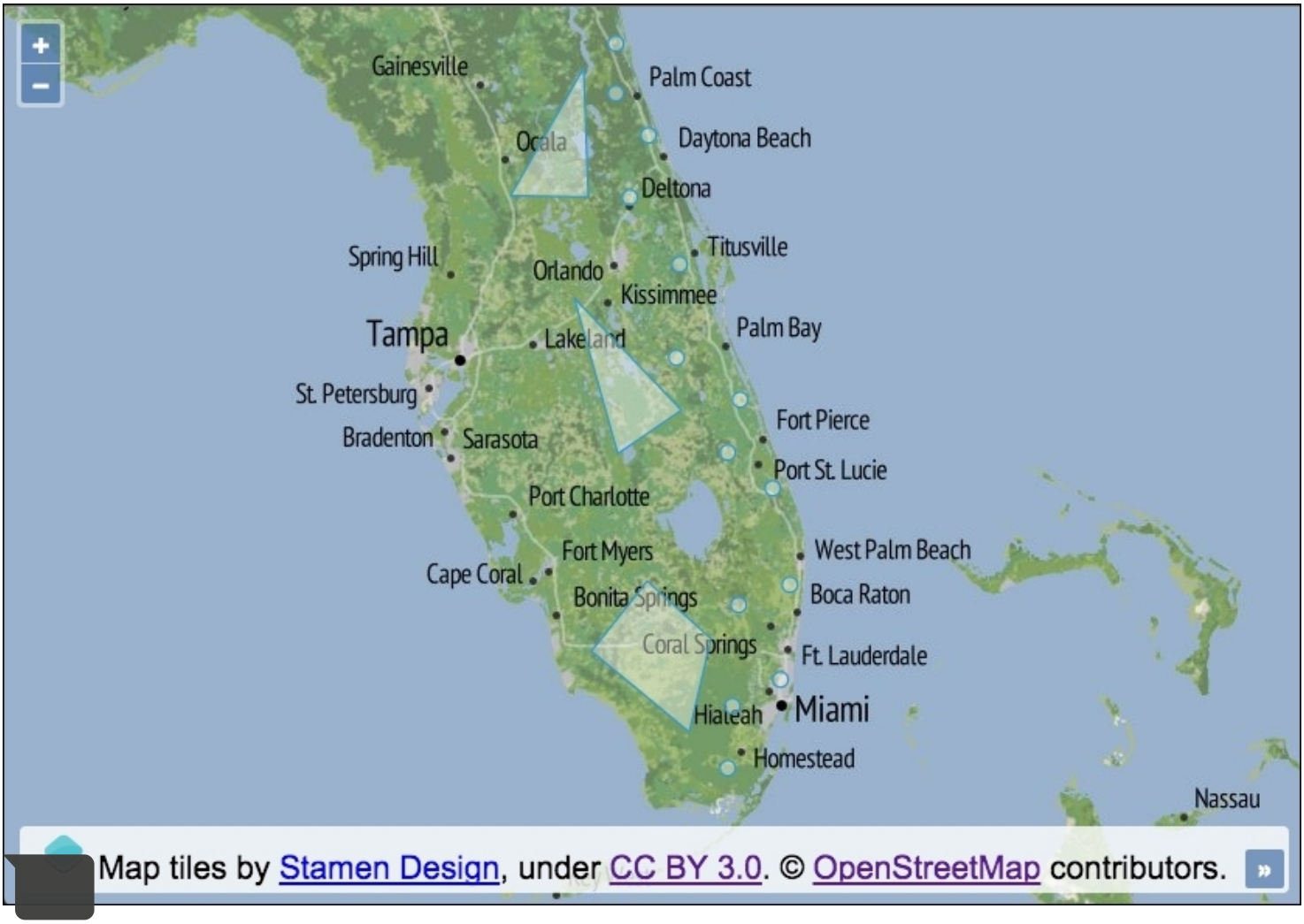
使用AJAX直接阅读功能

这个目标 配方是向我们展示如何直接与 AJAX从同一矢量层上的不同数据源请求并加载内容。

OpenLayers允许我们读取来自不同来源和来源的数据。我们已经看到我们可以连接到各种服务（例如WMS和WFS）和/或根据我们的需求自定义请求（例如提供直接指向GML文件的URL）。

我们已经看到向向量源提供URL和格式类型很方便，并且在许多情况下都可以使用。OpenLayers为我们发出AJAX请求并自动格式化响应。但是，有时我们需要对AJAX请求和响应进行更多控制，而这正是该 `loader` 功能发挥作用的地方。

我们选择使用jQuery执行AJAX请求。请求将提取两个不同格式的单独几何文件，并将它们都添加到同一矢量层。该食谱的源代码可以在中找到 `ch03/ch03-reading-features-from-ajax/` 。这是我们将在地图上整理并渲染的结果几何：



怎么做...

创建您的自定义 使用以下内容的AJAX加载程序 说明：

1. 创建一个HTML文件并添加OpenLayers依赖项和jQuery库。添加 `div` 元素以保存地图。
2. 接下来，初始化 `map` ，添加一个栅格图层，并将视口居中：

复制

```
var map = new ol.Map({
  view: new ol.View({
    zoom: 7,
    center: [-9039137, 3169996]
  }),
  target: 'js-map',
  layers: [
    new ol.layer.Tile({
      source: new ol.source.Stamen({layer: 'terrain'})
    })
  ]
});
```

3. 现在，创建一个带有源的矢量层，该源负责发出多个AJAX请求以获取两个几何文件。解析然后将返回的数据添加到向量源要素列表中：

复制

```
var vectorLayer = new ol.layer.Vector({
  source: new ol.source.Vector({
    loader: function() {
      $.ajax({
        type: 'GET',
        url: 'points.wkt',
        context: this
      }).done(function(data) {
        var format = new ol.format.WKT({splitCollection: true});
        this.addFeatures(format.readFeatures(data));
      });

      $.ajax({
        type: 'GET',
        url: 'polygons.json',
        context: this
      }).done(function(data) {
        var format = new ol.format.GeoJSON();
        this.addFeatures(format.readFeatures(data));
```

4. 添加 矢量层 到 `map` ：

复制

```
map.addLayer(vectorLayer);
```

怎么运行的...

让我们直接看一下并分解该食谱的重要部分：

复制

```
source: new ol.source.Vector({
  loader: function() {
```

我们使用 `loader` 向量源的属性来提供type的自定义函数 `ol.FeatureLoader` 。类似 `url` 属性（它需要一个函数式的 `ol.FeatureUrlFunction` ），则 `loader` 功能也被传递 `extent` ， `resolution` 以及 `projection` ，分别。我们已省略了这些内容，因为我们没有在请求中使用它们。

复制

```
$.ajax({
  type: 'GET',
  url: 'points.wkt',
  context: this
```

在jQuery的帮助下，我们建立了一个 `GET` 指向本地几何文件的HTTP 请求，该文件包含WKT格式的点。

在 `loader` 函数中， `this` JavaScript中的关键字指向矢量源。当我们从AJAX promise（ `done` ）中将返回的功能添加到源中时，我们打算使用此参考。jQuery提供了一个有用的属性 `context` ，该属性使我们能够指定 `this` 回调或Promise 中引用的值。这样可以节省我们编写临时变量或以其他方式引用向量源的麻烦。

复制

```
}).done(function(data) {
  var format = new ol.format.WKT({splitCollection: true});
  this.addFeatures(format.readFeatures(data));
});
```

一旦AJAX 请求已成功完成，我们的 `done` 承诺被解雇了。数据参数包含WKT文件的内容。

创建WKT格式对象时，我们传递 `splitCollection` 属性设置为的配置对象 `true` 。默认情况下，OpenLayers会将WKT几何图形包装到几何图形集合中，但是我们希望将每个点归为要素集合中的单个要素。对于我们的示例，这实际上只是次要的语义。

要素经过处理（ `format.readFeatures` ）并添加到矢量源（ `this.addFeatures` ）， `this` 与 `vectorLayer.getSource()` 此处等效。

复制

```
$.ajax({
  type: 'GET',
    url: 'polygons.json',
    context: this
  }).done(function(data) {
    var format = new ol.format.GeoJSON();
    this.addFeatures(format.readFeatures(data));
  });
```

该请求与上一个请求相同，尽管这次，我们获取一个包含GeoJSON格式数据的文件。要素被添加到同一矢量层。

使用该 `loader` 功能时，我们负责自行加载要素并将其添加到图层。

还有更多...

使用该 `loader` 功能，您有机会在之前过滤功能列表 将它们添加到图层。例如，如果您仅对功能列表的子集感兴趣，这将非常有用。您可以根据几何类型，大小，任意属性等进行过滤。

也可以看看

- 所述**添加GML层**配方
- 在**从WFS服务器添加功能**食谱
- 所述**添加KML层**配方

上一节 (/book/web_development/9781785287756/3/ch03lvl1sec36/using-the-cluster-strategy)

下一节 (/book/web_development/9781785287756/3/ch03lvl1sec38/creating-a-heat-map)