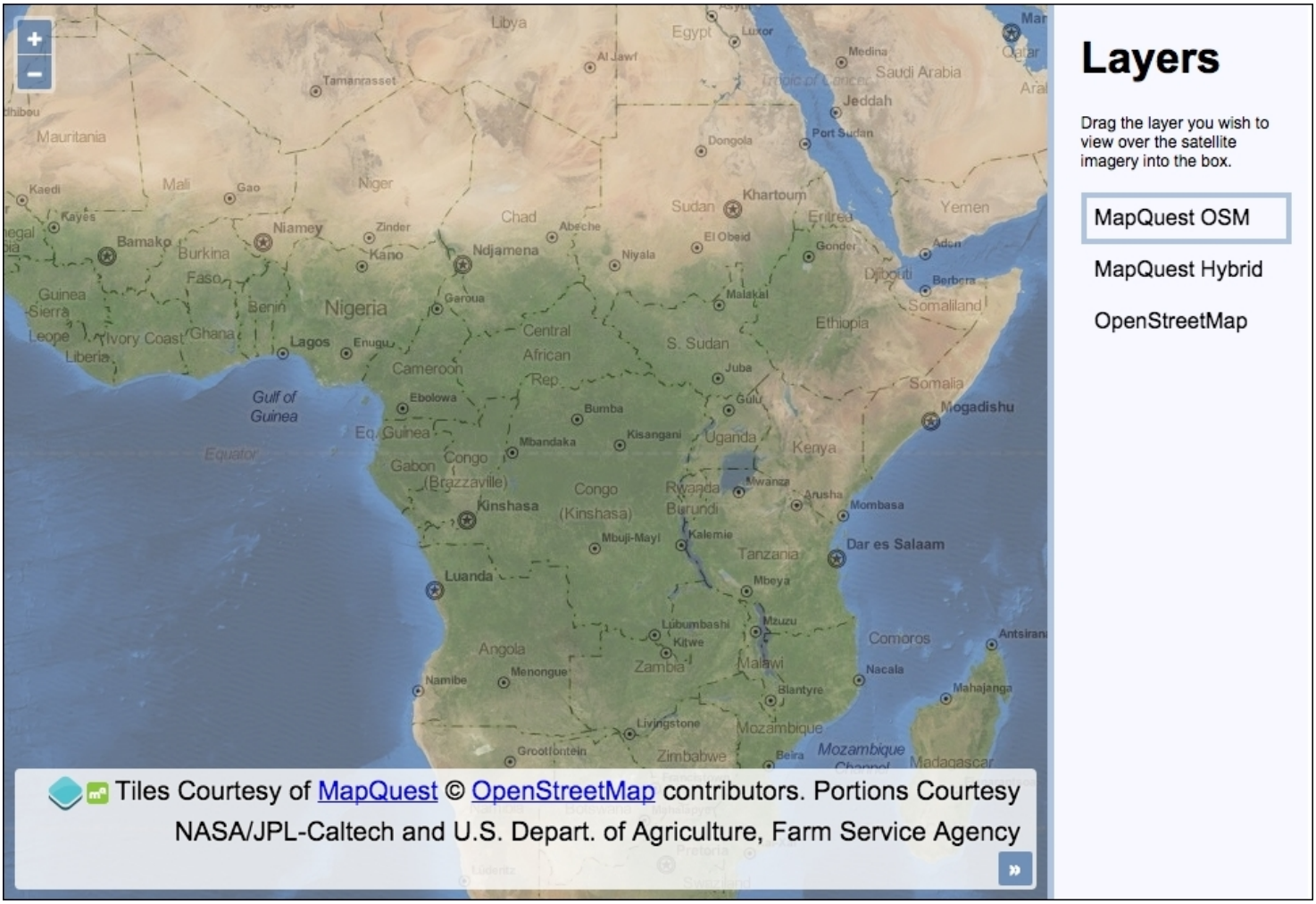
管理地图的堆栈层

OpenLayers地图使我们能够 可视化来自不同种类的层的信息，这为我们带来了管理与之相连的层的方法。

在本食谱中，我们将学习一些有关如何控制层的技术：添加，分组，管理堆栈顺序以及其他层操作。学习这些非常常见的操作非常重要，因为几乎每个Web映射应用程序都需要这些类型的任务。

该应用程序将在左侧显示地图，并在右侧显示控制面板，其中包含可以拖动的图层列表，您可以对其进行排序。这就是我们的最终结果：



您可以在中找到此食谱的源代码 `ch01/ch01-map-layers/` 。

注意

在此食谱中创建小部件（例如可排序列表）时，我们将使用jQuery UI库（https://jqueryui.com (https://jqueryui.com)），它对jQuery具有单一依赖性（https://jquery.com (https://jquery.com)）。这样做将有助于我们将注意力集中在OpenLayers代码上，而不是用于创建高级UI组件的常规JavaScript代码上。

怎么做...

1 我们从创建开始 一个HTML文件，用于组织应用程序布局并链接到资源：

复制

```
<head>
  <meta charset="utf-8">
  <title>Managing map's stack layers | Chapter 1</title>
  <link rel="stylesheet" href="ol.css">
  <link rel="stylesheet" href="style.css">
</head>
<body>
  <div id="js-map" class="map"></div>
  <div class="pane">
    <h1>Layers</h1>
    <p>Drag the layer you wish to view over the satellite imagery into the box.</p>
    <ul id="js-layers" class="layers"></ul>
  </div>
  <script src="ol.js"></script>
  <script src="jquery.js"></script>
  <script src="jquery-ui.js"></script>
  <script src="script.js"></script>
</body>
</html>
```

2 创建 CSS文件， `style.css` 并在其中添加以下内容：

复制

```
.map {
  position: absolute;
  top: 0;
  bottom: 0;
  left: 0;
  right: 20%;
}

.pane {
  position: absolute;
  top: 0;
  bottom: 0;
  right: 0;
  width: 20%;
  background: ghostwhite;
  border-left: 5px solid lightsteelblue;
  box-sizing: border-box;
  padding: 0 20px;
}
```

3 创建 在 `script.js` JavaScript文件，并添加了下列文件：

复制

```
var map = new ol.Map({
  layers: [
    new ol.layer.Tile({
      source: new ol.source.MapQuest({
        layer: 'sat'
      }),
      opacity: 0.5,
      zIndex: 1
    })
  ],
  view: new ol.View({
    zoom: 4,
    center: [2120000, 0]
  }),
  target: 'js-map'
});

var layerGroup = new ol.layer.Group({
  layers: [
```

怎么运行的...

HTML包含 地图和控制面板的标记。如本食谱前面所述，我们已链接到jQuery UI和jQuery的本地副本。如果您不使用提供的源代码，则需要自己下载这些库以进行后续操作。

CSS会组织布局，以使地图占据屏幕的80％的宽度，并为控制面板保留20％的空间。它还为图层列表提供样式，以便概述列表中的第一项以表示当前在视图中的图层。我们将不涉及CSS的更多细节，因为我们想花费更多时间来仔细研究OpenLayers代码。

让我们首先分解自定义JavaScript文件中的代码：

复制

```
var map = new ol.Map({
  layers: [
    new ol.layer.Tile({
      source: new ol.source.MapQuest({
        layer: 'sat'
      }),
      opacity: 0.5,
      zIndex: 1
    })
  ],
  view: new ol.View({
    zoom: 4,
    center: [2120000, 0]
  }),
  target: 'js-map'
});
```

我们在这里介绍了一个新的图层源 `ol.source.MapQuest` 。OpenLayers可以轻松访问此tile服务，该服务提供多种类型的 `layers` ，我们从中选择type `sat` ，这是Satellite的缩写。我们将使用这一层作为我们始终可见的背景。为了产生这种预期的效果，我们在某些性能已经传递给 `ol.layer.Tile` 到组 `opacity` 到50%（ `0.5` ），并 `zIndex` 到 `1` 。

我们设置 `zIndex` 为的原因 `1` 是为了确保该层不会被添加到该层顶部的层组隐藏。当我们继续浏览下一段代码时，将对此进行更好的解释，如下所示：

复制

```
var layerGroup = new ol.layer.Group({
  layers: [
    new ol.layer.Tile({
      source: new ol.source.MapQuest({
        layer: 'osm'
      }),
      title: 'MapQuest OSM'
    }),
    new ol.layer.Tile({
      source: new ol.source.MapQuest({
        layer: 'hyb'
      }),
      title: 'MapQuest Hybrid',
      visible: false
    }),
    new ol.layer.Tile({
      source: new ol.source.OSM(),
      title: 'OpenStreetMap',
      visible: false
```

我们实例化一个新的 的实例 `ol.layer.Group` ，它需要一个图层集合。创建图层组的一个有用好处是，您希望一次对多个图层应用相同的操作，例如设置属性。

我们实例化的三个新实例 `ol.layer.Tile` ，其中两个是 `ol.source.MapQuest` （ `osm` 和 `hyb` ）提供的不同层类型。另一个图块服务源是 `ol.source.OSM` 以前食谱中熟悉的图层源（OpenStreetMap）。

我们已将 `visible` 三个图块层中的两个上的属性设置为 `false` 。页面加载时，该 `MapQuest osm` 图层将是该图层组中唯一可见的图层。

（可选）我们可以将 `opacity` 不想显示的图层的设置为0。但是，将可见性设置为会带来性能上的好处 `false` ，因为OpenLayers不会对不可见的图层切片进行任何不必要的HTTP请求。

`title` 我们在每一层上设置的属性实际上并不是OpenLayers API的一部分。这是一个自定义属性，我们几乎可以为它命名。这使我们可以在 `layer` 对象上创建任意属性和值，稍后可以在应用程序中引用这些属性和值。我们将标题信息用于某些层切换逻辑，并将此文本显示在UI中。

最后， `layer` 通过将 `zIndex` 属性设置 `0` 为图层组实例，对组中的所有图层进行了自定义。但是，为什么我们要这样做呢？

在内部，OpenLayers存储 `layers` 在数组中，并且以与存储在数组中的顺序相同的顺序进行渲染（因此，第一个元素是底层）。您可以将地图想像为将图层存储在堆栈中，并且它们是从下到上渲染的，因此，根据不透明度和范围，上面的图层可以隐藏在下面的图层下面。

考虑到这一点，当将此图层组添加到地图后，它自然会在包含卫星图像的第一层上方进行渲染。由于组中的图层都是不透明的，因此将导致隐藏卫星图像图层。但是，通过手动操作地图图层堆栈顺序，我们可以通过将设置 `zIndex` 为来强制图层组位于堆栈的底部，通过将设置为来 `0`， 迫使卫星图像图层在堆栈的顶部， `zIndex` 以 `1` 使其在上方渲染这个图层组。

注意

无论如何， `zIndex` 图层组的默认属性是 `0` 。这意味着我们可以 `zIndex` 将卫星层的属性设置为 `1` ，这将给我们带来相同的结果。我们已在此处明确设置此值，以帮助解释发生了什么。

由于我们一直希望将卫星图像放在最上面，因此值得一提的是它 `ol.layer.Layer` 提供了一种 `setMap` 方法。tile图层（ `ol.layer.Tile` ）是的子类 `ol.layer.Layer` ，因此如果我们通过 `setMap` 方法将卫星图像tile图层添加到地图，则无需自己手动调整 `zIndex` 属性，因为该属性会自动显示在顶部。无论如何，这是一个很好的机会来展示 `zIndex` 实际行动。

复制

```
map.addLayer(layerGroup);
```

图层组仅添加到地图实例中。您会注意到，此方法可用于添加单个图层或一组图层。

复制

```
var $layersList = $('#js-layers');
layerGroup.getLayers().forEach(function(element, index, array) {
  var $li = $('<li />');
  $li.text(element.get('title'));
  $layersList.append($li);
});
```

现在，我们开始利用jQuery库来执行一些DOM操作。我们将 `js-layers` ID 的元素存储到一个变量中，即 `$layersList` 。在变量前面加上美元符号是一种约定，可以将结果表示为jQuery对象。此选择器将定位先前的HTML：

复制

```
<ul id="js-layers" class="layers"></ul>
```

为了在面板中动态填充图层列表，我们使用了来自图层组实例的方法 `getLayers` 。这会返回 `ol.collection` 给定组的所有层的列表（），然后我们将其链接到 `forEach` 方法（可从中使用另一种方法 `ol.collection` ）。

在内部， `forEach` 方法从Google Closure库调用实用程序方法。该 `forEach` 方法中可用的参数是元素，索引和数组。元素是迭代中的层，索引是该层在迭代中在组中的位置，而array是我们要遍历的一组层。在本例中，我们仅使用element参数。

我们使用jQuery创建一个 `li` 元素并设置文本内容。文本值源自图层的标题值，这是我们赋予组中每个图层的自定义属性，以标识它们。OpenLayers提供了一种方便的 `get` 方法来检索此值。然后，我们使用jQuery将这个 `li` 元素附加到 `ul` 元素上。

复制

```
$layersList.sortable({
  update: function() {
    var topLayer = $layersList.find('li:first-child').text();

    layerGroup.getLayers().forEach(function(element) {
      element.setVisible(element.get('title') === topLayer);
    });
  }
});
```

为了使列表项能够重新排序，我们使用jQuery UI可排序小部件并将其应用于HTML中的层列表。列表中的某个项目一旦移动，就会触发更新事件。这是我们执行一些OpenLayers逻辑的地方。

获取最顶层的文本内容，因为这是用户希望看到的层。文本存储在 `topLayer` 变量内。该文本将对应于图层标题之一。

我们使用相同的 `getLayers` 层组上的方法和 `forEach` 该方法 `ol.collection` 如前。根据文本是否与图层标题匹配，我们使用 `setVisible` 方法相应地切换图层可见性。

还有更多...

对于此配方，我们选择一次仅显示其他一层。如果需要使所有图层保持可见，而是动态更改图层的堆叠顺序，则可以使用layer `setZIndex` 方法来管理哪些图层位于其他图层之上。

使用图层的集合（例如随返回的内容） `ol.Map.getLayers()` ，您可以 `setAt` 在 `ol.collection` layers对象上使用方法对图层进行重新排序，随后更改其堆叠顺序。这实际上与更改 `zIndex` 属性相同。

还有许多其他方法可以操纵地图图层。在此配方中，我们仅看到了一些：添加，设置标准和任意属性，层堆栈排序等。但是，您可以找到更多方法，例如，删除图层/图层组，更改图层源等。

也可以看看

- 该**管理地图的控制**食谱
- 在**周围的地图视图搬家**食谱
- 的**制约地图范围**的配方

上一节 (/book/web_development/9781785287756/1/ch01lvl1sec11/playing-with-the-map-s-opt

下一节 (/book/web_development/9781785287756/1/ch01lvl1sec13/managing-the-map-s-contro