以手绘模式绘制

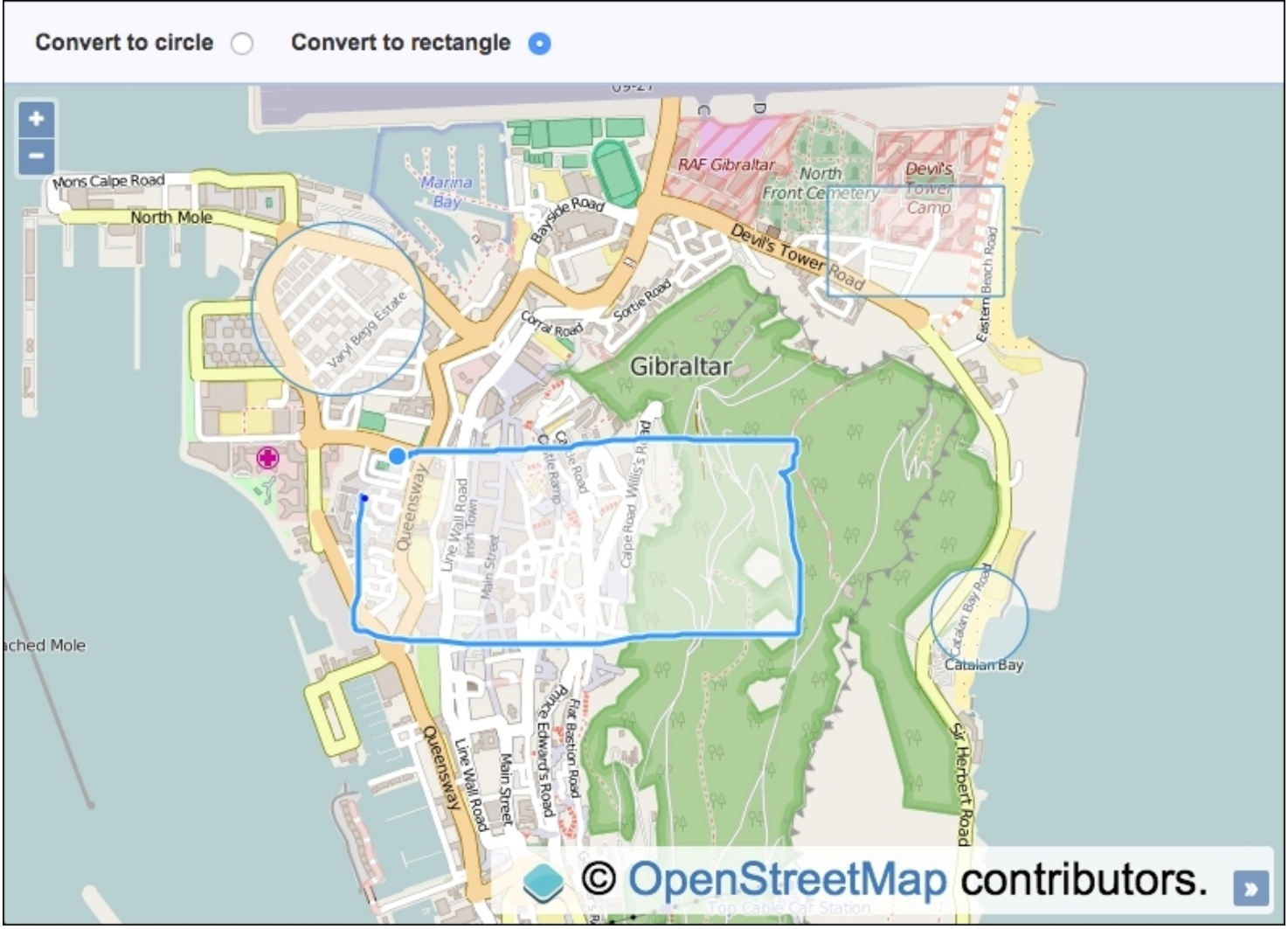
我们向您展示了如何绘制要素早在第5章 (/book/web_development/9781785287756/5)，**添加控件**在**绘图跨多个矢量层设有**配方。OpenLayers还提供了一种徒手绘制的机制，可以使用 `ol.interaction.Draw` 该类在矢量层上绘制要素。徒手模式可以定义为在地图上草绘某些东西，而没有在没有徒手模式下绘制几何图形类型时获得的定义精度和方向。

具有更大的自由度来进行绘制的灵活性在许多应用程序需求中很有用。例如，使用连续的徒手绘制流程沿着道路创建路径可能会比较麻烦，而不是每次移动时都必须单击以添加点。在徒手画模式下，当您将鼠标移到想要插入的方向时，会自动为您添加点。

OpenLayers支持多边形或直线的徒手绘制。在本食谱中，我们将以徒手模式探索绘图多边形。绘制完多边形后，我们将根据用户从无线电输入中选择的内容将其转换为外观更规则的形状（圆形或矩形）（请参阅以下屏幕截图）。

默认情况下，工程图通过按住**Shift**键，然后**按住**鼠标按钮并移动光标，可以启用徒手模式。完成后，松开**Shift**键并单击以完成绘图。我们将在食谱中使用这些默认值。

可以在中找到源代码 `ch07/ch07-freehand-drawing` ，最后得到的结果类似于以下屏幕截图：



怎么做...

1 创建一个HTML文件，并包含OpenLayers依赖项和一个 `div` 用于保存地图的元素。尤其是，请使用以下HTML作为 `radio` 输入：

复制

```
<input type="radio" name="convert" value="circle">
<input type="radio" name="convert" value="rectangle" checked>
```

2 创建一个自定义JavaScript文件，并 `map` 使用一个 `view` 实例，一个栅格图层以及一个矢量层实例化一个实例：

复制

```
var map = new ol.Map({
  view: new ol.View({
    zoom: 15, center: [-595501, 4320196]
  }), target: 'js-map',
  layers: [
    new ol.layer.Tile({source: new ol.source.OSM()}),
    new ol.layer.Vector({source: new ol.source.Vector()})
  ]
});
```

3 建立 `draw` 互动为多边形几何类型并将其添加到 `map` 实例中：

复制

```
var draw = new ol.interaction.Draw({
  source: map.getLayers().item(1).getSource(),
  type: 'Polygon'
});
map.addInteraction(draw);
```

4 订阅 `drawend` 事件并将草图转换为 `rectangle` 或 `circle` 几何：

复制

```
draw.on('drawend', function(event) {
  var feature = event.feature;
  var convertTo = document.querySelector('[type="radio"]:checked').value;

  if (convertTo === 'rectangle') {
    feature.setGeometry(
      new ol.geom.Polygon.fromExtent(
        feature.getGeometry().getExtent()
      )
    );
  } else {
    var extent = feature.getGeometry().getExtent();
    var centre = ol.extent.getCenter(extent);
    var width = ol.extent.getTopRight(extent)[0] - ol.extent.getTopLeft(extent)[0];
    var radius = width / 2;

    feature.setGeometry(
      new ol.geom.Circle(centre, radius)
    );
```

怎么运行的...

我们已经使用了 Bootstrap CSS框架可帮助设置顶部面板内容的样式，在本食谱的“ **如何做...**”部分中已将其排除在外。请查看完整图片的源代码。

大多数代码看起来与您以前的食谱相似，但是我们当然会仔细看看事件处理程序中发生的 `draw` 交互作用：

复制

```
draw.on('drawend', function(event) {
  var feature = event.feature;
  var convertTo =
      document.querySelector('[type="radio"]:checked').value;
```

写意草图完成后，将 `drawend` 发布事件，我们的先前处理程序将预订该事件。

在 `event` 对象保存参照绘制的特征，这是我们存储在一个变量，即 `feature` 。 `convertTo` 通过查询DOM以 `radio` 获取所选输入，我们将用户的转换选择存储在变量中，然后检索 `input` 元素的值。该值将为“ `rectangle` ”或“ `circle` ”。

复制

```
if (convertTo === 'rectangle') {
    feature.setGeometry(
      new ol.geom.Polygon.fromExtent(
        feature.getGeometry().getExtent()
      )
    );
```

如果用户选择了此工程图以将其转换为 `rectangle` ，我们将使用实例中的 `setGeometry` 方法来更新要素的几何形状 `ol.Feature` 。在此方法内部，我们提供了一个新的几何实例。

要根据草绘的特征范围创建新的几何实例，我们使用有用的方法 `ol.geom.Polygon.fromExtent` 。此方法采用范围的单个参数并返回的实例 `ol.geom.Polygon` ，该实例用作要素的替换几何。

复制

```
var extent = feature.getGeometry().getExtent();
var centre = ol.extent.getCenter(extent);
var width = ol.extent.getTopRight(extent)[0] -
    ol.extent.getTopLeft(extent)[0];
var radius = width / 2;

feature.setGeometry(
  new ol.geom.Circle(centre, radius)
);
```

但是，如果用户希望将草图转换为圆，则需要做更多的手工工作。

我们从检索开始 几何形状的范围与以前相同。使用 `extent` ，我们可以根据草图的意图估算出准确的圆。OpenLayers提供了许多方法，可在将扩展作为 `ol.extent` 对象的一部分使用时使用。

首先是不言自明的 `ol.extent.getCenter` 方法。 `centre` 从此方法返回的坐标也将用于定义 `centre` 圆的坐标。

然后 `width` ， `extent` 通过采用该 `ol.extent.getTopRight` 方法的最右边的点，然后使用该方法从最左边的点减去它，来计算of `ol.extent.getTopLeft` 。如果我们愿意，可以选择使用 `ol.extent.getBottomRight` 和 `ol.extent.getBottomLeft` 方法来提供相同的期望结果。

这两种方法都返回一个坐标， `ol.Coordinate` 即xy格式的类型的数组。我们仅对感兴趣 `width` ，因此我们通过访问第一个索引（）从每个数组中获取x值 `[0]` 。结果存储在 `width` 变量中，实质上为圆提供了直径。

当我们使用构造函数构造圆几何时， `ol.geom.Circle` 第二个参数需要一个 `radius` 圆，而不是直径，因此我们将 `width` 变量除以二。

就像之前一样，我们通过在特征上设置新几何来完成。

注意

如果您想实现与OpenLayers 2中的类似功能有关，请参见http://codechewing.com/library/create-circles-in-openlayers-using-freehand-mode/中 (http://codechewing.com/library/create-circles-in-openlayers-using-freehand-mode/)的教程。

也可以看看

- 该**绘图跨多个矢量层设有**配方在第5章 (/book/web_development/9781785287756/5)，**添加控件**
- 第5章 (/book/web_development/9781785287756/5)，**添加控件中的“修改功能”**配方 (/book/web_development/9781785287756/5)
- 第6章 (/book/web_development/9781785287756/6)，**样式化功能中**的**样式化交互呈现意图**配方 (/book/web_development/9781785287756/6)

上一节 (/book/web_development/9781785287756/7/ch07lvl1sec65/using-the-custom-openlaye

下一节 (/book/web_development/9781785287756/7/ch07lvl1sec67/modifying-layer-appearance