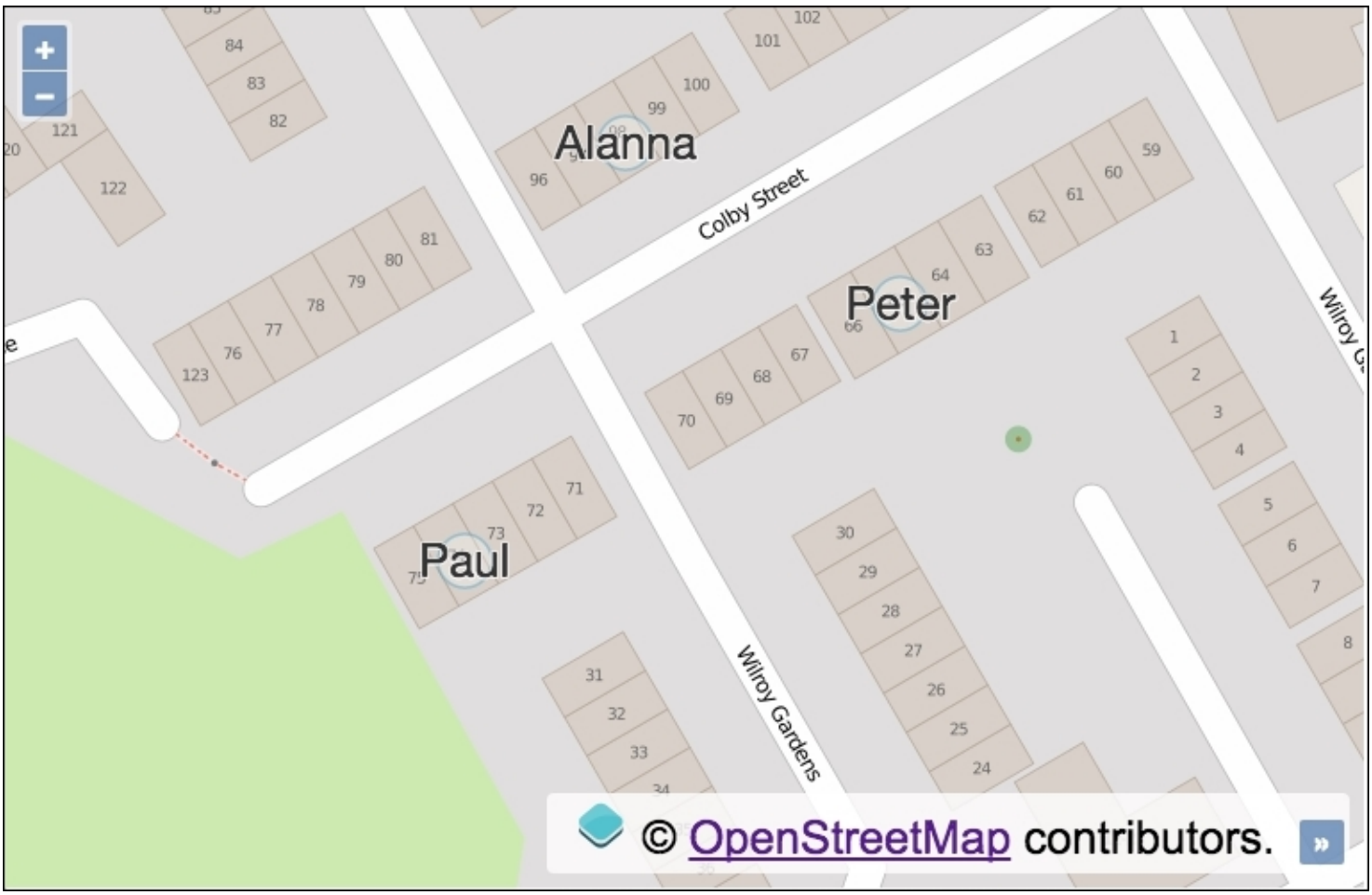
将文本标签添加到几何点

兴趣点 现实世界可以通过 各种形状（例如圆形和直线），但仅凭几何形状可能不足以用于各种目的。例如，您可能有一些表示区域的多边形，但您想将这些区域标记为区域A，区域B和区域C。或者，可能需要显示某个位置的人数，并用圆圈和包含计数的标签。

对于此配方，我们将一个人的名字分配给一个几何点。功能将放置在某些房屋中。您可以想像一下为什么这样做可能有用！可以在中找到源代码 `ch03/ch03-geometry-labels/` 。这是最终结果的屏幕截图：



怎么做...

我们将学习如何添加 使用的几何点的有用文本标签 以下步骤：

1. 创建一个具有OpenLayers依赖项的HTML文件以及一个 `div` 用于保存地图的元素。

2 创建自定义JavaScript文件和实例化 `map` , `view` 和光栅层：

复制

```
var map = new ol.Map({
  view: new ol.View({
    zoom: 19,
    center: [-161669, 6609321]
  }),
  target: 'js-map',
  layers: [
    new ol.layer.Tile({
      source: new ol.source.OSM()
    })
  ]
});
```

3 创建一个函数，该函数将用于生成带有文本标签的点要素：

复制

```
var createPoint = function(coords, resident) {
  var feature = new ol.Feature(new ol.geom.Point(coords));
  feature.set('resident', resident);
  return feature;
};
```

4 创建功能 这将用于定点样式 特征：

复制

```
var getStyle = function(feature) {
  return [
    new ol.style.Style({
      text: new ol.style.Text({
        text: feature.get('resident'),
        fill: new ol.style.Fill({
          color: '#333'
        }),
        stroke: new ol.style.Stroke({
          color: [255, 255, 255, 0.8],
          width: 2
        }),
        font: '26px "Helvetica Neue", Arial'
      }),
      image: new ol.style.Circle({
        fill: new ol.style.Fill({
          color: [255, 255, 255, 0.3]
        }),
        stroke: new ol.style.Stroke({
```

5 使用我们生成的点特征和样式函数将矢量层添加到地图中：

复制

```
map.addLayer(
  new ol.layer.Vector({
    source: new ol.source.Vector({
      features: [
        createPoint([-161705, 6609398], 'Alanna'),
        createPoint([-161659, 6609371], 'Peter'),
        createPoint([-161732, 6609328], 'Paul')
      ]
    }),
    style: getStyle
  })
);
```

怎么运行的...

此食谱的主要两个方面是如何针对功能设置自定义属性，以及如何将此标签呈现到地图上。我们将其分解如下：

复制

```
var createPoint = function(coords, resident) {
  var feature = new ol.Feature(new ol.geom.Point(coords));
  feature.set('resident', resident);
  return feature;
};
```

为方便起见，我们 由此创建点要素可重用的功能。我们的函数采用坐标数组（ `coords` ）和居民名称（ `resident` ）。这将基于传入的坐标创建点几何要素，然后将居民的名称分配给要素本身的自定义属性，即 `resident` 。该 `set` 方法从 `ol.Object` 类继承。

接下来是我们的 `getStyle` 函数，该函数采用OpenLayers要求的格式（ `ol.style.StyleFunction` ）。当OpenLayers调用它时，它将分别传递的实例 `ol.Feature` 和视图的分辨率。我们的功能将完全确定此功能的样式。为了满足OpenLayers函数的约定，我们的方法必须返回一个包含一个或多个 `ol.style.Style` 实例的数组。

复制

```
text: new ol.style.Text({
  text: feature.get('resident'),
  fill: new ol.style.Fill({
    color: '#333'
  }),
  stroke: new ol.style.Stroke({
    color: [255, 255, 255, 0.8],
    width: 2
  }),
  font: '26px "Helvetica Neue", Arial'
}),
```

在返回的数组的内联中， `ol.style.Style` 为简便起见，我们创建了一个新的实例，在前面的代码中省略了。在配置对象中，我们为 `text` 属性提供了一个值，它是的实例 `ol.style.Text` 。此构造函数的配置对象还具有一个 `text` 属性。当OpenLayers为我们传递功能时， `feature` 变量在 `getStyle` 调用时将在范围内，因此我们只需 `resident` 通过的 `get` 方法从功能本身中检索值 `ol.Object` 。

我们还提供填充 颜色，笔触颜色，宽度和字体的大小和家庭。还有许多其他属性和样式类型可用，例如 `lineDash` 笔触的属性和样式类型 `ol.style.RegularShape` 。

您会注意到颜色可以通过多种方式定义。对于 `fill` 属性，我们以十六进制格式将颜色作为字符串提供。对于 `stroke` ，我们传入 `ol.Color` 了RGBA（红色，绿色，蓝色，Alpha）格式的类型数组。

复制

```
image: new ol.style.Circle({
  fill: new ol.style.Fill({
    color: [255, 255, 255, 0.3]
  }),
  stroke: new ol.style.Stroke({
    color: [51, 153, 204, 0.4],
    width: 1.5
  }),
  radius: 15
})
```

`ol.style.Style` 实例的第二个也是最后一个属性是 `image` 。这就是在文本标签后面绘制圆圈的方法。我们使用RGBA阵列技术设置 `fill` 和 `stroke` 颜色，还提供了笔触宽度和圆半径。

复制

```
new ol.layer.Vector({
    source: new ol.source.Vector({
      features: [
        createPoint([-161705, 6609398], 'Alanna'),
        createPoint([-161659, 6609371], 'Peter'),
        createPoint([-161732, 6609328], 'Paul')
      ]
    }),
    style: getStyle
  })
```

创建矢量层后，我们将调用自定义 `createPoint` 函数3次以返回 `features` 数组内联的几何点。

我们的自定义 `getStyle` 方法已分配给 `style` 矢量层的属性。如前所述，当该层中的要素需要使用 `feature` 和 `resolution` 参数进行样式设置时，它将由OpenLayers调用。

我们探索 造型 在第6章 (/book/web_development/9781785287756/6)，**样式设置中**还有更多内容。

上一节 (/book/web_development/9781785287756/3/ch03lvl1sec33/zooming-to-the-extent-of-a

下一节 (/book/web_development/9781785287756/3/ch03lvl1sec35/adding-features-from-a-wfs