AngularJS ui-router (嵌套路由)

介绍

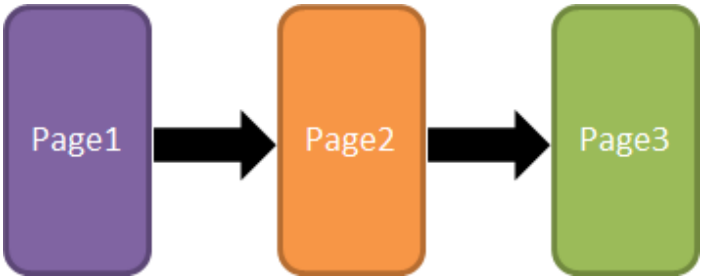
AngularJS 嵌套路由：这是我针对同一个主题（ui-router）的第二篇文章. 如果你对第一篇文章感兴趣的话，可以访问 这里. 好了，让我们继续吧，来看看嵌套的ui-router状态是怎么回事. ui-router和同属AngularJS框架一部分的ng-route一样强大. ui-router提供了让我们可以做路由嵌套和视图命名的特性. 我们将在示例中看到ui-router中存在的所有类型.

回到顶部

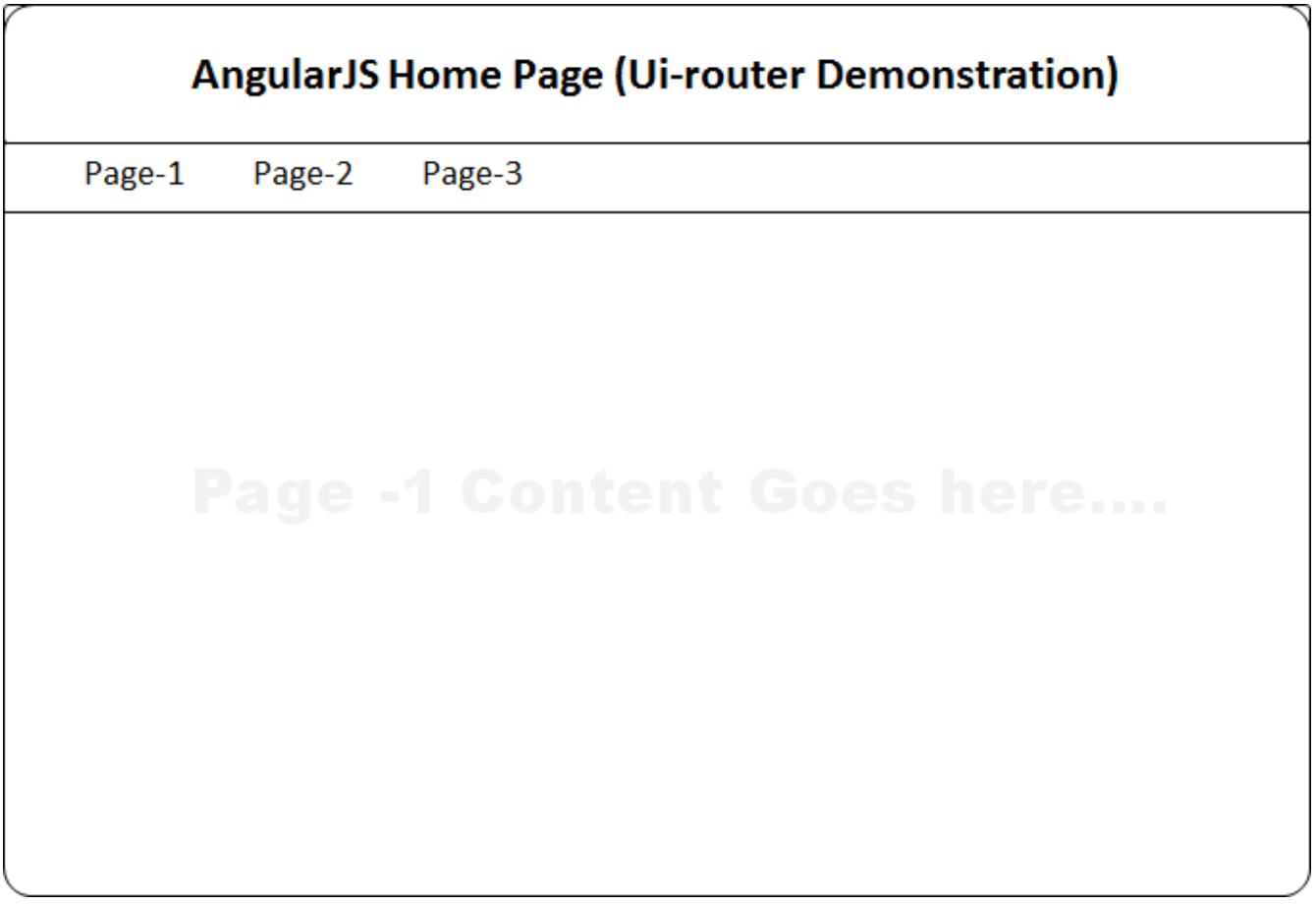
背景

引述我之前那篇文章开头给出的使用ui-router框架实现的简单路由，基于我们的业务需求，需要有不同类型的导航，一般像那种从一个页面到另外一个页面的导航非常的普通。但请想象一下在某些情况下，你需要在一个主页中有tab页或者菜单可以点击打开相应的页面.

好吧，让我们来看看一个典型的导航..



注意，这个导航我们已经在之前的文章中见过。针对现在的主题我们将看到该导航嵌套进视图中的形式.



根据上的界面设计，我们计划该页面能从一个页面导航到另外一个页面，当点击page-1时，我们将在下面显示page-1的内容，点击其它导航菜单也会有类似的效果. 我们希望这能够用一种很直接的方式被处理. 让我们开始写代码吧.

回到顶部

实战

针对该需求我们使用AngularJS框架来创建简单的html和JavaScript页面. 我们将创建3个HTML页面和一个JavaScript脚本文件。

一开始我们创建一个空的web应用程序，并加入三个HTML页面。如下所示. 这些页面都是片段视图，它们会在导航过程中展示。我们还要为了能展示应用程序的Tab，创建另外一个叫做PageTab.html的页面.

因此我们需要创建以下文件：

1. Page1.html
2. Page2.html

3. Page3.html

4. PageTab.html

注意：我们使用的是AngularJS 1.2，当我写这篇文章的时候，Angular 1.3已经发布了。

Page1.html

创建如下的html页面：

```
<div>
     <div>
         <h1>Page 1 content goes here...</h1>
     </div>
</div>
```

Page2.html

创建如下的html页面：

```
<div>
     <div>
         <h1>Page 2 content goes here...</h1>
     </div>
</div>
```

Page3.html

创建如下的html页面：

```
<div>
     <div>
         <h1>Page 3 content goes here...</h1>
     </div>
</div>
```

创建如下的html页面：

PageTab.html

创建如下的html页面：

```
<div>
     <div>
         <span style="width:100px">Page-1</span>
         <span style="width:100px">Page-2</span>
         <span style="width:100px">Page-3</span>
     </div>
</div>
```

这将会使页面文本处于侧边，哎呀，我忘了添加当用户将鼠标悬停在文本上的时候的超链接了。让我们这样做：

```
<div>
     <div>
         <span style="width:100px"><a href="">Page-1</a></span>
         <span style="width:100px"><a href="">Page-2</a></span>
         <span style="width:100px"><a href="">Page-3</a></span>
     </div>
</div>
```

我们没有指向任何超链接，只是为了把链接放在href中，实际上这是一种获取url的解决方法。

注意，到目前为止，我们还没有插入任何AngularJS路由或者其它任何框架。目前我们只是创建了一些页面片段，我们需要一个占位或者说父页面来装下这些东西. 让我们把这个页面叫做 Main.html.

现在我们就来创建它.

Main.html

用如下内容创建这个html页面.

```
<!DOCTYPE html>
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
    <title></title>
    <script src="Scripts/angular.js"></script>
    <script src="Scripts/angular-ui-router.js"></script>
    <script src="App.js"></script>

</head>
<body data-ng-app="myApp">
    <h1>AngularJS Home Page (Ui-router Demonstration)</h1>
    <div data-ui-view=""></div>
</body>
<html>
```

我们需要在主页中做一些事情, (i) 我们需要引入AngularJS框架 (ii) 我们需要引入ui-router框架. (iii) 引入AngularJS文件 App.js (之后我会谈到这个) (iv) 第四件事情就是让主页内容展示出来，然后显示出它里面的页面.

现在，让我们看一下App.JS文件的内容，我们声明了AngularJS模块和路由配置。当页面加载的时候我们会在Main.html中显示PageTab.html的内容。代码如下：

App.js

```
var myApp = angular.module("myApp", [&apos;ui.router&apos;]);

myApp.config(function ($stateProvider, $urlRouterProvider) {

     $urlRouterProvider.when("", "/PageTab");

     $stateProvider
        .state("PageTab", {
            url: "/PageTab",
            templateUrl: "PageTab.html"
        })
        .state("PageTab.Page1", {
            url:"/Page1",
            templateUrl: "Page-1.html"
        })
        .state("PageTab.Page2", {
            url:"/Page2",
            templateUrl: "Page-2.html"
        })
        .state("PageTab.Page3", {
            url:"/Page3",
            templateUrl: "Page3.html"
        });
});
```

我们一步步地来解释这做了什么。

Line-1: 第一行，声明AngularJS模块, 并把ui-router传入AngularJS主模块，所有的结合起来我们就得到了Angular模块。

```
var myApp = angular.module("myApp", [&apos;ui.router&apos;]);
```

这里叫做App模块，这将告诉HTML页面这是一个AngularJS作用的页面，它的内容由AngularJS引擎来解释。

代码行-2:这一行声明了把 $stateProvider 和 $urlRouteProvider 路由引擎作为函数参数传入，这样我们就可以为这个应用程序配置路由了.

```
myApp.config(function ($stateProvider, $urlRouterProvider) {
```

代码行-3: 好，那这一行做什么的呢，如果没有路由引擎能匹配当前的导航状态，那它就会默认将路径路由至 PageTab.html, 这个页面就是状态名称被声明的地方. 只要理解了这个，那它就像switch case语句中的default选项.

```
$urlRouterProvider.when("", "/PageTab");
```

语句块-1: 这一行定义了会在main.html页面第一个显示出来的状态，作为页面被加载好以后第一个被使用的路由.

```
$stateProvider
        .state("PageTab", {
            url: "/PageTab",
            templateUrl: "PageTab.html"
        })
```

这就向母版页的子页面，应用程序会首先加载这个main.html页面。

语句块-2: 现在，就由这一行来定义页面, 但是等一等，这里有点不同，我们之前为上面的状态名称加上了前缀，并且使用点“.“号进行了分隔. 这里很关键，它会告诉路由引擎我们在这里定义的是子页面/嵌入页面/嵌入(sub page / nested page / nested)状态的page/route.

```
.state("PageTab.Page1", {
            url:"/Page1",
            templateUrl: "Page-1.html"
        })
```

它将会在 "PageTab.html" 页面里面显示出来，那么它是什么意思呢. 想象一下当我们想要在母版页中管理所有的页面时，我们就会想要一个叫做”ui-view“的占位标记, 因此我们现在把PageTab.html叫做一个母版页，因为它会把我们需要在PageTab.html中用”ui-view“ 声明好的其它页面都管理起来. 现在让我们来修改一下它.

PageTab.html

```
<div>
     <div>
         <span style="width:100px"><a href="">Page-1</a></span>
         <span style="width:100px"><a href="">Page-2</a></span>
         <span style="width:100px"><a href="">Page-3</a></span>
     </div>
     <div>
          <div ui-view=""/>
     </div>
</div>
```

好了，再来下面一行..

```
<div>
    <div ui-view=""/>
</div>
```

也就是说 PageTab.html 将对装下所有的子页面.

现在一切就绪了。OK，可是现在谁来告诉程序应该显示哪个页面呢. 这就是我们要在路由引擎里面配置的东西，如下所示.

```
.state("PageTab.Page2", {
            url:"/Page2",
            templateUrl: "Page2.html"
        })
```

Page2.html 将会在被叫做PageTab的状态中展示，它就是 PageTab.html.

Ok, 但是我们还落下啥事没做，这事就是当我们在 Page-1 或者 Page-2 再或者 Page-3 菜单上点击的时候需要页面在占位标记那里显示出来，是不 ？

还真是把那一块给忘啦，我们还没有为路由和这种逻辑建立起联系, 想象一下如果那是href的话，就意味着我们可以指定将会锚向页面里面的ID名称, 如果它是简单的html本地引用就是这样，但我们则需要按照需求显示不同的页面.

关键的地方在这里. (ui-sref) 我们需要再一次修改 PageTab.html，如下所示.,

```
<div>
     <div>
         <span style="width:100px" ui-sref=".Page1"><a href="">Page-1</a>
         <span style="width:100px" ui-sref=".Page2"><a href="">Page-2</a>
         <span style="width:100px" ui-sref=".Page3"><a href="">Page-3</a>
     </div>
     <div>
          <div ui-view=""/>
     </div>
</div>
```

注意，只是上面高亮的部分发生了改变 ， 这里我们只是简单的将App.js中定义的状态同tab中定义的对应文本进行了关联. 当我们使用点符号对它进行了声明，程序就会认为页面时ui-view中的子页面或者说嵌入页面，它们就是路由配置中需要被展示的页面.

现在，我们要看看目前为止我们讨论过的那些页面的内容了.

Main.html

```
<!DOCTYPE html>
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
    <title></title>
    <script src="Scripts/angular.js"></script>
    <script src="Scripts/angular-ui-router.js"></script>
    <script src="App.js"></script>

</head>
<body data-ng-app="myApp">
    <h1>AngularJS Home Page (Ui-router Demonstration)</h1>
    <div data-ui-view=""></div>
</body>
<html>
```

PageTab.html

```
<div>
     <div>
         <span style="width:100px" ui-sref=".Page1"><a href="">Page-1</a>
         <span style="width:100px" ui-sref=".Page2"><a href="">Page-2</a>
         <span style="width:100px" ui-sref=".Page3"><a href="">Page-3</a>
     </div>
```

```
        <div>
            <div ui-view=""/>
        </div>
    </div>
```

Page1.html

```
<div>
    <div>
        <h1>Page 1 content goes here...</h1>
    </div>
</div>
```

Page2.html

```
<div>
    <div>
        <h1>Page 1 content goes here...</h1>
    </div>
</div>
```

Page2.html

```
<div>
    <div>
        <h1>Page 1 content goes here...</h1>
    </div>
</div>
```

App.js

```
var myApp = angular.module("myApp", [&apos;ui.router&apos;]);

myApp.config(function ($stateProvider, $urlRouterProvider) {

    $urlRouterProvider.when("", "/PageTab");

    $stateProvider
        .state("PageTab", {
            url: "/PageTab",
            templateUrl: "PageTab.html"
        })
        .state("PageTab.Page1", {
            url:"/Page1",
            templateUrl: "Page1.html"
        })
        .state("PageTab.Page2", {
            url:"/Page2",
            templateUrl: "Page2.html"
        })
        .state("PageTab.Page3", {
            url:"/Page3",
            templateUrl: "Page3.html"
        });
});
```

一切OK，现在让我们把这个应用程序运行起来吧.

本文地址：http://www.oschina.net/translate/angularjs-ui-router-nested-routes

原文地址：http://www.codeproject.com/Articles/842880/AngularJS-ui-router-nested-routes

相关资讯 — 更多

- 6个强大的AngularJS扩展应用
- 客户端JavaScript框架的五大痛点
- 让你的 Node.js 应用跑得更快的 10 个技巧
- 10个惊人的JavaScript框架
- 客户端 JavaScript 的 5 个弊端
- 你是否应该使用一个Javascript MVC框架?
- AngularJS 、Backbone.js 和 Ember.js 的比较
- 谷歌发布 AngularJS 1.0，允许扩展HTML语法
- 拥抱AngularJS
- AngularJS资源大集合
- 10 个实用的免费AngularJS开发资源
- AngularJS 1.3.0 正式发布，超光速发展!
- 15个非常有用Angular.js工具
- 25个超有用的 AngularJS Web 开发工具
- AngularJS 为什么成功了?
- 利用AngularJS构建应用的 5 个最好框架
- AngularJS准备好投入企业应用了吗?
- 针对Web开发人员的14 个实用的 AngularJS 工具
- AngularJS 1.4.0 RC1 发布
- 现在就开始使用AngularJS的三个重要原因

相关文档 — 更多

- AngularJS 理论与实战.ppt
- AngularJS 的介绍与研究.pptx
- AngularJS API 参考手册.chm
- AngularJS学习教程.pdf
- AngularJS 中文版.pdf
- AngularJS 中文版.pdf
- AngularJS 教程指南.pdf
- AngularJS 权威教程.pdf
- AngularJS 理论与实战.ppt
- AngularJS中文版.pdf
- AngularJS 进阶实践.pptx
- AngularJS 教程.pdf
- Cheat Sheet AngularJS.pdf
- AngularJS 速查表.pdf
- AngularJS in 60 Minutes.pdf
- AngularJS 入门教程.pdf
- AngularJS 速查表.pdf
- AngularJS 入门教程.pdf
- AngularJS 学习笔记.pdf
- AngularJS 学习笔记.pdf

内容信息
0.0
(已有0个人评价)

0%

0%

0%

0%

0%
收藏：0人 发布时间：2014-11-25 10:05:51

经验标签
AngularJS
同类热门经验

- 一个AngularJS的数据表格：Angular UI Grid
  23452次浏览
- AngularJS的输入框自动补全：Angucomplete
  6362次浏览
- AngularJS 2 学习教程：ng-book 2
  814次浏览
- Angular.js的对对话框：ngDialog
  19185次浏览
- Angular UI Tree 组件
  17711次浏览
- 使用 AngularJS 开发 2048 游戏
  17465次浏览

相关经验

- Angular2 路由模块简介
  0人评
- 实用的AngularJS资源集合：angular-education
  0人评
- AngularJS与RequireJS集成方案
  0人评
- 在 AngularJS 应用中通过 JSON 文件来设置状态
  0人评
- 在线动态几何编辑器：GeometryEditor
  0人评
- 移动开发框架：Ionic Framework

0人评

- AngularJS - 下一个大框架
  1人评
- Web App的零框架解决方案
  0人评
- AngularAdmin - AngularJS 管理 UI
  0人评
- AngularJS的UI框架：Suave UI
  0人评
- 基于 angularjs 的 UI 组件：WebUI4Angular
  0人评
- 纯 AngularJS 实现的 Bootstrap 组件：UI Bootstrap
  0人评
- React工具、资源、视频和实用内容集合：Awesome React
  0人评
- AngularJS 为 Semantic UI 设计的原生指令集：angular-semantic-ui
  0人评

相关讨论 - 更多

- 程序猿的鄙视链
- 成为高级程序员的 10 个步骤
- [淘宝玉伯]说说全栈工程师
- ★★杭州招聘：架构师、开发(云计算、技术框架)、前端架构等
- 程序员到高级程序员，只需要10个步骤！
- 玩转Android---UI篇 ZoomControls放大缩小图片
- 前端 MVVM 框架KnockOut.JS深入浅出

联系我们 - 问题反馈