天气预报图像之间的过渡

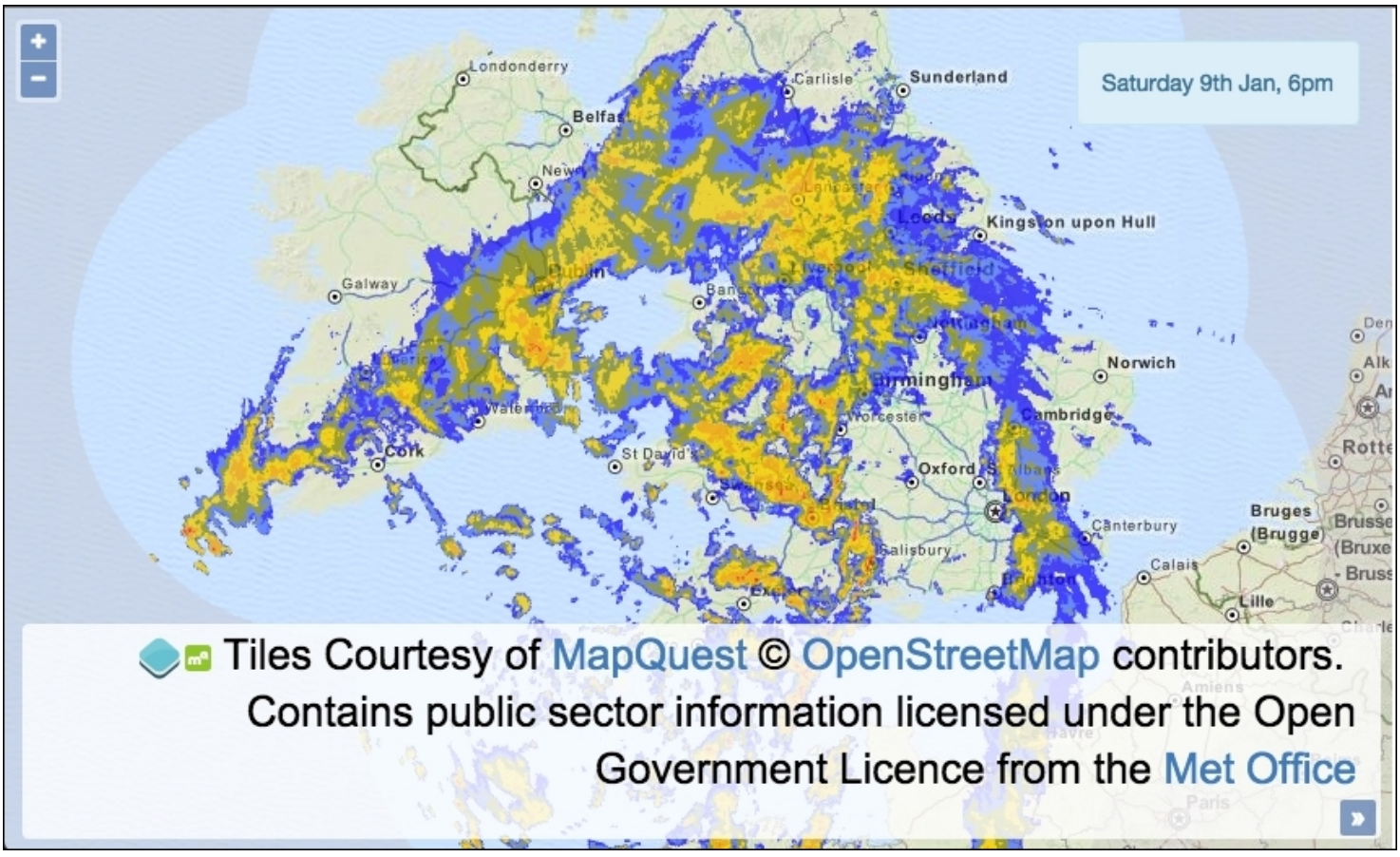
工作时 有了地理信息，它在空间上的几何表现并不是唯一重要的事情。我们还可以考虑时间的维度。人们对基于特定时刻的最新信息或数据越来越感兴趣。

这样，可视化可以向我们展示数据如何随时间变化：城市人口，疾病暴发，天气预报等。

在本食谱中，我们将向您展示如何通过在一天中不同时间的天气预报之间进行转换来在客户端创建动画。

我们将使用WMTS服务中的大都会办公室（http://www.metoffice.gov.uk (http://www.metoffice.gov.uk)），显示了不同时刻的降雨演变（如以下屏幕截图所示），我们将通过使先前的预测消失并创建新的动画来创建动画。淡入。可视化表示的时间将显示在地图的右上方。

可以在中找到源代码 `ch07/ch07-weather-forecast-imagery` 。



做好准备

在这食谱，我们将使用JavaScript中的日期来在特定时间请求数据。为了帮助解决此问题，并帮助将时间格式化为字符串以供显示，我们使用了一个非常好的库，该库是专门为这种类型的操作而设计的，它称为**moment**，可以下载该库。来自http://momentjs.com (http://momentjs.com)。

我们将通过Met Office（http://www.metoffice.gov.uk/datapoint/support/documentation/inspire-layers-detailed-documentation (http://www.metoffice.gov.uk/datapoint/support/documentation/inspire-layers-detailed-documentation)）连接到WMTS服务，该服务需要使用API密钥，就像我们用必应地图看到在第2章 (/book/web_development/9781785287756/2)，**添加栅格图层**中的**使用Bing图像**配方中。前往http://www.metoffice.gov.uk/datapoint (http://www.metoffice.gov.uk/datapoint)进行注册（免费）；注册后，您将可以获取API密钥。这是一个非常快速和自动化的过程。(/book/web_development/9781785287756/2) (http://www.metoffice.gov.uk/datapoint)

怎么做...

1. 创建一个新的HTML文件，添加OpenLayers依赖项，还添加JavaScript日期库 `moment` ，如本食谱的“**准备就绪**”部分所述。添加一个 `div` 元素来保存地图，尤其是一些标记来显示时间：

复制

```
<span id="js-time"></span>
```

2. 创建一个自定义JavaScript文件，并 `resolutions` 根据 `EPSG:4326` 投影手动生成每个缩放级别：

复制

```
var proj4326 = ol.proj.get('EPSG:4326');
var proj4326Extent = proj4326.getExtent();
var size = ol.extent.getWidth(proj4326Extent) / 256;
var resolutions = [], matrixIds = [];
for (var z = 0; z < 11; ++z) {
  resolutions[z] = size / Math.pow(2, z);
  matrixIds[z] = z;
}
```

3. 产生时间列表，从最后一个小时开始，再到现在的五个小时：

复制

```
var times = [], count = 0;
while (count <= 5) {
  var time = moment().startOf('hour');
  time.subtract(count, 'hour');
  times.push(time);
  count++;
}
```

4. 分配您对此变量的API密钥（有关更多详细信息，请参阅此食谱的“**准备就绪**”部分）：

复制

```
var apiKey = 'YOUR_API_KEY_HERE';
```

5 创建一个函数，该函数将返回WMTS层的新实例：

复制

```
var createWMTSLayer = function(time) {
  return new ol.layer.Tile({
    opacity: 0.7,
    source: new ol.source.WMTS({
      attributions:  [new ol.Attribution({
        html: '<br>Contains public sector information licensed' +
          'under the Open Government Licence from the ' +
          '<a href="http://www.metoffice.gov.uk">Met Office</a>'
      })],
      url:'http://datapoint.metoffice.gov.uk/' +
          'public/data/inspire/view/wmts?' +
          'key=' + apiKey + '&TileMatrix=EPSG:4326:6&' +
          'time=' + time.format('YYYY-MM-DDTHH:00:00') + 'Z',
      layer: 'RADAR_UK_Composite_Highres',
      matrixSet: 'EPSG:4326',
      format: 'image/png',
      style: 'Bitmap 1km Blue-Pale blue gradient 0.01 to 32mm/hr',
      projection: proj4326,
      tileGrid: new ol.tilegrid.WMTS({
```

6 实例化一个新 `map` 实例，该 `view` 实例的实例被限制为缩放级别6，并为背景映射添加了一个栅格图块层。呼叫 `createWMTSLayer` 为了根据列表中的第一时间添加天气预报图层：

复制

```
var map = new ol.Map({
  view: new ol.View({
    zoom: 6, minZoom: 6, maxZoom: 6, center: [-354667, 7254791]}),
  target: 'js-map', layers: [
    new ol.layer.Tile({source: new ol.source.MapQuest({
      layer: 'osm'
    })}), createWMTSLayer(times[0])]
});
```

7 缓存包含表示地图上当前可见数据的时间的DOM元素。将列表中的第一次添加到元素：

复制

```
var timeElem = document.getElementById('js-time');
timeElem.innerHTML = times[0].format('dddd Do MMM, ha');
```

8 设置一些将在预测转换过程中使用的变量：

复制

```
var rotateCount = 1, oldLayer, newLayer;
```

9 创建将使以前的天气预报图像淡出并删除它的函数：

复制

```
var fadeAndRemoveLayer = function() {
  var opacity = oldLayer.getOpacity();
  if (opacity > 0) {
    oldLayer.setOpacity(opacity - 0.1);
    setTimeout(fadeAndRemoveLayer, 100);
  } else {
    map.removeLayer(oldLayer);
    timeElem.innerHTML = times[rotateCount].format(
      'dddd Do MMM, ha'
    );
    if (rotateCount !== times.length - 1) {
      rotateCount++;
    } else {
      rotateCount = 0;
    }
    setTimeout(rotate, 7000);
  }
};
```

10 创建使新天气预报图像淡入的函数：

复制

```
var showLayer = function() {
  var opacity = newLayer.getOpacity();
  if (opacity < 0.7) {
    newLayer.setOpacity(opacity + 0.1);
    setTimeout(showLayer, 100);
  }
};
```

11 创建 功能会 `rotate` 天气预报图像：

复制

```
var rotate = function() {
  newLayer = createWMTSLayer(times[rotateCount]);
  newLayer.setOpacity(0);
  map.addLayer(newLayer);
  oldLayer = map.getLayers().item(1);
  setTimeout(function() {
    fadeAndRemoveLayer();
    showLayer();
  }, 3000);
};
```

12 最后，安排天气预报图像在10秒内开始转换：

复制

```
setTimeout(rotate, 10000);
```

怎么运行的...

如您所见，这里有很多代码。我们不会详细介绍如何实现HTML和CSS，但请查看本书的源代码以了解更多信息。我们将专注于JavaScript的新引入的概念：

复制

```
var proj4326 = ol.proj.get('EPSG:4326');
var proj4326Extent = proj4326.getExtent();
var size = ol.extent.getWidth(proj4326Extent) / 256;
var resolutions = [], matrixIds = [];
for (var z = 0; z < 11; ++z) {
    resolutions[z] = size / Math.pow(2, z);
    matrixIds[z] = z;
}
```

使用此WMTS服务时，我们需要根据服务的功能手动设置每个缩放级别的分辨率，您可以在找到 `http://datapoint.metoffice.gov.uk/public/data/inspire/view/wmts?REQUEST=getcapabilities&key=YOUR_API_KEY_HERE` 。如您所见，API密钥构成了URL的一部分，因此请确保将其添加。

WMTS服务支持 `EPSG:4326` 投影，据我们所知，OpenLayers默认情况下支持投影，因此这使实现更容易。

从投影范围（ `proj4326.getExtent()` ）的检索中，我们可以将其除以 `256` （图块的像素宽度）。下一个任务是根据每个缩放级别的大小分配分辨率（此WMTS服务支持 `11` 从开始的缩放级别 `0` ）。在循环的每次迭代中，都会将相关分辨率应用于该缩放级别。

如果缩放级别为7，则将基数2乘以7（ `Math.pow(2, 7)` ）的指数。换句话说，**2*2*2*2*2*2*2**。这等于128，缩放级别将其除以128，对于此矩阵ID为7，分辨率为0.010986328125。

当我们为WMTS源设置图块网格时，将使用此信息。在第2章“(/book/web_development/9781785287756/2)**添加栅格图层”中的“ 在WMS图层中设置拼贴大小”**配方中，我们演示了类似的拼贴网格构造。 (/book/web_development/9781785287756/2)

复制

```
var times = [], count = 0;
while (count <= 5) {
  var time = moment().startOf('hour');
  time.subtract(count, 'hour');
  times.push(time);
  count++;
}
```

我们收集了六个时间戳，这些时间戳将构成WMTS服务请求的一部分。该服务通常每隔15分钟提供一次数据，备份到前一天的午夜。但是，我们决定每小时收集一次数据。该 `moment` 库使我们可以方便地从最后一个小时（ `moment().startOf('hour')` ）开始。例如，如果当前时间是3:28 pm，则列表将以3:00 pm的时间戳记开始，并以10:00 am的最后时间输入结束。 `moment` 方法 `subtract` ，使用该 `count` 值（每次迭代增加）从开始时间中减去相关的小时数。

复制

```
var createWMTSLayer = function(time) {
  return new ol.layer.Tile({
    opacity: 0.7,
    source: new ol.source.WMTS({
      ...
      url: 'http://datapoint.metoffice.gov.uk/' +
           'public/data/inspire/view/wmts?' +
           'key=' + apiKey + '&TileMatrix=EPSG:4326:6&' +
           'time=' + time.format('YYYY-MM-DDTHH:00:00') + 'Z',
      layer: 'RADAR_UK_Composite_Highres',
      matrixSet: 'EPSG:4326',
      format: 'image/png',
      style: 'Bitmap 1km Blue-Pale blue gradient 0.01 to 32mm/hr',
      projection: proj4326,
      tileGrid: new ol.tilegrid.WMTS({
        origin: ol.extent.getTopLeft(proj4326Extent),
        resolutions: resolutions,
        matrixIds: matrixIds
      })
```

为简便起见，我们 尚未包含此方法的全部代码，因为在本书的此阶段我们不需要解释它的每个部分。

我们的 `createWMTSLayer` 函数采用一个参数 `time` 。这是因为URL的一部分是由URL构成的，`time` 因此我们可以获取特定时间的天气预报。您会看到，在 `url` 属性的值内，我们 `time.format('YYYY-MM-DDTHH:00:00') + 'Z'` 使用矩库将时间格式化为字符串（），矩库是WMTS服务所需的结构；例如2016-04-15T11：00：00Z。

注意

需要将自定义URL参数连接到URL字符串，因为 `ol.source.WMTS` 不会 `params` 为此提供属性，这与的不同 `ol.source.TileWMS` 。

该 `url` 属性不包含所述URL的整体，作为其他属性 `layer` ， `matrixSet` ， `format` ，和 `style` ，还由的OpenLayers URL字符串相结合。如果我们愿意，我们可以自己将这些键/值添加到URL字符串中，并避免使用额外的属性。

为 `tileGrid` 属性分配了的新实例 `ol.tilegrid.WMTS` ，该实例为我们设置了网格图案。这意味着可以从WMTS服务正确请求带有相关矩阵详细信息（例如，图块列和图块行）的图块，例如，这些信息将添加到URL中 `TileCol=66&TileRow=13` 。

您会看到我们的分辨率和矩阵ID分别用于构造OpenLayers通过 `resolutions` 和 `matrixIds` 属性为我们创建的图块网格。

复制

```
var fadeAndRemoveLayer = function() {
  var opacity = oldLayer.getOpacity();
  if (opacity > 0) {
    oldLayer.setOpacity(opacity - 0.1);
    setTimeout(fadeAndRemoveLayer, 100);
  } else {
    map.removeLayer(oldLayer);
    timeElem.innerHTML = times[rotateCount].format('dddd Do MMM, ha');
    if (rotateCount !== times.length - 1) {
      rotateCount++;
    } else {
      rotateCount = 0;
    }
    setTimeout(rotate, 7000);
  }
};
```

这个方法 优雅地使旧的WMTS图层淡出，然后将其从地图中删除。首先从获取 `opacity` 图层的最新值开始，然后检查其是否在之上 `0` 。如果是这种情况，则表明它仍然可见；因此，我们 `0.1` 从不透明度级别中去除了另一个（进一步使此淡出效果），并 `100` 使用JavaScript `setTimeout` 方法在毫秒内再次调用了此函数。

如果 `opacity` 不再大于 `0` ，我们将从地图上删除旧的WMTS图层。然后，我们将HTML时间显示更新为新时间，并相应地设置其格式。时间是从时间列表中收集的，这些时间基于 `rotateCount` 变量内的值。我们 `rotateCount` 在前面的条件块中将变量保持最新。

如果 `rotateCount` 仍然小于时间列表的最后一个索引，则我们增加该值，因为它推断我们每次输入都尚未转换。相反，如果 `rotateCount` 等于最后一个索引，则将值重置为零，以便从列表的开头再次循环。这是时间条目的无限循环。

我们通过调度下旋转在7秒内启动，与玩完 `setTimeout(rotate, 7000)` 。

复制

```
var showLayer = function() {
  var opacity = newLayer.getOpacity();
  if (opacity < 0.7) {
    newLayer.setOpacity(opacity + 0.1);
    setTimeout(showLayer, 100);
  }
};
```

另一方面，要淡入淡出并删除，此功能会使新的天气预报图像淡入。遵循类似的逻辑，但相反，我们获得了最新 `opacity` 值，并检查其是否低于 `0.7` 此值（这是我们最终的不透明度级别希望实现）。如果它小于此值，则我们将其增加 `0.1` 一并在 `100` 毫秒内再次调用此函数。这一直持续到该图层以所需的 `opacity` 值正常淡入淡入视图为止。

复制

```
var rotate = function() {
  newLayer = createWMTSLayer(times[rotateCount]);
  newLayer.setOpacity(0);
  map.addLayer(newLayer);
  oldLayer = map.getLayers().item(1);

  setTimeout(function() {
    fadeAndRemoveLayer();
    showLayer();
  }, 3000);
};
```

该 `rotate` 功能在应用程序首次加载后的10秒钟内启动，可协调预测图像之间的转换。

这首先创建 新的WMTS层，该层基于列表中的下一次输入。对新层的引用存储在 `newLayer` 变量中，这是 `showLayer` 动画方法对其进行访问的方式。图层的不透明度设置为零，然后添加到地图。我们这样做是因为将不透明度设置为零并不能阻止在地图上请求并加载图块，而它们现在还不可见。

当前层（必须将其删除）存储在 `oldLayer` 变量中，这就是 `fadeAndRemoveLayer` 访问该变量的方式。' `old` '图层始终 `ol.Collection` 是地图图层数组中的第二项，因此我们可以通过i进行访问 `tem(1)` 。

此方法的最终指令将在3秒内触发层的转换。我们将其延迟任意数量，以使新的瓷砖在淡入之前有机会加载。

也可以看看

- 第2章 (/book/web_development/9781785287756/2)，**添加栅格图层中**的**创建图像图层**配方 (/book/web_development/9781785287756/2)
- 第2章 (/book/web_development/9781785287756/2)，**添加栅格图层中**的**更改图层不透明度**配方 (/book/web_development/9781785287756/2)

上一节 (/book/web_development/9781785287756/7/ch07lvl1sec63/selecting-features-by-dragg

下一节 (/book/web_development/9781785287756/7/ch07lvl1sec65/using-the-custom-openlaye