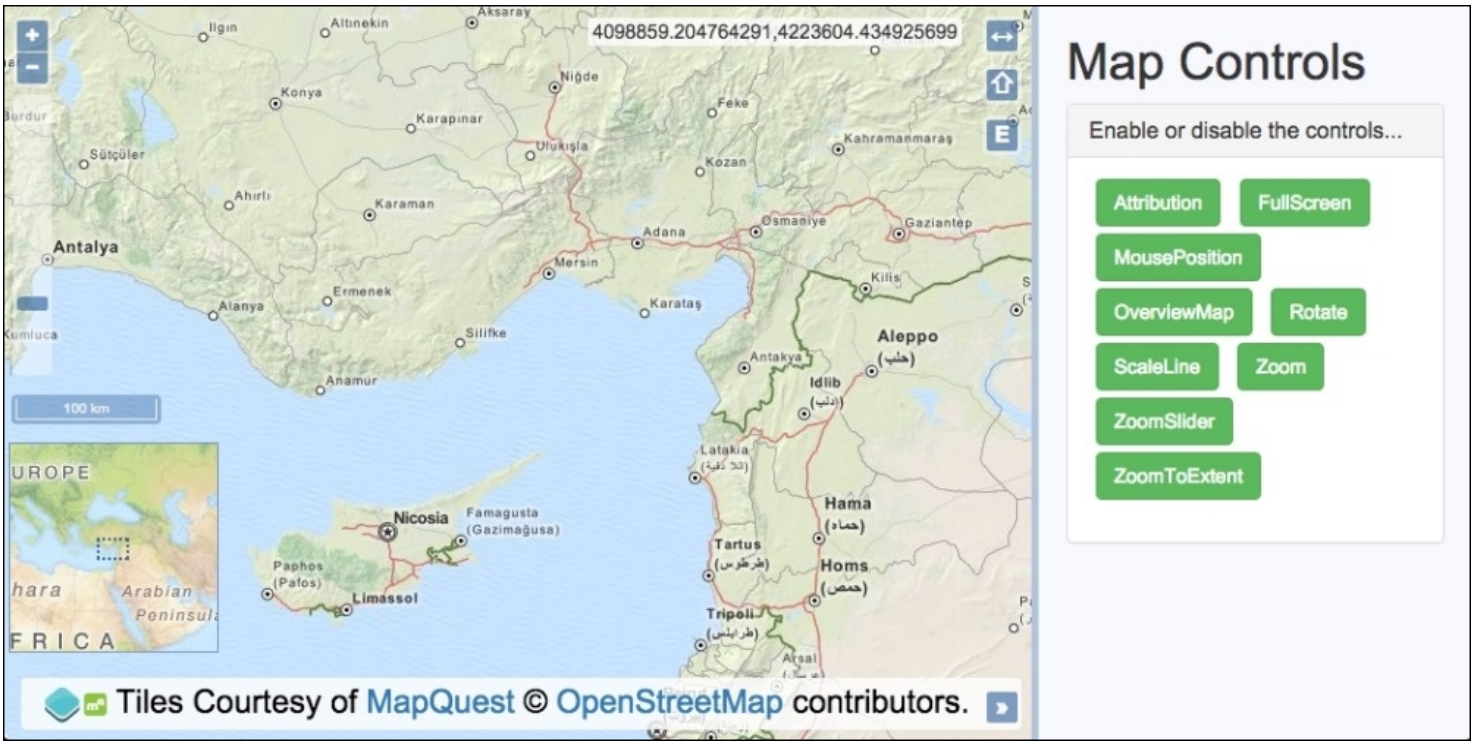
添加和删除控件

OpenLayers 提供大量控件，通常是 用于制图应用程序。

此食谱向我们展示了如何使用所有具有可视表示形式的可用控件。该列表包括 `OverviewMap` ， `ScaleLine` 和 `ZoomSlider` 控制，以及更多。

我们将在连接了所有控件的情况下初始化地图，但会提供一个侧面板，其中包含每个控件的按钮，以便用户可以根据需要切换控件。可以在中找到源代码 `ch05/ch05-adding-removing-controls` 。这是最终结果的屏幕截图：



怎么做...

请按照以下说明创建包含全部OpenLayers控件的地图， 可以根据需要打开和关闭：

1. 创建一个HTML文件并添加OpenLayers依赖项和一个 `div` 元素来保存地图。特别是，添加代表每个控件的按钮列表：

复制

```
<ul id="js-buttons">
<li><button class="btn btn-success">Attribution</button></li>
<li><button class="btn btn-success">FullScreen</button></li>
<li><button class="btn btn-success">MousePosition</button></li>
<li><button class="btn btn-success">OverviewMap</button></li>
<li><button class="btn btn-success">Rotate</button></li>
<li><button class="btn btn-success">ScaleLine</button></li>
<li><button class="btn btn-success">Zoom</button></li>
<li><button class="btn btn-success">ZoomSlider</button></li>
<li><button class="btn btn-success">ZoomToExtent</button></li>
</ul>
```

2 接下来，我们创建一个自定义JavaScript文件，首先创建一个包含所有OpenLayers控件的数组：

复制

```
var controls = [
  new ol.control.Attribution({collapsed: false}),
  new ol.control.FullScreen(),
  new ol.control.MousePosition(),
  new ol.control.OverviewMap({
    collapsed: false, collapsible: false
  }),
  new ol.control.Rotate({autoHide: false}),
  new ol.control.ScaleLine(),
  new ol.control.Zoom(),
  new ol.control.ZoomSlider(),
  new ol.control.ZoomToExtent()
];
```

3 使用创建一个 `map` 实例 `view` ，一个栅格图层，以及 `controls` ：

复制

```
var map = new ol.Map({
  view: new ol.View({
    zoom: 7,
    center: [3826743, 4325724]
  }),
  target: 'js-map',
  layers: [
    new ol.layer.Tile({
      source: new ol.source.MapQuest({layer: 'osm'})
    })
  ],
  controls: controls
});
```

4 缓存一个 引用按钮列表DOM元素并创建一个 正则表达式，供以后使用：

复制

```
var buttonList = document.getElementById('js-buttons');
var controlEnabledRegex = /btn-success/;
```

5 最后，创建一个单击处理程序以对 `click` 按钮上的事件做出反应。添加或删除 `control` ，相应地：

复制

```
buttonList.addEventListener('click', function(event) {
  var element = event.target;

  if (element.nodeName === 'BUTTON') {
    if (controlEnabledRegex.test(element.className)) {
      map.getControls().forEach(function(control) {
        if (control instanceof ol.control[element.innerHTML]) {
          map.removeControl(control);
        }
      });
      element.className = element.className.replace(
        'btn-success', 'btn-default'
      );
    } else {
      controls.forEach(function(control) {
        if (control instanceof ol.control[element.innerHTML]) {
          map.addControl(control);
        }
      });
```

怎么运行的...

我们使用了CSS 框架，Bootstrap，大部分样式以及一些为了简洁起见，省略了应用程序的HTML。我们还使用了一些自定义样式，将控件放置在地图上，以使它们不会重叠或显得过于混乱。请查看随附的源代码以全面了解实现。

数组 `controls` 提供具有可视表示形式的所有可用OpenLayers控件的列表。我们为每个实例创建了一个实例 `control` ，其中一些我们已自定义，以便始终显示。我们也可以为不可见的交互控件（ `ol.interaction` ）做一些非常相似的事情，但这将增加最小的学习价值。请看一下OpenLayers上可以进行哪些交互文档（http://openlayers.org (http://openlayers.org)）。

映射实例看起来会很熟悉，因此让我们继续进行单击处理程序和其中的逻辑：

复制

```
buttonList.addEventListener('click', function(event) {
  var element = event.target;
  if (element.nodeName === 'BUTTON') {
```

为了提高效率，我们将click事件处理程序添加到 `ul` 按钮所在的整个DOM元素中。因此，在处理程序中，我们首先检查该事件是否源自button类型的元素，如果是，则按以下步骤进行：

复制

```
  if (controlEnabledRegex.test(element.className)) {
```

我们检查一下单击的按钮是否 `'btn-success'` 在 `class` 属性中包含字符串。如果是这样，则意味着该控件当前已添加到地图，但是用户希望将其删除。为此，我们创建了一个正则表达式，该正则表达式用作 `test` JavaScript方法的一部分，该正则表达式检查DOM元素的类名中是否适用字符串。

例如， `<button class="btn btn-success">Rotate</button>` 按钮包含一个 `className` 字符串， `'btn btn-success'` 因此此正则表达式将成功匹配该类名。根据此信息，我们可以确定按钮和 `control` 已启用。

复制

```
map.getControls().forEach(function(control) {
  if (control instanceof ol.control[element.innerHTML]) {
    map.removeControl(control);
  }
});
```

为了处理从地图中删除控件，我们首先通过 `getControls` 方法从地图中获取所有控件。这将返回一个 `ol.Collection` 数组，并使用该 `forEach` 方法进行循环。

当我们实例化一个新控件时，即 `new ol.control.Zoom()` ，它是 `ol.control.Zoom` 构造函数的一个实例。当我们遍历所有控件时，使用此知识，我们获取按钮（ `element.innerHTML` ）的文本（例如，等于） `'Zoom'` ，并检查循环的当前迭代是否与此控件类型匹配。例如，在这种情况下 `ol.control[element.innerHTML]` 等于 `ol.control.Zoom` 。

找到匹配项后，必须是按钮代表的控件，因此已从中删除 通过 `removeControl` 方法绘制地图。

复制

```
element.className = element.className.replace(
  'btn-success', 'btn-default'
);
```

为了保持HTML同步与地图控件，我们从更新按钮类名 `'btn btn-success'` 来 `'btn btn-default'` 通过JavaScript的替代方法。

复制

```
} else {
  controls.forEach(function(control) {
    if (control instanceof ol.control[element.innerHTML]) {
      map.addControl(control);
    }
  });
  element.className = element.className.replace(
    'btn-default', 'btn-success'
  );
}
```

相反，如果该按钮当前处于禁用状态（不具有的类别 `'btn-success'` ），则用户希望启用该控件。为了实现这一点，我们遍历包含所有控件的controls数组，而不管当前是否将它们添加到地图中。

应用相同的逻辑，以便 `ol.control[element.innerHTML]` 在此迭代中如果按钮文本与控件对象（）匹配控件实例，则将其添加到映射（ `addControl` ）。

通过更新类名字符串，按钮HTML再次与JavaScript保持同步。

也可以看看

- **地图**配方**外的“放置”控件**
- 所述**跨多个矢量层绘制新功能**的配方

上一节 (/book/web_development/9781785287756/5/ch05lvl1sec45/introduction)

下一节 (/book/web_development/9781785287756/5/ch05lvl1sec47/working-with-geolocation)
