2023-2024学年第2学期

“混合移动开发与地图应用”创新实践项目系统开发文档

组号：1

成员：2116020016 孟天宇

2024年 6 月 18 日

**目 录**

目录

[1 系统说明 1](#_Toc166595122)

[1.1 混合移动地图应用项目的背景意义 1](#_Toc166595123)

[1.2混合移动地图应用项目的目标定位 1](#_Toc166595124)

[2 系统分析 2](#_Toc166595125)

[2.1混合移动地图应用系统基本功能 2](#_Toc166595126)

[2.2混合移动地图应用系统成果的价值 2](#_Toc166595127)

[3 系统设计与实现 3](#_Toc166595128)

[3.1模块一：TypeScript入门技术 3](#_Toc166595129)

[3.1.1 Ionic页面布局 3](#_Toc166595130)

[3.1.2 TypeScript组件 4](#_Toc166595131)

[3.1.3页面导航 5](#_Toc166595132)

[3.1.4模块化 5](#_Toc166595133)

[3.1.5路由配置 6](#_Toc166595134)

[3.1.6 Typescript基本用法部分（A1） 7](#_Toc166595135)

[3.1.7 Typescript基本基本数据类型（A2） 8](#_Toc166595136)

[3.1.8 Typescript联合与交叉训练（A3） 8](#_Toc166595137)

[3.1.9 Typescript对象的使用（A4） 9](#_Toc166595138)

[3.1.10模块一总结 9](#_Toc166595139)

[3.2模块二：Angular入门技术 10](#_Toc166595140)

[3.2.1 Angular基础知识 10](#_Toc166595141)

[3.2.2 ng插值表示用法（B1） 11](#_Toc166595142)

[3.2.3 ng单向绑定用法（B2） 12](#_Toc166595143)

[3.2.4 ng双向绑定用法（B3） 13](#_Toc166595144)

[3.2.5 ng的内置指令（B4） 14](#_Toc166595145)

[3.2.6模块二总结 15](#_Toc166595146)

[3.3模块三：Ionic 入门技术 15](#_Toc166595147)

[3.4模块四：百度地图应用技术 15](#_Toc166595148)

[4 任务分工 17](#_Toc166595149)

[5 经验总结 18](#_Toc166595150)

# 1 系统说明

## 1.1 混合移动地图应用项目的背景意义

本训练项目的指导思想是:通过指导教师的几次讲解和学生自学，让学生理解混合移动开发以及地图应用等涉及的概念和实现技术，并通过入门级的代码实现，训练学生的自学能力、创新思维能力和实际动手实践的能力。

本项目并不要求学生创新出新的算法，也不要求学生去分析底层算法，而是重在对应用模块功能实现代码的调试过程和体验，理解别人如何对底层算法进行封装并以组件或 API的形式提供给开发人员复用，从而提高应用项目的开发效率。

## 1.2混合移动地图应用项目的目标定位

本训练项目的基本目标定位是:通过项目训练和动手实践，让学生明白面对一个自己不熟悉的大项目，如何从以下几个方面去把握:

1. 了解如何尽快熟悉自己需要学习的新内容、了解这些内容可能涉及到哪些知识和技术、自己需要对这些新内容熟悉到什么程度。
2. 了解对一个相对较大的模块功能，如何从简单到复杂去逐步分析，然后将其剥离为一个个独立的小功能。
3. 了解如何利用别人做好的组件，来快速完成分配给自己的应用模块功能。

混合移动开发和地图应用本身都是很复杂的系统，本项目实现的示例功能界面看起来好像很复杂，但是要求实现的代码其实并不多。这是因为本项目并不要求学生自己去实现内部的各种底层算法，而是仅要求利用这些封装后架构或组件去实现一些入门级的功能。

学生在完成这些入门级功能的基础上，也可以在以后的工作或学习研究中，进步实现能真正用于商业用途的实用功能。

# 2 系统分析

## 2.1混合移动地图应用系统基本功能

该系统旨在实现一个基于混合移动开发的地图应用软件，具体会实现地图基本图层的呈现与查询功能。包括：地图呈现、定位、图层显示地图查询、路线规划、模拟导航等。

**地图呈现**：能通过模拟器观察指定城市的地图，比如开封市。

**地图定位：**能展示指定位置(地名或者鼠标点击处)的地图及其周边环境。

**图层显示：**能根据需要仅呈现用到的图层。

**地图查询：**能根据查询的内容在地图上呈现查询结果。能检索所当前所在位置的周边餐馆、住宿、加油站、商店等。

**路线规划：**输入起点与终点，可以通过地图和文字描述展示规划的路线，

**GPS功能**：能根据路径规划，实现地图的模拟导航。

## 2.2混合移动地图应用系统成果的价值

**(1)、外出前可以查询目标环境和大致行走的路线**

外出前可通过你开发的地图查询功能，观察旅行的目标位置及其周围环境(住宿、餐饮、加油站、商店等)。

外出前可以先通过你开发的“模拟导航”模块，了解可选的步行、骑行、驾车行走路线和大约需要多长时间。

**(2)、外出过程中可以用手机真实导航**

外出时，每个人都可以通过手机来运行你开发的“实际导航”模块，帮助自己快速到达目的地。

**(3)、能够在应用中了解景点信息**

该应用可以自定义相关的景点信息，例如标注并展示景点的图片，方便用户预览。可以框选相关地区的范围供用户查阅。

# 3 系统设计与实现

本系统设计共实现的功能分四大部分：TypeScript入门技术、Angular入门技术、Ionic 入门技术、百度地图应用技术。其中，前三部分主要练习地图应用中使用的相关基础知识，为顺利实现百度地图相关功能打下基础。

## 3.1模块一：TypeScript入门技术

本模块介绍如何在Ionic Angular应用程序中使用TypeScript编程语言。TypeScript是一种由微软开发的开源编程语言，它是JavaScript的超集，为JavaScript添加了静态类型检查和其他高级功能。在Ionic Angular项目中使用TypeScript可以带来更好的代码组织、更好的代码重用性和更可靠的类型检查。

### 3.1.1 Ionic页面布局

在Ionic Angular应用程序中，页面的布局通常由HTML模板和TypeScript组件共同完成。以下是一个示例Ionic页面布局的代码：

|  |
| --- |
| <**ion-header** [translucent]="true">    <**ion-toolbar**>      <**ion-title**>TypeScript</**ion-title**>    </**ion-toolbar**>  </**ion-header**>    <**ion-content** [fullscreen]="true" padding>    <**ion-list**>      <**ion-item** lines="none">        <**button** class="custom-button" (click)="showPage1()">Hello World</**button**>      </**ion-item**>      <**ion-item** lines="none">        <**button** class="custom-button" (click)="showPage2()">数据类型、数组、元组</**button**>      </**ion-item**>      <**ion-item** lines="none">        <**button** class="custom-button" (click)="showPage3()">字符串类型、联合、交叉</**button**>      </**ion-item**>      <**ion-item** lines="none">        <**button** class="custom-button" (click)="showPage4()">Function与箭头函数</**button**>      </**ion-item**>    </**ion-list**>  </**ion-content**>    <**style**>    /\* CSS样式省略，具体样式见原代码 \*/  </**style**> |

### 3.1.2 TypeScript组件

每个Ionic页面都有一个对应的TypeScript组件，用于控制页面逻辑和行为。以下是示例页面Tab1Page的TypeScript组件代码：

|  |
| --- |
| import { Component } from '@angular/core';  import { NavController } from '@ionic/angular';  @Component({    selector: 'app-tab1',    templateUrl: 'tab1.page.html',    styleUrls: ['tab1.page.scss']  })  export class Tab1Page {    constructor(private nav: NavController) { }      showPage1() {  **this**.nav.navigateForward('/tabs/tab1/a01');    }      showPage2() {  **this**.nav.navigateForward('/tabs/tab1/a02');    }      showPage3() {  **this**.nav.navigateForward('/tabs/tab1/a03');    }      showPage4() {  **this**.nav.navigateForward('/tabs/tab1/a04');    }  } |

### 3.1.3页面导航

Ionic应用程序包含多个页面，用户可以通过页面导航在不同页面之间进行切换。在Ionic中，可以使用NavController进行页面导航。上述TypeScript组件中的showPageX()方法用于导航到不同的页面。showPage1()方法将导航到/tabs/tab1/a01页面。同时为确保页面和组件的正确性，通常需要编写单元测试。以下是对Tab1Page组件的简单单元测试示例：

|  |
| --- |
| import { ComponentFixture, TestBed } from '@angular/core/testing';  import { IonicModule } from '@ionic/angular';  import { ExploreContainerComponentModule } from '../explore-container/explore-container.module';  import { Tab1Page } from './tab1.page';    describe('Tab1Page', () => {    let component: Tab1Page;    let fixture: ComponentFixture<Tab1Page>;      beforeEach(async () => {      await TestBed.configureTestingModule({        declarations: [Tab1Page],        imports: [IonicModule.forRoot(), ExploreContainerComponentModule]      }).compileComponents();        fixture = TestBed.createComponent(Tab1Page);      component = fixture.componentInstance;      fixture.detectChanges();    });      it('should create', () => {      expect(component).toBeTruthy();    });  }); |

### 3.1.4模块化

为了更好地组织和管理代码，Ionic应用程序通常使用模块化的方式。以下是Tab1Page组件的模块化代码示例：

|  |
| --- |
| import { NgModule } from '@angular/core';  import { IonicModule } from '@ionic/angular';  import { CommonModule } from '@angular/common';  import { FormsModule } from '@angular/forms';  import { Tab1Page } from './tab1.page';  import { ExploreContainerComponentModule } from '../explore-container/explore-container.module';  import { Tab1PageRoutingModule } from './tab1-routing.module';    @NgModule({    imports: [      IonicModule,      CommonModule,      FormsModule,      ExploreContainerComponentModule,      Tab1PageRoutingModule    ],    declarations: [Tab1Page]  })  export class Tab1PageModule {} |

### 3.1.5路由配置

最后，需要在应用程序中配置路由，以便Ionic应用程序知道如何根据用户导航操作显示不同的页面。以下是Tab1Page的路由配置示例：

|  |
| --- |
| import { NgModule } from '@angular/core';  import { RouterModule, Routes } from '@angular/router';  import { Tab1Page } from './tab1.page';    const routes: Routes = [    {      path: '',      component: Tab1Page,    },    {      path : 'a01', loadChildren: () => import('./a01/a01.module').then(m => m.A01PageModule)    },    {      path : 'a02', loadChildren: () => import('./a02/a02.module').then(m => m.A02PageModule)    },    {      path : 'a03', loadChildren: () => import('./a03/a03.module').then(m => m.A03PageModule)    },    {      path : 'a04', loadChildren: () => import('./a04/a04.module').then(m => m.A04PageModule)    }  ];    @NgModule({    imports: [RouterModule.forChild(routes)],    exports: [RouterModule]  })  export class Tab1PageRoutingModule {} |

上述代码可以将不同的组件进行导航。

### 3.1.6 Typescript基本用法部分（A1）

|  |  |  |  |
| --- | --- | --- | --- |
| 图 1基本文字显示 | 图 2 var声明变量 | 图 3var存在的问题 | 图 4 let的优点 |

### 3.1.7 Typescript基本基本数据类型（A2）

|  |  |  |
| --- | --- | --- |
| 图 5基本数据类型 | 图 6数组 | 图 7元组 |

### 3.1.8 Typescript联合与交叉训练（A3）

|  |  |  |
| --- | --- | --- |
| 图 8字符串自变量类型 | 图 9联合 | 图 10交叉 |

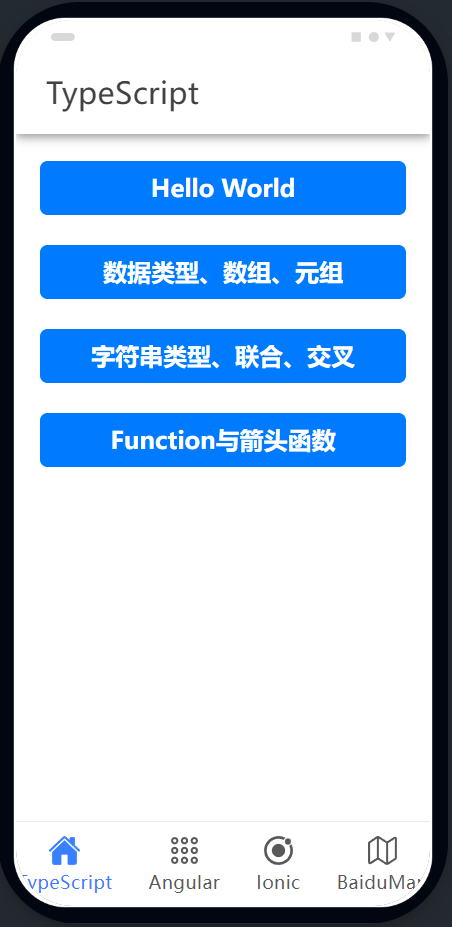
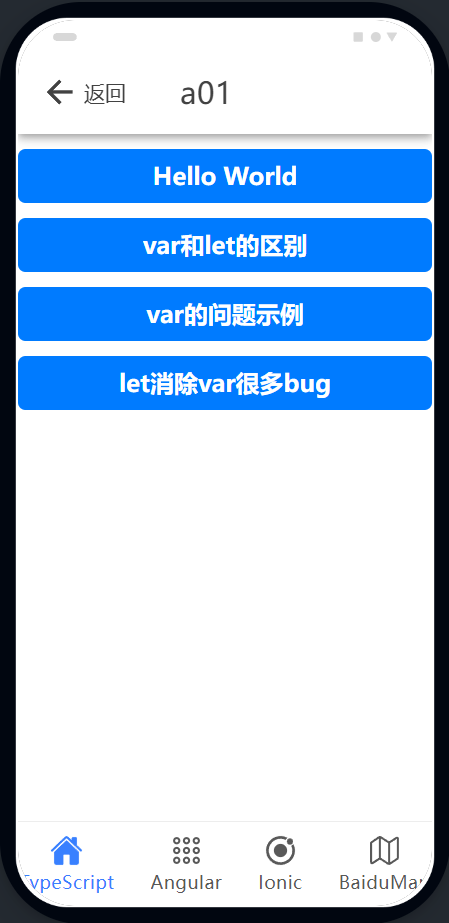
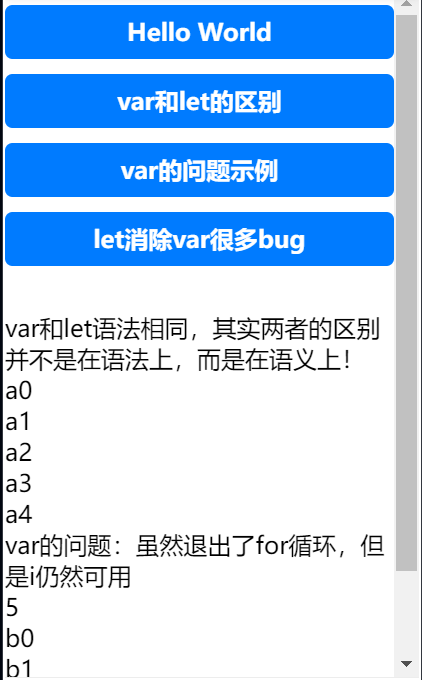
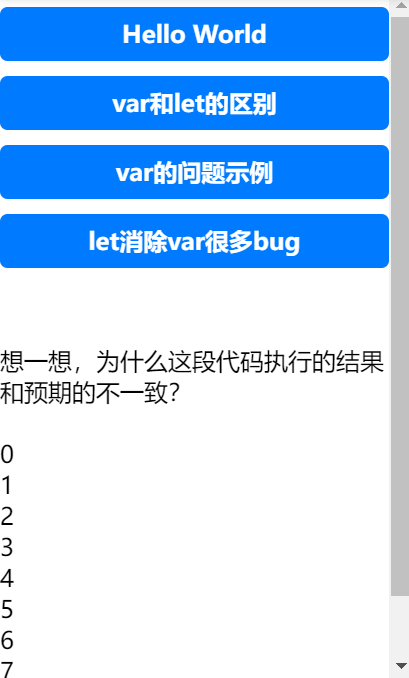
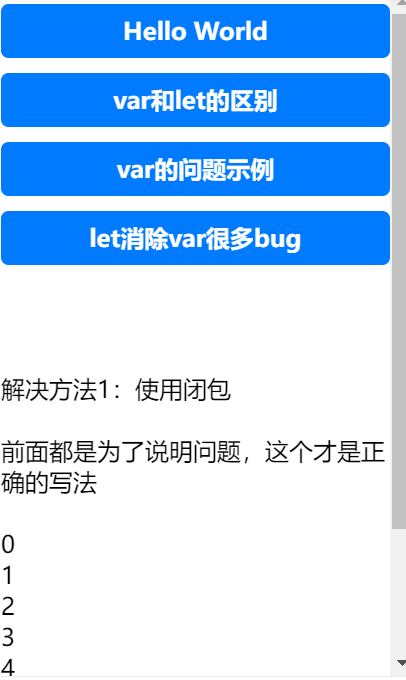
### 3.1.9 Typescript对象的使用（A4）

|  |  |
| --- | --- |
| 图 11 JavaScript的使用 | 图 12TypeScript的使用 |

### 3.1.10模块一总结

通过本模块的学习，学生了解如何在Ionic Angular应用程序中使用TypeScript编程语言，其中A1部分学会实现页面布局、页面导航、单元测试以及模块化等功能。A2部分学会声明基本的数据类型，A3部分学会了字符串字面变量类型、联合、交叉声明的基本用法。A4部分通过比较Javascript和Typescript的区别，来实现相同的功能，深入理解之间的区别联系。

这些技术和方法有助于开发高质量、可维护的Ionic应用程序。

## 3.2模块二：Angular入门技术

本模块介绍在Ionic Angular应用程序中使用Angular框架的基础知识。Angular是一个流行的前端框架，提供了丰富的功能和工具，用于构建现代化的Web应用程序。通过学习Angular基础知识，读者将能够更好地理解和利用Angular框架来开发Ionic应用程序。

### 3.2.1 Angular基础知识

本模块介绍Angular框架的基础知识，包括插值表达式、单向绑定、双向绑定和内置指令。

* **插值表达式：**通过插值表达式，可以将组件中的数据动态地显示在HTML模板中。
* **单向绑定：**单向绑定将组件中的数据绑定到HTML模板中，使得数据的变化可以自动更新到页面上。
* **双向绑定：**双向绑定不仅可以将组件中的数据绑定到HTML模板中，还可以将用户输入的数据反向绑定回组件中，实现数据的双向传递。
* **内置指令：**Angular提供了许多内置指令，用于控制HTML元素的显示和行为，例如\*ngFor用于循环，\*ngIf用于条件判断等。

通过学习这些基础知识，读者将能够更加灵活地使用Angular框架来开发复杂的应用程序。

### 3.2.2 ng插值表示用法（B1）

|  |
| --- |
| 图 13 插值表示 |

这部分实现了一个页面（b01.page.html），显示了当前时间，并且每秒钟更新一次。在页面的顶部显示当前时间的标题，下方有一个描述 Angular 框架的文本，以及一个装载时间的容器。

实现这个功能涉及到以下知识点和技术：

Ionic页面布局: 使用了Ionic提供的组件来构建页面布局，包括ion-header、ion-content等。

插值表达式: 在HTML模板中使用了插值表达式 {{current}} 来显示当前时间，并且动态更新。

CSS样式设计: 使用了CSS样式对页面元素进行美化，包括背景色、圆角边框等。

Angular组件和生命周期钩子: 实现了一个名为 B01Page 的Angular组件，并在其中实现了 ngOnInit 生命周期钩子。在 ngOnInit 方法中调用了 ionViewDidLoad 方法。

定时器和时间格式化: 在 ionViewDidLoad 方法中使用 setInterval 方法，每秒钟更新一次当前时间，并且通过调用 DateFormat 方法对时间进行格式化。

日期格式化: 实现了一个 DateFormat 方法，用于将时间戳格式化成指定格式的字符串。在其中使用了 Date 对象的方法来获取年、月、日、时、分、秒，并且根据指定的格式进行替换。

通过学习这部分内容，可以了解到在Ionic Angular应用程序中如何构建页面、使用插值表达式显示动态数据、处理时间和日期的相关操作，以及对Angular组件的生命周期钩子的使用。

### 3.2.3 ng单向绑定用法（B2）

|  |  |  |
| --- | --- | --- |
| 图 14启动定时器 | 图 15停止定时器 | 图 16显示和隐藏div |

这部分实现了一个名为 "单向绑定" 的页面（b02.page.html），包含了三个按钮和一个具有多种绑定方式的 <div> 元素。页面中有三个按钮分别用于启动定时器、停止定时器和切换 <div> 的显示和隐藏。<div> 中包含了一个图片、一个表格和一个段落，它们通过单向绑定绑定了一些属性或样式。

实现这个功能涉及到以下知识点和技术：

Ionic页面布局: 使用了Ionic提供的组件来构建页面布局，包括ion-header、ion-content等。

单向绑定: 使用了单向绑定来绑定了图片的 src 属性、表格的 colspan 属性和段落的 class 属性。这些属性在组件中发生变化时，会自动更新到HTML模板中。

事件绑定: 使用了 (click) 事件绑定来监听按钮的点击事件，并且调用组件中对应的方法。

定时器: 使用了 window.setInterval 方法来启动定时器，定时更新图片的 src 属性和段落的 class 属性。

条件判断: 在定时器中使用了条件判断语句，根据条件来切换图片和段落的属性值。

样式切换: 使用了 hidden 属性来控制 <div> 元素的显示和隐藏。

通过学习这部分内容，可以了解到在Ionic Angular应用程序中如何使用单向绑定来实现页面元素的动态更新，以及如何通过事件绑定来监听用户操作并做出响应。同时也学习了如何使用定时器来实现周期性的任务，以及如何通过条件判断和样式切换来控制页面元素的状态。

### 3.2.4 ng双向绑定用法（B3）

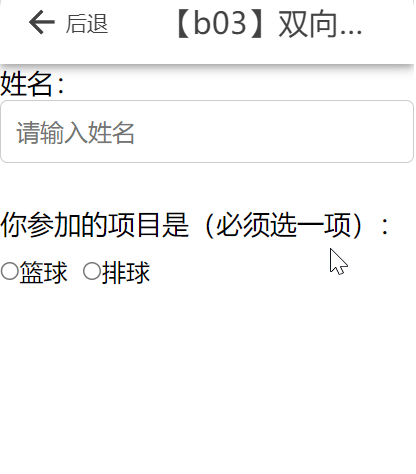


图 17双向绑定用法

这部分实现了一个名为 "双向绑定" 的页面（b03.page.html），其中包含了一个输入框和两个单选按钮，用于展示双向绑定的功能。页面中的输入框用于输入姓名，而单选按钮用于选择参加的项目（篮球或排球）。同时，页面下方展示了输入的姓名和选择的项目信息。

实现这个功能涉及到以下知识点和技术：

双向绑定: 使用了Angular中的双向绑定 [(ngModel)] 来实现数据的双向绑定。在输入框和单选按钮上分别应用了双向绑定，这意味着数据的变化会同步更新到视图上，同时用户在视图上的操作也会反映到数据模型中。

ngModelChange事件: 使用了 (ngModelChange) 事件来监听数据模型的变化，当输入框或单选按钮的值发生改变时，调用 showResult() 方法更新结果。

动态展示数据: 使用了插值表达式 {{result}} 和 [innerHTML] 属性来动态展示结果。当姓名或项目发生变化时，会动态更新显示的结果。

HTMLDivElement: 使用了 document.getElementById 方法来获取 HTML 元素，并且将其赋值给了类成员变量 myDiv，以便后续在组件中操作该元素。

样式设计: 使用了CSS样式来美化页面元素，包括输入框、单选按钮和文字标签等。

通过学习这部分内容，可以了解到在Angular应用中如何使用双向绑定来实现数据的同步更新，以及如何使用事件绑定来监听数据模型的变化。同时也学习了如何在视图中动态展示数据，并且通过操作HTML元素来实现更丰富的交互功能。

### 3.2.5 ng的内置指令（B4）



图 18内置指令

这部分实现了一个名为 "内置指令" 的页面（b04.page.html），展示了Angular中内置指令 ngFor 和 ngIf 的基本用法。页面中使用了这两个指令来展示学生列表和学生的年龄、成绩信息。

实现这个功能涉及到以下知识点和技术：

ngFor指令: 使用了 \*ngFor 指令来遍历 student 数组中的所有元素，并将每个学生的姓名展示在页面上。这样可以动态生成多个相同结构的HTML元素，使得显示列表更加灵活和动态化。

ngIf指令: 使用了 \*ngIf 指令来根据条件动态地添加或移除HTML元素。在页面中，根据 age 和 grade 是否存在，决定是否显示学生的年龄和成绩信息。这样可以根据不同的条件动态地显示不同的内容，增强了页面的交互性和可定制性。

样式设计: 使用了CSS样式来美化页面元素，包括标题、学生列表、学生信息容器等。

HTMLDivElement: 使用了 document.getElementById 方法来获取 HTML 元素，并且将其赋值给了类成员变量 myDiv，以便后续在组件中操作该元素。

动态数据: 在组件中定义了一个 student 数组，用于存储学生姓名；定义了 age 和 grade 属性，用于存储学生的年龄和成绩。这些数据会动态地反映在页面上，根据数组的内容和属性的值来动态渲染页面。

通过学习这部分内容，可以了解到Angular中内置指令的基本用法，包括如何使用 ngFor 来循环遍历数据并动态生成页面内容，以及如何使用 ngIf 来根据条件动态显示或隐藏HTML元素。同时也学习了如何在组件中操作HTML元素，并且如何通过数据绑定来实现动态数据的展示。

### 3.2.6模块二总结

展示了如何使用Angular中的组件、模板和样式来构建页面。

使用了Ionic中的导航组件进行页面间的导航。

展示了如何使用Angular中的事件绑定、属性绑定和双向绑定来实现交互功能。

通过Angular的ngFor指令和ngIf指令展示了数据的循环渲染和条件渲染。

展示了双向绑定在Angular应用中的基本用法和实现方式。

通过在输入框和单选按钮上应用双向绑定，实现了数据的动态绑定和同步更新。

使用了ngModelChange事件来监听数据模型的变化，并根据变化动态更新页面内容。

展示了Angular中内置指令ngFor和ngIf的基本用法。

使用ngFor指令动态生成学生列表，并展示了如何遍历数组中的元素。

使用ngIf指令根据条件动态显示或隐藏HTML元素，增强了页面的交互性和可定制性。

## 3.3模块三：Ionic 入门技术

## 3.4模块四：百度地图应用技术

描述本项目一共包含几个功能模块，每个功能模块包含什么功能。

【参考实践指导书P55页，3.4 系统功能模块的设计。】

从安装和配置开发环境、主菜单设计、主要功能代码展示（每个模块可以选1~2个页面展示）和运行效果等方面描述。

备注：在进行主要功能代码展示前，先给出整个系统在VSCode中的截图。运行效果部分可以多展示地图应用部分的功能。

# 4 任务分工

小组成员间的分工与合作情况描述。

备注：组长如何协调完成该项目的、每位同学都干了什么工作必须描述清楚。

# 5 经验总结

针对项目开发过程中的遇到的问题以及解决办法方面的心得体会。

比如环境配置、百度地图API调用方面、发现问题时调试方面、所用框架更新带来的一些新问题的解决方面。