

第二十讲 ionic 动态组件 \$ionicModal \$ionicActionSheet \$ionicPopup \$ionicPopover \$ionicLoading \$ionicBackdrop

学习要点：

- 1.模态对话框 : \$ionicModal
- 2.上拉菜单 : \$ionicActionSheet
- 3.弹出框 : \$ionicPopup
- 4.浮动框 : \$ionicPopover
- 5.载入指示器 : \$ionicLoading
- 6.背景幕 : \$ionicBackdrop

主讲教师：(树根)

合作网站: www.phonegap100.com (PhoneGap 中文网)

合作网站: www.itying.com (IT 营)

1.模态对话框 : \$ionicModal

模态对话框常用来供用户进行选择或编辑，在模态对话框关闭之前，其他 的用户交互行为被阻止。

在 ionic 中使用模态对话框有三个步骤：

1.声明对话框模板

使用 ion-modal-view 指令声明对话框模板，对话框模板通常置入 script 内以构造内联模板：

```
<script id="a.html" type="text/ng-template">
  <ion-modal-view>
    <!--对话框内容-->
  </ion-modal-view>
</script>
```

2.创建对话框对象

服务\$ionicModal 有两个方法用来创建对话框对象：

fromTemplate(templateString,options) - 使用字符串模板
fromTemplateUrl(templateUrl,options) - 使用内联模板

这两个方法返回的都是一个对话框对象。

3.操作对话框对象

对象框对象有以下方法用于显示、隐藏或删除对话框：

`show()` - 显示对话框
`hide()` - 隐藏对话框
`remove()` - 移除对话框
`isShown()` - 对话框是否可视？

2.上拉菜单 : `$ionicActionSheet`

上拉菜单是一个自屏幕底部向上滑出的菜单，通常用来让用户做出选择。

ionic 的上拉菜单由三种按钮组成，点击任何按钮都自动关闭上拉菜单：

取消按钮 - 取消按钮总是位于菜单的底部，用户点击该按钮将关闭。一个上拉菜单 最多有一个取消按钮。

危险选项按钮 - 危险选项按钮文字被标红以明显提示。一个上拉菜单最多有一个 危险选项按钮。

自定义按钮 - 用户定义的任意数量的按钮。

在 ionic 中使用上拉菜单需要遵循以下步骤：

1.定义上拉菜单选项

使用一个 JSON 对象定义上拉菜单选项，包括以下字段：

titleText - 上拉菜单的标题文本

buttons - 自定义按钮数组。每个按钮需要一个描述对象，其 **text** 字段用于按钮显示

cancelText - 取消按钮的文本。如果不设置此字段，则上拉菜单中不出现取消按钮

destructiveText - 危险选项按钮的文本。如果不设置此字段，则上拉菜单中不出现危险选项按钮

buttonClicked - 自定义按钮的回调函数，当用户点击时触发

cancel - 取消按钮回调函数，当用户点击时触发

destructiveButtonClicked - 危险选项按钮的回调函数，当用户点击时触发

cancelOnStateChange - 当切换到新的视图时是否关闭此上拉菜单。默认为 `true`

cssClass - 附加的 CSS 样式类名称

2. 创建上拉菜单

`$ionicActionSheet` 服务的 `show()` 方法用来创建上拉菜单，返回一个函数，调用该 返回函数可以关闭此菜单。

```
angular.module("ezApp", ["ionic"])
.controller("ezCtrl", function($scope, $ionicActionSheet, $timeout) {

    // Triggered on a button click, or some other target
    $scope.show = function() {

        // Show the action sheet
        var hideSheet = $ionicActionSheet.show({
            titleText: "操作当前文章",
            buttons: [
                { text: "<b>分享</b>文章" },
                { text: "移动到..." }
            ],
            buttonClicked: function(index) {
                return true;
            },
            cancelText: "取消",
            cancel: function() {
                // add cancel code..
            },
            destructiveText: "删除",
            destructiveButtonClicked: function(){
            }
        });

        // For example's sake, hide the sheet after two seconds
        $timeout(function() {
            // hideSheet();
        }, 2000);

    };
});
```

3. 弹出框 : `$ionicPopup`

弹出框通常用于提醒、警告等，在用户响应之前其他交互行为不能继续。与模态对话框覆盖整个屏幕空间不同，弹出框通常仅占据一部分屏幕空间。

在 **ionic** 中，使用 **\$ionicPopup** 服务管理弹出框：

```
$ionicPopup.show(options)
.then(function(){
    //这个函数在弹出框关闭时被调用
});
```

show()方法返回的是一个 **promise** 对象，当弹出框关闭后，该对象被解析，这意味着 **then()** 方法指定的参数函数此时将被调用。

show()方法的参数 **options** 是一个 **JSON** 对象，可以包括以下字段：

- title** - 弹出框标题文本
- subTitle** - 弹出框副标题文本
- template** - 弹出框内容的字符串模板
- templateUrl** - 弹出框内容的内联模板 URL
- scope** - 要关联的作用域对象
- buttons** - 自定义按钮数组。按钮总是被置于弹出框底部
- cssClass** - 附加的 CSS 样式类

简化的特定弹出框

除了 **show()**方法，**\$ionicPopup** 还针对一些特定场景提供了简化的方法，这些方法不需要自定义按钮，只需要设置 **title** 和 **template** 即可：

1. **alert(options)** - 警告弹出框，仅包含一个按钮供关闭弹出框

option 属性：

```
{
title: "", // String. 弹窗的标题。
subTitle: "", // String (可选)。弹窗的子标题。
template: "", // String (可选)。放在弹窗 body 内的 html 模板。
templateUrl: "", // String (可选)。放在弹窗 body 内的 html 模板的 URL。
okText: "", // String (默认: 'OK')。OK 按钮的文字。
okType: "", // String (默认: 'button-positive')。OK 按钮的类型。
}
```

2.confirm(options) - 确认弹出框，包含一个取消按钮和一个确认按钮

option 属性:

```
{  
title: "", // String. 弹窗标题。  
subTitle: "", // String (可选)。弹窗的副标题。  
template: "", // String (可选)。放在弹窗 body 内的 html 模板。  
templateUrl: "", // String (可选)。放在弹窗 body 内的一个 html 模板的 URL。  
cancelText: "", // String (默认: 'Cancel')。一个取消按钮的文字。  
cancelType: "", // String (默认: 'button-default')。取消按钮的类型。  
okText: "", // String (默认: 'OK')。OK 按钮的文字。  
okType: "", // String (默认: 'button-positive')。OK 按钮的类型。  
}
```

3.prompt(options) - 输入提示弹出框，包含一个文本输入框、一个取消按钮和一个确认按钮

option 属性:

```
{  
title: "", // String. 弹窗的标题。  
subTitle: "", // String (可选)。弹窗的副标题。  
template: "", // String (可选)。放在弹窗 body 内的 html 模板。  
templateUrl: "", // String (可选)。放在弹窗 body 内的 html 模板的 URL。  
inputType: // String (默认: 'text')。input 的类型。  
inputPlaceholder: // String (默认: "")。input 的 placeholder。  
cancelText: // String (默认: 'Cancel')。取消按钮的文字。  
cancelType: // String (默认: 'button-default')。取消按钮的类型。  
okText: // String (默认: 'OK')。OK 按钮的文字。  
okType: // String (默认: 'button-positive')。  
}
```

4.浮动框 : \$ionicPopover

如果需要从脚本中控制列表元素，可以使用\$ionicListDelegate 服务：

浮动框通常用以非侵入的方式提供当前视图的额外信息。

在 ionic 中使用浮动框的几个步骤：

1.声明模板

需要首先利用 ion-popover-view 指令声明一个模板内容：

```
<ion-popover-view>
  <!--模板内容-->
</ion-popover-view>
```

2.创建浮动框对象

使用\$ion-popover 服务的以下方法创建浮动框对象:

fromTemplate(templateString,options) - 使用模板字符串构造浮动框
fromTemplateurl(templateUrl,options) - 使用内联模板构造浮动框

注意: 这两个方法返回的都是 promise 对象, 在浮动框对象被构造 成功后得到解析, 这时可以获取浮动框对象:

```
$ionicPopover.fromTemplate(...)
.then(function(popover){
  //popover 参数是浮动框对象
});
```

3.操作浮动框对象

浮动框对象提供以下方法:

show() - 显示浮动框
hide() - 关闭浮动框
remove() - 移除浮动框
isShown() - 浮动框是否处于显示状态?

5.载入指示器 : \$ionicLoading

当进行耗时的操作时, 可以使用载入指示器提示用户操作进行中, 并暂时阻止交互。 载入指示器通常会叠加一个半透明的幕布层以便阻止用户的交互。

在 ionic 中, 使用\$ionicLoading 服务操作载入指示器:

show(options) - 显示载入指示器
hide() - 隐藏载入指示器

显示参数

`show()`方法的 `options` 参数是一个 JSON 对象，可以包含如下字段：

- `template` - 模板字符串
- `templateUrl` - 内联模板的 Url
- `scope` - 要绑定的作用域对象
- `noBackdrop` - 是否隐藏背景幕
- `hideOnStateChange` - 当切换到新的视图时，是否隐藏载入指示器
- `delay` - 显示载入指示器之前要延迟的时间，以毫秒为单位，默认为 0，即不延迟
- `duration` - 载入指示器持续时间，以毫秒为单位。时间到后载入指示器自动隐藏。默认情况下，载入指示器保持显示状态，知道显示的调用 `hide()` 方法

参数配置服务 : `$ionicLoadingConfig`

如果要在多处都使用载入指示器，统一对 `options` 参数进行配置是一个更好的方法，这样 在应用时直接调用 `show()` 方法而不必传递参数了。这通过定义一个 `constant provider` 来实现：

```
angular.module("ezApp", ["ionic"])
.constant("$ionicLoadingConfig", {
  template : "default loading template ..."
})
```

`$ionicLoading` 服务会通过注入器查找这个常量，如果存在就使用其值作为参数进行显示。

6.背景幕 : `$ionicBackdrop`

在浮动框、载入指示器中我们已经接触过背景幕。它是一个覆盖全屏的半透明图层，用来 阻止用户的交互行为。

我们可以使用 `$ionicBackdrop` 服务单独地使用背景幕：

- `retain()` - 保持背景幕
- `release()` - 释放背景幕

为什么不是 `show()` 和 `hide()` 呢？

在 UI 中可能有多个指令/元素都使用背景幕，为每个指令都创建单独的背景幕是不明智的。

事实上, `$ionicBackdrop` 服务在 DOM 中只保留有一个背景幕。每次当使用 `retain()` 方法时, 只是给背景幕加一次锁, `release()` 方法只是给背景幕解一次锁。如果 `retain()` 被调用三次, 背景幕将一直显示, 直到 `release()` 也被调用三次后才隐藏。

感谢收看本次教程

本教程由 **phonegap 中文网(phonegap100.com)**

www.itying.com 提供

我是主讲老师: 树根

我的邮箱: **phonegap100@qq.com**