ionic 模态窗口 ionic 平台

ionic 导航

ion-nav-view

当用户在你的app中浏览时，ionic能够检测到浏览历史。通过检测浏览历史，实现向左或向右滑动时可以正确转换视图。

采用AngularUI路由器模块等应用程序接口可以分为不同的$state(状态)。Angular的核心为路由服务，URLs可以用来控制视图。

AngularUI路由提供了一个更强大的状态管理，即状态可以被命名，嵌套， 以及合并视图，允许一个以上模板呈现在同一个页面。

此外，每个状态无需绑定到一个URL，并且数据可以更灵活地推送到每个状态。

以下实例中，我们将创建一个应用程序中包含不同状态的导航视图。

我们的标记中选择ionNavView作为顶层指令。显示一个页眉栏我们用 ionNavBar 指令通过导航更新。

接下来，我们需要设置我们的将渲染的状态值。

```
var app = angular.module('myApp', ['ionic']);
app.config(function($stateProvider) {
  $stateProvider
  .state('index', {
    url: '/',
    templateUrl: 'home.html'
  })
  .state('music', {
    url: '/music',
    templateUrl: 'music.html'
  });
});
```

再打开应用，$stateProvider 会查询url, 看是否匹配 index 状态值, 再加载index.html到<ion-nav-view>。

页面加载都是通过URLs配置的。在Angular中创建模板最一个简单的方式就是直接将他放到html模板文件中并且用<script type="text/ng-template"> 包含。

所以这也是一种方式将Home.html加入到我们的APP中来：

```
<script id="home" type="text/ng-template">
  <!--  ion-view 标题将显示在 navbar 中 -->
  <ion-view view-title="Home">
    <ion-content ng-controller="HomeCtrl">
      <!-- The content of the page -->
      <a href="#/music">Go to music page!</a>
    </ion-content>
  </ion-view>
</script>
```

尝试一下 »

这是一个很好的方法，因为模板会很快的加载并被缓存就不同通过网络再去加载。

缓存

通常情况下，视图都被缓存了能提升性能。当跳出视图时，他的元素被保留在Dom中，并且它的作用域也从 $watch 中移除。
当我们跳到一个已经被缓存了的视图，视图会被激活，它的作用域被重新连接上，Dom中也保存了他的元素。这也允许保持以前的视图滚动位置。

缓存也可以通过很多方式来开启和关闭的。默认Ionic将最大缓存页面数为10个，并且这并不是唯一可以定制的，应用程序可以显式状态来设置视图应不应该被缓存。

注意，因为我们是缓存这些视图，我们没有破坏作用域。相反, 它的作用域也从 $watch 中移除。

因为接下来的观看作用域并没有被摧毁和重建，控制器也没被再次加载。如果app/controller需要知道什么时候进入或离开一个视图，再视图事件从 ionView 作用内发出, 例如 $ionicView.enter， 可能是有用的。

全局禁用缓存

$ionicConfigProvider 可以用于设置最大允许缓存的视图数量，通过设置为0来禁用所有缓存。

```
$ionicConfigProvider.views.maxCache(0);
```

在STATE PROVIDER中禁用缓存

```
$stateProvider.state('myState', {
   cache: false,
   url : '/myUrl',
   templateUrl : 'my-template.html'
})
```

通过属性禁用缓存

```
<ion-view cache-view="false" view-title="My Title!">
  ...
</ion-view>
```

AngularUI 路由

请访问 AngularUI Router's docs 了解更多。

API

属性	类型	详情
name *(可选)*	字符串	一个视图的名字。这个名字应该是在相同的状态下其他视图中唯一的。你可以在不同的状态中有相同名称的视图。欲了解详细信息，查看ui-router的 ui-view 文档。

ion-view

隶属于ionNavView。 一个内容的容器，用于展示视图或导航栏信息。

下面是一个带有"我的页面"标题的导航栏载入页面的例子。

```
<ion-nav-bar></ion-nav-bar>
<ion-nav-view class="slide-left-right">
  <ion-view title="我的页面">
    <ion-content>
      你好!
    </ion-content>
  </ion-view>
</ion-nav-view>
```

API

属性	类型	详情
title *(可选)*	字符串	显示在父ionNavBar的标题。
hide-back-button *(可选)*	布尔值	默认情况下，是否在父ionNavBar隐藏后退按钮。
hide-nav-bar *(可选)*	布尔值	默认情况下，是否隐藏父ionNavBar。

ion-nav-bar

创建一个顶部工具栏，当程序状态改变时更新。

用法

```
<body ng-app="starter">
  <!-- 当我们浏览时，导航栏会随之更新。 -->
  <ion-nav-bar class="bar-positive nav-title-slide-ios7">
  </ion-nav-bar>

  <!-- 初始化时渲染视图模板 -->
  <ion-nav-view></ion-nav-view>
</body>
```

API

属性	类型	详情

属性	类型	详情
delegate-handle *(可选)*	字符串	该句柄用`$ionicNavBarDelegate`标识此导航栏。
align-title *(可选)*	字符串	导航栏标题对齐的位置。可用： 'left', 'right', 'center'。 默认为 'center'。

ion-nav-buttons

隶属于ionNavView

在ionView内的ionNavBar上用ionNavButtons设置按钮。

你设置的任何按钮都将被放置在导航栏的相应位置，当用户离开父视图时会被销毁。

用法

```
<ion-nav-bar>
</ion-nav-bar>
<ion-nav-view>
  <ion-view>
    <ion-nav-buttons side="left">
      <button class="button" ng-click="doSomething()">
        我是一个在导航栏左侧的按钮!
      </button>
    </ion-nav-buttons>
    <ion-content>
      这里是一些内容!
    </ion-content>
  </ion-view>
</ion-nav-view>
```

API

属性	类型	详情
side	字符串	在父`ionNavBar`中按钮放置的位置。 可用: 'left' 或 'right'。

ion-nav-back-button

在一个ionNavBar中创建一个按钮。

当用户在当前导航能够后退时，将显示后退按钮。

用法

默认按钮动作:

```
<ion-nav-bar>
  <ion-nav-back-button class="button-clear">
    <i class="ion-arrow-left-c"></i> 后退
  </ion-nav-back-button>
</ion-nav-bar>
```

自定义点击动作，用 $ionicNavBarDelegate:

```
<ion-nav-bar ng-controller="MyCtrl">
  <ion-nav-back-button class="button-clear"
    ng-click="canGoBack && goBack()">
    <i class="ion-arrow-left-c"></i> 后退
  </ion-nav-back-button>
</ion-nav-bar>
```

```
function MyCtrl($scope, $ionicNavBarDelegate) {
  $scope.goBack = function() {
    $ionicNavBarDelegate.back();
  };
}
```

在后退按钮上显示上一个标题，也用$ionicNavBarDelegate。

```
<ion-nav-bar ng-controller="MyCtrl">
  <ion-nav-back-button class="button-icon">
    <i class="icon ion-arrow-left-c"></i>{{getPreviousTitle() || 'Back'}}
  </ion-nav-back-button>
</ion-nav-bar>
```

```
function MyCtrl($scope, $ionicNavBarDelegate) {
  $scope.getPreviousTitle = function() {
    return $ionicNavBarDelegate.getPreviousTitle();
  };
}
```

---

nav-clear

nav-clear一个当点击视图上的元素时用到的属性指令，比如一个 <a href> 或者一个 <button ui-sref>。

当点击时，nav-clear将会导致给定的元素，禁止下一个视图的转换。这个指令很有用，比如，侧栏菜单内的链接。

用法

下面是一个侧栏菜单内添加了nav-clear指令的一个链接。点击该链接将禁用视图间的任何动画。

```
<a nav-clear menu-close href="#/home" class="item">首页</a>
```

ion-nav-title

ion-nav-title 用于设置 ion-view 模板中的标题。

用法

```
<ion-nav-bar>
</ion-nav-bar>
<ion-nav-view>
  <ion-view>
    <ion-nav-title>
      <img src="logo.svg">
    </ion-nav-title>
    <ion-content>
      Some super content here!
    </ion-content>
  </ion-view>
</ion-nav-view>
```

nav-transition

设置使用的过渡类型，可以是：ios, android, 和 none。

用法

```
<a nav-transition="none" href="#/home">Home</a>
```

nav-direction

设置导航视图中过渡动画的方向，可以是forward, back, enter, exit, swap。

用法

```
<a nav-direction="forward" href="#/home">Home</a>
```

$ionicNavBarDelegate

授权控制 ion-nav-bar 指令。

用法

```
<body ng-controller="MyCtrl">
  <ion-nav-bar>
```

```
      <button ng-click="setNavTitle('banana')">
        Set title to banana!
      </button>
    </ion-nav-bar>
  </body>
```

```
function MyCtrl($scope, $ionicNavBarDelegate) {
  $scope.setNavTitle = function(title) {
    $ionicNavBarDelegate.title(title);
  }
}
```

方法

```
align([direction])
```

在浏览历史中后退。

参数	类型	详情
event *(可选)*	DOMEvent	事件对象（如来自点击事件）

```
align([direction])
```

带有按钮的标题对齐到指定的方向。

参数	类型	详情
direction *(可选)*	字符串	标题文字对齐的方向。可用: 'left', 'right', 'center'。 默认: 'center'。

返回值: 布尔值， 后退按钮是否显示。

```
showBar(show)
```

设置或获取 ion-nav-bar 是否显示。

参数	类型	详情
show	布尔值	导航栏是否显示。

返回值: 布尔值， 导航栏是否显示。

```
showBackButton([show])
```

设置或获取 ion-nav-back-button 是否显示。

参数	类型	详情
show *(可选)*	布尔值	是否显示后退按钮

```
title(title)
```

为ion-nav-bar设置标题。

参数	类型	详情
title	字符串	显示新标题。

$ionicHistory

$ionicHistory 用于跟踪用户在 app 内的浏览记录。

方法

```
viewHistory()
```

用于查看历史记录。

```
currentView()
```

app 的当前视图。

```
currentHistoryId()
```

历史堆栈的 ID，是当前视图的父类容器。

```
currentTitle([val])
```

获取或设置当前视图的标题。

```
backView()
```

返回上次浏览的视图。

```
backTitle()
```

获取上次浏览的视图的标题。

```
forwardView()
```

返回历史堆栈中当前视图的上一个视图。

```
currentStateName()
```

返回当前状态名。

```
goBack([backCount])
```

app 回退视图，如果回退的视图存在。

← ionic 模态窗口 ionic 平台 →