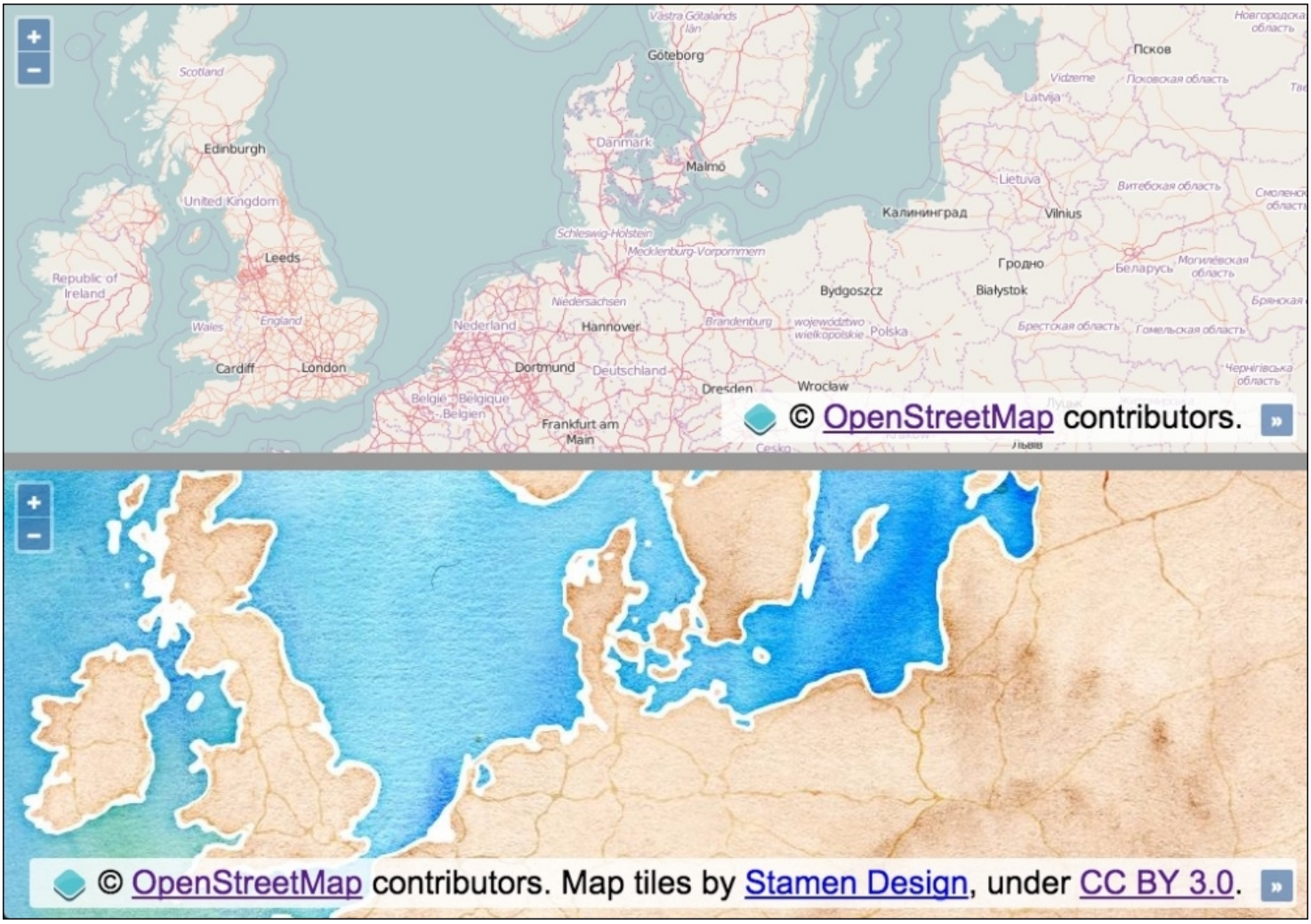
创建并排地图比较器

我们 将要创建一个地图比较器。目标是使两个地图来自不同的提供程序并排，并同步相同的位置和缩放级别。可以在中找到源代码 `ch04/ch04-map-comparator` 。这是我们两个同步地图的并排截图：



怎么做...

至 有两个地图同步工作，请执行以下步骤：

1 创建具有OpenLayers库依赖项的HTML文件。特别是，两个映射和分隔符的标记应如下所示：

复制

```
<div id="js-map1"></div>
<hr/>
<div id="js-map2"></div>
```

2 创建一个自定义JavaScript文件并设置共享 `view` ：

复制

```
var view = new ol.View({
  zoom: 5,
  center: [1252000, 7240000]
});
```

3 使用shared实例化第一个地图 `view` ，如下所示：

复制

```
var map1 = new ol.Map({
  view: view,
  target: 'js-map1',
  layers: [
    new ol.layer.Tile({source: new ol.source.OSM()})
  ]
});
```

4 完 通过使用相同的共享实例化第二张地图来关闭 `view` ：

复制

```
var map2 = new ol.Map({
  view: view,
  target: 'js-map2',
  layers: [
    new ol.layer.Tile({
      source: new ol.source.Stamen({layer: 'watercolor'})
    })
  ]
});
```

怎么运行的...

尽管CSS重要地并排显示了两个地图，但是我们将跳过样式实现的细节，以便专注于JavaScript。请查看随附的源代码以获得对布局背后的完整了解。

您可以看到，我们只需付出一点努力，就可以通过简单地共享视图来镜像地图的缩放和位置。我们抓住了多么容易，这是做的，当我们看着风**缓冲层数据时提高了地图导航**的食谱第2章(/book/web_development/9781785287756/2)，**添加栅格图层，**在解决方案还必须在同步两张地图。

OpenLayers 3已从任何给定的地图实例外部化了视图。这种松散的耦合设计决策使像这样的技术成为可能。 `view` 两个地图实例都引用了OpenLayers 对象。因此，当一张地图移动时，基础对象就会更新。这意味着新值也会在其他地图实例中显示。

作为学习练习，我们可以演示如何手动订阅缩放，以及如何从一个地图实例定位位置更改事件并将更改反映到另一个地图实例上。在此示例中，视图将不会共享：

复制

```
map1.getView().on(['change:resolution', 'change:center'], syncMap);

function syncMap(event) {
  if (event.type === 'change:resolution') {
    map2.getView().setZoom(event.currentTarget.getZoom());
  } else {
    map2.getView().setCenter(event.currentTarget.getCenter());
  }
};
```

对于在第一个地图实例中，我们握住了view（ `getView` ），订阅 `change:resolution` 和 `change:center` 事件，并提供了处理函数，即 `syncMap` 。在 `change:center` 当地图位置已经移动事件发布，并 `change:resolution` 在地图缩放级别已经改变事件发布。

这些事件发布后，我们的处理程序将被调用并传递一个 `event` 对象。该对象包含事件的类型（ `event.type` ）。这很重要，因为我们的处理程序捕获了多种类型的事件。因此，使用此信息，我们能够根据事件的类型确定要执行的操作。

该 `event` 对象还包含对已修改视图的引用，即 `event.currentTarget` 。我们使用对视图的引用来获取新的缩放值（ `getZoom` ）或新的中心坐标（ `getCenter` ）。我们使用第二个地图实例的对象的 `setZoom` 或 `setCenter` 方法将这些新值反映 `view` 在第二个地图实例上。

为简便起见，此代码已简化，因为您还需要从第二个地图实例中注册相同的事件类型，以便也可以更新第一个地图实例。

除了监听事件之外，如果愿意，我们也可以停止监听通知。

本 `ol.Observable` 类有 `un` 方法，以及该 `on` 方法。该 `un` 方法使我们可以从先前订阅的已发布事件中取消订阅侦听器功能。

如果继续我们的手动订阅示例，要取消订阅缩放级别的将来更改，我们可以执行以下操作：

复制

```
map1.getView().un('change:resolution', syncMap);
```

与 `on` 方法类似，该 `un` 方法允许您注销多个侦听器，因此 `on` ，如果我们想这样做，我们可以像为for那样传递一系列事件类型。

我们的 `syncMap` 事件处理程序将不再传递缩放级别事件类型。

还有一种 `ol.Observable` 称为的方法，该方法 `once` 提供订阅事件，仅侦听该类型的已发布事件，然后自动取消订阅将来通知的功能。

在某些情况下，这可能会非常有用。

也可以看看

- 第2章 (/book/web_development/9781785287756/2)，**添加栅格图层中**的**使用Bing图像**配方 (/book/web_development/9781785287756/2)
- 在**实现用于地图层的工作正在进行中的指示符**食谱
- 所述**矢量层听力设有事件**食谱

上一节 (/book/web_development/9781785287756/4/ch04lvl1sec39/introduction)

下一节 (/book/web_development/9781785287756/4/ch04lvl1sec41/implementing-a-work-in-pr

---