用MontageJS创建一个简单的Reddit客户端

大多数的数据驱动网页应用需要显示一组数据，表现形式各有不同。比如常见的例子：一个员工列表里面显示一组的员工姓名和电话号码，一个相册里面显示一组的图片和标题，一个聊天室里显示一组的消息并且包括发送者和时间戳。虽然上面的例子数据显示出来样式都不一样，但是它们都有统一的模式：重复显示一组元素

在MontageJS应用中，一组重复的数据可以用内置的Repetition组件来显示。这个组件能够让应用按照数据的顺序重复地用同一段HTML一个一个地显示出来。在MontageJS开发过程中使用Repetition组件来显示数据是一个重要的表现方式，功能跟传统模板语言中的循环表达式类似。典型的应用场景就是显示一组动态的用户界面数据，比如动态创建的列表。

这个教程向你展示如何用Repetition组件构建一个简单的reddit客户端。这个应用包括两个列表(如图1)：一个新闻列表（左侧）和一个新闻目录列表（右侧）。当你选择一个新闻目录之后，应用通过reddit API获取当前选中目录的新闻列表。新闻的显示包括标题、创建者和评分数。

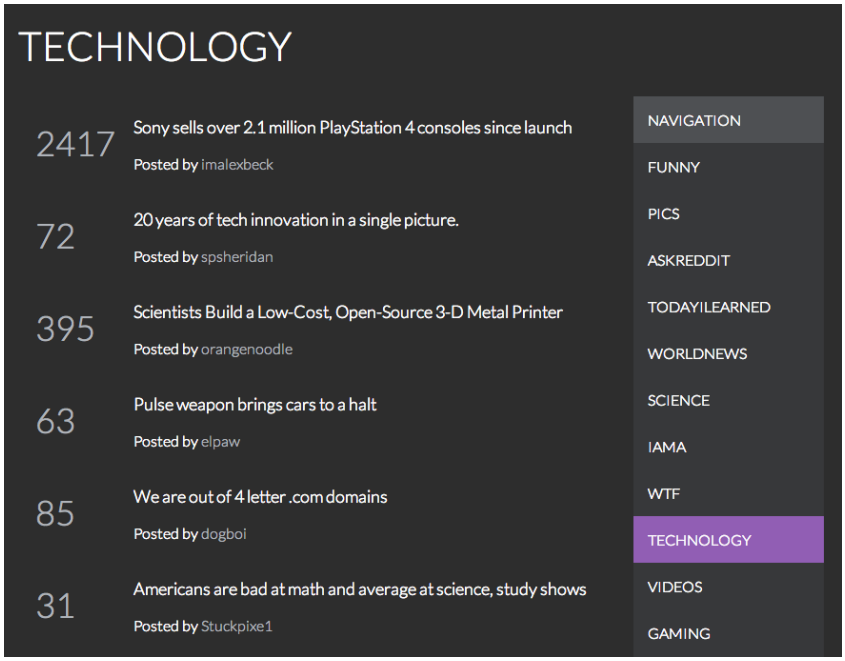

图 1. 一个用MontageJS构建的reddit简单客户端

查看示例

需要的基础知识

要完成这个教程，你需要熟悉MontageJS的基本概念和开发流程。如果你还不熟悉MontageJS框架，你可能需要从快速入门学习如何使用MontageJS组件。

开始构建应用

在这个教程中，你不需要重新创建一个MontageJS项目。你可以使用MFiddle，这是一个MontageJS组件的在线编辑器：在这个教程的每一步后面都有一个指向MFiddle的链接，你可以在这个MFiddle中查看应用在当前步骤的状态和源码。

> 备注：如果你想从一个新的项目开始一步一步地构建应用，你需要首先创建一个

> 备注：如果你想从一个新的项目开始一步一步地构建应用，你需要首先创建一个MontageJS项目（详细步骤在快速入门教程中）。接下来依照推荐方案，你需要创建一个RedditClient组件，然后在项目的Main组件中引用它。（我们已经提到过，Main是一个应用的主界面组件。就像一个网站的首页或者一个单页面应用的开始界面：它可以包含任意数量的子组件来绘制和实现应用的功能。）

在这个教程中，我们首先展示如何使用真实的数据构建新闻目录导航栏，然后展示如何显示热门新闻列表。

构建导航栏

在这个reddit客户端应用中包括两个用户界面元素：一个新闻目录（导航栏）列表和一个热门新闻列表。在这个教程中，我们首先开始创建新闻目录导航栏。

显示一个简单的重复列表

新闻目录列表严格的来讲就是一组重复的预先定义好的HTML元素（AKA模板）。要显示这个列表，你需要编辑你的HTML文件和声明。

1. 在HTML模板文件中添加下面两个元素:
    - 一个无序列表.
    - 列表中的一行.

    ```
    <ul data-montage-id="items">
        <li data-montxage-id="item"></li>
    </ul>
    ```

2. 在模板的声明中添加两个对象，然后把对象与模板中的DOM元素连接起来。在这个例子中:
    - `rep` 使用Repetition组件并且跟 `ul` 连接在一起；组件的 `content` 属性是用来定义需要Repetition重复绘制的数据。
    - `item` 使用Text组件并且跟 `li` 元素连接在一起。它的 `content` 属性定义当前行的值。

    ```
    {
        "owner": {
            "properties": {
                "element": {"#": "component"}
            }
        },
        "rep": {
            "prototype": "montage/ui/repetition.reel",
            "properties": {
                "element": { "#": "items" },
                "content": [ 1, 2, 3, 4, 5 ]
            }
        },
        "item": {
            "prototype": "montage/ui/text.reel",
            "properties": {
                "element": { "#": "item" },
                "value": "I am a list item."
            }
        }
    }
    ```

到现在为止，应用显示一列一共五个元素都是相同的值：I am a list item。 被嵌套在重复组件中的li元素会依次循环重复组件的content数据。（想象重复组件就是一个for循环：每

组件中的li元素会依次循环重复组件的content数据。（想象重复组件就是一个for循环：每一个重复组件里面的元素就是循环里面的一个实例。）

绑定列表子元素到循环的当前值

下一步，修改子元素对象让每个列表子元素能够显示数组中的值。Repetition组件有个叫做objectAtCurrentIteration的特殊属性，通过它可以获取到数组的当前值。为了让每一个列表子元素能够显示对应的数字，就需要把列表子元素对象的value绑定到重复组件的objectAtCurrentIteration属性。

> **备注:** MontageJS使用功能反应绑定（FRB）来完成用户界面与数据模型之间的同步。FRB是一种声明式的语言，可以绑定属性和集合查询结果，让它们保持数据的同步。

```
"item": {
    "prototype": "montage/ui/text.reel",
    "properties": {
        "element": { "#": "item" },
    },
    "bindings": {
        "value": {"<-": "@rep.objectAtCurrentIteration"}
    }
}
```

在这个例子中，重复组件的数据是一个静态的数字数组。在实际应用中，你可以在组件中声明一个数据模型属性，然后绑定到这个属性上面。

注意数组的内容可以是一个复杂的对象，不是仅仅像上面的列子那样只可以是简单的数字。在重复组件中的子元素也可以绑定到数据对象的一个属性。任何对数组数据的更改将会自动地更新到用户界面。

绑定**Repetition**到组件的属性

要绑定一个重复组件到当前组件的一个属性，你需要在当前组件的JS文件中添加一些JavaScript代码。 在这个例子中使用一个叫做 `subs` 的属性，它是一个包含新闻目录数据的数组。这个数组中的每个数据对象有两个属性： `display_name` 和 `url` 。

- `display_name` 是要显示在应用的新闻目录（导航下拉框）组件中的文本。
- `url` 是每条数据各自对应的数据地址。

```
var Component = require("montage/ui/component").Component;

exports.Owner = Component.specialize({
    subs: {
        value: [
            { display_name: "HTML", url: "/r/html5" },
            { display_name: "Programming", url: "/r/programming"
 },
            { display_name: "Coding", url: "/r/coding" },
            { display_name: "Comp Sci", url: "/r/compsci" },
            { display_name: "Web Dev", url: "/r/webdev" },
            { display_name: "Startups", url: "/r/startups" }
        ]
    }
});
```

下一步，你需要修改模板定义中的绑定表达式：

- 替换rep对象的content属性，从一个静态的数组到当前组件的subs属性。
- 编辑子对象的value绑定，让它连接到当前循环实例对象的display_name属性。

```
"rep": {
    "prototype": "montage/ui/repetition.reel",
    "properties": {
        "element": {"#": "items"}
    },
    "bindings": {
        "content": {"<-": "@owner.subs"}
    }
},
"item": {
    "prototype": "montage/ui/text.reel",
    "properties": {
        "element": {"#": "item"}
    },
    "bindings": {
        "value": {"<-": "@rep.objectAtCurrentIteration.display_n
ame"}
    }
}
```

现在为止，应用会显示新闻目录列表。

在MFiddle上查看源码

选择重复组件中的元素

Repetition有对元素选择功能的内置支持。当这个功能被开启之后，用户可以通过点击重复组件中的一个项目来标识它被选中。组件有个叫做selection属性（`isSelectionEnabled`打开之后）可以用来获取当前选中的值，也可以让程序修改或者绑定selection属性的方式来设置值。组件同时也自动地设置当前被选中项目的CSS值，这样就可以很容易地修改选中的样式。

设置isSelectionEnabled属性

当isSelectionEnabled属性被打开之后，默认的选择方式是单选，意味着只有一个项目会被选中。当选择一个新的项目之后组件会自动清空上一个被选中的项目。为了让项目可选，修改rep对象的属性：

1. 在模板定义中，设置`isSelectionEnabled`的值为`true`：

```
"rep": {
    "prototype": "montage/ui/repetition.reel",
    "properties": {
        "element": {"#": "items"},
        "isSelectionEnabled": true
    },
    "bindings": {
        "content": {"<-": "@owner.subs"}
    }
}
```

2. 添加一个简单的CSS规则让选中的项目高亮：

```
.selected { color: red; }
```

现在为止，点击选择列表中的一个项目。当项目被选中之后，项目变成红色。

显示被选中项目的名字

下一步，添加一个文本标题并且用绑定的方式来显示被选中项目的名字。（在最终的应用中标题显示在新闻列表的上部）

1. 在HTML模板中，添加h1元素：

```
<div data-montage-id="component">
    <h1 data-montage-id="currentsub"></h1>
    <ul data-montage-id="items">
        <li data-montage-id="item"></li>
    </ul>
</div>
```

2. 在模板定义中，添加一个新的名字叫做 `currentsub` 的对象，并且和 `h1` 元素连接在一起。对象的 `value` 属性被绑定到被选中对象的 `display_name` 属性：

```
"currentsub": {
    "prototype": "montage/ui/text.reel",
    "properties": {
        "element": {"#": "currentsub"}
    },
    "bindings": {
        "value": {"<-": "@rep.selection.0.display_name ?? 'Pleas
e Select a Sub'"}
    }
}
```

注意在对象的 `value` 绑定表达式中的 `0` 和 `??` 的用法。

- Repetition组件是允许用户一次选择多个项目的（虽然这个列子中用户只可以单选）。为了实现多选的功能，selection返回一个数组。 `0` 告诉绑定模块使用数组中的第一个项目，也就是被选中的那个项目。（和JavaScript不一样，FRB的语法不需要用中括号来访问数组的index值。）

> **备注:** 用数组的index来值来访问第一个值只有在数据类型为数组的时候才可以。你可以用FRB的one()函数来获取第一个元素，这是更通用的一种方法。

- `??` 运算符是用来控制使用运运算符之前还是之后的值，根据左边表达式的值来判断。如果值为真，使用??运算符左边的值。如果值是null或者undefined，运算符右边的值会被应用。

在这个列子中，如果selection还没有值，??运算符使h1的值为“Please Select a Sub”。 一旦selection有值，h1的值就是被选中的项目。

在MFiddle上查看源码

用实时数据来填充应用

现在，你已经学习了如何使用Repetition组件来显示一组静态数据。其实你想要的是根据reddit API返回值显示一组真实的数据。Reddit API可以返回JSON的数据格式，这使得它在web应用中可很容易地被获取和显示。

传递数据到应用

如何把从reddit服务器获取到的数据传递到应用，以下是具体步骤：

1. 在组件的JS文件中，移除掉测试数据，让 `subs` 属性的初始化值为一个空数组（subs: { value: [] }`）
2. 添加一个 `templateDidLoad` 方法。这个方法会在组件被加载完成后被自动执行，所以这是一个下载数据的合理地方。

在 `templateDidLoad` 方法中，应用从reddit服务器获取到一个新闻目录的列表数据。在回调方法中传入一个JSON格式的数组数据，然后把它赋值给subs属性。绑定方法监听到这个变化然后更新重复组件。

```
var Component = require("montage/ui/component").Component;

exports.Owner = Component.specialize({
    templateDidLoad: {
        value: function() {
            var script = document.createElement("script");
            script.src = "http://www.reddit.com/reddits.json?jso
np=subfn";

            var component = this;
            window["subfn"] = function(jsonData) {
                component.subs = jsonData.data.children;
            };

            document.head.appendChild(script);
        }
    },

    subs: { value: [] }
});
```

在这个例子中，从远程API获取到的原始JSON数据被直接传递到应用。绑定方法按照定义好的方式让数据在应用中呈现，当数据改变之后应用会做出相应的响应。如果应用重复地从API获取数据，赋值返回值到subs属性，绑定方法会自动不断地根据数据变化来绘制界面。

使用实时数据

跟很多其它的JavaScript MVC框架不一样，MontageJS不需要开发人员把数据抽取出来然后用一个特殊的方法包装起来实现监听；而是直接使用JavaScript原生的数据结构。通常从API返回的数据跟我们之前用的测试数据结构上有点不一样。从reddit API返回的列表里面的每一个数据跟下面的相似：

```
{
    "kind": "t5",
    "data": {
        "id": "2qh33",
        "submit_text": "",
        "display_name": "funny",
        "header_img": "http://f.thumbs.redditmedia.com/CzqvfNUiQGzm
MIOw.png",
        "description_html": "...",
        "title": "funny",
        "header_title": "Brought to you by Team Coco",
        "description": "...",
        "header_size": [ 160, 64 ],
        "subscribers": 4842178,
        "name": "t5_2qh33",
        "created": 1201246556,
        "url": "/r/funny/",
        "created_utc": 1201242956,
        "subreddit_type": "public",
        "submission_type": "any"
    }
}
```

在reddit API返回的JSON数据中，在新闻列表里面的每个数据是被嵌套在一个 `data` 属

性中。为了让这些数据能够在应用中显示，你需要修改你的模板定义中关于 `item` 和 `currentsub` 两个对象的 `bindings` 方式。

- 在 `item` 中，修改绑定表达式 `@rep.objectAtCurrentIteration.display_name` 为 `@rep.objectAtCurrentIteration.data.display_name`
- 在 `currentsub` 中，修改 `@rep.selection.0.display_name ?? 'Please Select a Sub'` 为 `@rep.selection.0.data.display_name ?? 'Please Select a Sub'`

你的模板定义就像这样：

```
{
    "owner": {
        "properties": {
            "element": {"#": "component"}
        }
    },
    "rep": {
        "prototype": "montage/ui/repetition.reel",
        "properties": {
            "element": {"#": "items"},
            "isSelectionEnabled": true
        },
        "bindings": {
            "content": {"<-": "@owner.subs"}
        }
    },
    "item": {
        "prototype": "montage/ui/text.reel",
        "properties": {
            "element": {"#": "item"}
        },
        "bindings": {
            "value": {"<-": "@rep.objectAtCurrentIteration.data.dis
play_name"}
        }
    },
    "currentsub": {
        "prototype": "montage/ui/text.reel",
        "properties": {
            "element": {"#": "currentsub"}
        },
        "bindings": {
            "value": {"<-": "@rep.selection.0.data.display_name ??
'Please Select a Sub'"}
        }
    }
}
```

到现在为止，你的应用已经显示一列真实的新闻目录。

列表排序

为了更接近真实应用一些，这里让新闻目录根据订阅者数量降序排列。

Reddit API返回的每个新闻目录数据中有一个 `subscriber` 属性可以用作排序字段。FRB提供一个简便的排序方法让你可以根据一个指定的属性排序。

```
"rep": {
    "prototype": "montage/ui/repetition.reel",
    "properties": {
        "element": {"#": "items"},
```

```
        "isSelectionEnabled": true
    },
    "bindings": {
        "content": {"<-": "@owner.subs.sorted{-data.subscribers}"}
    }
},
```

这个 - 运算符，就是放在绑定表达式的属性名前面的，告诉FRB列表反方向排序：从高到低。在这里我们可以使用 - 运算符，因为数据都是简单的数字。你也可以使用FRB的 `reversed` 方法来替换。FRB提供很多有用的数据处理方式。如果向 `subs` 数组中添加更多的数据或者用程序的方式修改其中一个新闻目录数据的 `subscribers` ，绑定方法会自动地根据变化重新排序数据。

在MFiddle上查看源码

显示新闻列表

现在导航栏已经完成，开始实现显示选中新闻目录对应的新闻列表功能。用跟新闻目录一样的方式获取和显示新闻：从reddit API获取JSON数据然后把它放到一个重复组件中。

1. 在模板的文件中添加一些必须的HTML片段。

   在这个列子中，应用会用一个两列的表格来显示reddit新闻：第一个列显示新闻的评分数；第二列显示标题和作者。

   ```
   <div data-montage-id="component">
       <h1 data-montage-id="currentsub"></h1>
       <ul data-montage-id="items">
           <li data-montage-id="item"></li>
       </ul>
       <table data-montage-id="stories">
           <tr>
               <td data-montage-id="score"></td>
               <td>
                   <p><a data-montage-id="title"></a></p>
                   <p>Posted by <span data-montage-id="author"></sp
   an></p>
               </td>
           </tr>
       </table>
   </div>
   ```

   - 这里的 `table` 标签是应用循环显示新闻的容器。
   - 这里的 `tr` 标签，包含新闻的具体内容，每次循环都会新建一个。
   - 这里表格中的 `score` 和 `author` 两个标签都是Text组件
   - 这里的 `title` 标签，被封装成了Anchor组件。

2. Update the template's declaration with four new objects: `stories` , `title` , `author` , and `score` .

3. 现在我们的模版有了4个新的组件： `stories` , `title` , `author` , 和 `score`

   ```
   "stories": {
       "prototype": "montage/ui/repetition.reel",
       "properties": {
           "element": { "#": "stories" }
       },
       "bindings": {
           "content": { "<-": "@owner.stories" }
       }
   },
   ```

```
"title": {
    "prototype": "matte/ui/anchor.reel",
    "properties": {
        "element": { "#": "title" }
    },
    "bindings": {
        "textContent": { "<-": "@stories.objectAtCurrentIteratio
n.data.title" },
        "href": { "<-": "@stories.objectAtCurrentIteration.data.
url" }
    }
},
"author": {
    "prototype": "montage/ui/text.reel",
    "properties": {
        "element": { "#": "author" }
    },
    "bindings": {
        "value": { "<-": "@stories.objectAtCurrentIteration.data
.author" }
    }
},
"score": {
    "prototype": "montage/ui/text.reel",
    "properties": {
        "element": { "#": "score" }
    },
    "bindings": {
        "value": { "<-": "@stories.objectAtCurrentIteration.data
.score" }
    }
}
```

每当一个不同的新闻目录被选中之后应用需要显示对应的新闻列表。最好的方式是添加一个监听器，一种每当指定的属性值改变之后自动调用一个方法的机制。

1. 在组件的JS文件中，在 `templateDidLoad` 方法里面，添加一行代码监听导航栏的选择事件：

```
this.addPathChangeListener("templateObjects.rep.selection.0.data
", this, "handleSelection");
```

2. 然后，创建一个名字叫做 handleSelection 的方法。在这个方法中会加载新的新闻数据并且赋值给stories属性：

```
handleSelection: {
    value: function(selected) {
        if (selected) {
            var script = document.createElement("script");
            script.src = "http://www.reddit.com/" + selected.url
 + ".json?sort=top&t=month&jsonp=storyfn";

            var component = this;
            window["storyfn"] = function(jsonData) {
                component.stories = jsonData.data.children;
            };

            document.head.appendChild(script);
        }
    }
},

stories: { value: [] }
```

可以看到在`handleSelection`方法中的代码跟前面的获取新闻目录的代码很相似。请求一个reddit服务器的API然后把返回的JSON数据赋值给组件的`stories`属性。同时这里跟新闻目录代码里面的`subs`属性也一样，`stories`的初始值是一个空数组。

到现在为止，应用的功能部分已经完成。当用户点击一个新闻目录之后，选中的目录数据会被传递到`handleSelection`方法，然后加载这个目录的新闻并且把返回值赋给组件的`stories`属性。当`stories`属性被重新赋值之后，Repetition组件又会自动地把新闻显示在页面上。

在MFiddle上查看源码

美化应用

MontageJS组件是使用标准HTML创建创建的，这就意味着开发者可以使用强大的CSS来美化应用。这个例子中浅色的文字显示在黑色的背景上，并用到高亮紫色的文本标记列表中选中的项目。

首先在HTML中添加一些`class`属性。也可以做一些有用的微小结构变化，比如用一个p标签包装一下评分数文本，让它在一个新的段落里面显示。下面显示的就是最终的HTML结构：

```
<div data-montage-id="component">
    <h1 data-montage-id="currentsub"></h1>
    <div class="navigation">
        <div class="header">Navigation</div>
        <ul data-montage-id="items">
            <li data-montage-id="item"></li>
        </ul>
    </div>
    <table data-montage-id="stories">
        <tbody><tr>
            <td>
                <p class="score" data-montage-id="score"></p>
            </td>
            <td>
                <p class="title"><a data-montage-id="title"></a></p
>
                <p class="author">Posted by <span data-montage-id="
author"></span></p>
            </td>
        </tr></tbody>
    </table>
</div>
```

你可以自己修改你的HTML结构，也可以修改组件CSS文件中的CSS规则。为了快速方便，你可以在GitHub查看应用的样式列表。

在MFiddle上查看源码

下一步

MontageJS灵活和强大的组件结构能够减少构建应用需要的大量代码。你可以用简单、声明式的绑定来定义复杂，相互关联的功能

如果要获取更多关于reddit客户端和MontageJS应用开发的信息，请查看下面的这些资源：