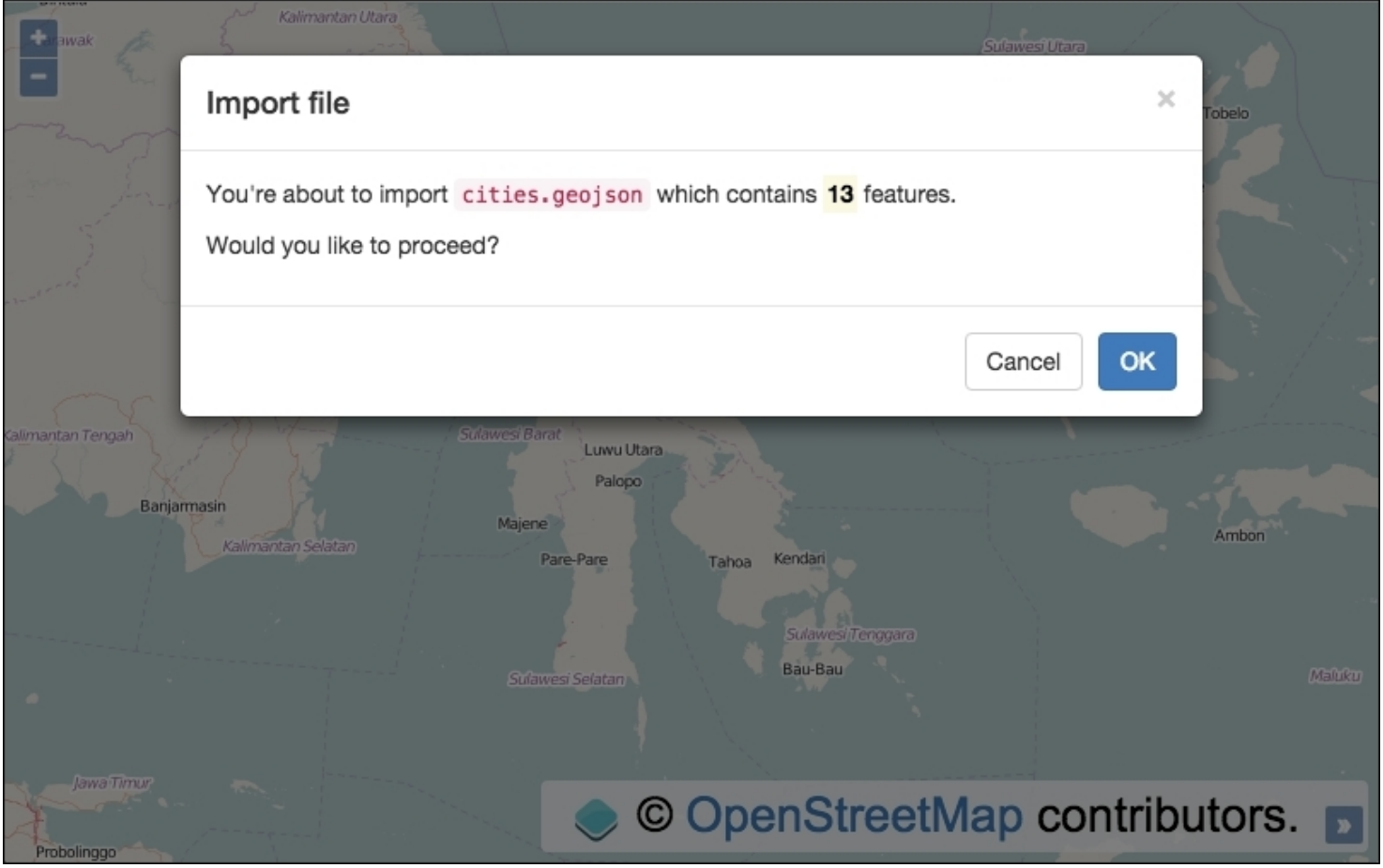
# 通过拖放将要素添加到矢量层

在这本书中，我们证明很多方法的功能添加到地图，如与**阅读功能，直接使用**AJAX的配方在第3章(/book/web_development/9781785287756/3)，**工作与矢量图层。**我们还看到了一个示例，该示例从GeoJSON格式的矢量图层中**导出要素**，**并**在第3章“ (/book/web_development/9781785287756/3)使用矢量图层”中的“将要素作为GeoJSON格式**导出”**配方的文本框中显示结果。对于本食谱，我们将向您展示如何通过从计算机中拖动文件并将其拖放到Web应用程序中来将包含GeoJSON的文件导入地图。

拖放是HTML5 API的一部分，该功能现已具有良好的跨浏览器支持。要阅读有关规范的更多信息，请参阅https://html.spec.whatwg.org/multipage/interaction.html#dnd (https://html.spec.whatwg.org/multipage/interaction.html#dnd)。要快速了解浏览器支持，请访问http://caniuse.com/#search=drag (http://caniuse.com/#search=drag)。

尽管您可以通过其他方法将文件从用户计算机导入地图，但在浏览器中处理客户端文件导入很方便。对于此配方，用户将能够将文件从其计算机拖到地图视图。在执行此操作时，系统将提示他们是否要继续导入，然后他们可以决定取消还是继续。它们将以模式显示导入的一些详细信息。

可以在中找到源代码 `ch07/ch07-drag-and-drop-import` ，这是用户在地图上拖动文件时的阶段的屏幕截图：




做好准备

对于模态 在屏幕截图中，我们使用了Bootstrap CSS和JavaScript库。请从 http://getbootstrap.com/getting-started (http://getbootstrap.com/getting-started)下载副本，或查看随附的源代码以获取依赖关系的副本。

Bootstrap有其自己的依赖项jQuery。因此，我们也需要将其拉入HTML文件。

怎么做...

1 创建一个HTML文件，并包含所有OpenLayers依赖项，jQuery和一个 `div` 包含地图的元素。如本食谱的“ **准备就绪**”部分所述，包括Bootstrap CSS和JavaScript依赖项。特别是，模态的一些标记如下：

复制

```
<div class="modal-body">
  <p>You're about to import <code id="js-filename"></code>
  which contains
  <mark><strong id="js-count"></strong></mark> features.</p>
  <p>Would you like to proceed?</p>
</div>
<div class="modal-footer">
  <button type="button" data-dismiss="modal">Cancel</button>
  <button id="js-confirmed" type="button" data-dismiss="modal">OK</button>
</div>
```

2 创建一个 自定义JavaScript文件，并 `map` 使用 `view` 实例和栅格图层实例化实例：

复制

```
var map = new ol.Map({
  view: new ol.View({
    zoom: 6, center: [13484714, -266612]
  }), target: 'js-map',
  layers: [
    new ol.layer.Tile({source: new ol.source.OSM()})
  ]
});
```

3 设置拖放交互并将其添加到 `map` ：

复制

```
var dragDrop = new ol.interaction.DragAndDrop({
  formatConstructors: [ol.format.GeoJSON]
});
map.addInteraction(dragDrop);
```

4 `click` 在 `OK` 模式按钮上订阅事件，并将导入的要素添加到新的矢量层：

复制

```
$('#js-confirmed').on('click', function() {
  var vectorSource = new ol.source.Vector({
    features: featuresToImport
  });

  map.addLayer(new ol.layer.Vector({
    source: vectorSource
  }));

  map.getView().fit(
    vectorSource.getExtent(),
    map.getSize()
  );
});
```

5 最后， `addfeatures` 从交互中订阅事件，更新一些值以在模式中显示，然后在屏幕上显示模式：

复制

```
var featuresToImport;
dragDrop.on('addfeatures', function(event) {
  featuresToImport = event.features;
  $('#js-filename').text(event.file.name);
  $('#js-count').text(featuresToImport.length);
  $('#js-modal').modal();
});
```

怎么运行的...

一些 为了简洁起见，省略了HTML，因此请查看随附的源代码以获取实现的完整详细信息。

让我们看一下新引入的OpenLayers交互和支持逻辑：

复制

```
var dragDrop = new ol.interaction.DragAndDrop({
  formatConstructors: [ol.format.GeoJSON]
});
map.addInteraction(dragDrop);
```

我们 `DragAndDrop` 从OpenLayers 实例化了一个新的交互实例。在幕后，将 `ol.interaction.DragAndDrop` 交互添加到地图后，OpenLayers使用Google Closure库构造了一个处理程序，用于使用构造函数从地图的视口检测到的拖放事件 `goog.events.FileDropHandler` 。此构造函数使用元素的单个参数来侦听拖放事件类型。OpenLayers将地图的 `div` 元素传递给构造函数 （ `map.getViewport()` ）。

该 `formatConstructors` 属性采用您希望接受的文件格式类型的数组。请注意，您在此处传递格式构造函数而不调用它。OpenLayers在处理导入的文件时，将为您调用格式构造函数，将其作为基础交互代码的一部分。

复制

```
$('#js-confirmed').on('click', function() {
  var vectorSource = new ol.source.Vector({
    features: featuresToImport
  });

  map.addLayer(new ol.layer.Vector({
    source: vectorSource
  }));

  map.getView().fit(
    vectorSource.getExtent(),
    map.getSize()
  );
});
```

我们 `click` 通过模式 `OK` 按钮订阅事件。我们的处理程序创建一个新的向量源并将该 `featuresToImport` 值分配给该 `features` 属性。该 `featuresToImport` 变量填充在 `addfeatures` 交互事件处理程序中，我们将在后面进行讨论。

然后使用之前的新来源创建新的矢量层，然后将其添加到地图。

为了方便用户，我们将地图更新到新导入的功能的范围。 `fit` from 的方法 `ol.View` 采用范围的第一个参数，我们从向量源中得出。第二个参数是我们适合范围的大小（宽度，高度）。我们希望提供这些尺寸的地图总大小，因为我们希望它利用所有可用空间。

复制

```
dragDrop.on('addfeatures', function(event) {
  featuresToImport = event.features;
  $('#js-filename').text(event.file.name);
  $('#js-count').text(featuresToImport.length);
  $('#js-modal').modal();
});
```

OpenLayers循环遍历放置在地图视口上的文件列表，从文件创建一系列要素，并调度 `ol.interaction.DragAndDropEventType.ADD_FEATURES` 我们订阅的事件。该 `event` 对象包含功能列表以及一些其他信息，例如文件名。

我们将特征存储在 `featuresToImport` 变量中，如在模式 `OK` 点击处理程序中所引用的那样。文件（ `event.file.name` ）的名称以及计算出的要导入的特征的长度都插入到DOM中。这些值构成模态内容的一部分。

最后，我们触发要显示的Bootstrap模式。如果用户不单击“ **确定”就**退出模态，则不会导入功能。但是，如果他们单击**OK**，则将 `click` 执行事件处理程序中的逻辑，将要素添加到新的矢量层。

## 也可以看看

- 第3章 (/book/web_development/9781785287756/3)，**使用矢量层中的**将**功能导出为GeoJSON**配方 (/book/web_development/9781785287756/3)
- 所述**变焦到层的程度**的配方在第3章 (/book/web_development/9781785287756/3)，**与矢量层工作**
- 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/9781785287756/7/ch07lvl1sec67/modifying-layer-appearance

下一节 (/book/web_development/9781785287756/7/ch07lvl1sec69/making-use-of-map-permal