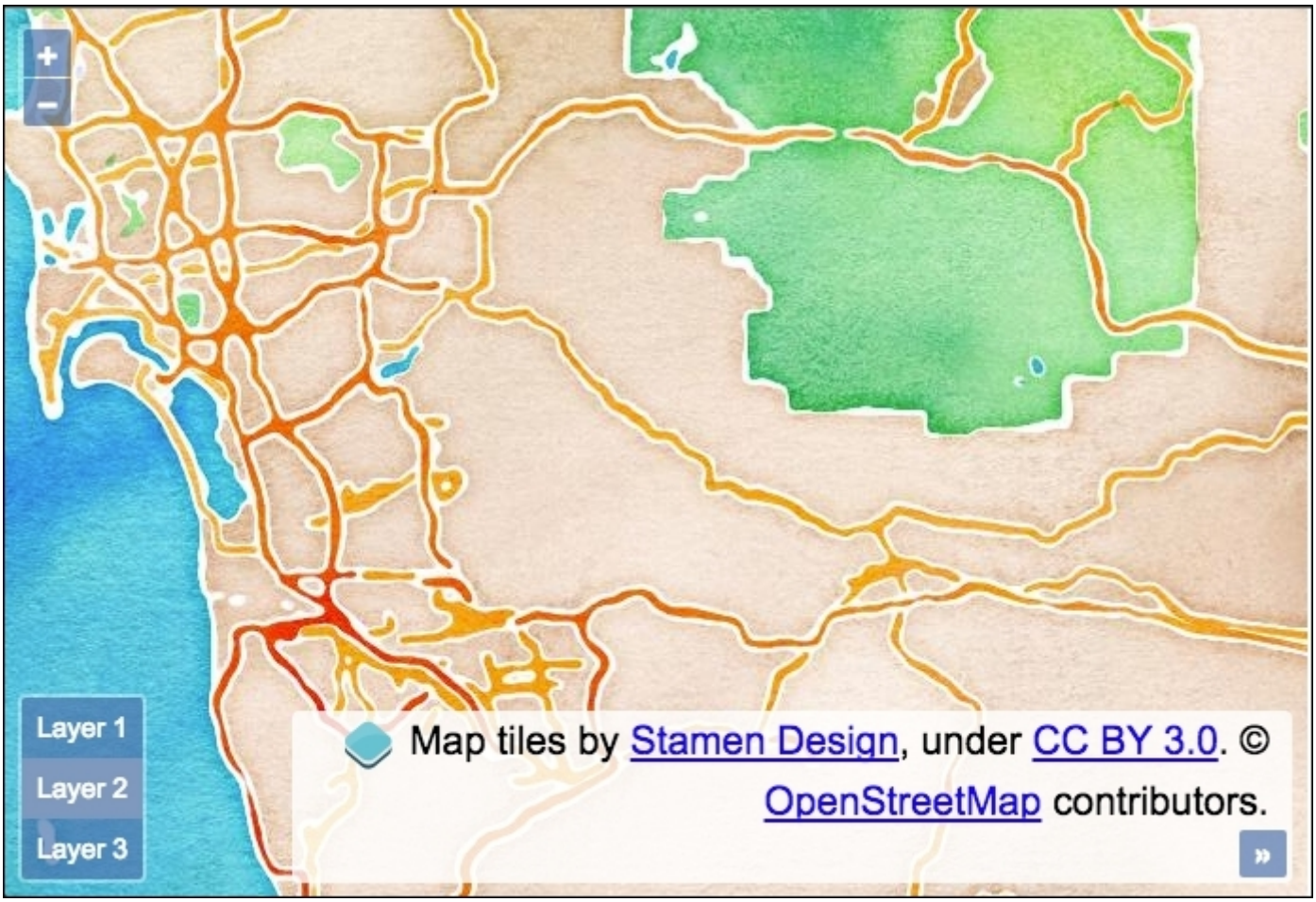
创建一个自定义控件

OpenLayers具有许多控件和交互功能，可以满足广泛的需求。但是，构建新的Web应用程序可能需要的要求不可避免地要求创建一个新的Web应用程序或扩展现有的Web应用程序。

OpenLayers 2来了 带有未在OpenLayers 3中使用的图层切换器控件。这不是问题，对于本食谱，我们将创建自己的图层切换器控件。除了使用户能够在层列表之间切换之外，此控件还将演示如何扩展基本控件类（ `ol.control.Control` ）以利用一些默认控件行为，然后在基本控件之上构建以提供用户可以单击的交互式控件。或点击以更改图层。

可以在中找到此食谱的源代码，这 `ch07/ch07-custom-control` 是位于地图左下角的自定义控件的预览：



怎么做...

1 创建一个具有OpenLayers依赖项的HTML文件，并使用一个 `div` 元素来保存地图。

2 创建一个自定义CSS文件，并添加以下内容以设置控件的样式：

复制

```
.ol-layer-switcher {
  bottom: 0.5em; left: 0.5em; }

.ol-layer-switcher ul {
  background-color: rgba(0,60,136,0.5);
  border: none; border-radius: 2px; color: #fff;
  margin: 0; padding: 0; list-style: none; font-size: 0.9em; }

.ol-layer-switcher li {
  display: block; padding: 8px; }

.ol-layer-switcher li:hover {
  background: #7b98bc; cursor: pointer; }

.ol-layer-switcher .active {
  background: #7b98bc; }
```

3 创建一个自定义 JavaScript文件并设置我们的自定义控件构造函数：

复制

```
var LayerSwitcher = function(options) {
  options = options || {};
  var className = options.className ?
    options.className : 'ol-layer-switcher';
  var cssClasses = className + ' ' + ol.css.CLASS_UNSELECTABLE +
     ' ' + ol.css.CLASS_CONTROL;
  var layers = options.layers;
  var list = document.createElement('ul');
  layers.forEach(function(layer, index, layers) {
    var li = document.createElement('li');
    li.setAttribute('data-layer-ref', ++index);
    li.innerHTML = 'Layer ' + index;
    if (index === layers.length) li.className = 'active';
    list.appendChild(li);
  });
  var controlDiv = goog.dom.createDom('div', cssClasses, list);
  controlDiv.addEventListener('click', function(event) {
    if (event.target.nodeName.toLowerCase() === 'li') {
      var itemNumber = parseInt(
```

4 让我们的自定义控件继承基本控件类，如下所示：

复制

```
ol.inherits(LayerSwitcher, ol.control.Control);
```

5 设置 `map` 实例具有一个 `view` 实例和三个栅格图块层之间进行切换，如下所示：

复制

```
var map = new ol.Map({
  view: new ol.View({
    zoom: 10, center: [-12987415, 3851814]
  }),
  target: 'js-map',
  layers: [
    new ol.layer.Tile({source: new ol.source.Stamen({
      layer: 'terrain'
    })}),
    new ol.layer.Tile({source: new ol.source.Stamen({
      layer: 'watercolor'
    })}),
    new ol.layer.Tile({source: new ol.source.Stamen({
      layer: 'toner'
    })})
  ]
});
```

6 实例化控件，传入一些图层，然后将其添加到 `map` 实例中：

复制

```
map.addControl(new LayerSwitcher({
  layers: map.getLayers()
}));
```

怎么运行的...

除了 `div` 用于保存地图的元素外，HTML仍然非常基础，因为控件标记是由我们的JavaScript生成的。我们向您展示了样式化控件所需的少量CSS，展示了实现该控件的容易程度。您会注意到，我们继承了OpenLayers提供的许多默认样式。

不出所料，此代码的核心围绕着自定义控件设置，跨约30行，因此出于说明目的，我们将其分成多个块：

复制

```
var LayerSwitcher = function(options) {
  options = options || {};
  var className = options.className ?
    options.className : 'ol-layer-switcher';
  var cssClasses = className + ' ' + ol.css.CLASS_UNSELECTABLE +
     ' ' + ol.css.CLASS_CONTROL;
  var layers = options.layers;
  var list = document.createElement('ul');
```

像任何其他OpenLayers控件一样，我们将其设计为可自定义的。该控件的任何用户定义选项都将出现在 `options` 参数中。

可自定义选项的第一个可以是类名，它将用作CSS规则的样式钩。如果提供了类名，请使用它；否则，我们默认使用自己的类名 `ol-layer-switcher` 。

什么时候我们 创建控件HTML包装元素后，我们将自定义类名称与OpenLayers的一些默认值（与其他控件一起使用）结合在一起。了解 `ol.css.CLASS_UNSELECTABLE` 和 `ol.css.CLASS_CONTROL` 分别引用了 `ol-unselectable` 和 `ol-control` 字符串（这对我们的控件应用了一些基本的CSS规则）非常有用。

我们将的引用存储 `options.layers` 在名为的变量中 `layers` ，创建一个无序列表（ `ul` ）DOM元素，并将其存储在该 `list` 变量中。

复制

```
layers.forEach(function(layer, index, layers) {
    var li = document.createElement('li');
    li.setAttribute('data-layer-ref', ++index);
    li.innerHTML = 'Layer ' + index;

    if (index === layers.length) li.className = 'active';
    list.appendChild(li);
  });
```

对于提供给控件的每一层，我们都会为其创建一个 `li` DOM元素。我们设置了一个属性， `data-layer-ref` 该属性使用数字标识图层，然后将一些文本与 `index` 数组（例如，图层1）一起添加到元素中。

如果这是循环的最终迭代，则 `active` 默认情况下，我们将最后一层设置为类名称。我们将这个列表项添加到父列表元素中以完成循环。我们已经成功地将图层列表转换为HTML列表，可以很快进行切换了。

复制

```
var controlDiv = goog.dom.createDom('div', cssClasses, list);
```

出于演示目的，我们使用了Google Closure库， `div` 使用先前的方法创建了包装元素。请记住，OpenLayers附带了Google Closure的许多实用程序方法，它们可能对您锻炼身体很有用。

此方法将必须创建的元素类型作为其第一个参数，然后将CSS类应用于该元素，然后将可变数量的子元素附加到父元素。在我们的情况下，这是 `list` 层的。

该 `controlDiv` 变量以以下内容结尾（以下是浏览器开发工具的屏幕截图）：

```
▼<div class="ol-layer-switcher ol-unselectable ol-control">
  ▼<ul>
     <li data-layer-ref="1">Layer 1</li>
     <li data-layer-ref="2">Layer 2</li>
     <li data-layer-ref="3" class="active">Layer 3</li>
   </ul>
 </div>
```

然后，我们带来 通过添加一些交互功能来实现对生活的控制：

复制

```
controlDiv.addEventListener('click', function(event) {
    if (event.target.nodeName.toLowerCase() === 'li') {
      var itemNumber = parseInt(
        event.target.getAttribute('data-layer-ref'), 10);

      list.querySelector('.active').className = '';
      list.querySelector('[data-layer-ref="' + itemNumber + '"]')
        .className = 'active';
      itemNumber--;

      layers.forEach(function(layer, index) {
        layers.item(index).setVisible(index === itemNumber);
      });
```

首先，在执行任何操作之前，检查我们的 `li` 元素之一是否是 `click` 事件的来源。一旦确定，我们将图层引用存储在 `itemNumber` 变量中。

我们通过从当前活动列表项中删除活动类来重置列表。然后，我们通过属性选择器再次查询该列表，以便选择 `li` 被单击的图层项目，并将该项目设置为类名称 `active` 。

`itemNumber` 然后，该数字将递减，因为 `index` 我们要迭代的层数组中的值从索引0开始。例如，列表中的第1层正确地与图层列表中的索引0相关联。

当我们遍历图层列表（一个 `ol.Collection` ）时，我们使用该 `item` 方法选择图层。然后 `visibility` ，我们设置为 `true` 或 `false` 通过评估迭代中的图层是否对应于所选图层来进行评估。

复制

```
ol.control.Control.call(this, {
  element: controlDiv
});
```

最后，如最后一步所示，当实例化此控件时，我们调用基本控件类，并将自定义元素分配给配置对象。这将调用基本 `control` 构造函数，以便使用OpenLayers默认设置来设置我们的控件。

复制

```
ol.inherits(LayerSwitcher, ol.control.Control);
```

本 `ol.inherits` 类只是一个别名 `goog.inherits` ，它从底座继承了原型 `ol.control.Control` 在我们的自定义构造函数 `LayerSwitcher` 的构造。例如，一个继承的原型方法是 `ol.control.Control.prototype.getMap` ，它获取 `map` 控件所附加到的实例。

我们确保我们的自定义控件与开发人员可能习惯的默认OpenLayers控件行为非常相似，从而使我们的控件熟悉，可扩展且易于使用。

我们将跳过在创建地图的部分中，因为这看起来非常熟悉。之后是我们的自定义控件的实例化：

复制

```
map.addControl(new LayerSwitcher({
  layers: map.getLayers()
}));
```

将其添加到 `map` 实例时，我们使用配置对象实例化控件，并将 `layers` 属性设置为地图中所有图层的值（雄蕊源中的三个栅格图层）。

这个层切换器当然不是万无一失的。但是，借助此样板，您将可以在OpenLayers的自定义控件中投入大量时间。

这是一个开源层切换器实现，您可能会对OpenLayers 3感兴趣：https (https://github.com/walkermatt/ol3-layerswitcher) : //github.com/walkermatt/ol3-layerswitcher (https://github.com/walkermatt/ol3-layerswitcher)。

也可以看看

- 在**添加和删除控制**在配方第5章 (/book/web_development/9781785287756/5)，**添加控件**
- 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)
- 第2章 (/book/web_development/9781785287756/2)，**添加栅格图层中**的**添加WMS图层**配方 (/book/web_development/9781785287756/2)

上一节 (/book/web_development/9781785287756/7/ch07lvl1sec61/working-with-projections)

下一节 (/book/web_development/9781785287756/7/ch07lvl1sec63/selecting-features-by-dragg