修改功能

正如我们在先前的食谱，“**在多个矢量层上绘制特征**”，为用户启用绘制功能非常简单。但是，如果用户需要编辑绘制的特征怎么办？也许他们想移动一些顶点或将整个特征移动到新位置。在本食谱中，我们将介绍这些类型的修改。

OpenLayers提供了 `ol.interaction.Modify` 用于移动或添加顶点的 `ol.interaction.Translate` 类以及用于移动整个要素的类。

可以在中找到源代码 `ch05/ch05-modifying-features` ，以下是完成后的屏幕截图：

怎么做...

了解如何修改 请按照以下步骤操作现有功能：

1. 创建具有OpenLayers依赖项的HTML文件并 `div` 保存地图。
2. 创建自定义的JavaScript文件，并通过实例开始 `map` 使用 `view` ，光栅片层和载体层检索从本地GeoJSON的文件的特点：

复制

```
var map = new ol.Map({
  view: new ol.View({
    zoom: 10, center: [-12035468, 4686812]
  }),
  target: 'js-map',
  layers: [
    new ol.layer.Tile({ source: new ol.source.Stamen({
      layer: 'terrain'
    })}),
    new ol.layer.Vector({
      editable: true,
      source: new ol.source.Vector({
        url: 'features.geojson',
        format: new ol.format.GeoJSON({
          defaultDataProjection: 'EPSG:3857'
        })
      })
    })
  ]
```

3 建立 `select` 互动这样就可以选择要修改的特定要素并将其添加到地图中：

复制

```
var select = new ol.interaction.Select({
  filter: function(feature, layer) {
    return layer.get('editable') &&
      /Polygon|LineString/.test(
        feature.getGeometry().getType()
      );
  },
  condition: ol.events.condition.click
});
map.addInteraction(select);
```

4 通过设置并添加修改和转换交互作用到地图来结束：

复制

```
map.addInteraction(new ol.interaction.Modify({
  features: select.getFeatures()
}));
map.addInteraction(new ol.interaction.Translate({
  features: select.getFeatures()
}));
```

怎么运行的...

如您所见，使用OpenLayers可以轻松添加这种丰富的功能。一旦用户从地图上选择了一个要素，他们就可以编辑该要素或四处移动。为了使事情更加有趣，我们在选择交互中添加了过滤器机制，该机制会 `layer` 在允许选择特征之前检查属性和几何类型。让我们仔细看看JavaScript中的情况：

复制

```
new ol.layer.Vector({
  editable: true,
  source: new ol.source.Vector({
    url: 'features.geojson',
    format: new ol.format.GeoJSON({
      defaultDataProjection: 'EPSG:3857'
    })
  })
})
```

在的实例化中 `map` ，我们设置了一个自定义属性为的矢量层，该属性 `editable` 用于标识是否可以编辑该层。我们还引入了一些GeoJSON，它将使用某些预制特征填充该图层。

复制

```
var select = new ol.interaction.Select({
  filter: function(feature, layer) {
    return layer.get('editable') &&
      /Polygon|LineString/.test(
        feature.getGeometry().getType()
      );
  },
  condition: ol.events.condition.click
});
```

该 `select` 相互作用配置了过滤功能。分配给该 `filter` 属性的函数必须是类型 `ol.interaction.SelectFilterFunction` 。此类函数分别通过 `feature` 和 `layer` 作为as参数传递，并且它必须返回true或false。如果返回true，则可以选择该特征，因此将其添加到集合中。如果返回false，则不会选择该功能，也不会将其添加到集合中。

我们的自定义过滤功能通过方法检索 `layer` 属性，并检查其结果是否为true。如果是这样，则对几何类型执行第二个条件检查。 `editable` `get`

`feature` 通过该 `getGeometry` 方法从中检索几何对象。有了几何图形后，我们可以使用一种称为的方法 `getType` ，该方法以字符串形式返回几何图形类型，例如 `Polygon` 。该结果被传递到JavaScript `test` 方法中，该方法针对正则表达式运行结果。该 `/Polygon|LineString/` 正则表达式中的任一个的字符串匹配 `Polygon` 或 `LineString` （几何类型）。 `|` 此特定正则表达式中的竖线（）表示OR逻辑条件。

如果将图层 `editable` 和图层都设置为可接受的类型之一，则return语句的值为true；否则为false。否则，这将是错误的。

换句话说，如果未将矢量层专门标记为 `editable` ，则无法修改要素。除了刚刚讨论的两种几何类型之外，它还不允许选择其他几何类型，然后再进行编辑。这演示了如何根据应用程序要求开始构造只读或读/写层。

该 `condition` 属性已设置为 `ol.events.condition.click` 避免接受 `ol.events.condition.singleClick` 该属性的默认值时出现250 ms的延迟。

复制

```
map.addInteraction(new ol.interaction.Modify({
  features: select.getFeatures()
}));
map.addInteraction(new ol.interaction.Translate({
  features: select.getFeatures()
}));
```

修改和翻译交互遵循相同的设置。的 `features` 每个属性被从通过当前选定的功能 `select` 经由交互 `select.getFeatures()` 。

虽然我们没有利用这个配方中的任何事件，则修改相互作用出版言自明 `modifyend` 和 `modifystart` 事件。平移交互会发布相似的事件，即 `translateend` 和 `translatestart` ，但也会发布一个 `translating` 事件以订阅之间的移动。

还有更多...

在OpenLayers 2中修改功能时，您还可以轻松旋转和调整它们的大小。在撰写本文时，此功能尚未直接移植到版本3中。

但是，有一个有趣的函数 `applyTransform` 从 `ol.geom.SimpleGeometry` 基类调用，该函数被大多数几何类型（例如）继承 `ol.geom.Polygon` 。此功能可用于手动变换几何图形的每个坐标，以便在适当位置修改特征。

该函数的类型为 `ol.TransformFunction` ，该类型将传递当前坐标数组，并且必须返回要使用的坐标输出数组。没有什么可以阻止您在此处进行一些创造性的转换来模仿您在OpenLayers 2中已经习惯的某些缺少的功能。但是，这是高级讨论，但是不幸的是，本书不会对此进行介绍。

也可以看看

- **跨多个矢量层**的**图形特征**配方
- 在**添加和删除控制**配方
- 第4章 (/book/web_development/9781785287756/4)，**使用事件中**的**侦听矢量层功能的事件**配方 (/book/web_development/9781785287756/4)

上一节 (/book/web_development/9781785287756/5/ch05lvl1sec49/drawing-features-across-mu

下一节 (/book/web_development/9781785287756/5/ch05lvl1sec51/measuring-distances-and-ar