

以编程方式创建特征

正在从中加载数据 外部源不是处理矢量图层的唯一方法。想象一下一个Web映射应用程序，用户可以在其中动态创建新功能，例如着陆区，周长，感兴趣的区域等，并将其添加到具有某种样式的矢量层中。这种情况要求能够以编程方式创建和添加功能。

在本食谱中，我们将介绍一些以编程方式创建一系列特征的方法。可以在[中找到源代码](#) `ch03/ch03-creating-features/`。以下屏幕截图显示了以编程方式创建的一些功能：



怎么做...

在这里，我们将创建一些功能以编程方式，无需导入任何文件。请按照以下说明来了解如何完成此操作：

- ¹ 首先创建一个具有所需OpenLayers依赖项的新HTML文件。特别是，添加 `div` 元素来保存地图：

<div id="js-map"></div>

复制

2 创建一个空的JavaScript文件，并 map 使用背景栅格图层实例化一个：

复制

```
var map = new ol.Map({
  view: new ol.View({
    zoom: 3,
    center: [-2719935, 3385243]
  }),
  target: 'js-map',
  layers: [
    new ol.layer.Tile({
      source: new ol.source.MapQuest({layer: 'osm'})
    })
  ]
});
```

3 创建 point 和 circle 功能，如下所示：

复制

```
var point = new ol.Feature({
  geometry: new ol.geom.Point([-606604, 3228700])
});

var circle = new ol.Feature(
  new ol.geom.Circle([-391357, 4774562], 9e5)
);
```

4 创建 line 和 polygon 功能：

复制

```
var line = new ol.Feature(
  new ol.geom.LineString([
    [-371789, 6711782], [1624133, 4539747]
  ])
);
var polygon = new ol.Feature(
  new ol.geom.Polygon([[
    [606604, 4285365], [1506726, 3933143],
    [1252344, 3248267], [195678, 3248267]
  ]])
);
```

5 创建矢量层并添加 features 到这一层：

复制

```
map.addLayer(new ol.layer.Vector({
  source: new ol.source.Vector({
    features: [point, circle, line, polygon]
  })
}));
```

尽管我们已经为该配方创建了一些随机特征，但制图应用程序中的特征通常会代表现实世界中的某种现象，并具有与其相关的适当几何形状和样式。

让我们研究一下程序化要素的创建以及如何将其添加到矢量层：

复制

```
var point = new ol.Feature({
  geometry: new ol.geom.Point([-606604, 3228700])
});
```

功能是的实例 `ol.Feature` 。此构造包含了许多有用的方法，如 `clone` ， `setGeometry` ， `getStyle` ，等。创建的实例时 `ol.Feature` ，我们必须传入类型为的几何图形 `ol.geom.Geometry` 或包含属性的对象。在本食谱中，我们将演示两种变化。

对于 `point` 功能，我们传入一个配置对象。我们提供的唯一属性是 `geometry` 。还有其他可用属性，例如和 `style` ，以及使用自定义 `feature` 属性自行设置属性的方法，这些属性随getter和setter一起提供。

的 `geometry` 实例属于 `ol.geom.Point` 。本 `ol.geom` 类提供了其他多种功能类型，我们没有得到在这个配方看，如 `MultiLineString` 和 `MultiPoint` 。点 `geometry` 类型只需要一个 `ol.Coordinate` 类型数组 (**xy**坐标) 。

复制

```
var circle = new ol.Feature(
  new ol.geom.Circle([-391357, 4774562], 9e5)
);
```

该 `circle` 功能几乎与该功能具有相同的结构 `point` 。但是，这次我们没有将配置对象传递给 `ol.Feature` ，而是直接实例化的 `ol.geom.Geometry` 类型 `Circle` 。构造函数采用坐标数组和半径的第二个参数。 `9e5` 或 `9e+5` 为90万的指数表示法。

该 `circle` 几何形状也有一些有用的方法，如 `getCenter` ，和 `setRadius` 。

复制

```
var line = new ol.Feature(
  new ol.geom.LineString([
    [-371789, 6711782], [1624133, 4539747]
  ])
);
```



注意

请记住在适当的投影中表达坐标，即视图使用的坐标，或者自己翻译坐标。我们将涵盖在功能款式工作第6章 ([/book/web_development/9781785287756/6](http://book.web_development/9781785287756/6))，**造型特点**。目前，所有功能都将使用默认的OpenLayers样式呈现。

与该 `LineString` 功能的唯一明显区别是 `ol.geom.LineString` 期望有一个坐标数组。对于更高级的线串，请使用 `ol.geom.MultiLineString` 几何类型（更多信息可以在OpenLayers API文档中找到在 <http://openlayers.org/en/v3.13.0/apidoc/> (<http://openlayers.org/en/v3.13.0/apidoc/>)）。

该 `LineString` 功能还具有有用的方法，例如 `getLength` 。

复制

```
var polygon = new ol.Feature(  
  new ol.geom.Polygon([[  
    [606604, 4285365], [1506726, 3933143],  
    [1252344, 3248267], [195678, 3248267]  
  ]])  
);
```

最终特征（`Polygon` 几何类型）与 `LineString` 特征略有不同，因为它期望 `ol.Coordinate` 另一个包装数组中的一个数组中的类型数组。这是因为构造函数（`ol.geom.Polygon`）需要一个环形数组，每个环形被表示为坐标数组。理想情况下，每个环都应关闭。

多边形要素还具有有用的方法，例如 `getArea` 和 `getLinearRing` 。

复制

```
map.addLayer(new ol.layer.Vector({  
  source: new ol.source.Vector({  
    features: [point, circle, line, polygon]  
  })  
}));
```



注意

OGC的**简单功能访问**规范（<http://www.opengeospatial.org/standards/sfa> (<http://www.opengeospatial.org/standards/sfa>)) 包含了深入的描述标准的。它还包含一个UML类图，您可以在其中查看所有几何类和层次结构。

最后，我们使用向量源实例创建向量层，并将所有四个要素添加到数组中并将其传递给 `features` 属性。

我们创建的所有功能都是的子类 `ol.geom.SimpleGeometry` 。此类提供了有用的基本方法，例如 `getExtent` 和 `getFirstCoordinate` 。

所有 `getType` 要素都有一种可用于标识要素类型的方法，例如“ `Point` ”或“ `LineString` ”。

还有更多...

有时，`polygon` 功能可能代表其中有孔的区域。要创建多边形的空心部分，我们使用 `LinearRing` 几何。以下屏幕截图可以最好地说明结果：



您可以看到该多边形具有一个部分切出。为了实现这种几何形状，我们必须以稍微不同的方式创建多边形。步骤如下：

1 创建 `polygon` 几何，如下所示：

复制

```
var polygon = new ol.geom.Polygon([[
  [606604, 4285365], [1506726, 3933143],
  [1252344, 3248267], [195678, 3248267]
]]);
```

2 创建线性环并将其添加到 `polygon` 几何体：

复制

```
polygon.appendLinearRing(
  new ol.geom.LinearRing([
    [645740, 3766816], [1017529, 3786384],
    [1017529, 3532002], [626172, 3532002]
  ])
);
```

创建完成的功能，如下所示：

复制

```
var polygonFeature = new ol.Feature(polygon);
```

4 通过将多边形要素添加到矢量层来结束：

复制

```
vectorLayer.getSource().addFeature(polygonFeature);
```

我们不会进一步细分这种逻辑，因为现在我们对几何图形的创建很满意，这是不言而喻的。



注意

该 `ol.geom.LinearRing` 特征只能与 `polygon` 几何体结合使用，而不能作为独立特征使用。

也可以看看

- 🔗 在添加标记到地图配方
- 🔗 通过WKT食谱阅读和创建功能
- 🔗 该造型特点基于几何类型的配方在第6章 (/book/web_development/9781785287756/6)，**造型特点**

◀ 上一节 (/book/web_development/9781785287756/3/ch03lvl1sec27/adding-a-kml-layer)

下一节 ▶ (/book/web_development/9781785287756/3/ch03lvl1sec29/exporting-features-as-geojs)

