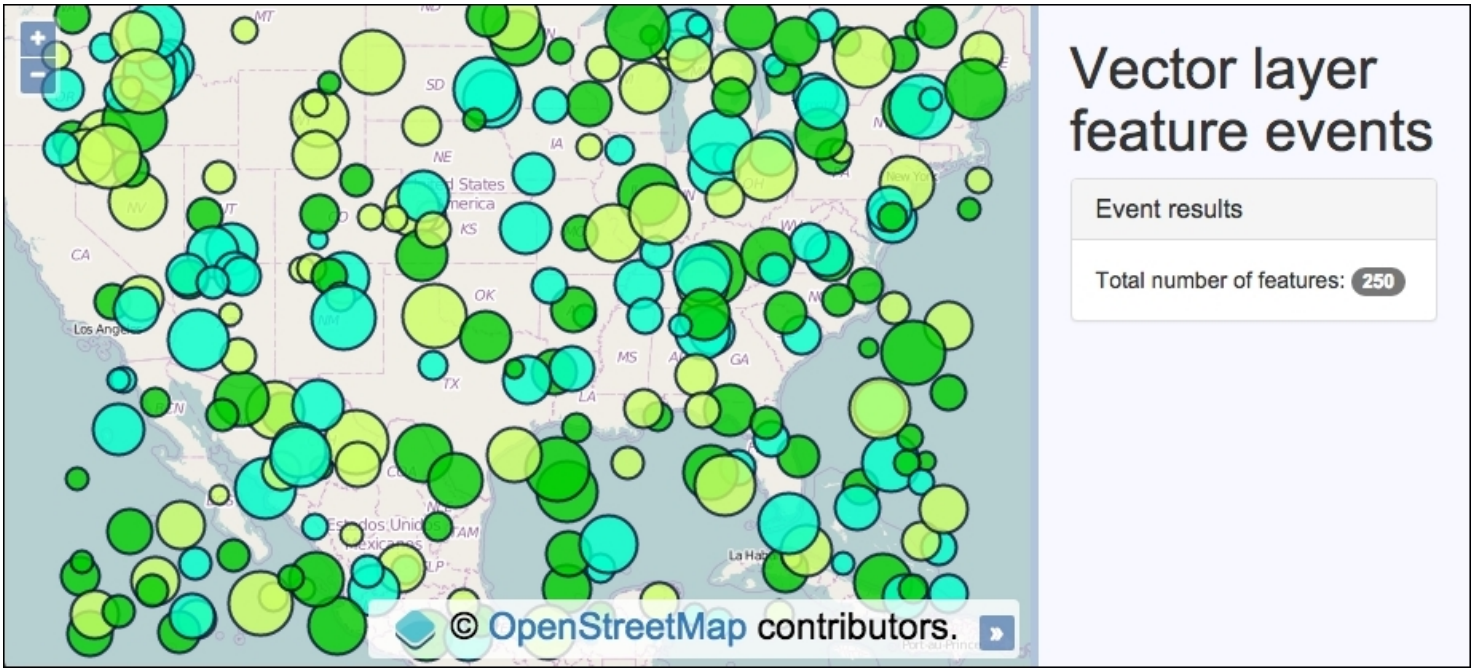
侦听矢量图层要素的事件

什么时候 在使用矢量图层时，通常会发现需要响应图层上发生的情况的情况，例如添加，修改或删除新功能时等。幸运的是，这些类型的事件可从矢量源获得，我们可以轻松地订阅它们。

本食谱的目的是向您展示侦听矢量源上的事件并使用此信息执行某些操作有多么简单。

我们将加载几何点的外部GeoJSON文件，并将根据要素属性设置填充颜色和半径的样式。我们还将跟踪矢量源图层上的要素数量，并允许用户通过单击要素来删除要素，随后更新要素计数。可以在中找到源代码 `ch04/ch04-vector-feature-events` ，这是我们的最终结果：



怎么做...

在 为了了解矢量层要素事件如何工作，请按照以下说明自行编写此食谱：

1. 创建一个HTML文件并添加OpenLayers依赖项。特别是，添加一个 `div` 元素来保存 `map` 实例以及侧面板和内容的标记：

复制

```
<div id="js-map"></div>
<div class="pane">
  <h1>Vector layer feature events</h1>
  <div class="panel panel-default">
    <h3 class="panel-title">Event results</h3>
      Total number of features:
      <span id="js-feature-count">0</span>
  </div>
</div>
```

2 使用 `map` 基础层和初始化实例 `view` ：

复制

```
var map = new ol.Map({
  view: new ol.View({
    zoom: 4,
    center: [-10703629, 2984101]
  }),
  target: 'js-map',
  layers: [
    new ol.layer.Tile({source: new ol.source.OSM()})
  ]
});
```

3 创建 向量图层并读取GeoJSON文件：

复制

```
var vectorLayer = new ol.layer.Vector({
  source: new ol.source.Vector({
    url: 'points.geojson',
    format: new ol.format.GeoJSON({
      defaultDataProjection: 'EPSG:3857'
    })
  })
});
```

4 创建地图选择交互，以便可以选择要删除的要素。还要缓存显示要素总数的DOM元素：

复制

```
var select = new ol.interaction.Select({
  condition: ol.events.condition.click,
  layers: [vectorLayer]
});
map.addInteraction(select);

select.on('select', function(event) {
  if (event.selected[0]) {
    vectorLayer.getSource().removeFeature(event.selected[0]);
    select.getFeatures().clear();
  }
});
var featureCount = document.getElementById('js-feature-count');
```

5 订阅添加功能事件并根据功能的属性为功能设置样式。也更新功能计数显示：

复制

```
vectorLayer.getSource().on('addfeature', function(event) {
  event.feature.setStyle(new ol.style.Style({
    image: new ol.style.Circle({
      fill: new ol.style.Fill({
        color: event.feature.get('colour')
      }),
      stroke: new ol.style.Stroke({
        color: [0, 13, 51, 0.8],
        width: 2
      }),
      radius: event.feature.get('size')
    })
  }));
  featureCount.innerHTML =
    vectorLayer.getSource().getFeatures().length;
});
```

6 订阅到删除功能事件并更新功能计数。通过将矢量层添加到地图来结束：

复制

```
vectorLayer.getSource().on('removefeature', function() {
    featureCount.innerHTML =
      vectorLayer.getSource().getFeatures().length;
});
map.addLayer(vectorLayer);
```

怎么运行的...

我们已经使用CSS框架Bootstrap来帮助进行侧面板样式设计。请查看本书的源代码以获取实现的完整详细信息。让我们继续OpenLayers代码：

复制

```
select.on('select', function(event) {
  if (event.selected[0]) {
    vectorLayer.getSource().removeFeature(event.selected[0]);
    select.getFeatures().clear();
  }
});
```

我们以前使用的选择交互第3章 (/book/web_development/9781785287756/3)，**工作与矢量图层**，在**卸下或利用覆盖克隆功能**配方，所以我们不会赘述。选择（单击或点击）某个功能后，该功能会立即从矢量源（ `removeFeature` ）中删除，并清除选择。此操作将导致发布矢量层要素事件，即 `removefeature` 我们订阅的事件。稍后我们将介绍该处理程序。

复制

```
vectorLayer.getSource().on('addfeature', function(event) {
  event.feature.setStyle(new ol.style.Style({
    image: new ol.style.Circle({
      fill: new ol.style.Fill({
        color: event.feature.get('colour')
      }),
      stroke: new ol.style.Stroke({
        color: [0, 13, 51, 0.8],
        width: 2
      }),
      radius: event.feature.get('size')
    })
  }));
  featureCount.innerHTML =
    vectorLayer.getSource().getFeatures().length;
});
```

我们 使用 `on` 向量来源的方法订阅 `addfeature` 事件。矢量源检索并解析了GeoJSON文件后，就会将每个要素一次添加到地图上。这为我们提供了一个在将要素添加到图层之前对其进行操作的机会。

文件中的每个几何点都有两个自定义属性，即 `color` 和 `size` 。我们使用此嵌入信息来相应地对功能进行样式设置。

事件对象包含 `event.feature` 对即将添加到图层的要素（）的引用。类型为的特征 `ol.Feature` 具有一种 `setStyle` 方法，该方法用于根据需要分别为每个特征设置样式。图像的填充颜色是从要素的 `colour` 属性（ `event.feature.get('colour')` ）导出的，半径是从要素的 `size` 属性（ `event.feature.get('size')` ）确定的。

我们在处理程序中所做的最后工作是更新新功能计数。这是使用 `getFeatures` 向量源的方法实现的，该方法返回所有特征的数组。我们可以获取新的长度（ `getFeatures().length` ），并使用该数字用 `innerHTML` DOM元素上的JavaScript方法替换HTML内容。

复制

```
vectorLayer.getSource().on('removefeature', function() {
    featureCount.innerHTML =
      vectorLayer.getSource().getFeatures().length;
});
```

值得庆幸的是， `removefeature` 事件发布后没有太多工作要做。我们需要做的就是像以前一样使用新功能更新DOM元素。

还有更多...

向量源事件在的命名空间下 `ol.source.VectorEvent` 。除了我们在此食谱中查看的两个事件之外，还有其他事件，例如 `changefeature` 和 `clear` 。分别在功能更新（例如设置新样式）和从矢量源清除所有功能时发布。

载体层（未矢量源）也出版事件，如 `change:extent` ， `change:opacity` ，和 `change:minResolution` 。在处理矢量层时，其中一些可能会变得有用。我鼓励您查看文档和/或OpenLayers源代码，以发现可以增强用户体验的可用事件。

也可以看看

- 该**造型特点基于使用symbolizers几何类型**的配方，从第6章 (/book/web_development/9781785287756/6)，**造型特点**
- 第3章 (/book/web_development/9781785287756/3)，**使用矢量层中**的**添加GML层**配方 (/book/web_development/9781785287756/3)
- 所述**创建并排侧地图比较**配方

上一节 (/book/web_development/9781785287756/4/ch04lvl1sec41/implementing-a-work-in-pr

下一节 (/book/web_development/9781785287756/4/ch04lvl1sec43/listening-for-mouse-or-touc

---