string): string {
        return value == undefined ? value :
             value.replace(new RegExp('\r\n', 'g'), '<br />')
              .replace(new RegExp('\n', 'g'), '<br />');
    }
}
```

uppercase.ts

```
import { Pipe } from '@angular/core';
/**
 * pipe to uppercase a string
 */
@Pipe({ name: 'upper' })
export class Uppercase{
    transform(value: string): string {
        return value == undefined ? value : value.toUpperCase( );
    }
}
```

pipes.module.ts

```
import { NgModule } from '@angular/core';
import { BreakLine } from './breakLine';
import { Uppercase} from './uppercase';

@NgModule({
    declarations: [
        BreakLine,
        Uppercase
    ],
    imports: [

    ],
    exports: [
        BreakLine,
        Uppercase
    ]
    ,
})
export class PipesModule {}
```

my.component.ts

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

@Component({
  selector: 'my-component',
  template: `{{ value | upper | br}}`
})
export class MyComponent {

    public value: string;

}
```

my.module.ts

```
import { NgModule } from '@angular/core';
import { BrowserModule } from '@angular/platform-browser';

import { MyComponent } from './my.component';
import { PipesModule} from './pipes.module';

@NgModule({
  imports: [
    BrowserModule,
    PipesModule,
  ],
  declarations: [
    MyComponent,
  ],
})
```

https://riptutorial.com/zh-TW/angular/topic/9824/

8:

Examples

app.routing.tsapp.module.ts。WebPackSystemJS。

///。。Route。app.module.ts

Angularchildren。URLURL。

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

import { HomeComponent } from "./components/home/home.component";
import { FetchDataComponent } from "./components/fetchdata/fetchdata.component";
import { CounterComponent } from "./components/counter/counter.component";

const appRoutes: Routes = [
    {
        path: "",
        redirectTo: "home",
        pathMatch: "full"
    },
    {
        path: "home",
        children: [
            {
                path: "",
                component: HomeComponent
            },
            {
                path: "counter",
                children: [
                    {
                        path: "",
                        component: CounterComponent
                    },
                    {
                        path: "fetch-data",
                        component: FetchDataComponent
                    }
                ]
            }
        ]
    },
    {
        path: "**",
        redirectTo: "home"
    }
];

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

```
        RouterModule
    ]
})
export class AppRoutingModule { }
```

Siraj

。

Angular -

```
<base href='/'>
```

index.html。app。Angular。

1. package.json/Angular`npm install` -

   ```
   "dependencies": {
       "@angular/router": "^4.2.5"
   }
   ```

2. ```
   interface Route {
     path?: string;
     pathMatch?: string;
     component?: Type<any>;
   }
   ```

3. `routes/app.routing.ts` 。 。“”。

   ```
   import { Routes, RouterModule } from '@angular/router';
   import { ModuleWithProviders } from '@angular/core';
   import { BarDetailComponent } from '../components/bar-detail.component';
   import { DashboardComponent } from '../components/dashboard.component';
   import { LoginComponent } from '../components/login.component';
   import { SignupComponent } from '../components/signup.component';

   export const APP_ROUTES: Routes = [
       { path: '', pathMatch: 'full', redirectTo: 'login' },
       { path: 'dashboard', component: DashboardComponent },
       { path: 'bars/:id', component: BarDetailComponent },
       { path: 'login', component: LoginComponent },
       { path: 'signup',   component: SignupComponent }
   ];
   export const APP_ROUTING: ModuleWithProviders = RouterModule.forRoot(APP_ROUTES);
   ```

4. `app.module.ts @NgModule([])imports`

   ```
   // Alternatively, just import 'APP_ROUTES
   import {APP_ROUTING} from '../routes/app.routing.ts';
   @NgModule([
       imports: [
           APP_ROUTING
           // Or RouterModule.forRoot(APP_ROUTES)
   ```
   ```

```
    ]
])
```

5. /。 <router-outlet>。

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

@Component({
    selector: 'demo-app',
    template: `
        <div>
            <router-outlet></router-outlet>
        </div>
    `
})
export class AppComponent {}
```

6. 。 RouterOutletRoutes。 RouterLinkhtml anchor。 RouterLinkhref。

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

@Component({
    selector: 'demo-app',
    template: `
        <a [routerLink]="['/login']">Login</a>
        <a [routerLink]="['/signup']">Signup</a>
        <a [routerLink]="['/dashboard']">Dashboard</a>
        <div>
            <router-outlet></router-outlet>
        </div>
        `
})
export class AnotherComponent { }
```

◦ RouterLinkRouterLink。

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

@Component({
  selector: 'demo-app',
  template: `
        <ul>
          <li *ngFor="let bar of bars | async">
            <a [routerLink]="['/bars', bar.id]">
              {{bar.name}}
            </a>
          </li>
        </ul>
    <div>
      <router-outlet></router-outlet>
    </div>
  `
})
export class SecondComponent { }
```

RouterLink。

https://riptutorial.com/zh-TW/angular/topic/9827/

S. No		Contributors
1	Angular	aholtry, Anup Kumar Gupta, BogdanC, Community, daddycool, Edric, Fahad Nisar, Faisal, Hendrik Brummermann, Philipp Kief, Tom
2	RXJSObservables	aholtry
3		aholtry
4		0mpurdy, BogdanC, JGFMK, Nehal
5		aholtry
6		aholtry, Saka7
7		aholtry, Amr ElAdawy
8		0mpurdy, aholtry, Edric