



项目实战

创建一个新项目

1. 首先打开控制台（cmd, terminal）,执行cd命令挂载到你想创建项目的地方

比如： `cd /home/git`

2. 使用tui命令工具创建项目,根据提示输入项目文件夹名称

`yo tui:new`

```
at process: clickatback (internal/process/next_tick.js:188:7)
xiaojian@xiaojian-virtual-machine:~/Git/test$ yo tui:new
? 请输入要创建的项目目录名称 my-app
```

3. cd进入项目目录,假设项目名称为my-app,然后执行npm install命令安装项目

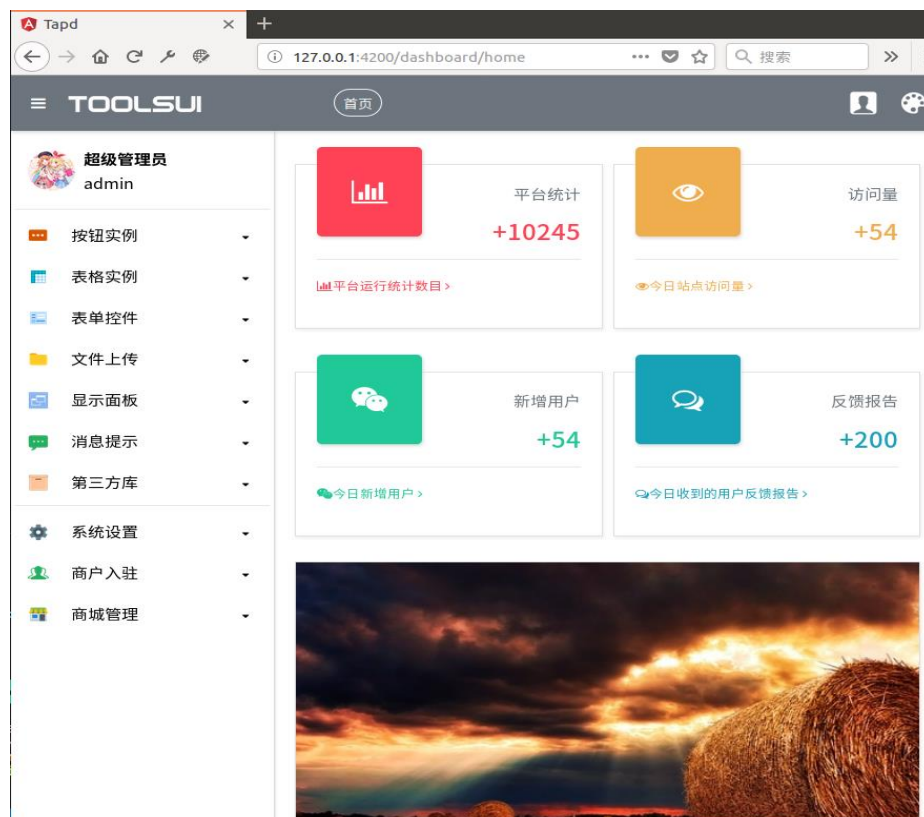
`cd my-app`

`npm install`

```
xiaojian@xiaojian-virtual-machine:~/Git/test$ cd my-app/
xiaojian@xiaojian-virtual-machine:~/Git/test/my-app$ npm install
```

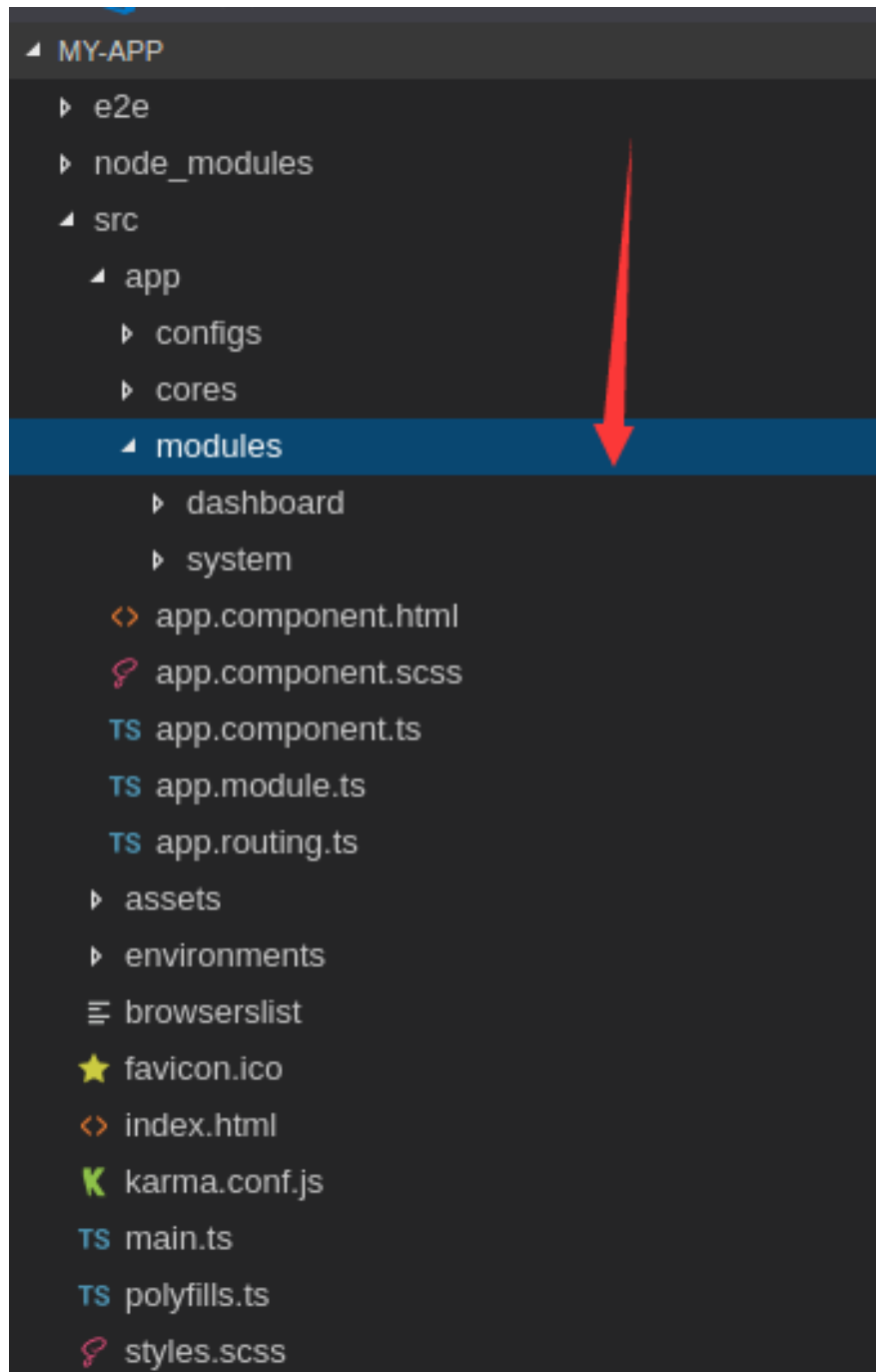
运行项目

1. 安装好的项目直接执行 `npm run serve` 即可运行项目
2. 在浏览器中输入 <http://127.0.0.1:4200> 可以查看项目运行结果
3. 打包部署项目之前，使用 `npx ng serve --prod` 运行测试下项目有无异常
4. 使用 `npm run build` 打包项目



查看项目目录

我们需要开发的模块全部存放在右图中
Modules文件夹里面



项目界面调整-系统登入页面

登入页面配置参数在app/configs/app.config.ts文件中

```
// 登入页面配置参数
LOGIN_PAGE: {
  LOGO: 'favicon.ico',
  TITLE: 'Angular Tools UI',
  SUBJECT: '管理员登入',
  DESCRIPTION: '欢迎使用ng5-dashboard,为angular爱好者打造的精简模板',
  FIRST_LABEL: '账户',
  FIRST_PLACEHOLDER: '请输入您的账户',
  SECOND_LABEL: '密码',
  SECOND_PLACEHOLDER: '请输入您的密码',
  BACKGROUND_IMAGE_SRC: 'url(assets/images/background/3.jpg)'
},|
```

ICON



Angular Tools UI

TITLE

SUBJECT

管理员登入

欢迎使用ng5-dashboard,为angular爱好者打造的精简模板

DESCRIPTION

BACKGROUND_IMAGE_SRC

账户



请输入您的账户

密码



请输入您的密码

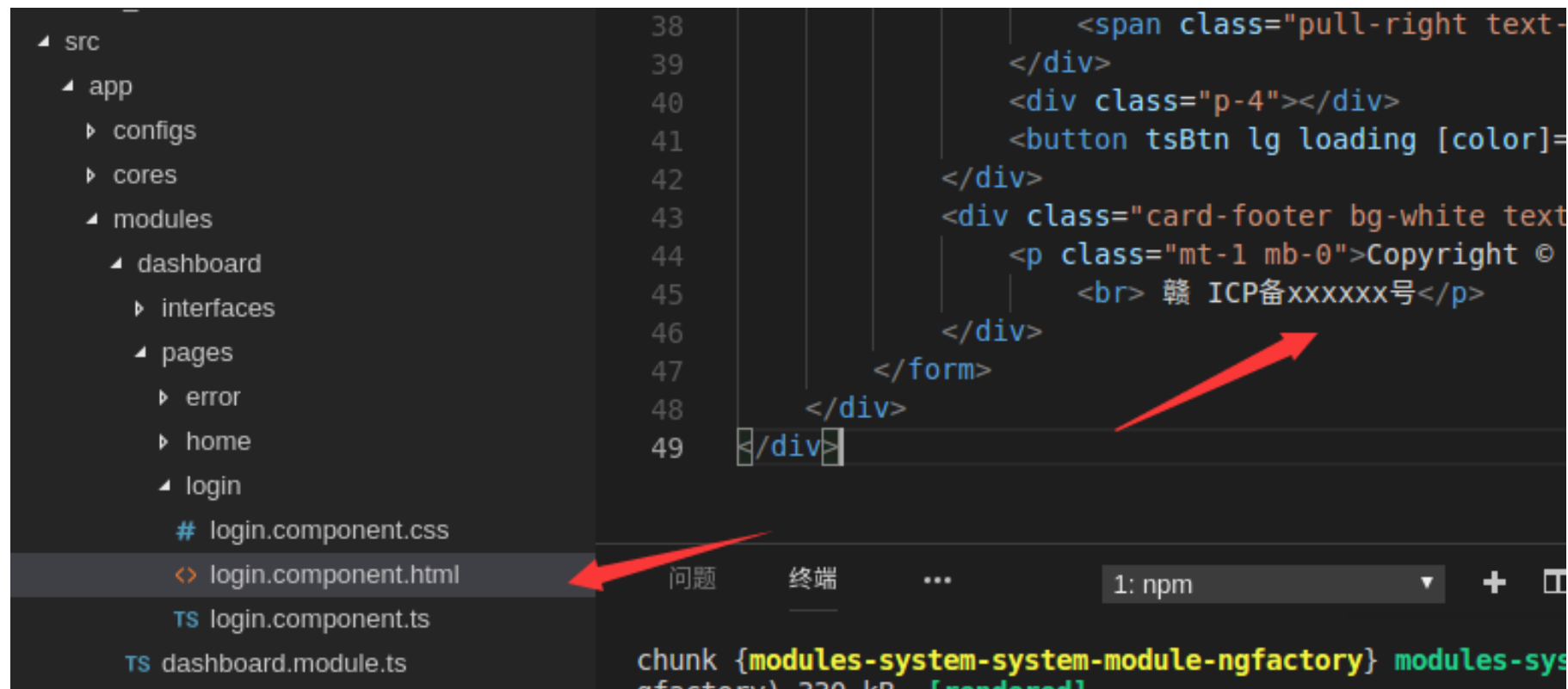
[忘记密码?](#)

登入

底部请到
login.component.html中
修改

© 2016 XiaoJian. All Rights Reserved.
赣 ICP备16010587号

其他相关样式需要自行到app/modules/dashboard/pages/login文件夹里面修改



项目界面调整-菜单配置

左侧菜单配置参数在app/configs/app.config.ts文件中

```
// 系统菜单配置参数
MENU_CONFIG: {
  // url(assets/images/background/1.jpg)可以设置菜单的背景图片
  BACKGROUND_IMAGE_SRC: '',
  BACKGROUND_COLOR: 'rgba(255,255,255,0.9)',
  DEFAULT_TEXT_COLOR: 'black',
  LINE_COLOR: 'rgb(0, 0, 0,.1)'
},
```



超级管理员

admin



按钮实例



表格实例



表单控件



文件上传



显示面板



消息提示



第三方库



系统设置



商户入驻



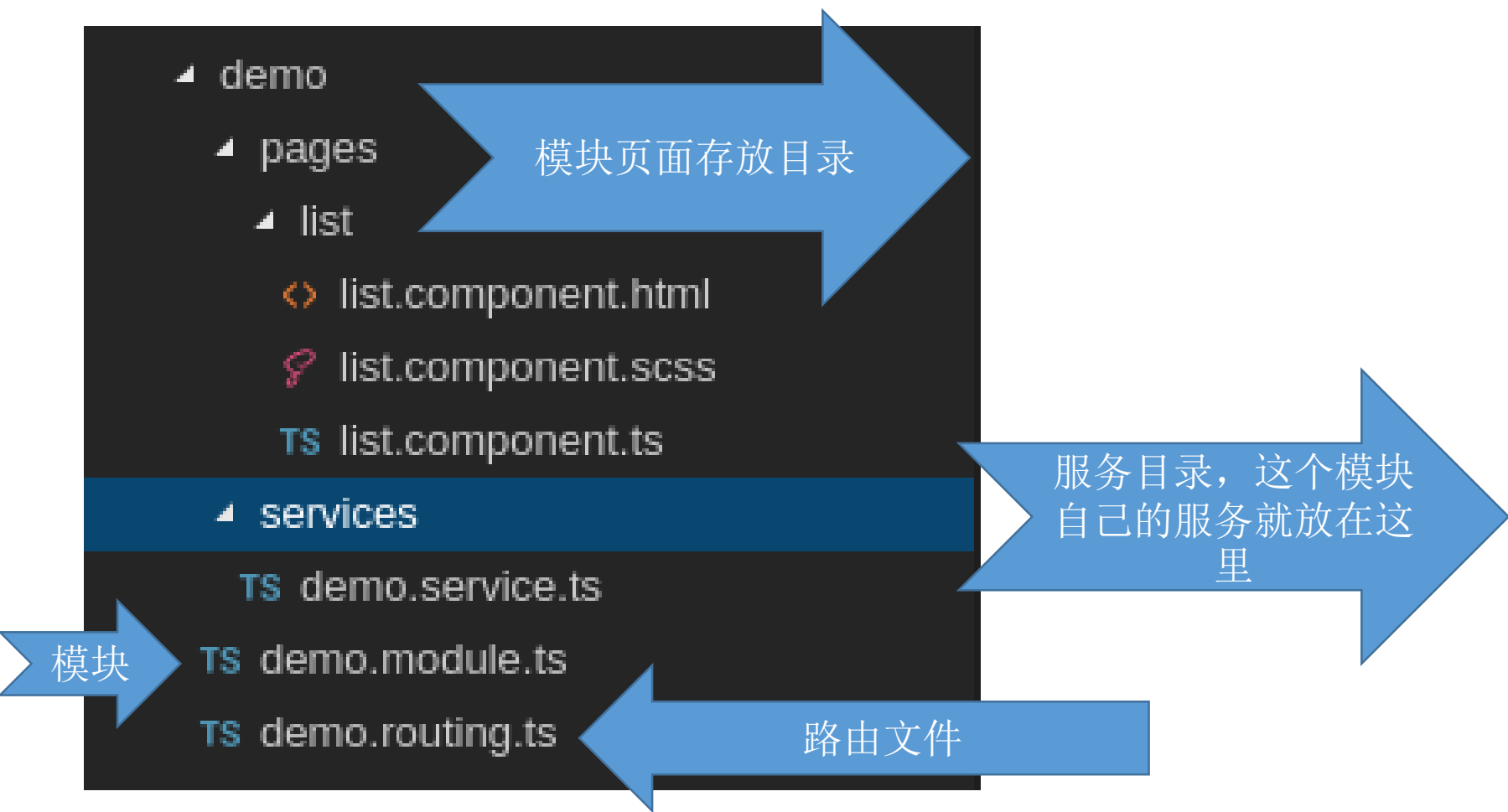
商城管理

开始开发-创建一个模块

1.使用 `yo tui:module` 命令创建一个模块，比如demo模块

```
xiaojian@xiaojian-virtual-machine:~/Git/test/my-app$ yo tui:module
? 要创建的模块名称 demo
? 要不要创建一个页面例子? Yes
? 要创建的页面名称 list
? 要不要在模块里面建一个服务? Yes
app.routing.ts中的路由规则: { path: 'demo', loadChildren: './modules/demo/demo.module
DemoModule', canActivate: [GuardService] }
  create src/app/modules/demo/demo.module.ts
  create src/app/modules/demo/demo.routing.ts
  create src/app/modules/demo/pages/list/list.component.ts
  create src/app/modules/demo/pages/list/list.component.scss
  create src/app/modules/demo/pages/list/list.component.html
  create src/app/modules/demo/services/demo.service.ts
xiaojian@xiaojian-virtual-machine:~/Git/test/my-app$
```

按上一张幻灯片的操作截图执行完后，我们得到一个创建好的demo目录



模块文件

注意写作者，注释必须完善

这里引入组件模块，
如日期，下拉等组件模块

组件，服务都不会写在在这里了
，统一写在路由文件里面

demo.routing.ts 路由文件

路由配置

页面，组件，管道声明数组

模态框，窗口 声明数组

服务提供商数组

```
/**
 * 页面组件
 */
import { ListComponent } from './pages/list/list.component';

/**
 * 相关服务
 */
import { DemoService } from './services/demo.service';

const routes: Routes = [
  { path: 'list', component: ListComponent },
];

/**
 * 指令、组件、管道声明
 */
export const declarationComponents = [
  ListComponent
];

/**
 * 动态组件（模态框，窗口）
 */
export const entryComponents = [];

/**
 * 服务列表
 */
export const providers = [
  DemoService
];

@NgModule({
  imports: [RouterModule.forChild(routes)],
  exports: [RouterModule]
})
export class DemoRoutingModule { }
```

把创建的模块配置到根路由中，进行加载 **app/app.routing.ts**

```
import { NgModule } from '@angular/core';
import { RouterModule, Routes, PreloadAllModules } from '@angular/router';
import { GuardService } from '../cores/services';

const routes: Routes = [

  // 此处设置网站首页
  { path: '', redirectTo: 'dashboard/home', pathMatch: 'full' },

  // 懒加载子模块
  { path: 'system', loadChildren: './modules/system/system.module#SystemModule', canActivate: [GuardService] },
  { path: 'demo', loadChildren: './modules/demo/demo.module#DemoModule', canActivate: [GuardService] },

  // 最后全局匹配其他链接
  { path: '**', redirectTo: 'dashboard/error', pathMatch: 'full' },

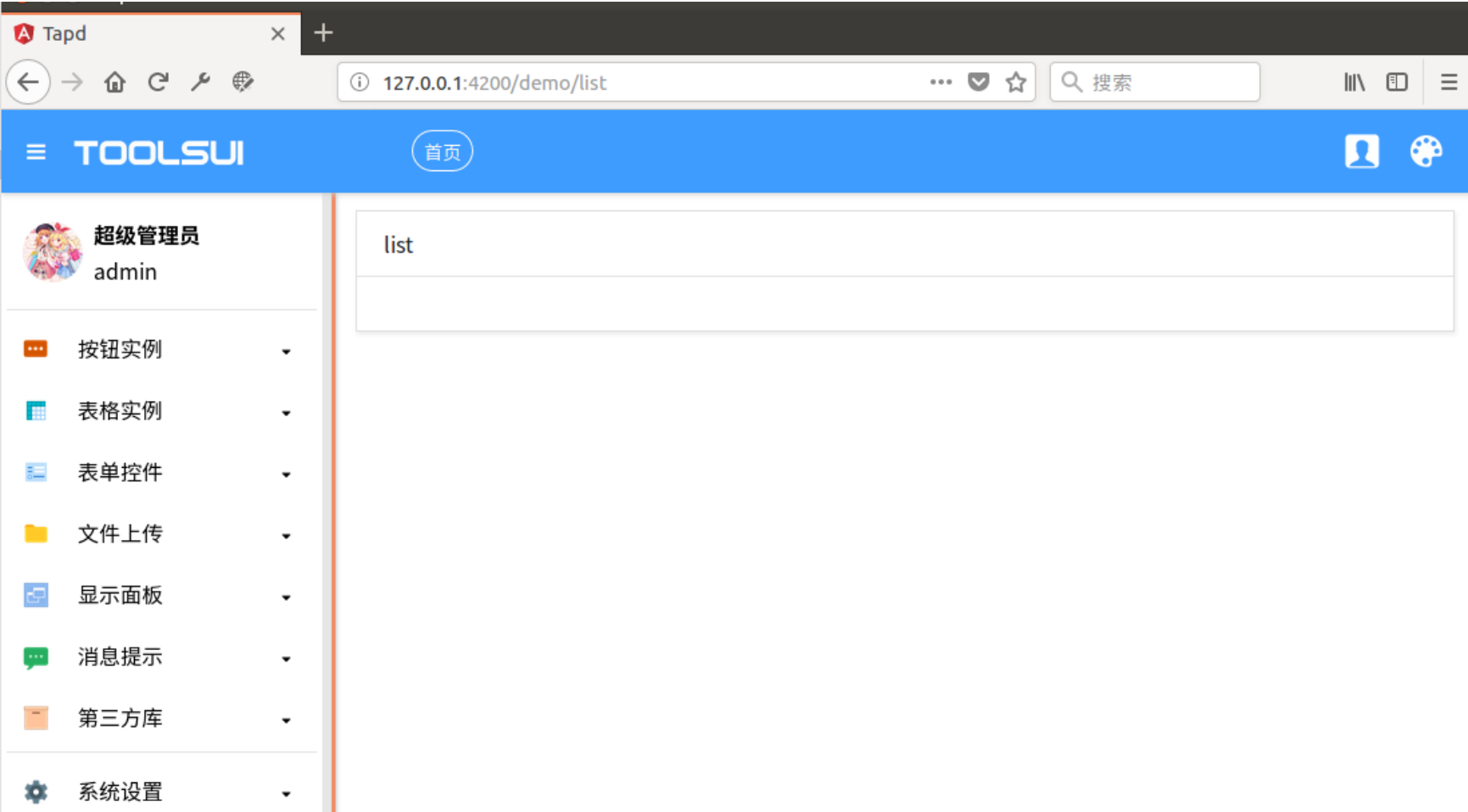
];

@NgModule({
  imports: [
    RouterModule.forRoot(routes, {
      enableTracing: false,
      // preloadingStrategy: PreloadAllModules,
      useHash: false
    })
  ],
  exports: [
    RouterModule
  ]
})
export class AppRoutingModule { }
```



新增了一条路由模块装载规则

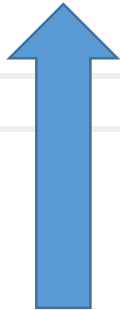
在浏览器中访问 <http://127.0.0.1:4200/demo/list>



在新创建的list页面中新增一个按钮

打开 app/modules/demo/pages/list/list.component.html 文件，看到代码如下：

```
login.component.html  TS demo.routing.ts  list.component.html x
1  <div class="card">
2      <div class="card-header bg-white">
3          list
4      </div>
5      <div class="card-body">
6
7      </div>
8  </div>
```



我们在这里插入一段模板代码，变成下一张幻灯片的代码

```
<div class="card">
  <div class="card-header bg-white">
    list
  </div>
  <div class="card-body">
    <button tsBtn color="danger">一个危险的按钮</button>
  </div>
</div>
```

你将会看到这样的效果，出现了一个红色的按钮



给按钮添加一个点击事件 click

```
<div class="card">
  <div class="card-header bg-white">
    list
  </div>
  <div class="card-body">
    <button (click)="showModal()" tsBtn color="danger">一个危险的按钮</button>
  </div>
</div>
```

对应的，我们在app/modules/demo/pages/list/list.component.ts中新增showModal方法

```
export class ListComponent implements OnInit {

  constructor() { }

  ngOnInit() {

  }

  /**
   * 显示一个模态框
   */
  showModal() {

  }

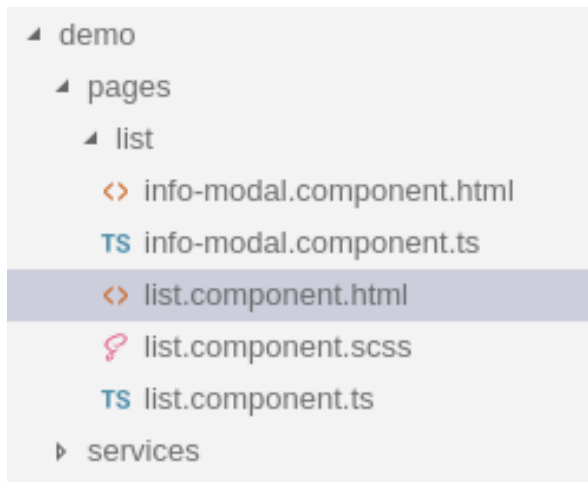
}
```

开始开发-创建一个模态框

1.使用 `yo tui:modal` 命令创建一个模态框，比如info-modal

```
xiaojian@xiaojian-virtual-machine:~/Git/test/my-app$ yo tui:modal
? 模态框属于哪个模块? demo
? 模态框属于哪个页面? list
? 要创建的模态框名称: info-modal
? 要不要单独创建一个html模板文件? Yes
  create src/app/modules/demo/pages/list/info-modal.component.ts
  create src/app/modules/demo/pages/list/info-modal.component.html
xiaojian@xiaojian-virtual-machine:~/Git/test/my-app$
```

2.如图操作后，你会发现你的list文件夹里面多了一个模态框的ts和html文件



3.在demo.routing.ts文件的声明数组和入口组件数组中加入InfoModalComponent

```
/**
 * 指令、组件、管道声明
 */
export const declarationComponents = [
    ListComponent,
    InfoModalComponent,
];

/**
 * 动态组件（模态框，窗口）
 */
export const entryComponents = [
    InfoModalComponent,
];
```

4. 要使用模态框服务，我们需要在demo.module.ts中导入ModalModule

```
/**
 * 请编写模块文件说明
 *
 * @author 填写作者
 * @file    demo.module.ts
 * @date    2018-6-26 17:17:46
 */
import { NgModule } from '@angular/core';
import { FormsModule } from '@angular/forms';
import { DemoRoutingModule, declarationComponents, entryComponents, providers } from './demo.routing';
import { ButtonModule, ModalModule } from 'ng-tools-ui';

@NgModule({
  imports: [
    FormsModule,
    ButtonModule,
    ModalModule,
    DemoRoutingModule,
  ],
  declarations: [declarationComponents],
  entryComponents: [entryComponents],
  providers: [providers]
})
export class DemoModule { }
```



5.在app/modules/demo/pages/list/list.component.ts注入ModalService

6.我们在showModal()方法中使用这个服务打开刚才创建的模态框

```
import { Component, OnInit } from '@angular/core';
import { ModalService } from 'ng-tools-ui';
import { InfoModalComponent } from './info-modal.component';

@Component({
  templateUrl: './list.component.html',
  styleUrls: ['./list.component.scss']
})
export class ListComponent implements OnInit {

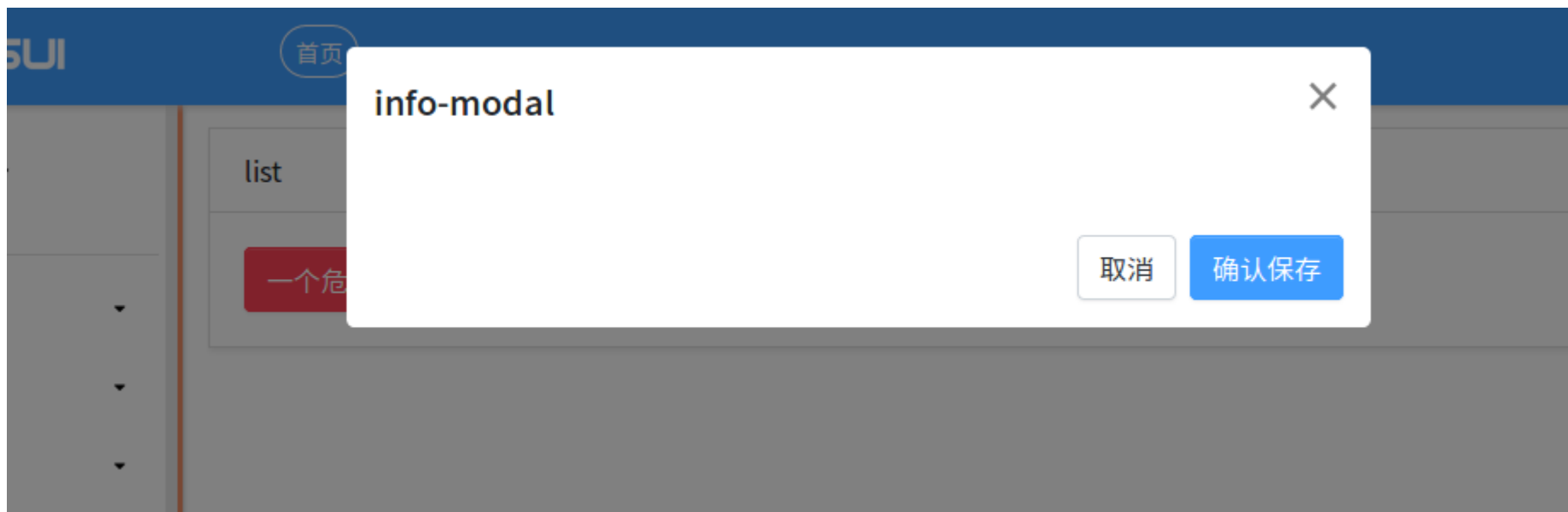
  constructor(
    private modal: ModalService
  ) { }

  ngOnInit() {

  }

  /**
   * 显示一个模态框
   */
  showModal() {
    const modal = this.modal.create(InfoModalComponent);
    modal.open();
  }
}
```

7. 点击之前的红色按钮，你可以看到弹出了一个模态框

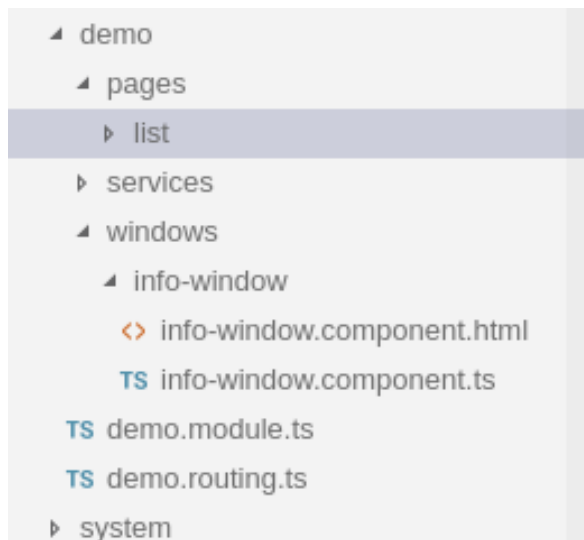


开始开发-创建一个窗口

1.使用 `yo tui:window` 命令创建一个窗口，比如info-window

```
xiaojian@xiaojian-virtual-machine:~/Git/test/my-app$ yo tui:window
? 窗口属于哪个模块? demo
? 要创建的窗口名称: info-window
? 要不要单独创建一个html模板文件? Yes
  create src/app/modules/demo/windows/info-window/info-window.component.ts
  create src/app/modules/demo/windows/info-window/info-window.component.html
xiaojian@xiaojian-virtual-machine:~/Git/test/my-app$
```

2.如图操作后，你会发现你的demo文件夹里面多了一个windows文件夹，这是用了存放窗口代码的



3.在demo.routing.ts文件的声明数组和入口组件数组中加入InfoWindowComponent

```
/**
 * 指令、组件、管道声明
 */
export const declarationComponents = [
    ListComponent,
    InfoModalComponent,
    InfoWindowComponent,
];
```

```
/**
 * 动态组件（模态框，窗口）
 */
export const entryComponents = [
    InfoModalComponent,
    InfoWindowComponent,
];
```



```
<div class="card">
  <div class="card-header bg-white">
    list
  </div>
  <div class="card-body">
    <button (click)="showModal()" tsBtn color="danger">一个危险的按钮</button>
    <button (click)="showWindow()" tsBtn color="success">一个成功的按钮</button>
  </div>
</div>
```

```
/**
 * 显示一个模态框
 */
showModal() {
  const modal = this.modal.create(InfoModalComponent);
  modal.open();
}

/**
 * 显示一个窗口
 */
showWindow() {
  const window = this.window.push(InfoWindowComponent);
  window.present();
}
```

info-window

✕ 关闭

