35分钟掌握Angular核心概念

大漠穷秋 2017-08-18

内容提纲

- **第一块内容：集成开发环境@angular/cli**
- **第二块内容：Angular三大核心概念**

  Component,Module,Route
- **第三块内容：Angular架构特色**

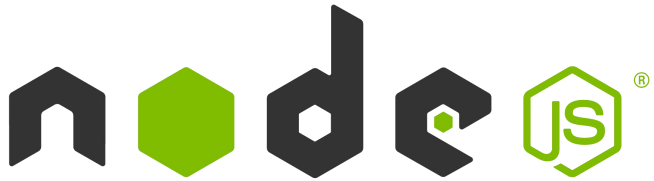
  依赖注入、数据绑定
- **第四块内容：UI库**

  PrimeNG,NG-Zorro,Clarity,Angular-Material,Jigsaw,ionic
- **第五块内容：参考资源推荐**

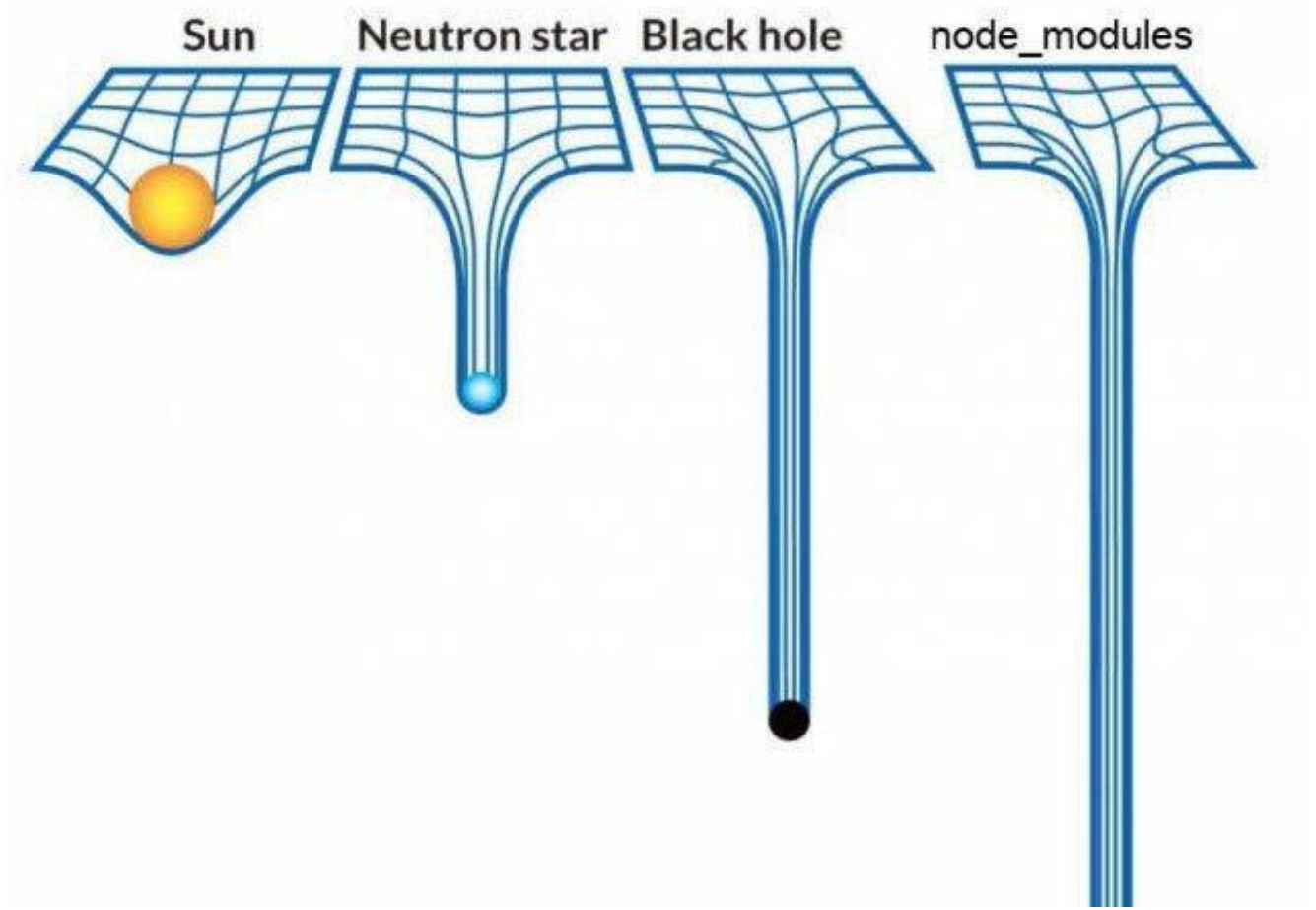
第一块：集成开发环境@angular/cli


node
JS
2009年，前端Big Bang!

Tool Chains based on NodeJS



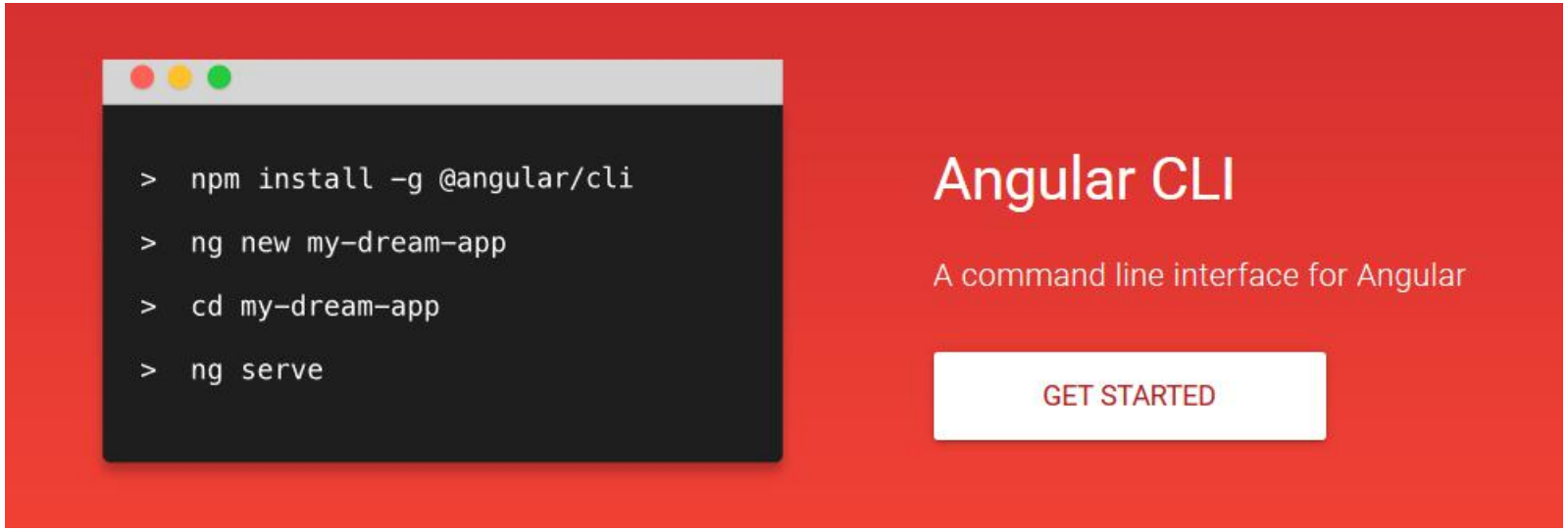
npm：世界上最糟糕的模块管理器



@angular/cli

把所有零散的Node工具都集成起来多好啊！

@angular/cli


> npm install -g @angular/cli
> ng new my-dream-app
> cd my-dream-app
> ng serve
Angular CLI
A command line interface for Angular
GET STARTED

@angular/cli

```
>npm install -g @angular/cli

>ng --version

>ng help

>ng new my-project1

>ng serve
```

@angular/cli

>ng generate

cl:class

c:component

d:directive

e:enum

m:module

p:pipe

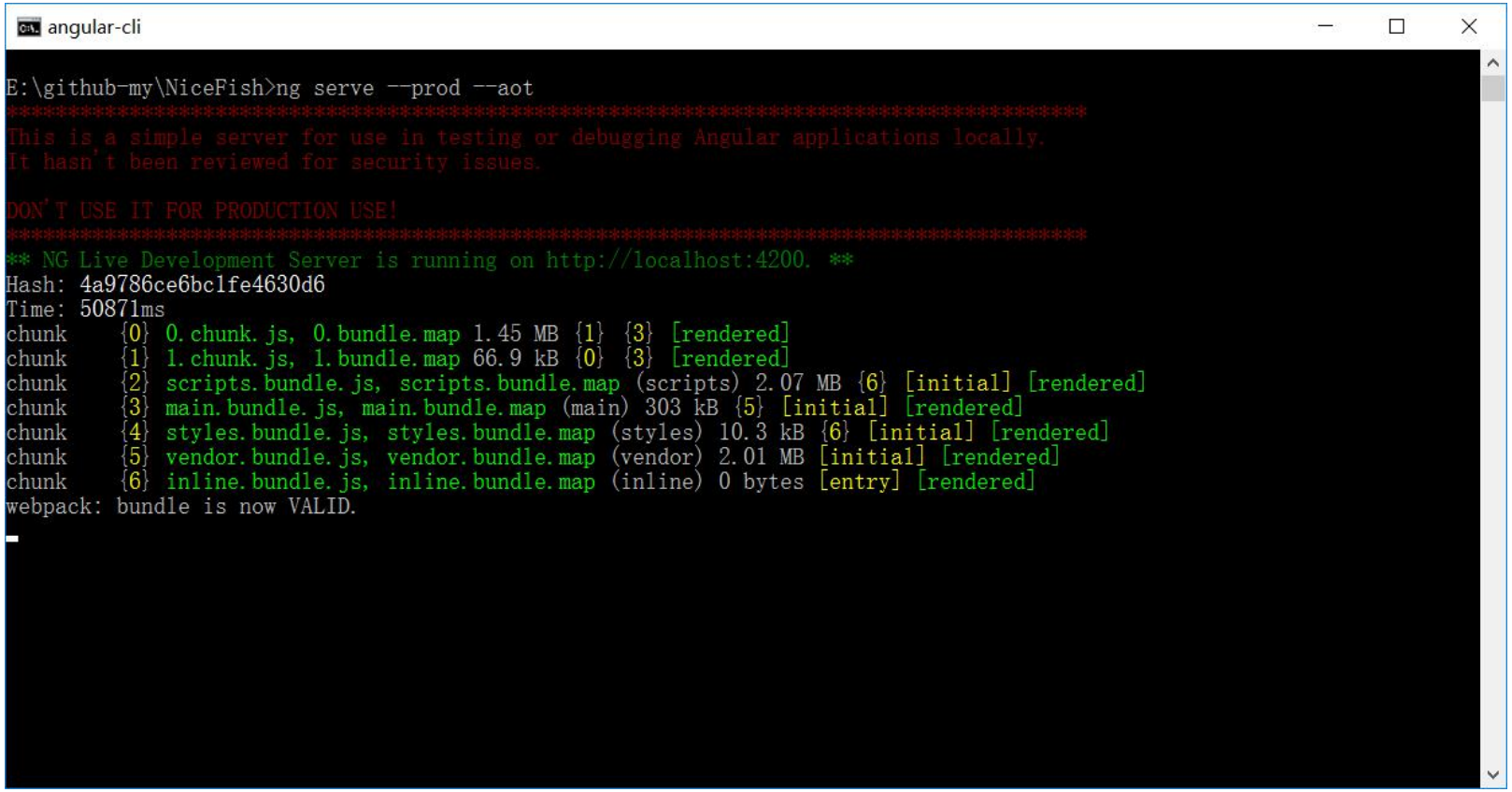
s:service

命令有简化版本：ng g c User

@angular/cli

- 在Windows下面，node-gyp这个包依赖于Visual Studio
- node-sass这个node模块被墙掉了，原因不明
- 所以强烈推荐采用cnpm安装

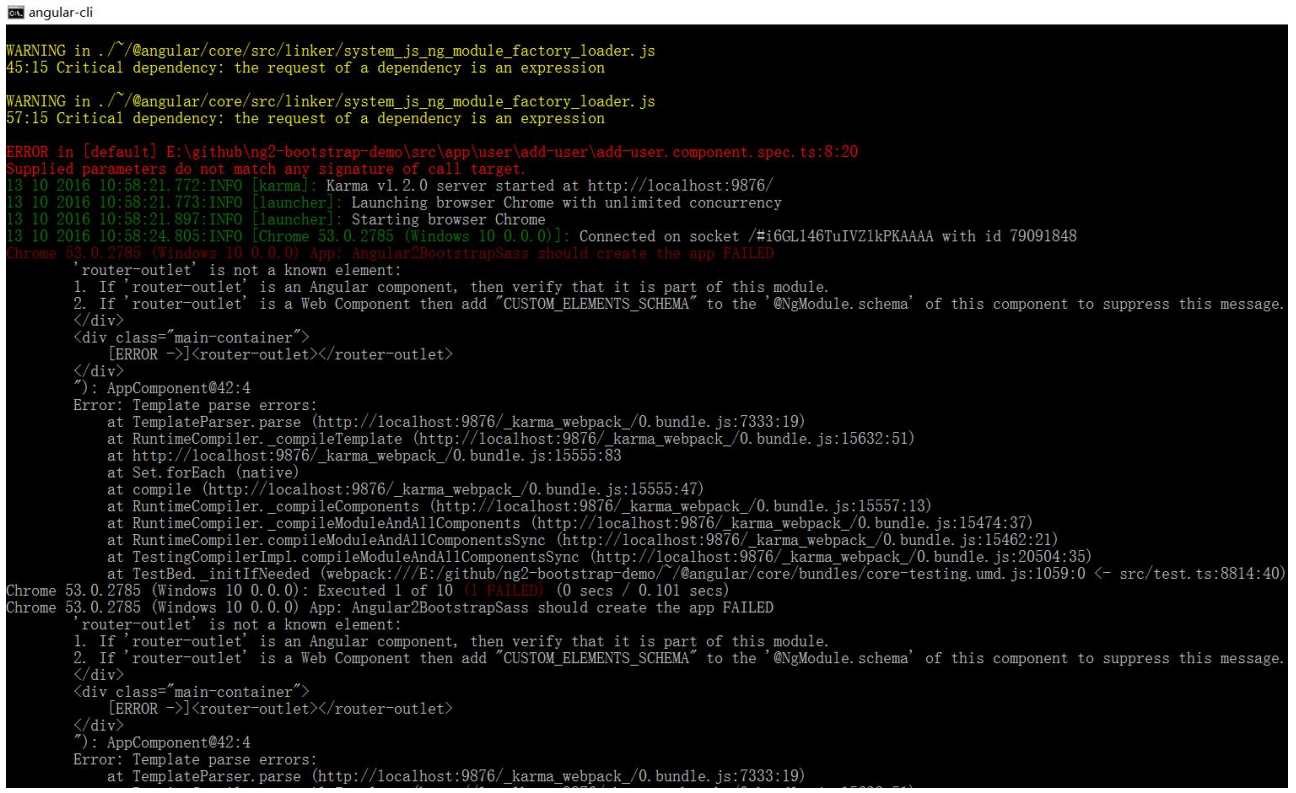
@angular/cli



ng serve --prod

注意：最新版的CLI加上--prod参数就自动AOT了，官网上的那篇文档过时了！

@angular/cli

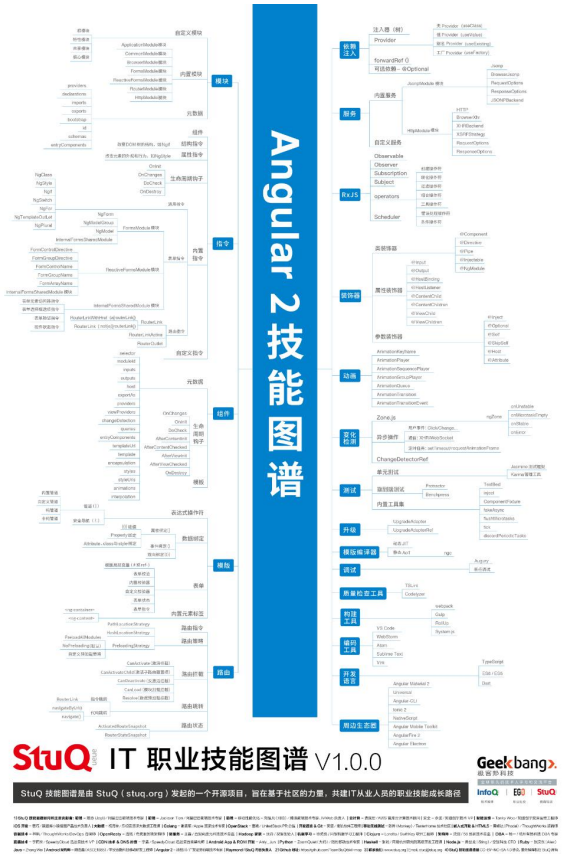


ng test

@angular/cli

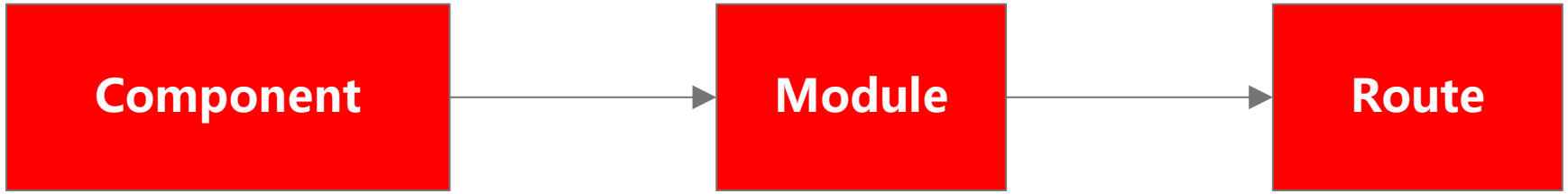
第二块：Angular中的3大核心概念

一张图覆盖Angular所有概念



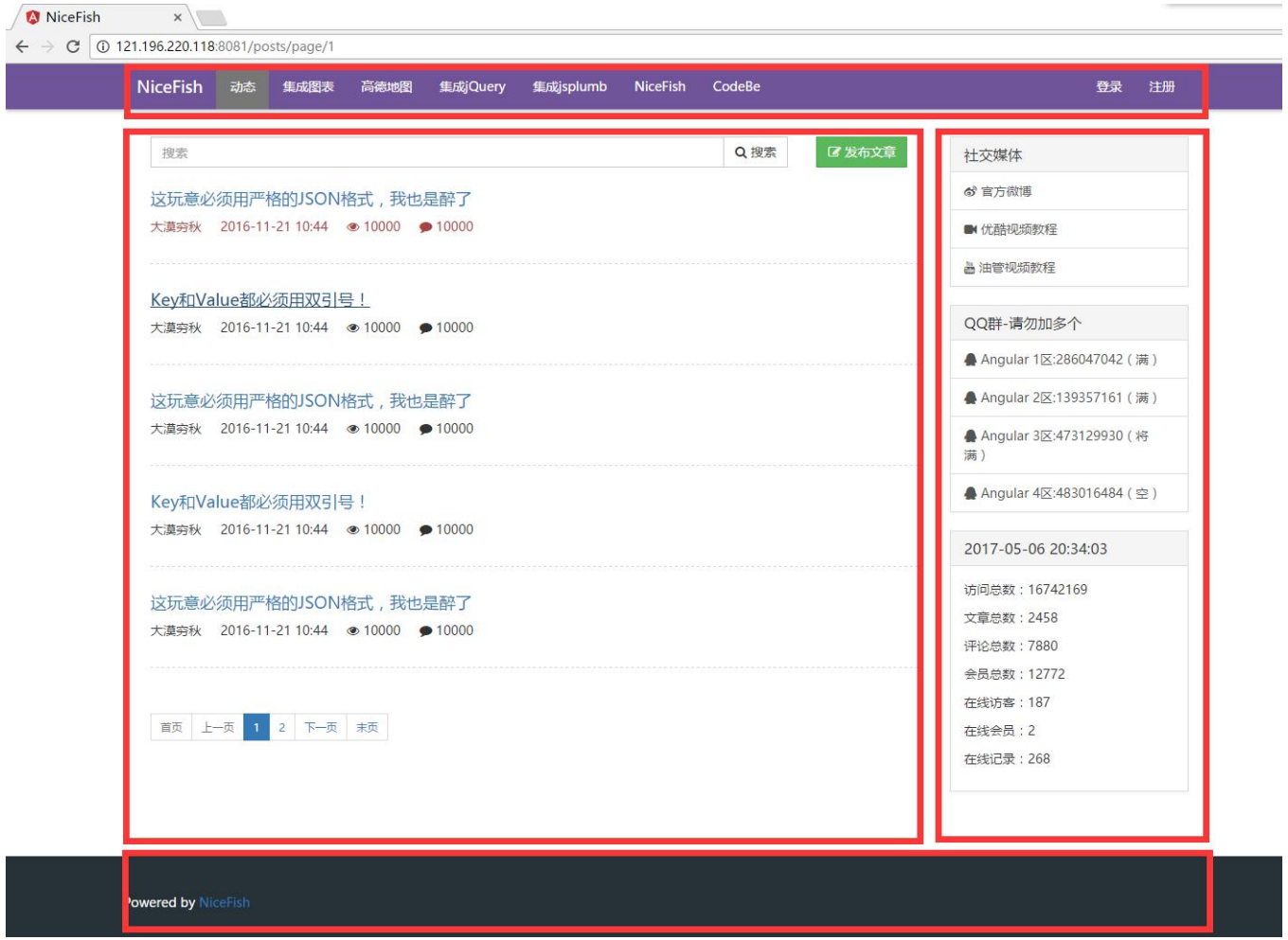
感谢《揭秘Angular2》的汤桂川老师提供如此详细的脑图！

Angular最核心的3个概念



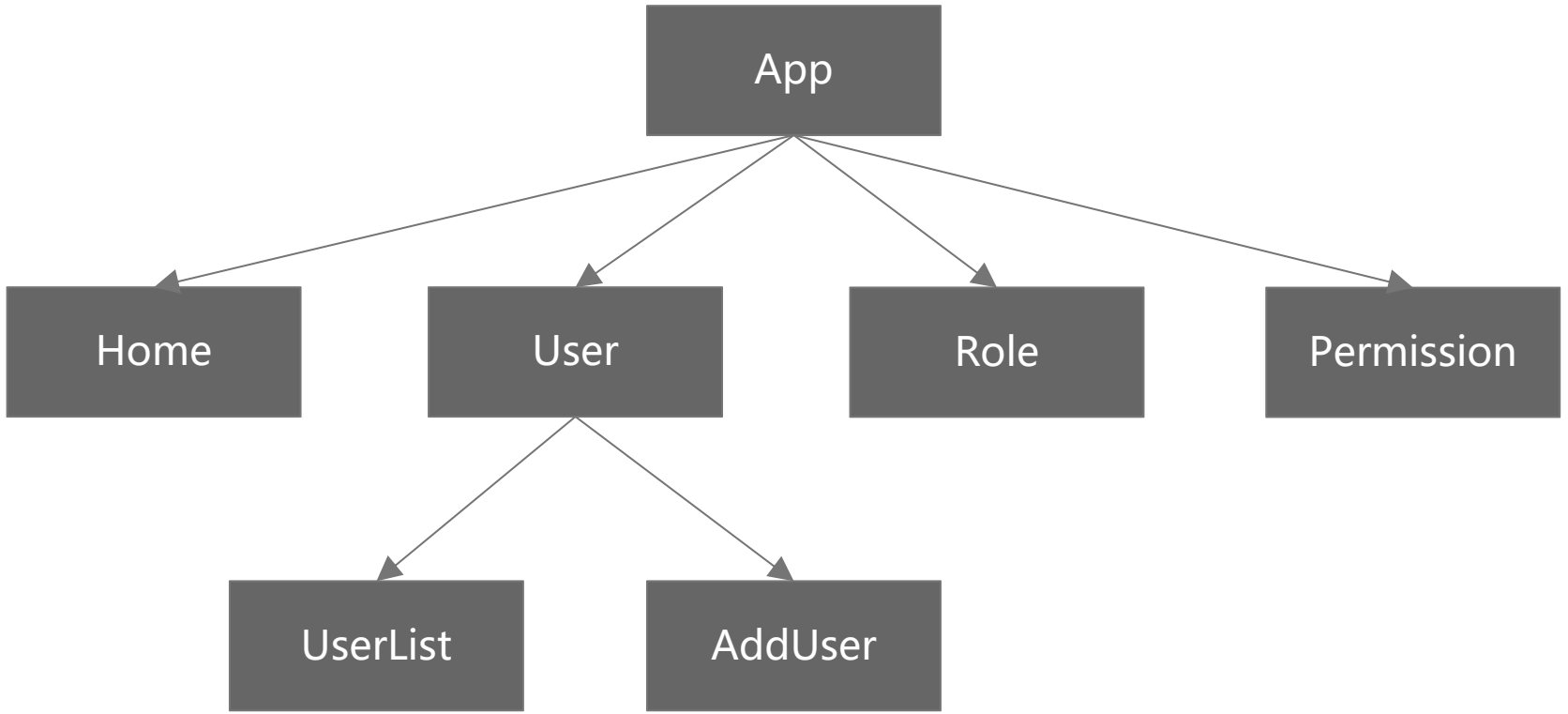
Angular最核心的概念是“组件化”

第一个核心概念：Component



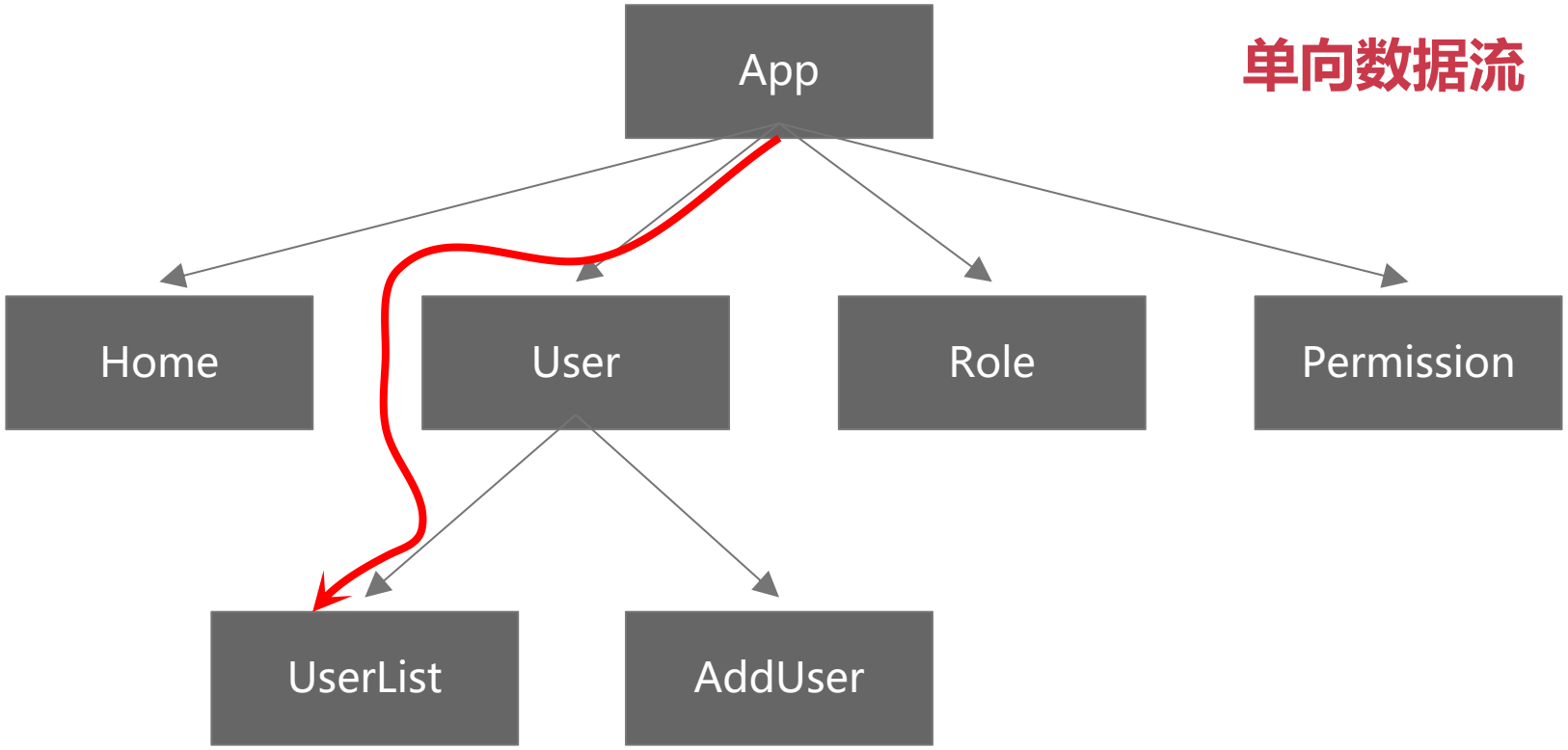
http://git.oschina.net/mumu-osc/NiceFish

Component Tree



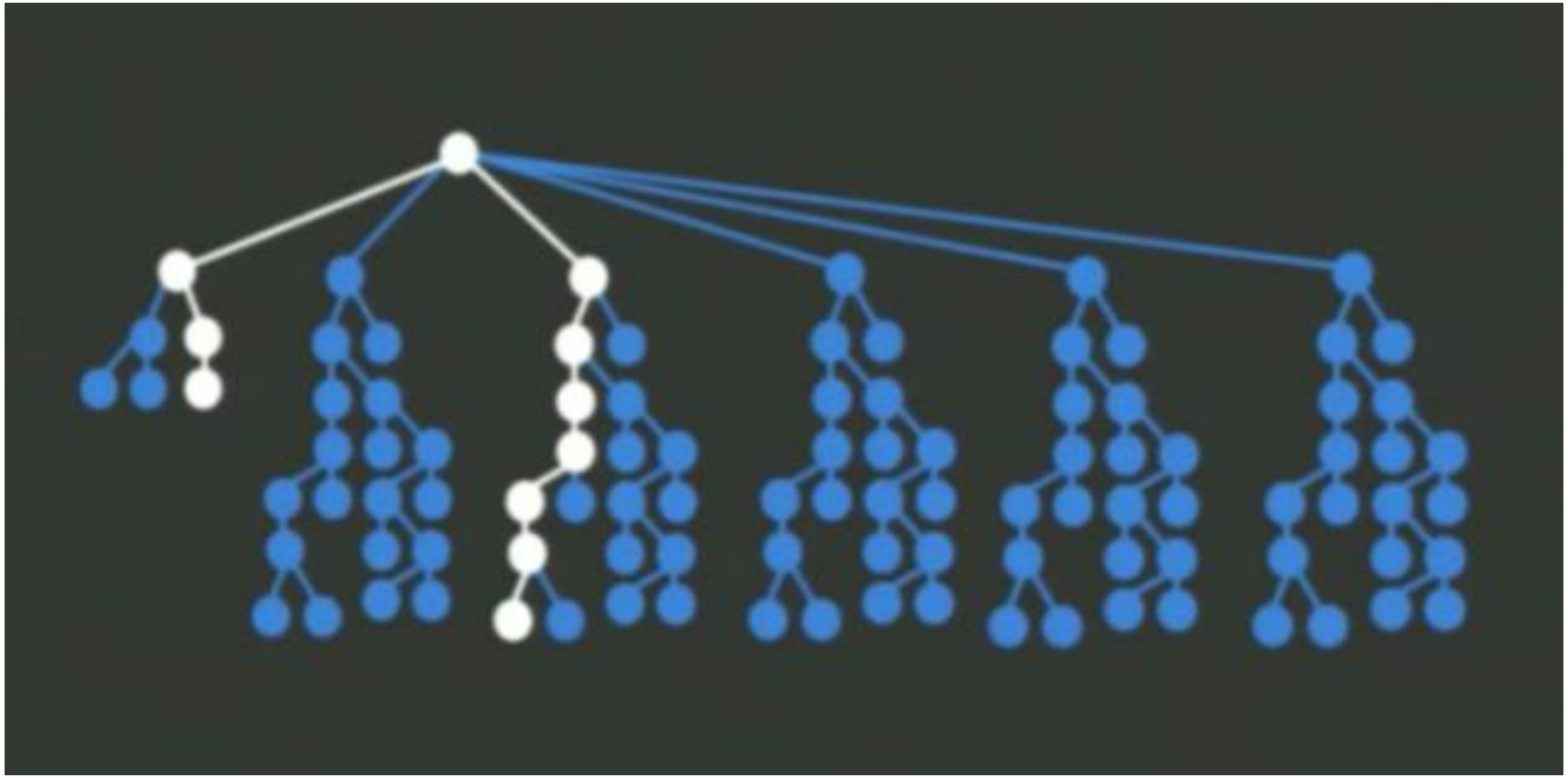
NiceFish实例项目的组件树结构

单向数据流



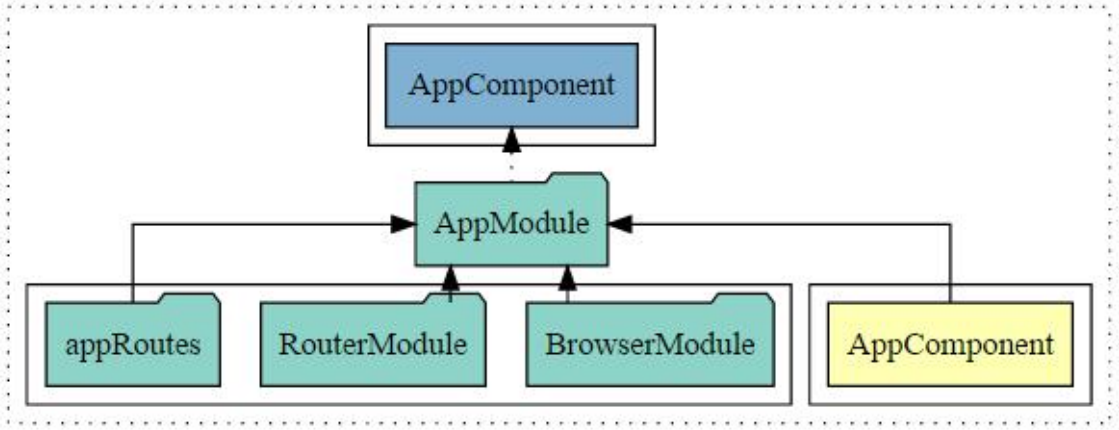
https://github.com/modern-javascript/angular2-data-flow

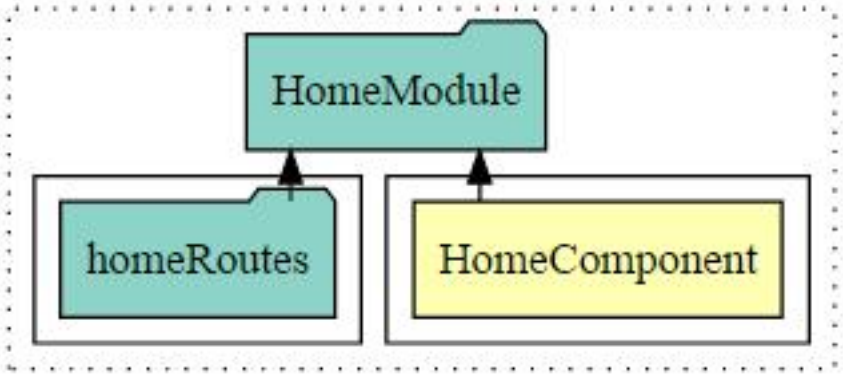
实际项目中的组件树

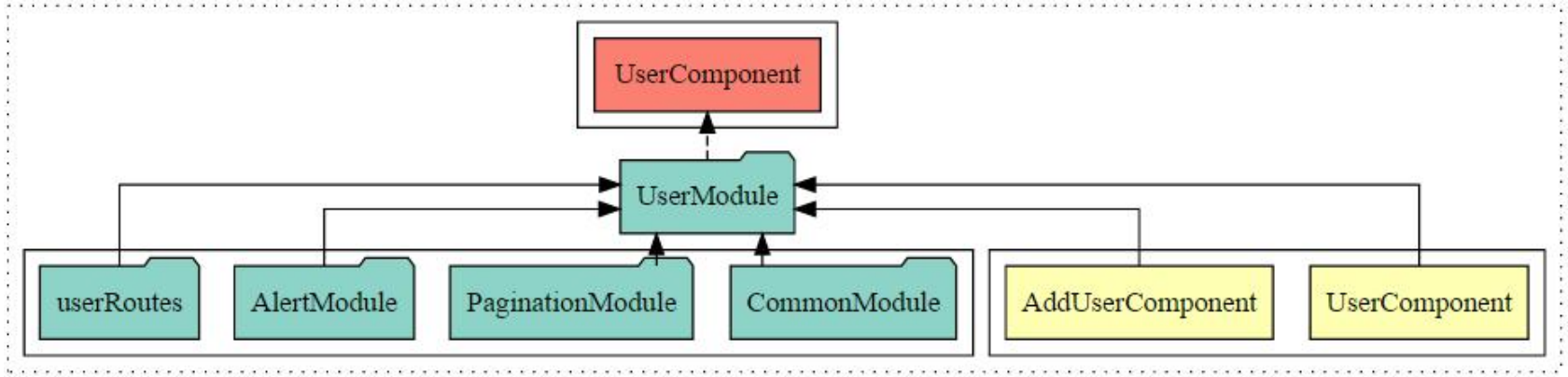


不可变数据类型

组件树生成器

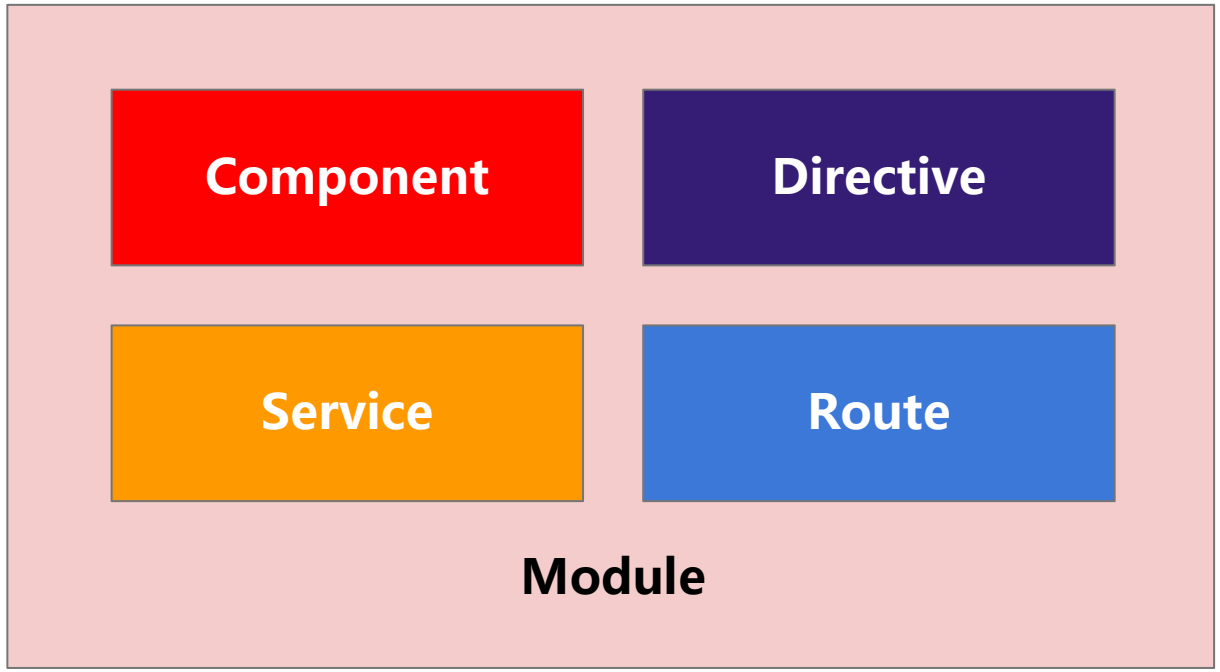






https://github.com/manekinekko/angular2-dependencies-graph

第二个核心概念：NgModule



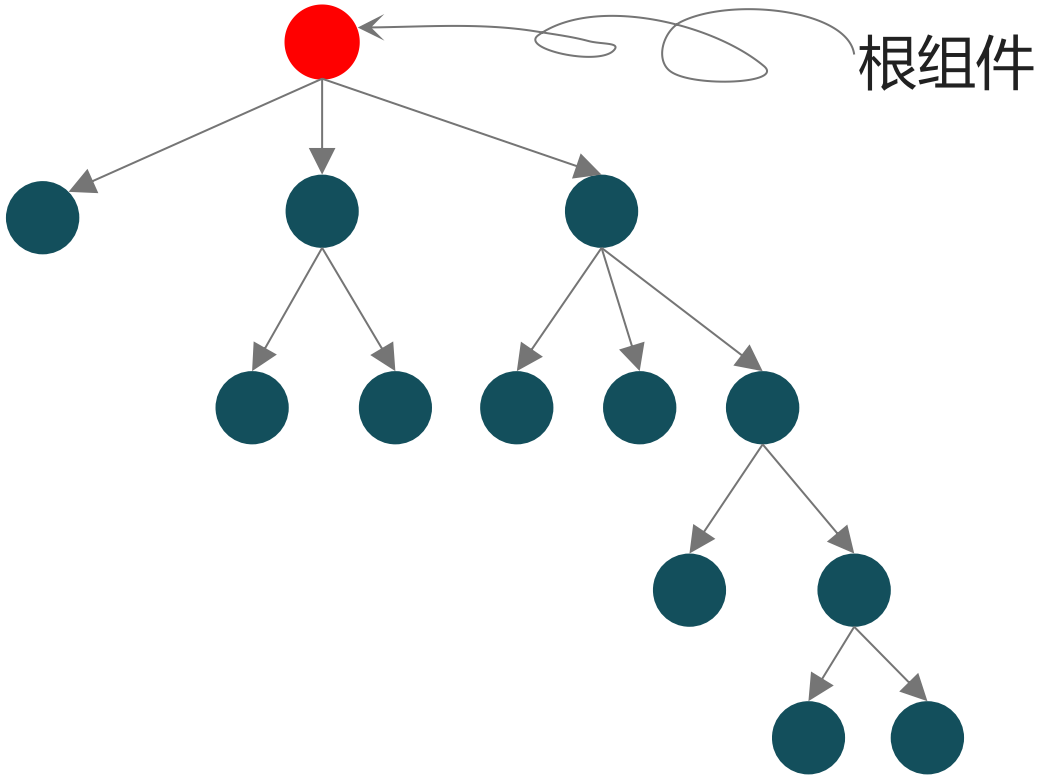
什么是模块？

第二个核心概念：NgModule

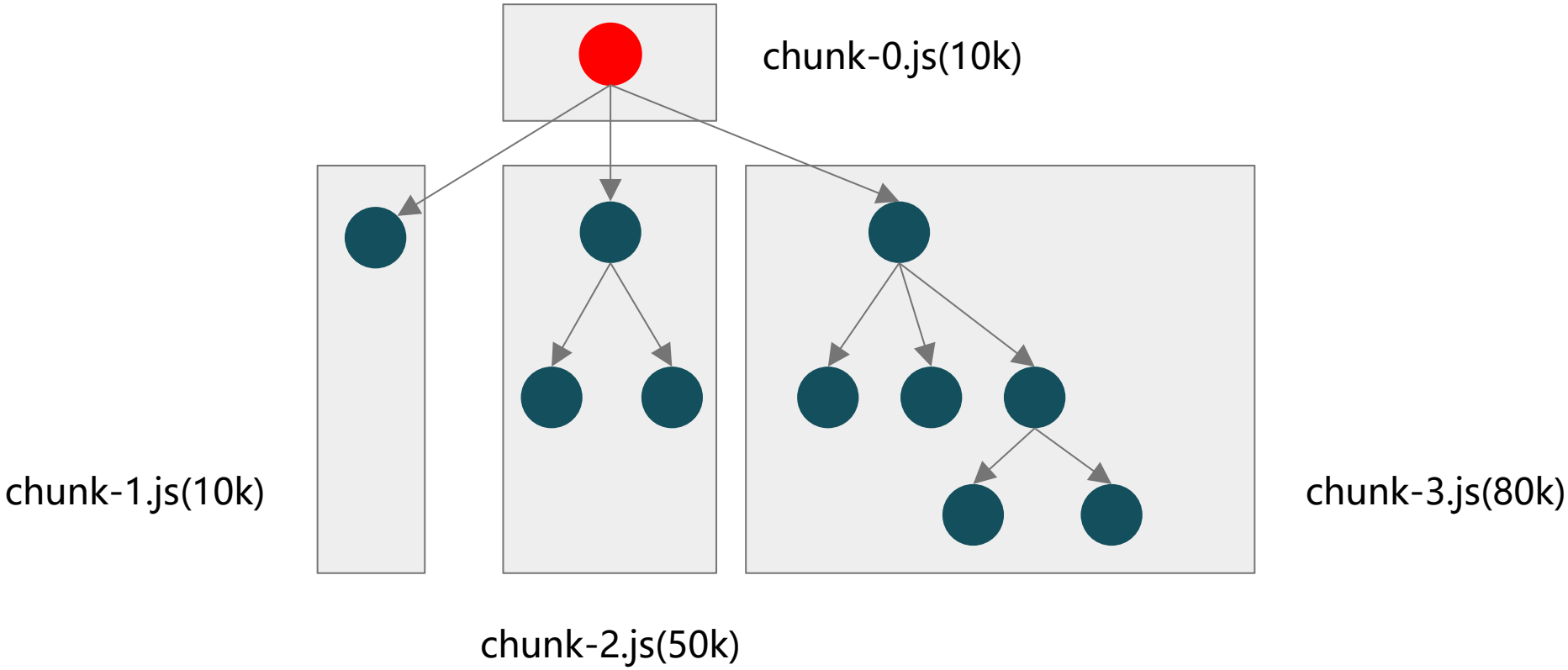
问题：为什么需要NgModule？

https://angular.cn/docs/ts/latest/cookbook/ngmodule-faq.html

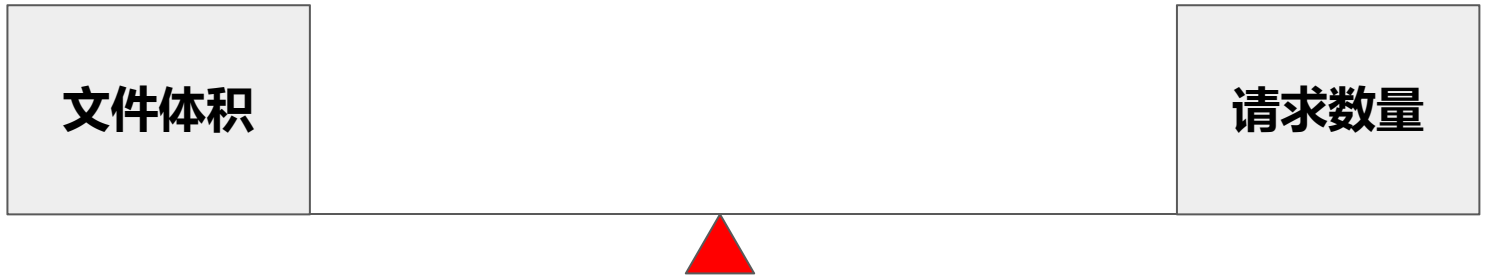
NgModule



NgModule



NgModule



这里需要取得一个平衡

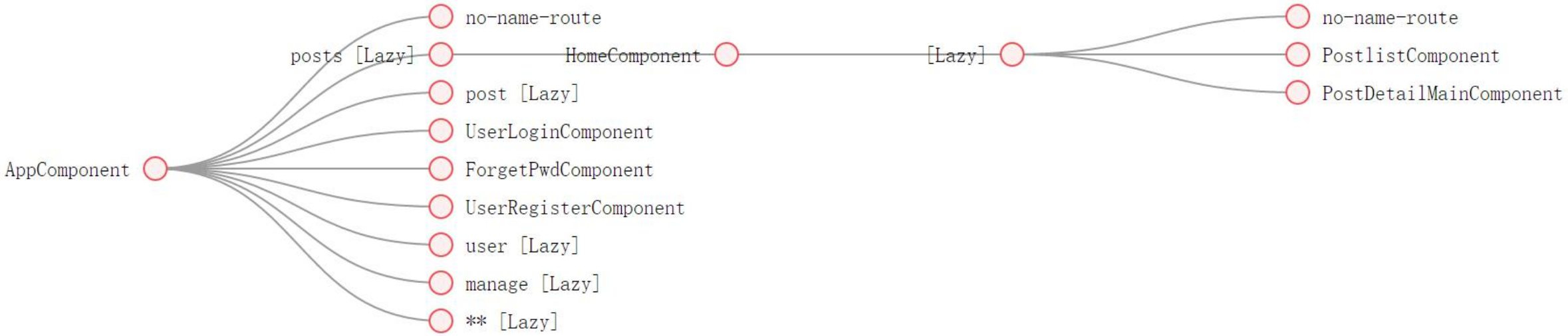
第三个核心概念：Router

问题：前端为什么需要Router？

前端为什么需要Router？

- 如果没有Router，浏览器的前进后退按钮没法用
- 如果没有Router，你将无法把URL拷贝并分享给你的朋友

Angular应用中路由形成的Tree形结构



Angular路由用法：静态路由

home.routes.ts

```
1 import {RouterModule} from "@angular/router";
2 import {HomeComponent} from "./home.component";
3
4 const homeRoutes= [
5   {path: '', component: HomeComponent}
6 ];
7
8 export default RouterModule.forChild(homeRoutes);
```

Angular路由用法：异步路由

app.routes.ts

```
1  import { RouterModule } from '@angular/router';
2
3  const appRoutes=[
4      {
5          path:'',
6          loadChildren:'app/home/home.module'
7      },{
8          path:'home',
9          loadChildren:'app/home/home.module'
10     },{
11         path:'user',
12         loadChildren:'app/user/user.module'
13     },{
14         path:'role',
15         loadChildren:'app/role/role.module'
16     },{
17         path:'**',//fallback router must in the last
18         loadChildren:'app/home/home.module'
19     }
20 ];
21 export default RouterModule.forRoot(appRoutes);
```

Angular路由用法：路由守卫

manage.routes.ts

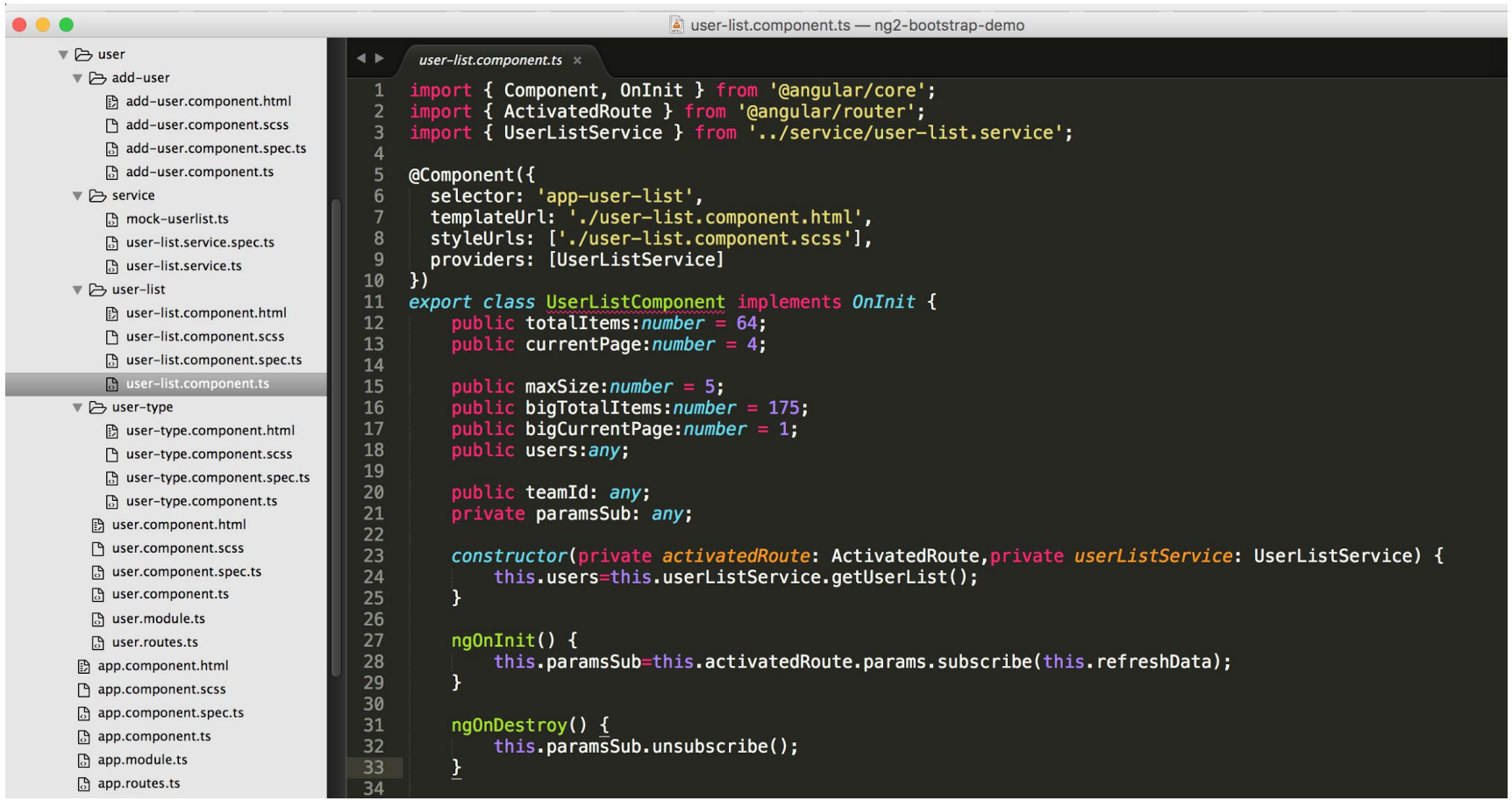
```
import { ManageMainComponent } from './manage-main/manage-main.component';
import { PostTableComponent } from './post-table/post-table.component';
import { CommentTableComponent } from './comment-table/comment-table.component';
import { UserTableComponent } from './user-table/user-table.component';
import { UserProfileComponent } from '../user/user-profile/user-profile.component';
import { SysParamComponent } from './sys-param/sys-param.component';
import { AuthGuard } from './auth-guard';

export const manageRoutes = [
    {
        path:'',
        component:ManageMainComponent,
        canActivate: [AuthGuard],
        children: [
            { path: '',redirectTo:'posttable/page/1',pathMatch:'full'},
            { path: 'posttable/page/:page', component: PostTableComponent },
            { path: 'commenttable/page/:page', component: CommentTableComponent },
            { path: 'usertable/page/:page', component: UserTableComponent },
            { path: 'usertable/edituser/:userId', component: UserProfileComponent },
            { path: 'usertable/newuser', component: UserProfileComponent },
            { path: 'sysparam', component: SysParamComponent },
            { path: '**', redirectTo:'posttable/page/1' }
        ]
    }
];
```

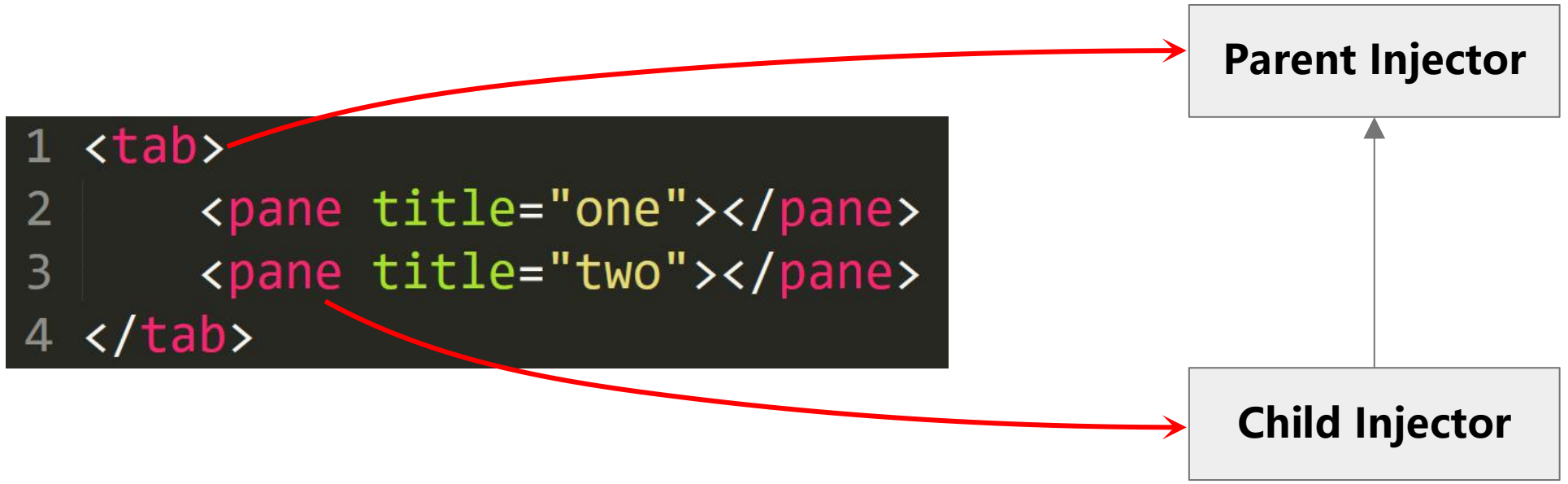
防止未授权访问

第三块：Angular架构特色

依赖注入


user-list.component.ts — ng2-bootstrap-demo
user
add-user
add-user.component.html
add-user.component.scss
add-user.component.spec.ts
add-user.component.ts
service
mock-userlist.ts
user-list.service.spec.ts
user-list.service.ts
user-list
user-list.component.html
user-list.component.scss
user-list.component.spec.ts
user-list.component.ts
user-type
user-type.component.html
user-type.component.scss
user-type.component.spec.ts
user-type.component.ts
user.component.html
user.component.scss
user.component.spec.ts
user.component.ts
user.module.ts
user.routes.ts
app.component.html
app.component.scss
app.component.spec.ts
app.component.ts
app.module.ts
app.routes.ts
user-list.component.ts
import { Component, OnInit } from '@angular/core';
import { ActivatedRoute } from '@angular/router';
import { UserListService } from '../service/user-list.service';
@Component({
selector: 'app-user-list',
templateUrl: './user-list.component.html',
styleUrls: ['./user-list.component.scss'],
providers: [UserListService]
})
export class UserListComponent implements OnInit {
public totalItems:number = 64;
public currentPage:number = 4;
public maxSize:number = 5;
public bigTotalItems:number = 175;
public bigCurrentPage:number = 1;
public users:any;
public teamId: any;
private paramsSub: any;
constructor(private activatedRoute: ActivatedRoute,private userListService: UserListService) {
this.users=this.userListService.getUserList();
}
ngOnInit() {
this.paramsSub=this.activatedRoute.params.subscribe(this.refreshData);
}
ngOnDestroy() {
this.paramsSub.unsubscribe();
}

依赖注入



注射器（Injector）也是一个树型结构

依赖注入


<top frame>
Preserve log
EXCEPTION: No provider for NgForm (ShowError -> NgForm)
browser_adapter.js:110
STACKTRACE:
browser_adapter.js:106
Error: DI Exception
browser_adapter.js:106
at NoBindingError.BaseException (lang.js:163)
at NoBindingError.AbstractBindingError (exceptions.js:48)
at new NoBindingError (exceptions.js:70)
at Injector.execute.$__export._throwOrNull (injector.js:640)
at Injector.execute.$__export._getByKeySelf (injector.js:645)
at Injector.execute.$__export._getByKey (injector.js:629)
at Injector.execute.$__export._getByDependency (injector.js:621)
at Injector.execute.$__export._instantiate (injector.js:518)
at Injector.execute.$__export._new (injector.js:491)
at InjectorInlineStrategy.execute.$__export.instantiateBinding (injector.js:283)
ERROR CONTEXT:
browser_adapter.js:106
browser_adapter.js:106
_Context {element: show-error.ng-binding, componentElement: template-driven-forms, injector: Injector}
componentElement: template-driven-forms
element: show-error.ng-binding
injector: Injector
__proto__: _Context
The injector of the element where
Angular failed to instantiate a directive

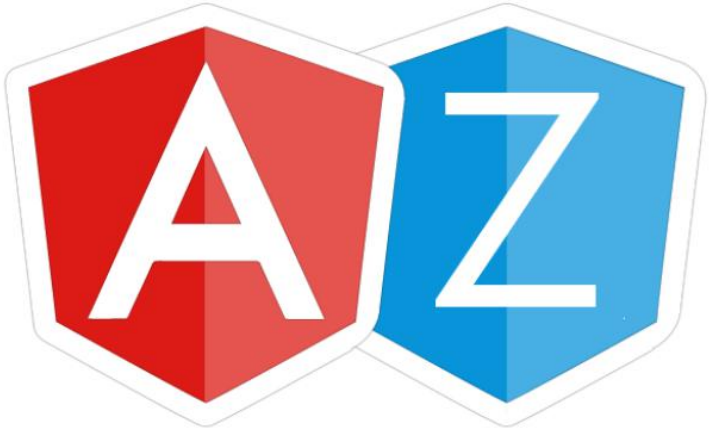
依赖注入

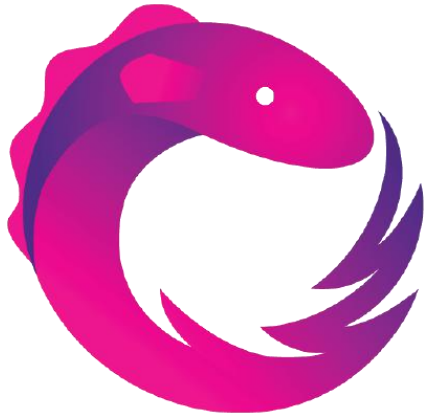
- 每一个HTML标签上面都会有一个注射器实例
- 注射是通过constructor进行的
- @Component是@Injectable的子类

https://vsavkin.com/dependency-injection-in-angular-1-and-angular-2-d69589979c18#.3ptlx48zf

https://my.oschina.net/mumu/blog/775695

数据绑定

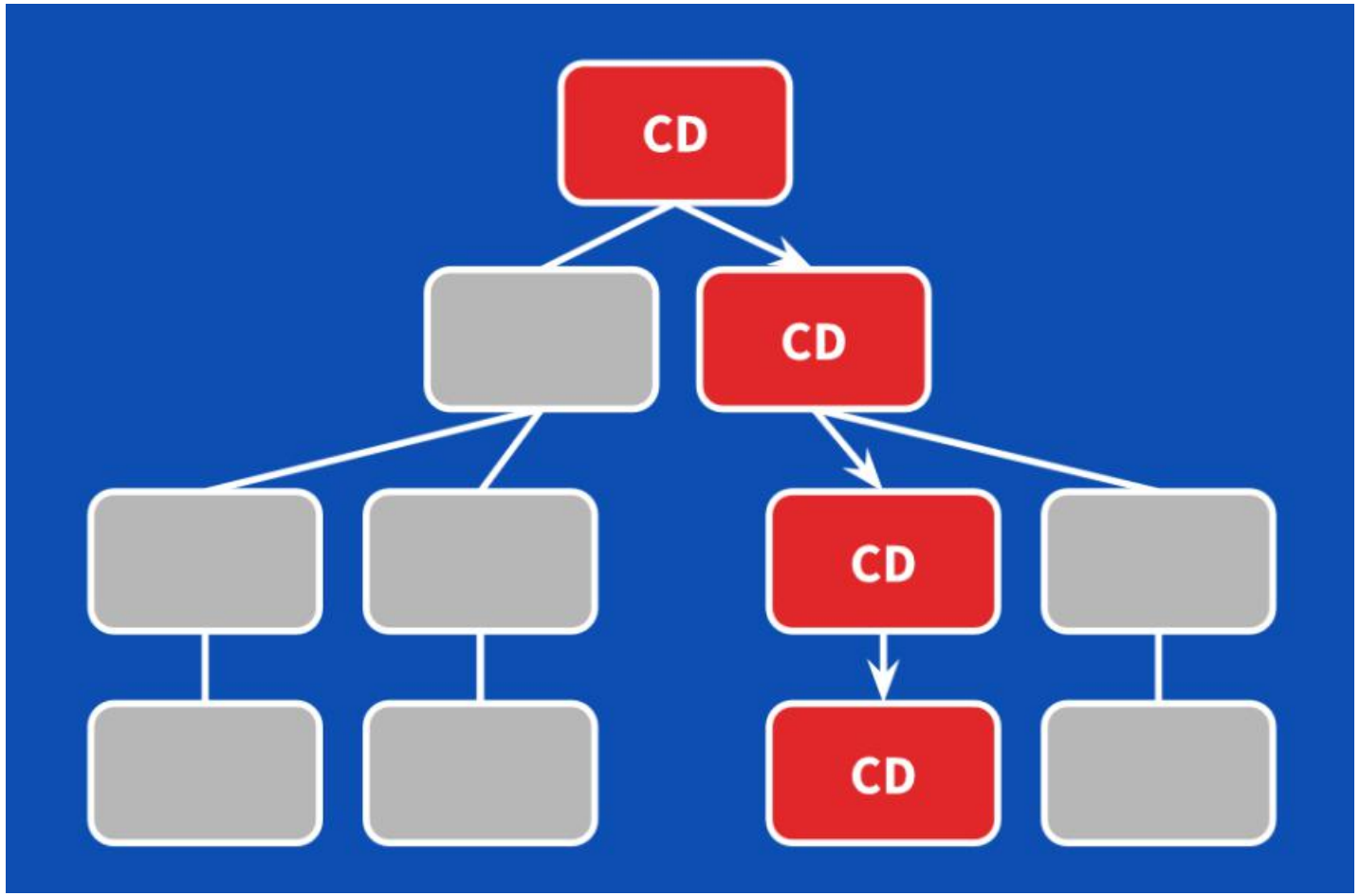




Immutable Data

简而言之，新版本的Angular已经彻底重写了脏检查机制

数据绑定



极其高效的脏检查机制

数据绑定

http://blog.thoughtram.io/angular/2016/02/22/angular-2-change-detection-explained.html

http://teropa.info/blog/2015/03/02/change-and-its-detection-in-javascript-frameworks.html

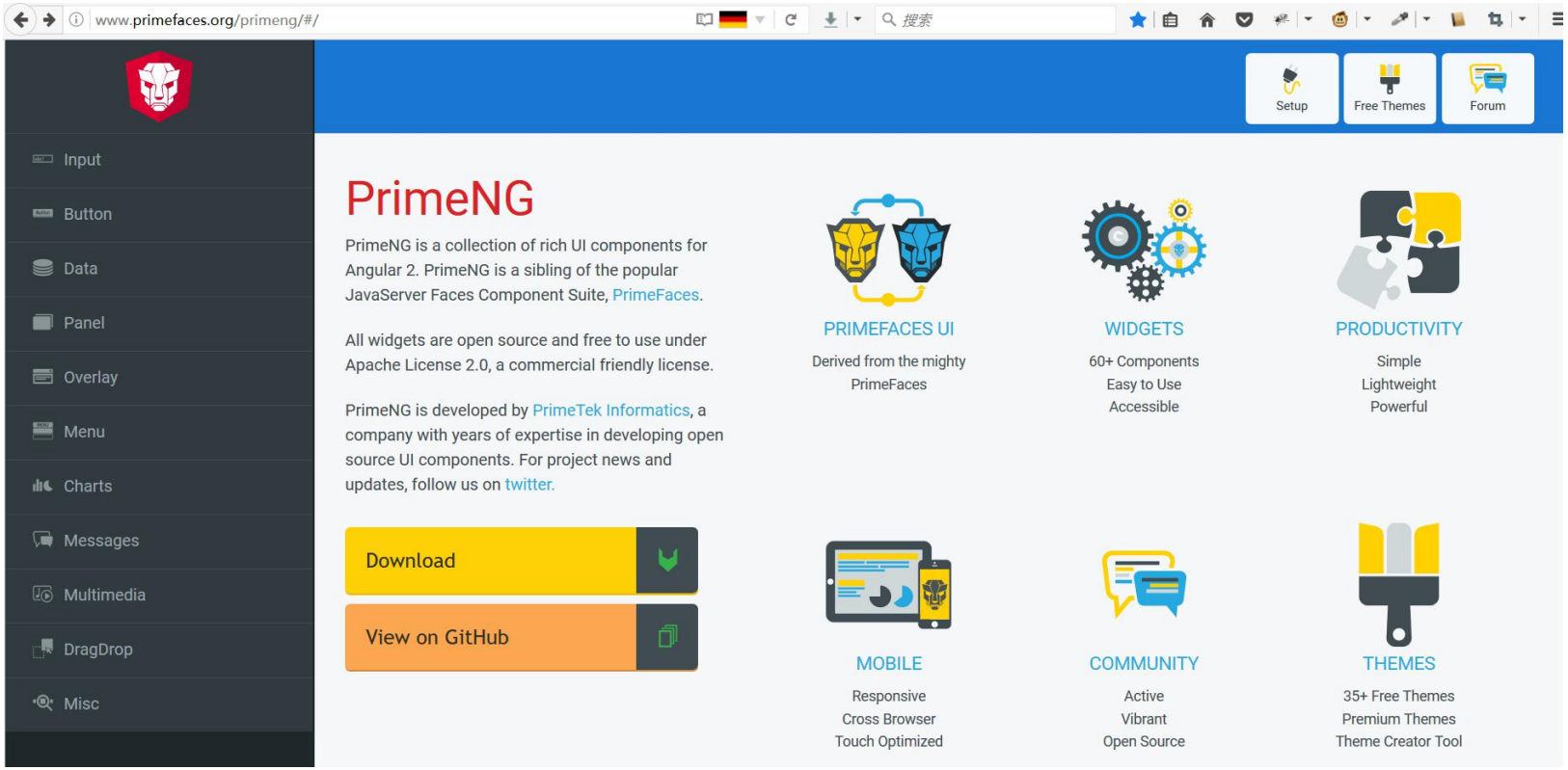
https://vsavkin.com/change-detection-in-angular-2-4f216b855d4c

第四块：UI库

UI Libraries

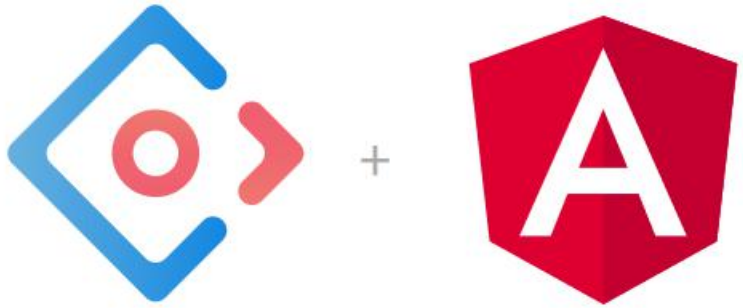
- PrimeNG ：到目前为止最完善的开源免费UI组件库
- NG-Zorro：来自阿里云的开源组件库
- Jigsaw：来自ZTE中兴通讯的开源组件库
- Clarity：来自Vmware的组件库
- Angular-Material：Angular官方提供的组件库
- Ionic ：专门为Angular设计的移动端组件库

UI Libraries



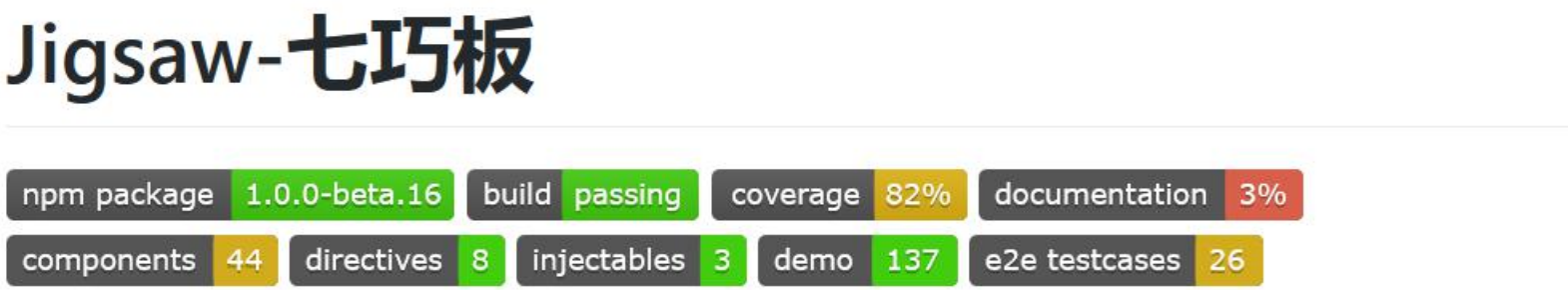
PrimeNG

UI Libraries



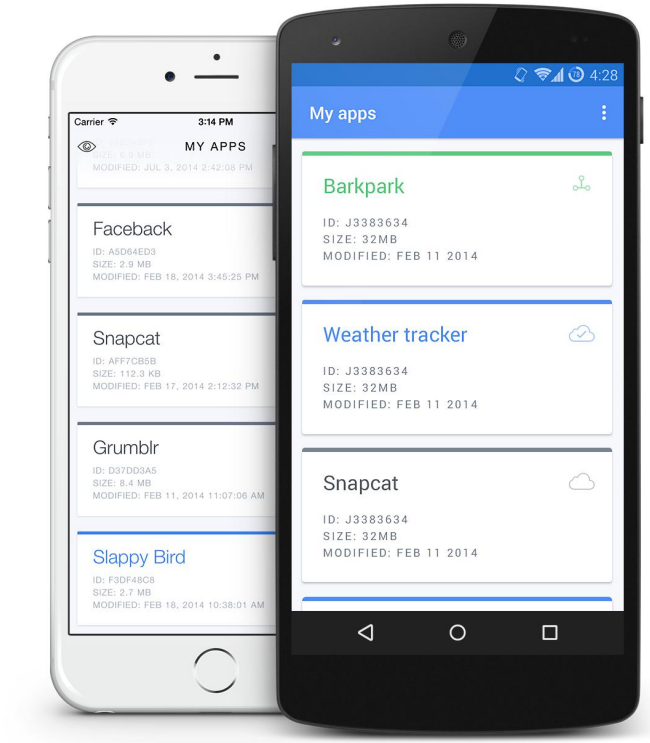
NG-Zorro

UI Libraries



Jigsaw

移动端开发神器



ionic，底层是NativeScript

第五块：参考资源推荐

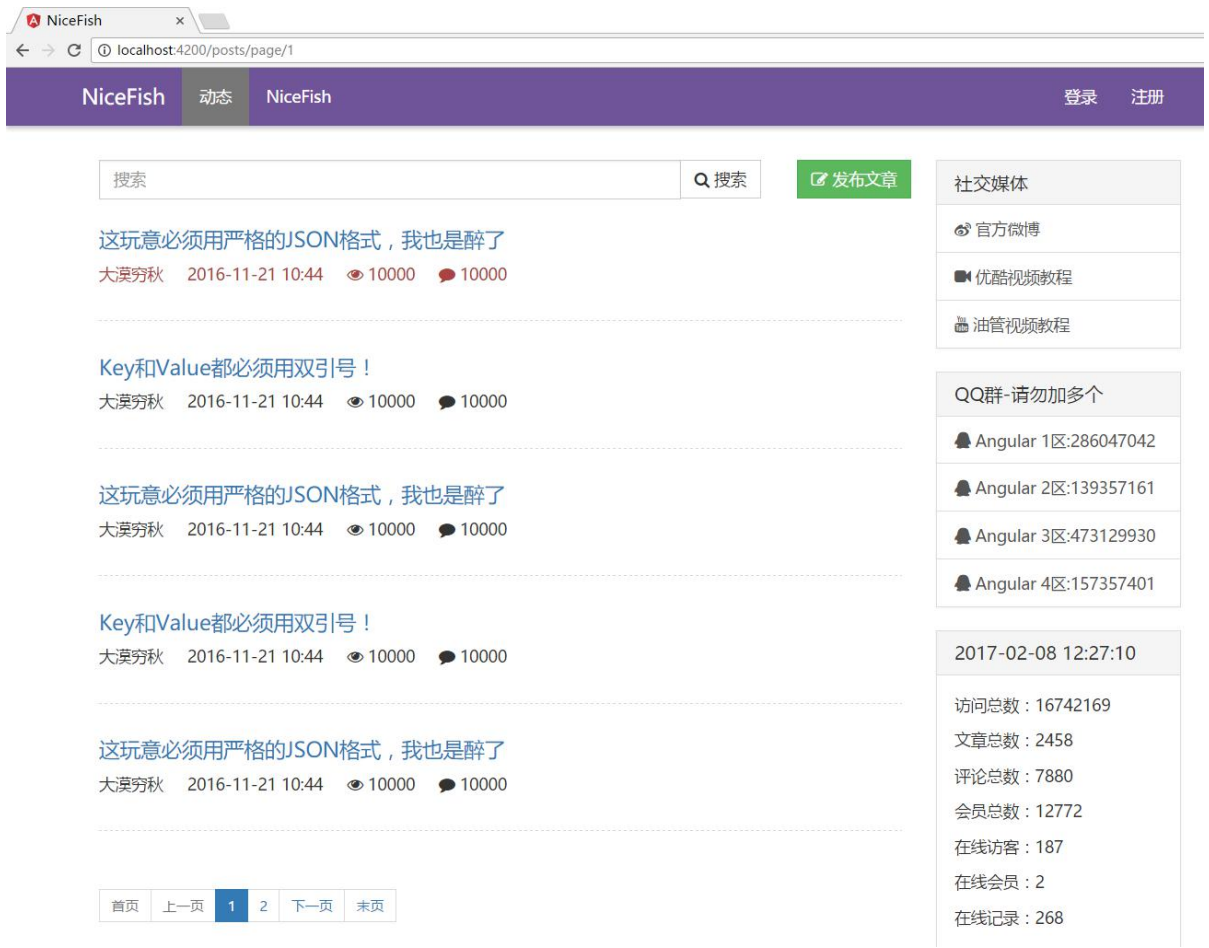
特别推荐：ng2-admin

https://github.com/akveo/ng2-admin

特别推荐：JHipster-后端基于SpringMVC

https://jhipster.github.io/

一般推荐：NiceFish



http://git.oschina.net/mumu-osc/NiceFish

谢谢

https://damoqiongqiu.github.io/