第10章 ionic JavaScript(上)

在本书第 1 章 ionic 简介中提到过 ionic 框架的组成部分，分为 ionic CSS、ionic JavaScript 和 ionic CLI（命令行）。前文中已经讲解了 ionic CLI 和 ionic CSS，从本章开始讲解 ionic JavaScript 部分的内容。

ionic JavaScript 中提供了许多组件，按照功能分类可大致分为 5 类，包括基本布局组件、ionic 路由、界面组件、动态组件和手势事件，本章将针对 ionic JavaScript 中的基本布局组件、ionic 路由和界面组件进行详细的讲解。

【教学导航】

学习目标	1. 了解 ionic JavaScript 中指令式组件和服务式组件的使用 2. 掌握 ionic JavaScript 的基本布局 3. 掌握 ionic JavaScript 导航组件的使用 4. 掌握 ionic JavaScript 界面组件的使用
教学方式	本章内容以理论讲解、案例和过程演示为主。
重点知识	1. ionic JavaScript 的基本布局 2. ionic JavaScript 的导航组件的使用 3. ionic JavaScript 的界面组件的使用
关键词	滚动条、滚动刷新、下拉刷新、路由、列表、幻灯片、侧边栏菜单、选项卡

10.1 ionic JavaScript 概述

ionic 对 AngularJS 进行了扩展，将移动端开发中常见的 UI 组件抽象成 AngularJS 的指令与服务，便于我们在开发中快速构建移动 App 界面。ionic JavaScript 是对 AngularJS 的扩展，其内置的 JavaScript 组件与 AngularJS 组件类似，按照使用方式可以将 ionic JavaScript 组件分为两大类，指令式组件和服务式组件。

10.1.1 ionic 指令式组件

ionic JavaScript 的指令式组件通常以元素、属性或者 CSS 类的形式在 HTML 文件使用。

1.元素形式

以元素形式使用的指令都带有“ion-”前缀，例如，使用 ion-tabs 指令实现一个功能完备的选项卡，示例代码如下所示。

```
<ion-tabs>
    <ion-tab title="本周热卖">...</ion-tab>
    <ion-tab title="销量最高">...</ion-tab>
    <ion-tab title="评分最高">...</ion-tab>
```

```
</ion-tabs>
```

ionic 中以元素形式使用的指令覆盖了移动端大部分的开发需求，包含基本布局、视图路由、列表、表单输入、选项卡、侧边栏、幻灯片等。

2.属性形式

ionic 中，以属性形式使用的指令没有固定前缀，而是使用多个单词来描述组件功能的，多个单词之间使用“-”符号连接，例如 nav-direction（导航描述）。

ionic 的手势事件功能也是通过属性形式使用的，例如长按事件 on-hold，读者可以使用“<any on-hold=“…”>…</any>”的方式在任何元素上使用这个指令挂载监听函数。

3.CSS 类形式

目前官方以 CSS 类形式使用的指令只有 1 个，即 hide-on-keyboard-open（键盘打开时隐藏元素），使用方式是“<any class=“hide-on-keyboard-open”>…</any>”。

10.1.2 ionic 服务式组件

ionic 的服务式组件通常带有“$ionic”前缀，例如$ionicLoading，ionic 服务式组件本质上是 AngularJS 服务对象，可以在 AngularJS 代码中以依赖注入的方式被应用，用于直接创建页面视图组件或执行与页面视图组件交互的任务。

ionic 服务式组件中包含几个常用的动态组件，如下所示。

- 模态对话框：$ionicModal
- 上拉菜单：$ionicActionSheet
- 弹出框：$ionicPopup
- 浮动框：$ionicPopover
- 载入指示器：$ionicLoading
- 背景幕：$ionicBackdrop

如果 ionic 服务组件名称带有后缀“delegate”，那么它的类型为代理类服务组件，例如$ionicTabsDelegate，代理类服务组件在使用上与普通服务组件有所差别，这类组件通常含有$getByHandle（delegateHandle）方法，该方法可以用来获取页面上对应指令式组件的操作对象，继而使用代码控制这些组件外观和行为的目的。

10.2 基本布局组件

ionic 的基本布局组件用于搭建移动 App 基本的单页面框架，包括固定标题栏、内容展示、内容滚动和下拉刷新等功能。

10.2.1 固定标题栏

在移动 App 界面中，固定标题栏经常位于页面的头部区域或底部区域，ionic JavaScript 提供了两个指令 ion-header-bar 和 ion-footer-bar，分别用于声明头部固定标题栏和底部固定标题栏。

1.ion-header-bar

在 ionic 中，ion-header-bar 指令用于定义固定在屏幕顶部的 header 标题栏，基本格式如

下所示。

```
<ion-header-bar>...</ion-header-bar>
```

如果在<ion-header-bar>标签上添加.bar-subheader 样式，便可以作为头部标签栏下面的次级顶栏，示例代码如下所示。

```
<ion-header-bar  class="bar-subheader">次级顶栏</ion-header-bar>
```

ion-header-bar 指令有两个可选的属性，如表 10-1 所示。

表10-1　ion-header-bar 属性

属性	取值类型	描述
align-title	字符串	用于设置标题的对齐方式，属性值：left | right | center，分别对应左对齐、右对齐和居中对齐。
no-tap-scroll	布尔值	当点击标题时，是否将内容区域自动滚动到最起始位置，属性值：true |false，默认为 true。

2.ion-footer-bar

ion-footer-bar 指令用于定义固定在屏幕底部的 footer 标题栏，基本格式如下所示。

```
<ion-footer-bar>...</ion-footer-bar>
```

如果在<ion-footer-bar>标签上添加.bar-subfooter 样式，便可以作为底部标签栏上面的次级底栏，示例代码如下所示。

```
<ion-footer-bar  class="bar-subfooter">次级顶栏</ion-footer-bar>
```

ion-footer-bar 指令也支持表 10-1 中的 align-title 属性。

接下来通过一个案例来演示 ion-header-bar 和 ion-footer-bar 指令的使用，具体步骤如下：

（1）创建 chapter10 目录，将 chapter09 目录下的 lib 文件夹拷贝到该目录下。

（2）在 chapter10 目录下创建 js 目录，在该目录下创建 app.js 文件，并且在 app.js 文件中添加如下代码。

```
//定义 starter 模块，并注入 ionic 模块
angular.module("starter", ["ionic"]);
```

上述代码用于定义页面入口模块，ionic 页面与 AngularJS 相同，需要使用 ng-app 指定根元素才能运行 AngularJS 代码。如果是使用命令行方式下载的 ionic 项目模板，项目模板中已经包含上述功能代码，读者可直接使用项目模板中的代码。

（3）在 chapter10 目录下创建 demo10-1.html，在 demo10-1.html 中添加如下代码。

```
1  <!DOCTYPE html>
2  <html>
3    <head>
4      <meta charset="utf-8">
5      <meta   name="viewport"   content="initial-scale=1,   maximum-scale=1,
   user-scalable=no, width=device-width">
6      <title>头部和底部</title>
7        <link href="lib/ionic/css/ionic.css" rel="stylesheet">
8        <script src="lib/ionic/js/ionic.bundle.js"></script>
9        <script src="js/app.js"></script>
10   </head>
11   <body ng-app="starter">
12       <ion-header-bar align-title="left" class="bar-positive">
13           <button class="button">消息</button>
14           <h4 class="title">这里是顶部</h4>
```

```
15          <button class="button">签到</button>
16      </ion-header-bar>
17          <ion-header-bar class="bar-subheader bar-energized">
18              <h4>次级顶栏</h4>
19          </ion-header-bar>
20           <!--添加内容的位置-->
21          <ion-footer-bar class="bar-subfooter  bar-energized">
22              <h4>次级底栏</h4>
23          </ion-footer-bar>
24      <ion-footer-bar align-title="right" class="bar-positive">
25              <button class="button">推荐</button>
26              <h4 class="title">这里是底部</h4>
27              <button class="button">设置</button>
28      </ion-footer-bar>
29   </body>
30 </html>
```

上述代码中，将页面分为四个区域，包括头部、底部、次级顶栏和次级底栏，并且使用 align-title 为头部和底部设置了不同的标题位置。

第 7~9 行引入了 3 个依赖文件，其中 ionic.css 和 ionic.bundle.js 文件是使用 ionic JavaScript 功能需要引入的基本文件，app.js 文件的作用是使 ionic 组件中的 AngularJS 代码生效。第 11 行使用 ng-app 指令绑定 body 元素，这样可以保证 body 元素的子元素能够使用 ionic JavaScript 组件。

使用 Chrome 浏览器访问 demo10-1.html，页面效果如图 10-1 所示。

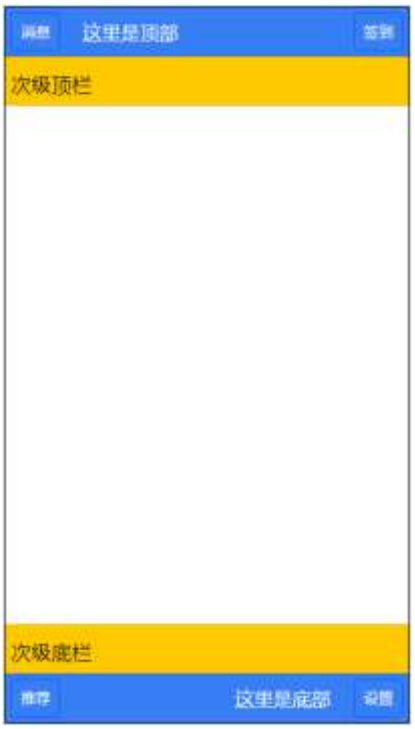

图 10-1　demo10-1.html 页面效果

10.2.2 内容区域

ionic 使用 ion-content 指令来声明内容展示容器，基本格式如下所示。

```
<ion-content>...</ion-content>
```

ion-content 提供了一些可选的属性和事件，其中常用属性如表 10-2 所示。

表10-2 ion-content API

类型	名称	取值类型	描述
属性	delegate-handle	字符串	用于标识带有$ionicScrollDelegate 的滚动视图。
	padding	布尔值	用于设置是否在内容上添加内边距。iOS 默认是 true，Android 是 false。
	scroll	布尔值	用于设置是否允许内容滚动，默认值为 true。
	overflow-scroll	布尔值	用于设置是否使用浏览器本身内置的溢出滚动功能代替 ionic 滚动。默认值为 false。
	has-bouncing	布尔值	用于设置是否允许内容滚动反弹到边缘。iOS 默认是 true，Android 默认为 false。
事件	on-scroll	表达式	当内容滚动时触发该事件，对表达式求值。
	on-scroll-complete	表达式	一个滚动动作完成时触发该事件，触发事件时可以通过传入 scrollLeft 和 scrollTop 两个变量来获取当前滚动位置。

在表 10-2 中，最常用的两个属性是 on-scroll 和 on-scroll-complete，接下来通过一个案例来演示这两个属性的用法，如 demo10-2.html 所示。

demo10-2.html

```
1  <!DOCTYPE html>
2  <html>
3    <head>
4      <meta charset="utf-8">
5      <meta   name="viewport"   content="initial-scale=1,   maximum-scale=1,
   user-scalable=no, width=device-width">
6      <title>内容区域:ion-content</title>
7        <link href="lib/ionic/css/ionic.css" rel="stylesheet">
8        <script src="lib/ionic/js/ionic.bundle.js"></script>
9    </head>
10   <body ng-app="starter">
11       <ion-header-bar  class="bar-positive">
12           <h4 class="title">这里是顶部</h4>
13       </ion-header-bar>
14       <ion-content ng-controller="myCtrl"  on-scroll="onScroll()"
              on-scroll-complete="onScrollComplete(scrollLeft,scrollTop)">
15             <h1>美丽的海岛</h1>
16             <img src="img/demo10-2/海岛.jpg">
17       </ion-content>
18       <ion-footer-bar class="bar-positive">
19             <h4 class="title">这里是底部</h4>
```

```
20      </ion-footer-bar>
21   </body>
22 <script>
23 angular.module("starter",["ionic"]).controller("myCtrl",function($scope){
24         $scope.onScroll=function(){
25             console.log("正在滚动");
26         }
27         $scope.onScrollComplete=function(scrollLeft,scrollTop){
28             console.log("当前滚动位置"+scrollLeft+","+scrollTop);
29         }
30    });
31 </script>
32 </html>
```

上述代码中，使用内嵌的 AngularJS 代码定义了控制器 myCtrl，并在该控制器中定义了两个函数 onScroll()和 onScrollComplete()，分别用于绑定 on-scroll 和 on-scroll-complete 事件。滚动页面时，将触发 on-scroll 事件，调用 onScroll()函数，在控制台输出“正在滚动”，滚动结束时，将触发 on-scroll-complete 事件，调用 onScrollCompete()函数，在控制台输出当前的滚动位置。

使用 Chrome 浏览器访问 demo10-2.html，页面效果如图 10-2 所示。

图 10-2　demo10-2.html 页面效果

在图 10-2 的页面中，内容区域包含一个标题和一张图片，由于图片大于浏览器窗口所以没有展示完全，这样做的目的是方便测试 ionic 的滚动事件。使用鼠标滚轮将页面向上滚动，然后单击屏幕（通知浏览器滚动结束），这时查看控制台输出效果，如图 10-3 所示。

图 10-3 控制台输出结果

从图 10-3 的输出结果可以看出，当页面开始滚动时，控制台输出了“正在滚动”，当滚动结束时，控制台输出了当前滚动位置，需要注意的是，滚动结束时，控制台输出当前位置后还会再调用一次 onScroll()函数，输出“正在滚动”。

10.2.3 滚动条

ion-scroll 指令用来创建一个包含所有内容的滚动容器，支持同一页面放置多个滚动容器和控制缩放等功能，也可以嵌入到 ion-content 指令中只让页面实现部分滚动，基本格式如下所示。

```
<ion-scroll>...</ion-scroll>
```

ion-scroll 中提供了一些可选的属性和事件，如表 10-3 所示。

表10-3 ion-scroll API

类型	名称	取值类型	描述
属性	delegate-handle	字符串	用于标识带有$ionicScrollDelegate 的滚动视图。
	direction	字符串	用于设置滚动的方向，取值为 'x' 或 'y'，默认 'y' ，其中'x'代表水平方向坐标值，'y'代表垂直方向坐标值。
	paging	布尔值	用于设置滚动是否限制分页。
	scrollbar-x	布尔值	是否显示水平滚动条，默认为 true。
	scrollbar-y	布尔值	是否显示垂直滚动条，默认为 true。
	zooming	布尔值	是否支持双指缩放。
	min-zoom	数字	允许的最小缩放量（默认为 0.5）。
	max-zoom	数字	允许的最大缩放量（默认为 3）。
事件	on-refresh	表达式	下拉刷新时触发事件，由 ionRefresher 触发，与 ion-refresher 配合使用。
	on-scroll	表达式	当内容滚动时触发该事件，对表达式求值。

ion-scroll 指令组件可以指定组件高度和内部内容元素的高度，接下来通过一个案例来演示 ion-scroll 的具体用法，如 demo10-3.html 所示。

demo10-3.html

```
1  <!DOCTYPE html>
2  <html>
3    <head>
4      <meta charset="utf-8">
5      <meta   name="viewport"   content="initial-scale=1,   maximum-scale=1,
   user-scalable=no, width=device-width">
6      <title>滚动条:ion-scroll</title>
```

```
7       <link href="lib/ionic/css/ionic.css" rel="stylesheet">
8       <script src="lib/ionic/js/ionic.bundle.js"></script>
9    </head>
10   <body ng-app="starter">
11      <ion-header-bar  class="bar-positive">
12         <h4>这里是头部</h4>
13      </ion-header-bar>
14         <ion-content>
15             <div>
16                <h4 class="title">中国地图展示</h4>
17                <p style="width: 300px; height: 150px">
18                   中国陆地面积约 960 万平方公里,在世界各国中,
19                   仅次于俄罗斯、加拿大,居第三位，差不多同整个欧洲面积相等。
20                </p>
21             </div>
22             <ion-scroll direction="xy" scrollbar-y ="true"
                          style="width: 500px; height: 300px">
23              <div style="width: 1005px; height: 568px;
                     background: url('img/demo10-3/map.jpg') no-repeat">
                 </div>
24             </ion-scroll>
25         </ion-content>
26      <ion-footer-bar class="bar-positive">
27             <h4>这里是底部</h4>
28      </ion-footer-bar>
29   </body>
30   <script>angular.module("starter", ["ionic"]);</script>
31 </html>
```

上述代码中，在 ion-content 中嵌入了一个 div 作为描述区域，该区域固定显示在屏幕上半部分，然后在 ion-conten 中嵌入了 ion-scroll 滚动条，在该滚动条中添加了一个带有背景的 div，背景为中国地图，设置滚动条的宽高时，要小于背景图的宽高，并使用 direction=“xy”设置滚动条可以水平和垂直方向滚动。使用 Chrome 浏览器访问 demo10-3.html，页面效果如图 10-4 所示。

图 10-4　demo10-3.html 页面效果

在图 10-4 中，按住鼠标左键上下左右拖动，可以查看地图的不同位置，如图 10-5 所示。

图 10-5　滚动效果

10.2.4 滚动刷新

滚动刷新在移动 App 中的应用十分广泛，例如实现商品列表时，由于移动 App 页面大小有限，不能一次性将所有商品全部展示，这时，如果想浏览更多商品，可以通过滚动刷新

的方式来加载数据，当没有更多商品时，对用户做出提示没有更多商品。

ionic 中提供了 ion-infinite-scroll 指令用于滚动刷新功能，该功能适用于瀑布流式（无限数据查询）页面，示例代码如下所示。

```
<ion-infinite-scroll on-infinite="loadMore()" distance="1%">
    ...
</ion-infinite-scroll>
```

使用 ion-infinite-scroll 指令时，当容器滚动到或接近页面底部会触发获取数据的事件 on-infinite，事件处理函数完成新内容数据的加载后，需要调用 scroll.infiniteScrollComplete 事件广播来通知页面更新滚动视图，该事件的功能类似于在 AngularJS 中$scope.$apply()脏数据检查，作用是通知页面中所有组件数据已经加载完成，可以更新到页面。

ion-infinite-scroll 指令组件中提供的属性和事件如表 10-4 所示。

表10-4 ion-infinite-scroll API

	名称	取值类型	描述
属性	distance	字符串	可选，从底部滚动到触发 on-infinite 表达式的距离，默认：1%。
	icon	字符串	可选，当加载时显示的图标。默认值: 'ion-loading-d'，该属性已不推荐使用，建议用 spinner 属性代替。
	spinner	字符串	可选，加载时显示轮转等待指示框。
事件	on-infinite	表达式	必选，当滚动到底部时触发的事件。

接下来通过一个案例来演示 ion-infinite-scroll 的使用方法，如 demo10-4.html 所示。

demo10-4.html

```
1  <!DOCTYPE html>
2  <html>
3    <head>
4      <meta charset="utf-8">
5      <meta  name="viewport"  content="initial-scale=1,  maximum-scale=1,
   user-scalable=no, width=device-width">
6      <title>滚动刷新:ion-infinite-scroll</title>
7      <link href="lib/ionic/css/ionic.css" rel="stylesheet">
8      <script src="lib/ionic/js/ionic.bundle.min.js"></script>
9    </head>
10   <body ng-app="starter">
11     <ion-header-bar  class="bar-positive">
12         <h4>这里是头部</h4>
13     </ion-header-bar>
14     <ion-content ng-controller="myCtrl">
15         <ul class="list">
16         <li class="item" ng-repeat="item in items">{{item}}</li>
17          <li class="item" ng-if="!moreDataCanBeLoaded()">没有更多数据喽</li>
18         </ul>
19         <ion-infinite-scroll ng-if="moreDataCanBeLoaded()"
20                              on-infinite="loadMore()" distance="1%">
21         </ion-infinite-scroll>
22     </ion-content>
23     <ion-footer-bar class="bar-positive">
```

```
24              <h4>这里是底部</h4>
25      </ion-footer-bar>
26  </body>
27  <script>
28      //定义控制器
29     angular.module("starter",["ionic"])
                    .controller("myCtrl",function($scope,$timeout){
30          //初始化加载更多数据的次数
31          $scope.loadTimes=1;
32          //定义判断是否可以加载更多数据的函数
33          $scope.moreDataCanBeLoaded=function(){
34              return $scope.loadTimes>0;
35          }
36          //初始化数据
37          $scope.items=[];
38          for(var i=1;i<=20;i++){
39              $scope.items.push("第"+i+"条数据");
40          }
41          //页面滚动到底部需要调用的方法
42          $scope.loadMore = function() {
43              //定义定时器，延时加载可以看到加载图标效果
44              $timeout(function () {
45                  $scope.loadTimes=$scope.loadTimes-1;
46                  //加载完毕通知容器更新滚动视图（收起图标）
47                  $scope.$broadcast('scroll.infiniteScrollComplete');
48              }, 1000);
49          }
50      });
51  </script>
52 </html>
```

上述代码中，使用模拟延时加载数据的效果，来演示 ion-infinite-scroll 组件的使用方法，ion-infinite-scroll 组件必须嵌套在 ion-content 中。

第 31 行定义的 loadTimes 属性代表加载次数，第 37~40 行用于定义初始化数据。本案例定义了 20 条数据，共加载两次，每次加载 10 条数据。

第 42 行定义 loadMore()函数，在第 19 行为该函数绑定 on-infinite 事件，当页面滚动到底部时，调用 loadMore()函数，在 loadMore()函数中添加定时器是为了实现延时加载，看到加载图标的效果。

使用 Chrome 浏览器访问 demo10-4.html，页面默认加载了 10 条数据如图 10-6 所示。

图 10-6　demo10-4.html 页面效果

在图 10-6 的页面向下滑动鼠标，可以看到加载时显示图标效果，如图 10-7 所示。

图 10-7　加载图标

加载完毕后，页面底部会提示没有更多数据，该效果为自定义设置，页面效果如图 10-8 所示。

图 10-8　加载完毕后隐藏图标

10.2.5 下拉刷新

下拉刷新功能即用户通过向下拉动页面，实现重新加载、刷新页面的内容，该功能适用于新闻类页面，例如今日头条等。

ion-refresher 指令可以实现内容区域的下拉刷新功能，该指令可以作为 ion-content 或者 ion-scroll 的第一个子元素，示例代码如下所示。

```
<ion-content>
  <ion-refresher pulling-text="下拉刷新..." on-refresh="doRefresh()">
  </ion-refresher>
</ion-content>
```

上述代码中，on-refresh 的事件处理函数 doRefresh()完成内容数据的加载后，需要调用 scroll.refreshComplete 事件，通知包含 ion-refresher 的内容容器更新滚动视图。

ion-refresher 指令组件中提供了一些可选的属性和事件，如表 10-5 所示。

表10-5　ion-refresher API

类型	名称	取值类型	描述
属性	pulling-icon	字符串	当用户向下拉动数据时显示的图标。默认值：'ion-arrow-down-c'。
	pulling-text	字符串	当用户向下拉动数据时显示的文字。
	refreshing-icon	字符串	当用户向下拉动数据松开后显示的图标，该属性已不推荐使用，建议用 spinner 属性代替。
	spinner	字符串	加载时显示轮转等待指示框。
事件	on-refresh	表达式	当用户向下拉动数据到一定程度，松开后开始刷新时触发。
	on-pulling	表达式	当用户开始向下拉动内容区域时触发。

为了读者更好的理解，接下来通过一个案例来演示 ion-refresher 指令组件的具体使用方法，如 demo10-5.html 所示。

demo10-5.html

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8">
    <meta   name="viewport"   content="initial-scale=1,   maximum-scale=1, user-scalable=no, width=device-width">
    <title>下拉刷新:ion-refresh</title>
      <link href="lib/ionic/css/ionic.css" rel="stylesheet">
      <script src="lib/ionic/js/ionic.bundle.min.js"></script>
  </head>
  <body ng-app="starter">
     <ion-header-bar  class="bar-positive">
        <h4>下拉刷新功能</h4>
     </ion-header-bar>
     <ion-content ng-controller="myCtrl">
           <!-- 添加下拉刷新的组件 -->
           <ion-refresher  pulling-text="正在刷新页面数据..."
                  on-refresh="doRefresh()">
           </ion-refresher>
        <ul class="list">
           <li class="item" ng-repeat="item in items">{{item}}</li>
        </ul>
        <ion-content>
  </body>
  <script>
     //定义控制器
angular.module("starter",["ionic"])
               .controller("myCtrl",function($scope,$timeout){
        //初始化数据
        $scope.items=[];
        for(var i=1;i<=10;i++){
           $scope.items.push("首页第"+i+"条数据");
        }
      // on-refresh 事件调用函数
        $scope.doRefresh=function () {
           $timeout(function () {
           //下来载入数据
           $scope.items=[];
           for(var i=1;i<=10;i++){
              $scope.items.push("下拉载入第"+i+"条数据");
           }
           // 停止广播 ion-refresher
```

```
41              $scope.$broadcast('scroll.refreshComplete');
42              }, 1000);
43          }
44      });
45   </script>
46 </html>
```

上述代码中，将下拉组件 ion-refresher 作为第一个子元素嵌入到 ion-content 组件中，使用 pulling-text 设置向下拉动页面要显示的文字，在没有自定义图标的情况下，会显示默认的图标效果。使用 on-refresh 事件绑定 doRefresh()函数，在该函数中实现重新载入数据的效果。需要注意的是，doRefresh()函数中添加定时器，是为了看到页面刷新前显示的图标，默认与向下拉动的图标样式不同。

使用 Chrome 浏览器访问 demo10-5.html，页面默认加载了 10 条初始数据，如图 10-9 所示。

图 10-9　demo10-5.html 页面效果

在图 10-9 中，使用鼠标轻轻向下拉动页面，可以看到提示文字和箭头朝下的图标，如图 10-10 所示。

图 10-10 拉动效果

看到箭头朝下的图标后，再向下拉会看到图标箭头变为朝上，这时松开鼠标，会看到加载数据的图标，加载完成后，会显示下拉加载的数据效果，如图 10-11~10-13 所示。

图 10-11 图标箭头朝上

图 10-12 显示加载图标

图 10-13 下拉载入数据

10.2.6 手动控制滚动视图

使用 ion-content 或 ion-scroll 指令时，如果容器内部的内容是可动态更新的，那么在更新内容后，需要调用$ionicScrollDelegate 服务代理的 resize()方法重新计算滚动容器的大小并更新滚动视图。通过$ionicScrollDelegate 服务代理还可以手动控制滚动视图的滚动位置。

作为 ion-content 或 ion-scroll 指令的代理服务，$ionicScrollDelegate 提供了很多方法用来手动控制滚动视图，常用方法如表 10-6 所示。

表10-6 $ionicScrollDelegate 的常用方法

方法	取值类型	描述

resize()	无	通知滚动视图由于内容更新，当前滚动容器的大小需要重新计算。
scrollTop ([shouldAnimate])	布尔值	滚动到内容顶部，参数 shouldAnimate 表示是否应用滚动动画。
scrollBottom ([shouldAnimate])	布尔值	滚动到内容底部，参数 shouldAnimate 表示是否应用滚动动画。
scrollTo(left,top, [shouldAnimate])	数值 布尔值	滚动到指定位置，left 和 top 分别表示要滚动到的 x 坐标和 y 坐标。参数 shouldAnimate 表示是否应用滚动动画。
scrollBy(left, top, [shouldAnimate])	数值 布尔值	滚动到指定偏移量，let 和 top 分别表示要滚动到的 x 和 y 偏移量。参数 shouldAnimate 表示是否应用滚动动画。
$getByHandle (handle)	字符串	返回匹配 handle 字符串所指定的滚动视图实例。在滚动视图上，使用 delegate-handle 属性绑定该字符串。

接下来将通过一个案例为读者演示使用$getByHandle(handle)方法控制特定滚动视图的效果，如 demo10-6.html 所示。

demo10-6.html

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8">
    <meta   name="viewport"   content="initial-scale=1,   maximum-scale=1, user-scalable=no, width=device-width">
    <title>$getByHandle()</title>
      <link href="lib/ionic/css/ionic.css" rel="stylesheet">
      <script src="lib/ionic/js/ionic.bundle.min.js"></script>
  </head>
  <body ng-app="starter">
    <ion-header-bar  class="bar-positive">
        <h4>$getByHandle()功能演示</h4>
    </ion-header-bar>
    <ion-content ng-controller="myCtrl" delegate-handle="mainScroll">
        <ul class="list">
            <li class="item royal-bg">整个内容区域顶部</li>
            <li class="item" ng-repeat="item in items">{{item}}</li>
        </ul>
        <ion-scroll delegate-handle="small" style="height: 300px;">
            <ul class="list">
             <li class="item royal-bg">小区域顶部</li>
           <li class="item calm-bg" ng-repeat="item in items">{{item}}</li>
            </ul>
        </ion-scroll>
     <div class="bar">
      <button ng-click="scrollSmallToTop()">滚动到小区域顶部</button>
```

```
27        <button ng-click="scrollMainToTop()">滚动到整个内容区域到顶部</button>
28        </div>
29      </ion-content>
30  </body>
31  <script>
32      //定义控制器
33      angular.module("starter",["ionic"])
                .controller("myCtrl",function($scope, $ionicScrollDelegate){
34          //初始化数据
35          $scope.items=[];
36          for(var i=1;i<=10;i++){
37              $scope.items.push("第"+i+"条数据");
38          }
39          //滚动到整个内容区域顶部
40          $scope.scrollMainToTop = function() {
41              $ionicScrollDelegate.$getByHandle('mainScroll').scrollTop();
42          };
43          //滚动到小区域顶部
44          $scope.scrollSmallToTop = function() {
45              $ionicScrollDelegate.$getByHandle('small').scrollTop();
46          };
47      });
48  </script>
49 </html>
```

上述代码中，使用 ion-content 定义了一个内容区域，在该区域中包含一个数据列表和一个 ion-scroll 滚动小区域，该小区域中包含另外一个数据列表。

第 40~46 行代码定义了 scrollSmallToTop() 和 scrollMainToTop() 函数，使用 $ionicScrollDelegate.$getByHandle()方法来指定滚动视图名称，然后调用 scrollTop()方法，第 14 和第 19 行分别使用 delegate-handle 属性绑定了视图名称。

第 26 和 27 行定义了两个按钮，使用 ng-click 指令分别绑定 scrollSmallToTop()和 scrollMainToTop()函数，单击“滚动到小区域顶部”按钮后，将调用 scrollSmallToTop()函数，视图滚动到小区域顶部，单击“滚动到整个内容区域到顶部”按，将调用 scrollMainToTop()函数，视图滚动到整个内容区域的顶部。

使用 Chrome 浏览器访问 demo10-6.html，页面初始效果如图 10-14 所示。

图 10-14　demo10-6.html 页面效果

在图 10-14 中，由于列表数据条数较多，所以滚动小区域在列表的下方，滚动到页面底部便可以看到小区域和两个按钮，为了方便测试按钮效果，需要将小区域的数据也拉到最下方，页面效果如图 10-15 所示。

图 10-15　两个区域视图拉到最下方

在图 10-15 中，单击“滚动小区域到顶部”按钮，页面效果如图 10-16 所示。

图 10-16　滚动到小区域顶部

在图 10-16 中，单击“滚动整个内容区域到顶部”按钮，可以看到视图回到整个区域的最顶部，效果同图 10-14，到这里该案例的效果演示完毕，关于$ionicScrollDelegate 服务提供的更多方法，可以查看 ionic 官方文档。

10.3　ionic 路由

在本书第 5 章中讲解的 AngularJS 路由是 AngularJS 官方的 ngRoute 模块提供的，通过学习 ngRoute，读者了解了实现路由的基本思路，ionic 的路由没有基于 ngRoute 模块实现，而是选择了 AngularUI 项目的 angular-ui-router 模块，由于 ionic.bundle.js 中打包了 angular-ui-route 模块，所以在使用 ionic 路由时不需要单独引入依赖文件，本节将为读者介绍 ionic 框架路由的使用方法。

10.3.1　路由状态机

在 angular ui-router 中有 3 个关键词：状态（state）、URL、HTML 模板（template），也就是说，angular-ui-router 在 ngRoute 的基础之上，增加了“状态”的概念，将动态加载的 HTML 模板集合抽象为一个状态机，通过状态的切换来实现路由的导航功能，一个完整的路由状态机如图 10-17 所示。

图 10-17　路由状态机

在图 10-17 中，angular-ui-router 的$state 服务用于定义一个状态机实例，home，contact，option，about 代表 HTML 模板，每一个 HTML 模板，存在于一个特定的 URL，同时也对应于一个独一无二的状态。在不同的状态下，ionic 渲染对应的 HTML 模板就实现了路由导航的功能。

10.3.2 模板视图与视图容器

ionic 路由中有两个重要组成部分——模板视图与视图容器，其中模板视图用于渲染 HTML 模板，视图容器作为 HTML 模板的最外层容器，用于存放 HTML 代码片段。

ionic 中使用 ion-nav-view 指令来定义模板视图，示例代码如下所示。

```
<ion-nav-view><!--模板内容将被插入此处--></ion-nav-view>
```

尽管在模板视图中可以随便写 HTML，但是在 ionic 中，为了代码的规范性，需要使用 ion-view 指令来定义视图容器，示例代码如下所示。

```
<script id="templates/index.html" type="text/ng-template">
    <ion-view>
      <!--模板内容-->
    </ion-view>
</script>
```

上述代码中，AngularJS 将 script 元素的 type 属性定义为 text/ng-template，该 script 代码块则被称为内联模板，使用内联模板，可以把这些零散的 HTML 片段模板都集中在一个文件里，方便维护和开发。

10.3.3 路由的实现

了解了路由状态机、模板视图和视图容器后，接下来为读者介绍 ionic 路由的实现步骤，具体如下所示。

1. 配置路由状态机

首先需要配置路由状态机。状态的划分以及每个状态的元信息（例如模板、URL 等）是在配置阶段通过$stateProvider 完成的，示例代码如下所示。

```
angular.module("starter",["ionic"])
      .config(function($stateProvider){
                $stateProvider.state("state1",{
                        url: '/url1',
```

```
                templateUrl: 'page1.html'
            })
            .state("state2",{
                    url: '/url2',
                templateUrl: 'page2.html'
            })
            .state("state3",{
                    url: '/url3',
                templateUrl: 'page3.html'
        });
});
```

上述代码中，$stateProvider 的 state()方法用于声明一个状态，state()方法的第 1 个参数为状态名称，第 2 个参数该状态的元信息。

2. 触发状态迁移

在 angular-ui-router 中定义的 ui-sref 指令，该指令用于触发状态迁移，示例代码如下所示。

```
<a ui-sref="url1">Go State 1</a>
```

当用户点击上述链接时，ui-serf 的值对应的是状态的 url 值，$state 服务将根据状态名（如 url1）找到对应的元信息，提取、编译模板 page1.html，并将其显示在 ui-view 指令指定的视图窗口中。

ionic 使用 ion-nav-view 指令代替了 ui-view 指令，与 ui-view 类似，$state 服务根据状态的变化来提取对应的 HTML 模板，将其显示在 ion-nav-view 中。

ion-nav-view 指令有一个属性 name，用于指定视图容器的名字，在相同状态下的所有视图容器的名字都是唯一的，不同状态下可以有相同名称的视图容器。通过设置 ion-view 指令的属性，可以定制因状态变化而被动态载入的 HTML 模板视图。

ion-view 指令的常用可选属性，如表 10-7 所示。

表10-7 ion-view 属性

属性	取值类型	描述
title	字符串	显示在父 ion-nav-bar 的标题。
hide-back-button	布尔值	默认情况下，是否在父 ion-nav-bar 隐藏后退按钮。
hide-nav-bar	布尔值	默认情况下，是否隐藏父 ion-nav-bar。
cache-view	布尔值	设置视图是否为被缓存。 有关详细信息，请参阅 ionNavView 中的缓存部分。默认值 true。
can-swipe-back	布尔值	是否允许视图使用滑动手势返回上一级。 如果正在运行的平台不支持滑动返回功能，或者没有之前的视图，则不会被滑动返回。默认值 true。

ionic 中，视图是可以被缓存的（最多可以缓存 10 个视图），这意味着控制器通常只加载一次。为了监听视图何时进入或离开，ion-view 指令组件中提供了一些事件，这些事件包含有关视图的数据，如表 10-8 所示。

表10-8 ion-view 事件

事件	描述

$ionicView.loaded	视图加载完毕。这个事件只会在每次创建视图并添加到 DOM 时发生一次。如果视图离开但缓存，则事件不会在后续查看时再次触发。加载事件通常用于是放置视图的设置代码，不推荐用于视图变为活动时监听。
$ionicView.enter	视图进入完毕，现在是活动视图。此事件将触发，无论是第一次加载还是缓存视图。
$ionicView.leave	视图离开完毕，并且不再是活动视图。此事件将会触发，无论是缓存还是销毁。
$ionicView.beforeEnter	视图即将进入，并成为活动视图。
$ionicView.beforeLeave	视图即将离开，不再是活动视图。
$ionicView.afterEnter	视图进入完毕之后，现在是活动视图。
$ionicView.afterLeave	视图离开完毕之后，并且不再是活动视图。
$ionicView.unloaded	视图的控制器已被销毁，元素已从 DOM 中删除。

接下来通过一个案例来演示使用 ion-nav-view 和 ion-view 实现 ionic 路由的方法，如 demo10-7.html 所示。

demo10-7.html

```
<html>
<head>
    <meta charset="utf-8">
    <meta name="viewport" content="initial-scale=1, maximum-scale=1, user-scalable=no, width=device-width">
    <title>模板视图与视图容器</title>
    <link href="lib/ionic/css/ionic.css" rel="stylesheet">
    <script src="lib/ionic/js/ionic.bundle.min.js"></script>
</head>
<body ng-app="starter">
  <ion-nav-view><i>模板内容将会加载到此处</i></ion-nav-view>
<script id="templates/page1.html" type="text/ng-template">
    <ion-view>
        <ion-content class="calm-bg">
            <p>  This is page 1.  </p>
            <a class="button" ui-sref="two">go to page2</a>
        </ion-content>
    </ion-view>
</script>
<script id="templates/page2.html" type="text/ng-template">
    <ion-view>
        <ion-content  class="royal-bg">
            <p>   This is page 2 </p>
            <a class="button" ui-sref="three">go to page3</a>
        </ion-content>
    </ion-view>
</script>
<script id="templates/page3.html" type="text/ng-template">
    <ion-view>
```

```
29        <ion-content class="energized-bg">
30            <p>    This is page 3. </p>
31            <a class="button" ui-sref="one">go to page1</a>
32        </ion-content>
33    </ion-view>
34 </script>
35 </body>
36 <script type="text/javascript">
37        var app = angular.module('starter', ['ionic']);
38        app.config(function($stateProvider,$urlRouterProvider) {
39            // $stateProvider 用来配置状态路由
40            $stateProvider
41                  .state('one', { //"one"是页面 1 的状态
42                     url: '/one',  //"/one"是页面 1 的 Url
43              templateUrl: 'templates/page1.html'//"page1.html"是页面 1 的模板
44                  })
45                  .state('two', {
46                     url: '/two',
47              templateUrl: 'templates/page2.html'
48                  })
49                  .state('three', {
50                     url: '/three',
51              templateUrl: 'templates/page3.html'
52                  });
53            // 以上匹配都失败的情况下，进行下面的匹配
54            $urlRouterProvider.otherwise('/one');
55        });
56    </script>
57 </html>
```

在上述代码中，第 11~34 行定义了三个不同的页面模板，第 10 行的 ion-nav-view 指令的位置用于加载页面模板；第 38~52 行代码用于定义视图状态，$urlRouterProvider.otherwise('/one');用于页面重定向，当没有匹配的 URL 时会重定向到“/one”。使用 Chrome 浏览器访问 demo10-7.html，页面效果如图 10-18 所示。

图 10-18　demo10-7.html 页面效果

在图 10-18 中，由于页面下半部分为空，所以只截取的页面上半部分的效果，单击按钮“go to page2”会跳转到第 2 页，页面效果如图 10-19 所示。

图 10-19　page2

同样，在图 10-19 中单击按钮“go to page3”，便会跳转到第 3 页，到这里，一个基本的 ionic 路由功能已经完成了。

10.4 界面组件

前文介绍了 ionic JavaScript 组件中的基本布局和导航组件，本节将开始介绍 ionic 中基本的界面组件，包括顶部导航栏、列表、表单输入、幻灯片、侧边栏菜单、选项卡等。

10.4.1 顶部导航栏

在移动 App 中，顶部导航栏位于页面的最顶部，通常包含本页面的标题、页面间的跳转按钮等。

本节将为读者介绍顶部导航栏的使用方法，包括声明导航栏、嵌入后退按钮、声明按钮组、定制导航栏标题以及如何使用$ionicNavBarDelegate 代理服务来控制顶部导航栏。

1.声明导航栏

ionic 中可以使用 ion-nav-bar 指令为页面添加顶部导航栏。顶部导航栏通常与路由配合使用，它会根据视图状态的变化来更新导航栏显示的内容。ion-nav-bar 组件的可选属性如表 10-9 所示。

表10-9　ion-nav-bar 组件属性

属性	取值类型	描述
delegate-handle	字符串	定义一个用$ionicNavBarDelegate 获取本地组件对象的句柄名称。
align-title	字符串	导航栏标题对齐的位置。可用： 'left', 'right', 'center'。默认为 'center'。

2.嵌入后退按钮

顶部导航栏内部可以嵌入 ion-nav-back-button 指令作为后退按钮，该按钮需要在当前导航能够后退时才会显示出来，示例代码如下所示。

```
<ion-nav-bar class="bar-positive">
    <ion-nav-back-button>返回</ion-nav-back-button>
</ion-nav-bar>
```

将上述代码添加到 demo10-7.html 中第 10 行的 ion-nav-view 指令上面，访问该页面后，单击“go to page2”按钮跳转到第 2 页，便会显示返回按钮，如图 10-20 所示。

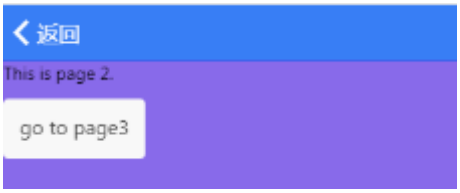

图 10-20　返回按钮

在图 10-20 中，按钮的图标为预定义效果，单击“返回”按钮后，会跳转到上一个页面，如图 10-21 所示。

图 10-21 page1

在图 10-21 中，由于跳转到 page1 后，页面不能在后退，所以按钮被隐藏。

3.声明按钮组

移动 App 的导航栏有时需要随着视图的切换显示不同的内容，为此，ionic 提供了 ion-nav-buttons 指令，用于在不同状态下显示不同的按钮组。该指令提供一个 side 属性，其作用是设置导航栏中按钮组的位置，取值范围包括：primary、secondary、left、right。

当 ion-bar-buttons 指令组件作为 ion-nav-bar 或者 ion-view 的直接后代元素，那么定义的按钮会依次根据 side 属性创建。

4.定制导航栏标题

ion-nav-titile 指令用于定制顶部导航栏的内容，该内容可以是任意的 HTML 代码片段，需要注意的是，ion-nav-title 必须是 ion-nav-bar 或者 ion-view 的直接后代元素。

5.$ionicNavBarDelegate 代理服务

$ionicNavBarDelegate 服务是用于控制顶部导航栏的服务组件，由于 App 中只能有一个顶部导航栏，所以不需要通过$getByHandle 方法获取导航栏对象，访问$ionicNavBarDelegate 即可。

该组件中提供的方法如表 10-10 所示。

表10-10 $ionNavBarDelegate 方法

方法	参数类型	描述
title(title)	字符串	用于设置 ion-nav-bar 显示的标题文本。
align([direction])	字符串	用于设置标题文字对齐的方向。可用：'left'，'right'，'center'。 默认: 'center'。
showBar(show)	布尔值	用于设置导航栏是否显示。
showBackButton([show])	布尔值	用于设置是否显示后退按钮。

接下来通过一个案例来演示定制顶部导航栏的效果，如 demo10-8.html 所示，

demo10-8.html

```
1  <html>
2  <head>
3      <meta charset="utf-8">
4      <meta   name="viewport"   content="initial-scale=1,   maximum-scale=1,
   user-scalable=no, width=device-width">
5      <title>定制顶部导航栏</title>
6      <link href="lib/ionic/css/ionic.css" rel="stylesheet">
7      <script src="lib/ionic/js/ionic.bundle.min.js"></script>
8  </head>
9  <body ng-app="starter" ng-controller="myCtrl">
10 <ion-nav-bar class="bar-positive">
11     <ion-nav-back-button>返回</ion-nav-back-button>
12 </ion-nav-bar>
13 <ion-nav-view>
```

```
14     <i>模板内容将会加载到此处</i>
15 </ion-nav-view>
16 <script id="templates/page1.html" type="text/ng-template">
17     <ion-view>
18         <ion-nav-title>page1</ion-nav-title>
19         <ion-nav-buttons side="secondary">
20             <a class="button">收藏</a>
21             <a class="button" ui-sref="two">下一页</a>
22         </ion-nav-buttons>
23         <ion-content class="calm-bg">
24             <p>    This is page 1.  </p>
25         </ion-content>
26     </ion-view>
27 </script>
28 <script id="templates/page2.html" type="text/ng-template">
29     <ion-view>
30         <ion-content  class="royal-bg">
31             <p>   This is page 2.  </p>
32             <a class="button" ng-click="setTitle()">设置标题</a>
33         </ion-content>
34     </ion-view>
35 </script>
36 </body>
37   <script type="text/javascript">
38         var app = angular.module('starter', ['ionic']);
39         app.config(function($stateProvider,$urlRouterProvider) {
40             // $stateProvider 用来配置状态路由
41             $stateProvider
42                    .state('one', {
43                        url: '/one',
44                        templateUrl: 'templates/page1.html'
45                    })
46                    .state('two', {
47                        url: '/two',
48                        templateUrl: 'templates/page2.html'
49                    });
50
51             // 以上匹配都失败的情况下，进行下面的匹配
52             $urlRouterProvider.otherwise('/one');
53         });
54         app.controller("myCtrl",function($scope,$ionicNavBarDelegate) {
55             $scope.setTitle = function () {
56                 $ionicNavBarDelegate.title("我是 page2");
57             }
58         });
```

```
59     </script>
60 </html>
```

上述代码中，定义了两个 HTML 模板 page1.html 和 page2.html，并且使用 ionic 路由配置了导航。page1.html 模板中使用了 ion-nav-title 定义了导航栏显示的标题内容，使用 ion-nav-buttons 定义了按钮组，side 属性取值为 secondary，代表按钮组显示在导航栏的右侧；page2.html 中“设置标题”按钮绑定了事件，单击按钮后调用 55~57 行定义的 setTitle()方法，该方法中使用了$ionicNavBarDelegate 服务代理的 title()方法，用于设置 page2 的标题。使用 Chrome 浏览器访问 demo10-8.html，页面效果如图 10-22 所示。

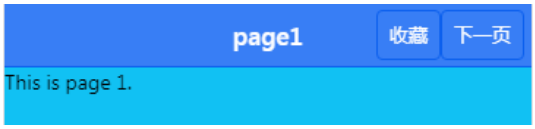

图 10-22　demo10-8.html 页面效果

在图 10-22 中单击“下一页”按钮，页面效果如图 10-23 所示。

图 10-23　page2

在图 10-23 中单击“设置标题”按钮，可以看到导航栏显示标题“我是 page2”，页面效果如图 10-24 所示。

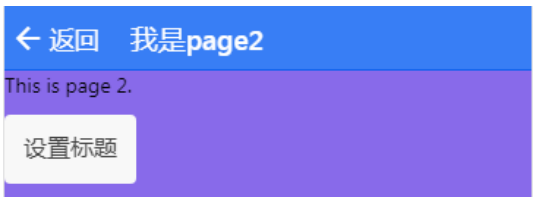

图 10-24　显示标题

10.4.2 列表

在本书第 9 章讲解过 ionic CSS 定义列表的方式，本节将为读者介绍可动态配置的 ionic 列表，该列表通过 ionic JavaScript 组件来实现，支持多种多样的交互模式，例如通过移除列表的某一项，拖动列表项重新排序，滑动编辑列表项等等。

本节将为读者介绍 ionic JavaScript 列表的使用方法，包括声明列表、嵌入成员按钮以及用于控制列表元素的代理服务$ionicListDelegate。

1. 声明列表

ionic JavaScript 使用指令 ion-list 和 ion-item 两个指令来声明列表，示例代码如下所示。

```
<ion-list>
  <ion-item ng-repeat="item in items">
    {{item}}
  </ion-item>
</ion-list>
```

ion-list 指令中提供了一些可选的属性，如表 10-11 所示。

表10-11　ion-list 属性

属性	取值类型	描述
delegate-handle	字符串	用于定义带有$ionicListDelegate 的列表。
show-delete	布尔值	如果在成员内有 delete 按钮（ion-delete-button），使用这个属性来通知列表是否显示元素删除按钮。允许的值为：true | false。
show-reorder	布尔值	如果在成员内有 reorder 按钮（ion-reorder-button），使用这个属性来通知列表是否显示元素重排序按钮。允许的值为：true | false。
can-swipe	布尔值	如果在成员内有 option 按钮（ion-option-button），使用这个属性来允许或禁止通过向左滑动成员来打开 option 按钮。允许的值为：true | false ，默认为 true。
type	字符串	可用来设置列表的种类：list-inset | card。这两种列表都产生内嵌的效果，区别在于 card 列表有边框的阴影效果。

2.嵌入成员按钮

在移动 App 中，列表的操作通常需要一些按钮的支持，例如删除某个列表项或者列表项重新排序等。ionic 中提供了三个指令为列表项添加按钮功能，如下所示。

- ion-option-button：选项按钮。

一个 ion-item 内可以包含多个选项按钮。选项按钮是隐藏的，需要用户向左滑动成员，以显示选项按钮。ion-tabs 的 can-swipe 属性用于设置允许或禁止滑动开启选项按钮。

- ion-delete-button：删除按钮。

一个 ion-item 内最多有一个删除按钮。删除按钮在显示时总是位于成员的最左端。ion-tabs 的 show-delete 属性用于显示或隐藏删除按钮。

- ion-reorder-button ：重排按钮。

一个 ion-item 内最多有一个重排按钮。重排按钮在显示时总是位于成员的最右端。可以使用 ion-tabs 的 show-reorder 属性用于显示或隐藏重排按钮。

3. $ionicListDelegate 服务代理

ionic 中，如果需要在脚本中控制列表元素，可以使用$ionicListDelegate 服务，该服务中提供的方法如表 10-12 所示。

表10-12 $ionicListDelegate 方法

方法	描述
showReorder([showReorder])	显示/关闭排序按钮。showReorder 的允许值为：true | false。可以使用一个作用域上的表达式。
showDelete([showDelete])	显示/关闭删除按钮。showDelete 的允许值为： true | false。可以使用一个作用域上的表达式。
canSwipeItems([canSwipeItems])	是否允许通过滑动方式来显示成员选项按钮。canSwipeItems 的允许值为：true | false。可以使用一个作用域上的表达式。
closeOptionButtons()	关闭所有选项按钮。

接下来通过一个案例来演示 ionic JavaScript 列表的使用方法，如 demo10-9.html 所示。

demo10-9.html

```
1  <html>
2  <head>
3      <meta charset="utf-8">
```

```
4     <meta   name="viewport"   content="initial-scale=1,   maximum-scale=1,
   user-scalable=no, width=device-width">
5     <title>ionic JS 列表</title>
6     <link href="lib/ionic/css/ionic.css" rel="stylesheet">
7     <script src="lib/ionic/js/ionic.bundle.min.js"></script>
8  </head>
9  <body ng-app="starter" ng-controller="myCtrl">
10 <ion-header-bar class="bar-positive">
11    <div class="buttons">
12       <button class="button button-icon icon ion-ios-minus-outline"
                 ng-click="data.showDelete =
                !data.showDelete; data.showReorder = false"></button>
13    </div>
14    <h1 class="title">通讯录</h1>
15    <div class="buttons">
16       <button class="button"
                ng-click="data.showDelete = false;
                data.showReorder = !data.showReorder">排序 </button>
17    </div>
18 </ion-header-bar>
19 <ion-content>
20    <ion-list show-delete="data.showDelete"
              show-reorder="data.showReorder">
21       <ion-item ng-repeat="item in items" item="item"
22                href="#/item/{{item.id}}" class="item-remove-animate">
23                好友{{ item.id }}
24          <ion-delete-button class="ion-minus-circled"
                               ng-click="onItemDelete(item)">
25          </ion-delete-button>
26          <ion-option-button class="button-assertive">
27                星标朋友
28          </ion-option-button>
29          <ion-option-button class="button-balanced">
30                修改备注
31          </ion-option-button>
32          <ion-reorder-button class="ion-navicon"
33             on-reorder="moveItem(item, $fromIndex, $toIndex)">
               </ion-reorder-button>
34       </ion-item>
35    </ion-list>
36 </ion-content>
37 </body>
38 <script type="text/javascript">
39       var app = angular.module('starter', ['ionic']);
40          app.controller('myCtrl', function($scope) {
```

```
41                  $scope.data = {
42                      showDelete: false
43                  };
44                  //排序
45                  $scope.moveItem = function(item, fromIndex, toIndex) {
46                      $scope.items.splice(fromIndex, 1);
47                      $scope.items.splice(toIndex, 0, item);
48                  };
49                  //删除好友
50                  $scope.onItemDelete = function(item) {
51                      $scope.items.splice($scope.items.indexOf(item), 1);
52                  };
53    $scope.items = [{ id: 0 }, { id: 1 }, { id: 2 }, { id: 3 },
                      { id: 4 }, { id: 5 }, { id: 6 }, { id: 7 }, { id: 8 },
                      { id: 9 }, { id: 10 }  ];
54  });
55     </script>
56 </html>
```

上述代码中，使用 ion-list 和 ion-item 定义一个列表，模拟通讯录的功能。

第 10~18 行定义了页面的头部区域，并在头部区域添加了两个按钮，其中第 12 行的按钮用于控制每个列表项前面删除按钮的显示和隐藏，第 16 定义的按钮用于排序功能。

第 24~25 行使用 ion-delete-button 在每个列表项上定义删除按钮，该按钮显示在列表项的前方，默认隐藏。

第 26~31 行使用两个 ion-option-button 定义星标朋友和修改备注功能的按钮，向左滑动列表项会显示这两个按钮，该案例中两个按钮没有绑定事件。

第 32~33 行使用 ion-reorder-button 定义排序拖动按钮，当单击“排序”按钮时显示拖动按钮，拖动按钮可以实现列表项的重新排序。

使用 Chrome 浏览器访问 demo10-9.html，页面效果如图 10-25 所示。

图 10-25　demo10-9.html 页面效果

在图 10-25 中，单击左上角“”按钮，可以显示列表项前面的删除按钮，再次单击隐藏删除按钮，单击删除按钮删除前两项后，页面效果如图 10-26 所示。

图 10-26　删除列表项

在图 10-26 中，单击“排序”按钮，会显示列表项后面的拖动按钮（再次单击隐藏），将前两项拖到最后，如图 10-27 所示。

图 10-27　列表项排序

在任意列表项上单击并向左滑动，会显示“修改备注”和“星标朋友”两个按钮，如图 10-28 所示。

图 10-28　左滑动显示按钮

10.4.3 表单输入

HTML5 中提供的表单控件在移动设备上的显示已趋于完善，因此，ionic 并没有大量

的包装定制表单输入组件。

ionic JavaScript 中主要提供三个表单输入组件，如下所示。

（1）ion-checkbox：用于定义复选框。

与 HTML5 的 checkbox 相比，使用 ionic 的 ion-checkbox 可以使用如下属性：

- ng-model：使用可选的 ng-model 属性，可以直接将选中状态绑定到作用域上的变量。

（2）ion-radio：用于定义单选按钮。

与 HTML5 的 radio 相比，ion-radio 的改进也是明显的，可以使用如下属性：

- ng-model：使用可选的 ng-model 属性，实现与作用域变量的数据绑定。
- ng-value：使用可选的 ng-value 属性，可以使用作用域变量设置单选按钮对应的值。

（3）ion-toggle：用于定义开关。

ion-toggle 有两个可选的属性：

- ng-model：和复选按钮一样，开关按钮也可以使用可选的 ng-model 属性实现与作用域变量的双向绑定。
- toggle-class ：同样可以使用可选的 toggle-class 属性为开关按钮声明额外的样式。比如：toggle-{color} 用来声明配色方案。

使用上述组件定义表单输入组件与使用 ionic CSS 定义的表单输入功能基本一致，只是除开关外，复选框和单选按钮在样式上更接近原生输入控件。

接下来通过一个案例来演示 ionic JavaScript 表单输入组件的用法，如 demo10-10.html 所示。

demo10-10.html

```
<html>
<head>
    <meta charset="utf-8">
    <meta name="viewport" content="initial-scale=1, maximum-scale=1, user-scalable=no, width=device-width">
    <title>ionic JS 表单输入</title>
    <link href="lib/ionic/css/ionic.css" rel="stylesheet">
    <script src="lib/ionic/js/ionic.bundle.min.js"></script>
</head>
<body ng-app="starter" ng-controller="myCtrl">
<ion-content>
    <div class="list">
        <div class="item item-divider calm-bg" >复选框</div>
        <ion-checkbox ng-repeat="item in checkList"
                      ng-model="item.checked"
                      ng-checked="item.checked">
            {{ item.text }}
        </ion-checkbox>
        <div class="item">
            <pre ng-bind="checkList | json"></pre>
        </div>
        <div class="item item-divider calm-bg">
            单选按钮选中的值为: {{ data.clientSide }}
```

```
23        </div>
24        <ion-radio ng-repeat="item in clientSideList"
25                   ng-value="item.value"
26                   ng-model="data.clientSide">
27            {{ item.text }}
28        </ion-radio>
29    <div class="item item-divider calm-bg" >开关</div>
30    <ion-toggle ng-repeat="item in settingsList"
31                ng-model="item.checked"
32                ng-checked="item.checked">
33               {{ item.text }}
34    </ion-toggle>
35        <div class="item">
36            <pre ng-bind="settingsList | json"></pre>
37        </div>
38    </div>
39 </ion-content>
40 </body>
41 <script type="text/javascript">
42        var app = angular.module('starter', ['ionic']);
43            app.controller('myCtrl', function($scope) {
44                //复选框数据
45                $scope.checkList = [
46                    { text: "Angular", checked: true },
47                    { text: "Backbone", checked: false }
48                ];
49                //单选按钮数据
50                $scope.clientSideList = [
51                    { text: "Angular", value: "Angular" },
52                    { text: "Backbone", value: "Backbone" }
53                ];
54                $scope.data = {
55                    clientSide: 'Backbone'
56                };
57                //开关数据
58                $scope.settingsList = [
59                    { text: "Angular", checked: true },
60                    { text: "Backbone", checked: false }
61                ];
62            });
63    </script>
64 </html>
```

上述代码中，定义了复选框、单选按钮和开关三个组件，将三个组件放入列表中，并使用列表分隔符进行分隔。

为了更直观的展示组件和数据的关系，在第 19 和第 36 行定义复选框和开关按钮的代

码中，使用“list|json”的方法在页面输出组件对象的数据结构，这样在选中某项时，checked 属性会变为 true；在单选按钮的代码中，使用 data.clientSide 属性绑定选中的值，并在页面做输出，可以直接看到单选按钮选中的值。使用 Chrome 浏览器访问 demo10-10.html，页面效果如图 10-29 所示。

图 10-29　demo10-10.html 页面效果

10.4.4 幻灯片

幻灯片也是一种常见的 UI 表现方式，它从一组元素中选择一个投射到屏幕可视区域，用户可以通过滑动方式（向左或向右）进行切换。一般作为移动 App 的启动页面。效果如图 10-30 所示。

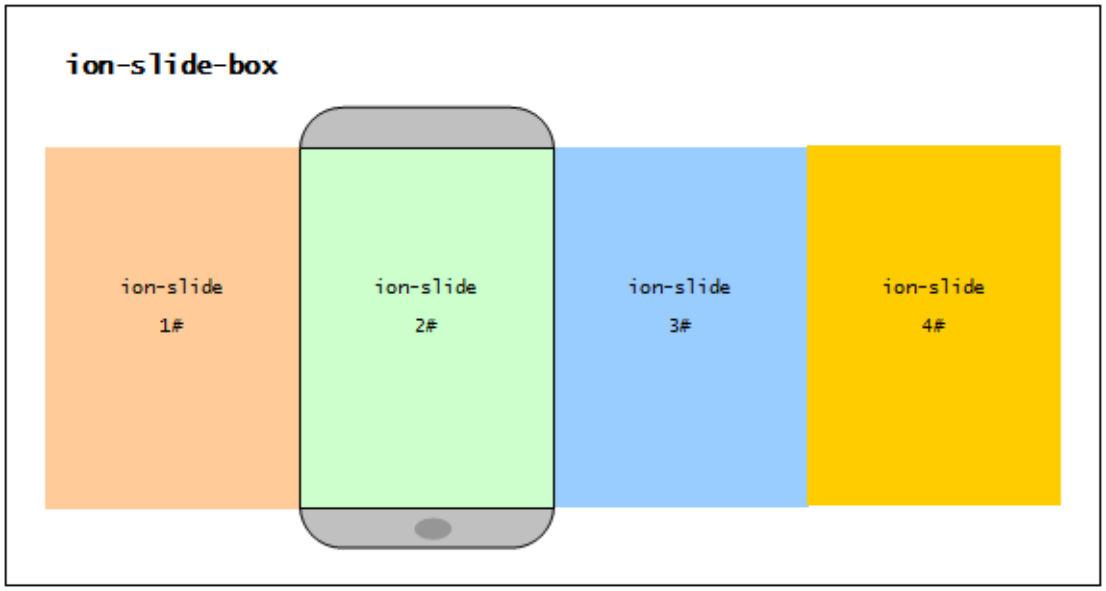

图 10-30 幻灯片效果

在 ionic 中，使用 ion-slide-box 指令声明幻灯片元素， 使用 ion-slide 指令声明幻灯页元素，示例代码如下所示。

```
<ion-slide-box>
    <ion-slide>...</ion-slide>
    <ion-slide>...</ion-slide>
    ...
</ion-slide-box>
```

ion-slide-box 指令有一些可选属性可以定制其播放行为，如表 10-13 所示

表10-13 ion-list 属性

属性	取值类型	描述
delegate-handle	字符串	该句柄用$ionicSlideBoxDelegate 来标识这个滑动框。
does-continue	布尔值	是否循环切换。由于开头的幻灯页只能向左滑动，最后的幻灯页只能向右滑动。将 does-continue 属性值设为 true，就可以让幻灯页组首尾连接起来，循环切换。
auto-play	布尔值	是否自动播放。通过将 auto-play 属性设置为 true，可以让幻灯片自动切换。切换的间隔默认是 4000ms，可以通过属性 slide-interval 进行调整。
slide-interval	数字	自动播放的间隔时间，默认为 4000ms。
show-pager	布尔值	是否显示分页器。分页器用来指示幻灯页的选中状态，位于幻灯片的底部。允许值为：true | false。

ion-slide-box 指令还提供了可选事件，如表 10-14 所示。

表10-14 ion-list 属性

事件	取值类型	描述
pager-click	表达式	当点击页面时，触发该表达式（如果 shou-pager 为 true），传递一个'索引'变量。
on-slide-changed	表达式	当滑动时，触发该表达式，传递一个'索引'变量。
active-slide	表达式	将模型绑定到当前滑动框。

读者可以使用服务$ionicSlideBoxDelegate 在脚本中操作幻灯片对象，该服务中提供的方法如表 10-15 所示。

表10-15 $ionicSlideBoxDelegate 方法

方法	描述
update()	重绘幻灯片。当容器尺寸发生变化时，需要调用 update()方法重绘幻灯片。
slide(to[,speed])	切换到指定幻灯页。参数 to 表示切换的目标幻灯页序号，参数 speed 是可选的，表示以毫秒为单位的切换时间。
enableSlide([shouldEnable])	幻灯片能使参数 shouldEnable 的允许值为：true | false 。
previous()	切换到前一张幻灯页。
next()	切换到后一张幻灯页。
currentIndex()	获得当前幻灯页的序号。
slideCount()	获得全部幻灯页的数量。

接下来通过一个案例来演示幻灯片的使用方法，如 demo10-11.html 所示。

demo10-11.html

```
1  <html>
```

```
<head>
    <meta charset="utf-8">
    <meta name="viewport" content="initial-scale=1, maximum-scale=1, user-scalable=no, width=device-width">
    <title>ionic JS 幻灯片</title>
    <link href="lib/ionic/css/ionic.css" rel="stylesheet">
    <script src="lib/ionic/js/ionic.bundle.min.js"></script>
    <style>
        .blue {
            background-color: deepskyblue;
        }
        .yellow {
            background-color: yellow;
        }
        .pink {
            background-color: pink;
        }
        .box{
            height:100%;
        }
        .box h1{
           text-align: center;
            position: relative;
            top:50%;
        }
    </style>
</head>
<body ng-app="starter" ng-controller="myCtrl">
<ion-slide-box auto-play="true">
    <ion-slide>
        <div class="box blue"><h1>全世界的好东西</h1></div>
    </ion-slide>
    <ion-slide>
        <div class="box yellow"><h1>天天五折起</h1></div>
    </ion-slide>
    <ion-slide>
        <div class="box pink"><h1>立即开启</h1></div>
    </ion-slide>
</ion-slide-box>
</body>
<script> angular.module('starter',['ionic']);</script>
</html>
```

上述代码中，第 29~39 行定义了一个幻灯片，该幻灯片中包含了三个 ion-slide 元素。第 29 行使用 auto-play 属性设置了婚登怕自动播放；第 8~26 行设置了幻灯片的样式，包括满屏（只需设置高度）、文字居中显示、三个元素不同的背景色等。使用 Chrome 浏览器访

问 demo10-11.html，页面效果如图 10-31 所示。

图 10-31　demo10-11.html 页面效果

该幻灯片默认 4000ms 自动播放一次，也可滑动播放，如图 10-32 所示。

图 10-32　幻灯片播放

10.4.5 侧边栏菜单

侧边栏菜单是一个最多包含三个子容器的元素：左边栏、右边栏、主内容区域，通过把主要内容区域从一边滑动到另一边，来让左侧或右侧的侧边栏菜单进行切换。如图 10-33 所示。

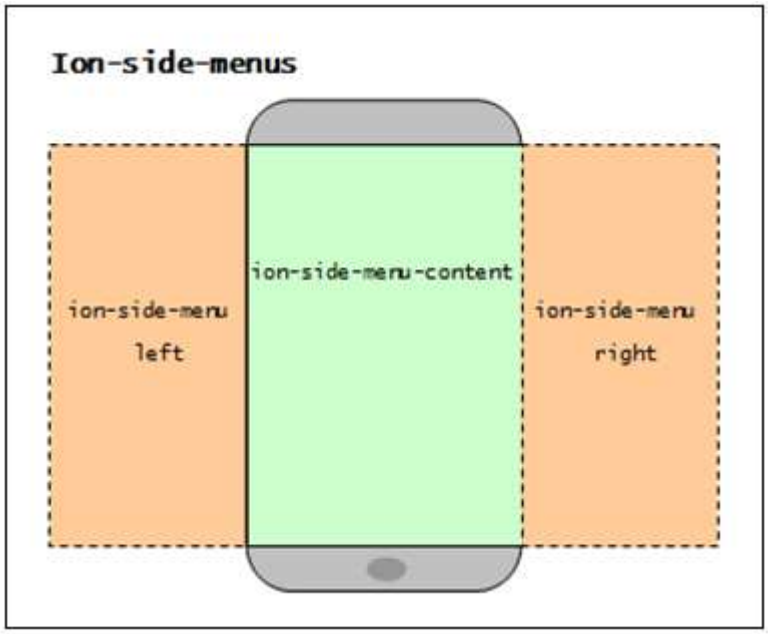

图 10-33　侧边栏结构

默认情况下，侧边栏菜单将只显示 ion-side-menu-content 容器的内容。向左滑动时，将显示右边栏 ion-side-menu 容器的内容，向右滑动时，将显示左边栏 ion-side-menu 容器的内容。

接下来将为读者介绍侧边栏菜单的具体使用方法，包括声明侧边栏菜单、侧边栏菜单的显示设置以及如何使用$ionicSideMenuDelegate 代理服务来控制侧边栏菜单。

1. 声明侧边栏菜单

ionic 中声明侧边栏菜单，需要添加一个父元素 ion-side-menus，一个中间内容 ion-side-menu-content 和一个或更多 ion-side-menu 指令，示例代码如下所示。

```
<ion-side-menus>
  <!-- 中间内容 -->
  <ion-side-menu-content ng-controller="ContentController">
  </ion-side-menu-content>
  <!-- 左侧菜单 -->
  <ion-side-menu side="left">
  </ion-side-menu>
  <!-- 右侧菜单 -->
  <ion-side-menu side="right">
  </ion-side-menu>
</ion-side-menus>
```

上述代码中，ion-side-menu-content 用于设置侧边栏菜单的主内容区域，使用指令 ion-side-menu 用于声明侧边栏区域容器。ion-side-menu-content 指令中有一些可选的属性，如表 10-16 所示。

表10-16　ion-side-menu-content 属性

属性	取值类型	描述
drag-content	布尔值	是否允许拖动内容打开侧栏菜单，默认为 true。当设置为 false 时，将禁止通过拖动内容打开侧栏菜单。

edge-drag-threshold	布尔值	是否启用边距检测，默认为 false。 1）如果设置为一个正数，那么只有当拖动发生在距离边界小于这个数值的情况下，才触发侧栏显示。 2）当设置为 true 时，使用默认的 25px 作为边距阈值。如果设置为 false 或 0，则意味着禁止边距检测，可以在内容区域的任何地方拖动来打开侧栏。
	数字	

ion-side-menu 指令的属性如表 10-17 所示。

表10-17 ion-side-menu 属性

属性	取值类型	描述
side	字符串	侧栏菜单当前在哪一边。可选的值有: 'left' 或 'right'。
is-enabled	布尔值	可选，该侧栏菜单是否可用。
width	数字	可选，侧栏菜单应该有多少像素的宽度，默认为 275。

2.侧边栏菜单显示设置

（1）显示状态设置

menu-toggle 指令用来给元素增加切换侧栏内容显示状态的功能，示例代码如下所示。

```
<!--切换左侧栏显示状态-->
<any menu-toggle="left"></any>
<!--切换右侧栏显示状态-->
<any menu-toggle="right"></any>
```

（2）关闭侧栏内容

menu-close 指令用来给元素增加关闭侧栏内容的功能，示例代码如下所示。

```
<any menu-close=""></any>
```

与 menu-toggle 指令不同，menu-close 不需要指定要关闭的侧栏位置，而是直接将当前打开的侧栏关闭。

（3）侧边栏自动显示条件表达式

默认情况下，侧边栏是隐藏的，需要用户向左或向右拖动内容，或者通过一个切换按钮来打开。但是在有些场景下，比如，横放的平板，当屏幕宽度足够大时，页面自动地显示侧边栏内容会更合理。

为此，ionic 提供了 expose-aside-when 属性，用来处理这种情况。与 CSS3 的媒体查询类似，expose-aside-when 的取值通常是一个 CSS 表达式，例如 expose-aside-when="(min-width:768px)"，这意味着当屏幕宽度大于 768px 时，将自动显示侧栏菜单。ionic 为 expose-aside-when 提供了一个快捷选项：large，设置 expose-aside-when="large"等同于 expose-aside-when="(min-width:768px)"。

3.$ionicSideMenuDelegate 代理服务

如果需要在脚本中控制侧边栏菜单，可以使用$ionicSideMenuDelegate 服务代理。$ionicSideMenuDelegate 提供了很多组件方法，常用方法如表 10-18 所示。

表10-18 $ionicSideMenuDelegate 方法

方法	描述
toggleLeft([isOpen])	是否打开左侧栏菜单。参数 isOpen 是可选的，默认为 true ，表示打开左侧栏菜单。
toggleRight([isOpen])	是否打开右侧栏菜单。参数 isOpen 是可选的，默认为 true ，表示打开右侧栏菜单。

getOpenRatio()	侧栏菜单打开的宽度占其总宽度比例。例如，一个 100px 宽的侧栏菜单，如果打开 50px，那么其比例为 50%，getOpenRatio()将返回 0.5 。
isOpen()	当前侧栏菜单是否打开。不管是左侧栏菜单，还是右侧栏菜单，只要处于打开状态，isOpen()都返回 true。
isOpenLeft()	左侧栏菜单是否打开。当左侧栏菜单处于打开状态时，isOpenLeft()返回 true。
isOpenRight()	右侧栏菜单是否打开。当右侧栏菜单处于打开状态时，isOpenRight()返回 true。
canDragContent([canDrag])	是否允许拖拽内容以打开侧栏菜单。canDrag 参数是可选的，如果 canDrag 为 true，表示允许通过拖拽内容打开侧栏菜单。
edgeDragThreshold(value)	设置边框距离阈值。当参数 value 为 false 或 0 时，意味着在内容区域任何位置进行拖拽都可以打开侧栏菜单。 如果参数 value 为一个数值，意味着只有当拖拽发生的位置距边框不大于此数值时，才能打开侧栏菜单。参数 value 为 true 等同于将 value 设置为 25。
$getByHandle(handle)	返回匹配 handle 字符串所指定的侧栏菜单实例。

接下来通过一个案例来演示侧边栏菜单的使用方法，如 demo10-12.html 所示。

demo10-12.html

```
<html>
<head>
    <meta charset="utf-8">
    <meta name="viewport" content="initial-scale=1, maximum-scale=1, user-scalable=no, width=device-width">
    <title>侧边栏菜单</title>
    <link href="lib/ionic/css/ionic.css" rel="stylesheet">
    <script src="lib/ionic/js/ionic.bundle.min.js"></script>
</head>
<body ng-app="starter" >
<ion-side-menus>
    <!-- 中间内容 -->
    <ion-side-menu-content ng-controller="myCtrl">
        <ion-header-bar align-title="left" class="bar-positive">
            <button class="button button-icon icon ion-navicon"
                    ng-click="toggleLeft()"></button>
            <h1 class="title" style="text-align: center;">小蓝书</h1>
        <button class="button button-icon icon ion-android-cart"></button>
        </ion-header-bar>
        <ion-content>
            商品展示区域
        </ion-content>
    </ion-side-menu-content>
    <!-- 左侧菜单 -->
    <ion-side-menu side="left" width="180">
        <header class="bar bar-header bar-stable">
```

```
25          <h1 class="title">个人中心</h1>
26      </header>
27      <ion-content class="has-header">
28          <ion-list>
29              <ion-item nav-clear menu-close class="item-avatar" href="#">
30                    <img src="img/demo10-12/abby.jpg">Abby
31              </ion-item>
32              <ion-item nav-clear menu-close href="#">
33                  我的收货地址
34              </ion-item>
35              <ion-item nav-clear menu-close href="#">
36                  账户与安全
37              </ion-item>
38              <ion-item nav-clear menu-close href="#">
39                   隐私
40              </ion-item>
41              <ion-item nav-clear menu-close href="">
42                  通用
43              </ion-item>
44          </ion-list>
45      </ion-content>
46    </ion-side-menu>
47 </ion-side-menus>
48 </body>
49 <script type="text/javascript">
50    angular.module('starter',['ionic'])
51          .controller('myCtrl',function($scope, $ionicSideMenuDelegate){
52              $scope.toggleLeft = function() {
53                  $ionicSideMenuDelegate.toggleLeft();
54              };
55          });
56 </script>
57 </html>
```

上述代码中，定义了一个侧边栏菜单，模拟了购物类 App 的个人中心效果，中间的主内容区域头部包含左侧按钮、标题和右侧按钮，内容区域添加了“商品展示区域”文字提示。

左侧按钮的 ng-click 事件绑定了 toggleLeft() 函数，该函数中调用 $ionicSideMenuDelegate.toggleLeft()方法实现的菜单显示，读者还可以尝试在按钮上添加 menu-toggle="left"也可以实现同样的效果。使用 Chrome 浏览器访问 demo10-12.html，页面效果如图 10-34 所示。

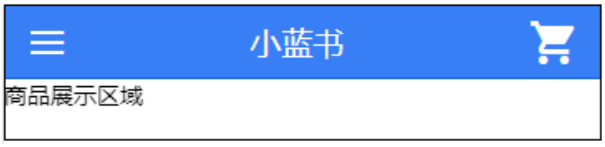

图 10-34　demo10-12.html 页面效果

在图 10-34 中单击左侧图标按钮，便会显示左侧菜单，如图 10-35 所示。

图 10-35　demo10-12.html 页面效果

需要注意的是，由于以上页面内容没有满屏，所以都是截取了页面的上半部分。

10.4.6 选项卡

在本书第 9 章讲解过 ionic CSS 定义的选项卡，读者了解了选项卡的外观以及通过预定义的 CSS 类进行选项卡的外观定制，然而通过 ionic CSS 定义的选项卡无法完成选项卡之间切换的功能，该功能需要通过 ionic JavaScript 实现的选项卡结合路由的功能来实现。

本节将为读者介绍 ionic 选项卡的使用，包括声明选项卡以及如何使用 $ionicTabsDelegate 代理服务来控制选项卡。

1．声明选项卡

ionic 中使用 ion-tabs 指令声明选项卡，使用 ion-tab 声明选项页，示例代码如下所示。

```
<ion-tabs>
    <ion-tab title="...">...</ion-tab>
    <ion-tab title="...">...</ion-tab>
       ...
</ion-tabs>
```

在上述代码中，每个 ion-tab 元素的 title 属性值将作为选项页的标题，可以使用 CSS 类.tabs-top 和.tabs-bottom 来定义选项卡位于页面的顶部还是底部。也可以通过 $ionicConfigProvider，在配置阶段设置选项卡的位置，示例代码如下所示。

```
app.config(function($ionicConfigProvider){
    $ionicConfigProvider.tabs.position("top");
    //参数可以是：top | bottom
});
```

需要注意的是，不要把 ion-tabs 指令放在 ion-content 之内，应当将其放入 ion-view 指令内，否则 ionic 在计算布局时可能出错。

ion-tab 指令中提供了一些可选的属性和事件用于选项卡的基本操作，如表 10-19 所示。

表10-19　ion-tab 属性

属性	取值类型	描述
href	字符串	当用户触碰的时候，该选项卡将会跳转的链接。
icon	字符串	该属性值将用来在标题文字旁边添加一个指定的图标。这个属性

		的值将被作为 icon-on 和 icon-off 的默认值。
icon-on	字符串	被选中标签的图标如果一个选项页被选中，ion-tabs 将使用 icon-on 属性的值绘制图标。如果 icon-on 没有设置，那么 ion-tabs 就使用 icon 属性的值绘制图标。
icon-off	字符串	没被选中标签的图标。如果一个选项页没有被选中，ion-tabs 将使用 icon-off 属性的值绘制图标。如果 icon-off 没有设置，那么 ion-tabs 就使用 icon 属性的值绘制图标。
badge	表达式	选项卡上的徽章，通常是一个数字，可以是一个具体的值，也可以是当前作用域上的一个变量。
badge-style	表达式	选项卡上微章的样式（例如，tabs-positive ）。
on-select	表达式	选项卡被选中时触发。
on-deselect	表达式	选项卡取消选中时触发。
ng-click	表达式	通常，点击时选项卡会被选中。如果设置了 ng-Click，它将不会被选中。可以用$ionicTabsDelegate.select()来指定切换标签。
hidden	表达式	隐藏标签页。hidden 属性是当前作用域上的表达式。当其值为 true 时，选项页将不可见。
disabled	表达式	禁用标签页。 disabled 属性是当前作用域上的表达式。当值为 true 时，选项页将不响应用户的点击。

2．$ionicTabsDelegate 代理服务

使用$ionicTabsDelegate 服务，可以在脚本中控制选项卡对象，该服务中提供的方法如表 10-20 所示。

表10-20 $ionicTabsDelegate 方法

方法	描述
select(index)	选中指定的选项页。index 参数从 0 开始，第一个选项页的 index 为 0，第二个为 1，依次类推。
selectedIndex()	返回当前选中选项页的索引号。 如果当前没有选中的选项页，则返回 -1。
$getByHandle(handle)	返回匹配 handle 字符串所指定的选项卡栏实例。

接下来通过一个案例来演示 ionic JavaScript 选项卡是具体用法，如 demo10-13.html 所示。

demo10-13.html

```
1  <html>
2  <head>
3      <meta charset="utf-8">
4      <meta   name="viewport"   content="initial-scale=1,   maximum-scale=1,
   user-scalable=no, width=device-width">
5      <title>选项卡</title>
6      <link href="lib/ionic/css/ionic.css" rel="stylesheet">
7      <script src="lib/ionic/js/ionic.bundle.min.js"></script>
8  </head>
9  <body ng-app="starter">
10 <ion-nav-bar class="bar-stable">
11     <ion-nav-back-button>
12     </ion-nav-back-button>
```

```
13 </ion-nav-bar>
14 <ion-tabs class="tabs-icon-top tabs-color-active-positive">
15    <!-- Dashboard Tab -->
16    <ion-tab title="Status" icon-off="ion-ios-pulse"
                icon-on="ion-ios-pulse-strong" href="#/dash">
17       <ion-nav-view name="tab-dash"></ion-nav-view>
18    </ion-tab>
19    <!-- Chats Tab -->
20    <ion-tab title="Chats" icon-off="ion-ios-chatboxes-outline"
                icon-on="ion-ios-chatboxes" href="#/chats">
21       <ion-nav-view name="tab-chats"></ion-nav-view>
22    </ion-tab>
23    <!-- Account Tab -->
24    <ion-tab title="Account" icon-off="ion-ios-gear-outline"
                icon-on="ion-ios-gear" href="#/account">
25       <ion-nav-view name="tab-account"></ion-nav-view>
26    </ion-tab>
27 </ion-tabs>
28 <!--模板-->
29 <script id="templates/tab-dash.html" type="text/ng-template">
30    <ion-view view-title="Dashboard">
31       <ion-content class="calm-bg"></ion-content>
32    </ion-view>
33 </script>
34 <script id="templates/tab-chats.html" type="text/ng-template">
35    <ion-view view-title="Chats">
36       <ion-content  class="royal-bg"></ion-content>
37    </ion-view>
38 </script>
39 <script id="templates/tab-account.html" type="text/ng-template">
40    <ion-view view-title="Account">
41       <ion-content class="energized-bg"></ion-content>
42    </ion-view>
43 </script>
44 </body>
45 <script type="text/javascript">
46   var app=angular.module('starter',['ionic']);
47  app.config(function($stateProvider, $urlRouterProvider) {
48      $stateProvider
49            .state('dash', {
50               url: '/dash',
51               views: {
52                  'tab-dash': {
53                     templateUrl: 'templates/tab-dash.html',
54                  }
```

```
55                  }
56              })
57              .state('chats', {
58                  url: '/chats',
59                  views: {
60                      'tab-chats': {
61                          templateUrl: 'templates/tab-chats.html',
62                      }
63                  }
64              })
65              .state('account', {
66                  url: '/account',
67                  views: {
68                      'tab-account': {
69                          templateUrl: 'templates/tab-account.html',
70                      }
71                  }
72              });
73      $urlRouterProvider.otherwise('/dash');
74    });
75 </script>
76 </html>
```

上述代码中，结合 ionic 路由完成了选项卡的切换功能，其中第 14~27 行为选项卡代码，该段代码是通过修改 ionic 项目代码中 www\templates\tabs.html 的代码完成的，需要注意选项卡图标的使用。第 29~43 行定义了三个选项卡页对应加载的 HTML 模板，其中使用 view-title 属性定义的标题会显示在 ion-nav-bar 元素中。由于第 10.3.2 小节讲解过 ionic 路由的使用，这里不再赘述。使用 Chrome 浏览器访问 demo10-13.html，页面效果如图 10-36 所示。

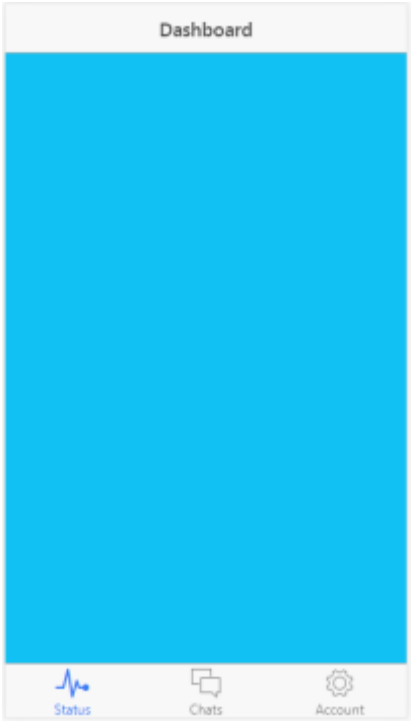

图 10-36　demo10-13.html 页面效果

在图 10-36 中，单击图标可以切换选项卡，同时，顶栏显示的内容会随之切换，如图 10-37 所示。

图 10-37　选项卡切换效果

多学一招： $ionicHistory 服务管理历史记录

浏览器的历史记录是按照用户访问顺序存储的，因此，在 Web 页面中可以利用这个特点定制返回上一页面的功能。但是在移动 App 中，有些情况是浏览器历史记录无法满足的，例

如选项卡内嵌套侧边栏，选项卡和侧边栏切换了多次后，顶部导航栏的返回按钮应该返回到上一个选项卡的视图，而不是侧边栏管理的视图，这时就需要使用 ionic 的$ionicHistory 服务对历史记录进行管理。

ionic 的导航框架会自动维护用户的访问历史栈，$ionicHistory 服务可以让选项卡间单独维护自己的浏览历史记录，这样在一个选项卡中，如果访问了多个页面，那么单击导航栏的返回按钮后，页面还将属于该选项卡。

$ionicHistory 服务中提供的一些方法需要读者了解，如表 10-21 所示。

表10-21 $ionicHistory 方法

方法	描述
viewHistory()	返回视图访问历史数据。
currentView()	返回当前视图对象。
currentHistoryId()	返回历史 ID。
currentTitle([val])	设置或读取当前视图的标题，参数 val 是可选的。无参数调用 currentTile()方法则返回当前视图的标题。
backView()	返回历史栈中前一个视图对象，如果从视图 A 导航到视图 B，那么视图 A 就是视图 B 的前一个视图对象。
backTitle()	返回历史栈中前一个视图的标题。
forwardView()	返回历史栈中的下一个视图对象。
currentStateName()	返回当前所处状态名。
goBack()	切换到历史栈中前一个视图（存在的情况下）。
clearHistory()	清空历史栈，除了当前的视图记录，将清空应用的全部访问历史。
clearCache()	清空视图缓存，将每一个 ion-nav-view 缓存的视图都清空，包括移除 DOM 及绑定的作用域对象。
nextViewOptions(options)	设置后续视图切换的选项，options 参数是一个 JSON 对象，目前支持的选项字段如下所示： 1）disableAnimate: true 在后续的转场中禁止动画； 2）disableBack: true 后续的视图将不能回退； 3）historyRoot: true 下一个视图将作为历史栈的根节点；

10.5 本章小结

本章首先介绍了 ionic JavaScript 组件的使用分类，分为指令组件和服务组件。然后按照功能分类，讲解了基本布局组件、导航组件和界面组件使用。

学习本章内容后，要求读者了解 ionic JavaScript 的指令组件和服务组件，掌握基本布局组件、导航组件和界面组件的使用，能够参考 API 完成书中案例代码。

【思考题】

1. 请列举 ionic JavaScript 中主要提供的三个表单输入组件。
2. 请列举 ionic 中提供的为列表项添加按钮功能的指令，并简要描述其作用。