从WFS服务器添加功能

该**Web要素服务**（**WFS**）是一个OGC标准，提供独立的平台调用以向服务器请求地理特征。在实际上，这意味着客户端会向实现该协议的服务器发出HTTP请求 WFS标准，并获得各种格式的功能，通常GML（地理标记语言，http：//en.wikipedia.org/wiki/Geography_Markup_Language (http://en.wikipedia.org/wiki/Geography_Markup_Language)）。

注意

如果您想了解更多相关信息，请在OGC网站上找到 (http://www.opengeospatial.org/standards/wfs)完整的规范，网址为http://www.opengeospatial.org/standards/wfs (http://www.opengeospatial.org/standards/wfs)。从OpenLayers的角度来看，WFS就是我们可以读取以填充矢量层的另一个数据源。

在继续之前，需要考虑一个重要的点。OpenLayers在加载数据时发出的大多数请求（例如GML，KML或GeoJSON文件）都是通过AJAX请求异步发出的。

任何JavaScript调用都受到浏览器强加的安全模型的限制，这避免了跨域请求。这意味着您只能向网页最初来自的同一服务器发出请求。

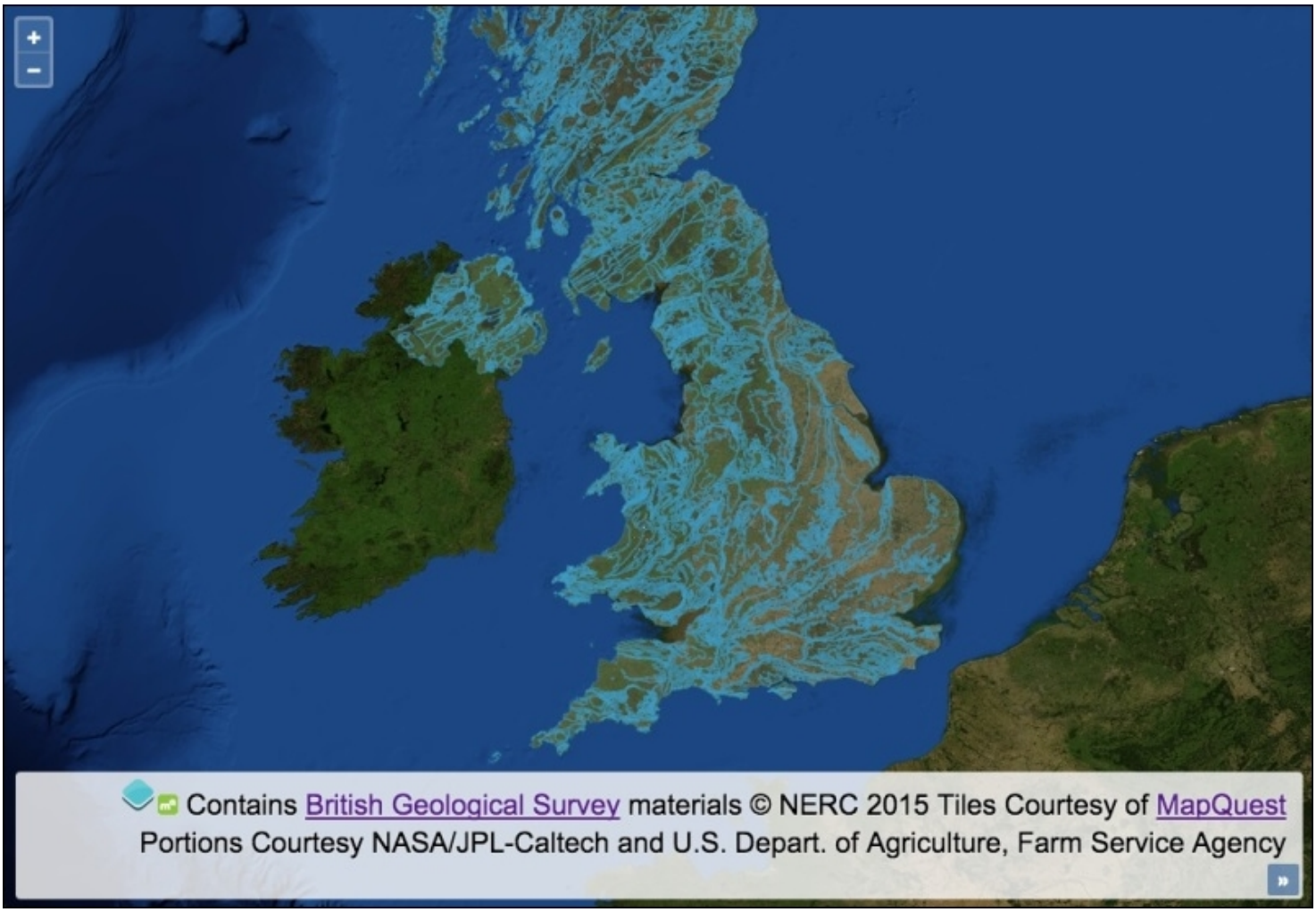
有多种方法可以绕过此限制。此类技术包括JSONP（https://en.wikipedia.org/wiki/JSONP (https://en.wikipedia.org/wiki/JSONP)）和调整CORS权限（https://en.wikipedia.org/wiki/Cross-origin_resource_sharing (https://en.wikipedia.org/wiki/Cross-origin_resource_sharing)），或在服务器上使用代理-侧。

注意

您可以在http://developer.yahoo.com/javascript/howto-proxy.html上 (http://developer.yahoo.com/javascript/howto-proxy.html)阅读有关代理实现的更清晰的说明。

代理的想法很简单。我们不是直接向跨域发出请求，而是向同一域上的脚本发出请求，该脚本负责为我们转发跨域请求并返回结果。服务器上的脚本不受浏览器供应商施加的跨域请求的限制。

对于此食谱（位于中 `ch03/ch03-wfs-layer/` ），我们将使用用Node.js编写的代理服务器来转发请求并返回响应（请参阅 `server.js` 根目录中的文件这本书的源代码）。我们将结束渲染的WFS图层看起来类似于以下屏幕截图：



怎么做...

按照以下说明连接到外部WFS服务器并在地图上渲染要素：

1. 创建一个包含OpenLayers依赖项的HTML文件，并添加一个 `div` 元素来保存地图。
2. 创建一个自定义JavaScript文件并启动 `map` ：

复制

```
var map = new ol.Map({
  view: new ol.View({
    zoom: 6,
    center: [-415817, 6790054]
  }),
  target: 'js-map',
  layers: [
    new ol.layer.Tile({
      source: new ol.source.MapQuest({layer: 'sat'})
    })
  ]
});
```

3 创建向量源将负责WFS请求：

复制

```
var vectorSource = new ol.source.Vector({
  format: new ol.format.WFS(),
  url: function(extent, resolution, projection) {
    return [
      '/proxy?proxyHost=ogc.bgs.ac.uk',
      'proxyPath=/digmap625k_gsml32_cgi_gs/wfs?',
      'service=WFS',
      'version=1.1.0',
      'request=GetFeature',
      'typename=test:uk_625k_mapped_feature',
      'srsname=' + projection.getCode(),
      'bbox=' + extent.join(',') + ',' + projection.getCode(),
      'outputformat=gml3'
    ].join('&');
  },
  strategy: ol.loadingstrategy.tile(ol.tilegrid.createXYZ()),
  attributions: [
    new ol.Attribution({
      html: 'Contains <a href="http://bgs.ac.uk/">British Geological Survey</a> ' +
```

4 设置矢量图层，向其添加矢量源，然后将该图层添加至地图：

复制

```
var vectorLayer = new ol.layer.Vector({
  source: vectorSource,
  opacity: 0.4
});
map.addLayer(vectorLayer);
```

怎么运行的...

创建WFS层时，总会涉及一些前期工作，因为您需要解释要从中检索信息的服务的功能，然后相应地调整HTTP请求。

为了帮助我们了解WFS服务，服务器通常会提供一个XML文档，概述该服务的功能。例如，可以在http://ogc.bgs.ac.uk/digmap625k_gsml32_cgi_gs/wfs?service=WFS&request=GetCapabilities中(http://ogc.bgs.ac.uk/digmap625k_gsml32_cgi_gs/wfs?service=WFS&request=GetCapabilities)找到我们在此食谱中访问的WFS服务的功能文档。

让我们关注向量源的内容以及它如何影响此配方的结果：

复制

```
var vectorSource = new ol.source.Vector({
  format: new ol.format.WFS(),
```

当此向量源 从WFS服务检索内容，我们 通知OpenLayers响应的预期格式是什么，以便成功解析数据并将其添加到地图。数据的格式将为WFS格式，功能将以GML3（我们指定为URL字符串的一部分，即输出格式）呈现。

您可以将一个可选的配置对象传递给WFS格式的构造函数，该构造函数具有一些值得一提的属性，如下所示：

- **gmlFormat**：为默认GML格式 `ol.format.WFS` 是 `ol.format.GML3` 的，但你可以指定 `ol.format.GML2` 是否需要
- **featureType**：这是要素类型；你有机会在这里有选择性
- **featureNS**：这是用于功能的名称空间URI

让我们继续进行向量源配置的下一个属性：

复制

```
url: function(extent, resolution, projection) {
    return [
      '/proxy?proxyHost=ogc.bgs.ac.uk',
      'proxyPath=/digmap625k_gsml32_cgi_gs/wfs?',
      'service=WFS',
      'version=1.1.0',
      'request=GetFeature',
      'typename=test:uk_625k_mapped_feature',
      'srsname=' + projection.getCode(),
      'bbox=' + extent.join(',') + ',' + projection.getCode(),
      'outputformat=gml3'
    ].join('&');
  },
```

我们通过分配给 `url` 属性的函数来手动构造将构成AJAX请求的URL 。该函数必须符合的函数类型 `ol.FeatureUrlFunction` ，如您所见，该函数的类型为 `extent` （类型 `ol.Extent` ），地图的 `resolution` （类型编号）和地图的 `projection` （类型 `ol.proj.Projection` ）。我们可以使用这些可用参数来动态构建URL。我们的函数必须将URL作为字符串返回。

我们正在使用代理 机制来完成对我们的请求。代理人实现要求我们传递主机名作为 `proxyHost` 参数，并传递路径名作为 `proxyPath` 参数。到目前为止，如果有帮助，我们可以使用以下端点：`ogc.bgs.ac.uk/digmap625k_gsml32_cgi_gs/wfs?` 。我们将附加条件附加到该字符串作为查询键/值对。

一些键/值对是与WFS服务提供的功能匹配的静态值。一旦您使用了一些不同的Web服务（例如1.1.0版）和请求（例如），这些内容就会开始变得更加熟悉 `GetFeature` 。该 `typename` 键指定什么类型的功能设置，我们感兴趣的问题。这将不可避免地供应商之间会有所不同。

我们动态设置了 `srsname` 关键点（空间参考），该关键点是从地图的当前投影（例如 `EPSG:3857` ）中检索到的。将 `bbox` 被动态地从视图程度的组合（其是最初经由的JavaScript转换成逗号分隔的字符串的数组确定 `join` 法）以及地图的投影（ `projection.getCode()` ）。

此函数必须将URL作为字符串返回，因此我们使用 `join` 方法再次将数组转换为字符串，该方法在数组的每个项目之间插入一个＆符号，符合标准查询字符串结构，即 `service=WFS&version=1.0.0` 。

复制

```
strategy: ol.loadingstrategy.tile(ol.tilegrid.createXYZ()),
```

我们选择了使用tile的非默认数据加载策略（ `ol.loadingstrategy.all` 默认设置）。这会使用XYZ切片方案在基于切片网格的要素中进行请求和加载。要了解有关XYZ平铺方案的更多信息，请参阅http://wiki.openstreetmap.org/wiki/Slippy_map_tilenames (http://wiki.openstreetmap.org/wiki/Slippy_map_tilenames)。还有一个很好的资源可以解释XYZ和**Tile Map Service**（**TMS**）切片方案之间的区别，请参见https://gist.github.com/tmcw/4954720 (https://gist.github.com/tmcw/4954720)。

该WFS服务仅覆盖英国，因此我们可以最佳地将图块网格的范围限制在英国的范围内，从而避免浪费的请求。的 `ol.tilegrid.createXYZ` 类可任选地给定的配置对象，该对象的 `extent` 属性可以被分配一个 `ol.Extent` 数组，将限制在请求的范围。我们还可以指定最小和最大缩放级别以及图块大小。

还有更多...

您可能会发现，支持WMS和WFS协议的地图服务器可以栅格和矢量格式提供相同的信息。

想象一下一组存储在PostgreSQL / PostGIS和地图服务器（例如GeoServer）中的区域，其中有一些国家/地区被配置为通过WMS请求或矢量GML格式（通过WFS请求）用作栅格图像。

也可以看看

- 的**添加WMS层**从食谱第2章 (/book/web_development/9781785287756/2)，**添加栅格图层**
- 在**利用覆盖卸下或克隆特征**食谱

- 在**使用点要素作为标记**配方
- 在**阅读功能，直接使用**AJAX食谱

上一节 (/book/web_development/9781785287756/3/ch03lvl1sec34/adding-text-labels-to-geom

下一节 (/book/web_development/9781785287756/3/ch03lvl1sec36/using-the-cluster-strategy)
