# WorldWind读取shapefile的API

|  |
| --- |
| **import** gov.nasa.worldwind.formats.shapefile.Shapefile;  **import** gov.nasa.worldwind.formats.shapefile.ShapefileRecord;  **import** gov.nasa.worldwind.geom.LatLon;  **import** gov.nasa.worldwind.util.VecBuffer;  **import** java.io.File;  **import** java.util.Map;  **public** **class** ReadShapefileTest {  /\*\*  \* 利用WorldWind相关API读取ESRI Shapefile(\*.shp)文件的例子。  \* **@param** args  \*/  **public** **static** **void** main(String[] args) {  // 从File构建Shapefile对象。  String userDir = System.*getProperty*("user.dir");  String filePath = userDir + "\\testData\\shapefiles\\bou2\\bou2\_4l.shp";  File shpFile = **new** File(filePath);    Shapefile sf = **new** Shapefile(shpFile);    // 遍历Shapefile对象中的所有记录  **try**  {  **while** (sf.hasNext())  {  // 拿到一条记录  ShapefileRecord r = sf.nextRecord();  **if** (r == **null**)  **continue**;  // 输出该条记录  *printShapefileInfo*(r);  }  }  **finally**  {  sf.close();  }  }    **public** **static** **void** printShapefileInfo(ShapefileRecord r)  {  System.*out*.printf("%d, %s: %d parts, %d points",  r.getRecordNumber(), /\*记录序号\*/  r.getShapeType(), /\*要素类型\*/  r.getNumberOfParts(), /\*要素分段数\*/  r.getNumberOfPoints() /\*要素包含点的个数\*/  );  // 输出属性数据  System.*out*.print("\n\t属性数据:\n\t\t");  **for** (Map.Entry<String, Object> entry : r.getAttributes().getEntries())  {  System.*out*.printf("%s = %s, ", entry.getKey(), entry.getValue());  }  System.*out*.println();  // 输出空间数据  System.*out*.println("\n\t空间数据:\n\t\t");  VecBuffer vb = r.getPointBuffer(0);  **for** (LatLon ll : vb.getLocations())  {  System.*out*.printf("\t\t经度: %f, 纬度: %f\n", ll.getLongitude().degrees, ll.getLatitude().degrees);  }  }  } |

# 从2D到3D

从shapefile读取到的数据都是平面上的点坐标，对于一个用经纬度描述的四边形，它的2D形式是这样的：



用5个点来描述一个四边形，其中第一个点和最后一个点是重合的，这样表示一个封闭的四边形。如果这个图形表示一幅平面图上某建筑物，而该建筑物的高度为H，那么便可以计算出该建筑物顶部5个点的坐标，从而这些坐标点就可以组成一个六面体，该六面体就可以用于在三维场景中展示此建筑。



# 一种拉伸算法

1. 假设建筑物高度为H，选边界点的第一个点为参考点；
2. 遍历所有边界点，如果该点是参考点执行（3），如果不是执行（4）；
3. 参考点的location即为底部点的坐标vBottom，然后计算参考点处的球面单位法向量**N**，再计算参考点到原点的距离vaLength，计算参考点处附加地形后的那一点vert到原点的距离vertLength，计算两者之差delta，按照如下公式确定顶部点坐标：
4. 计算该点附加地形后的那一点坐标vert到原点的距离vertLength，计算两者之差delta，按照如下公式确定顶部点坐标：

# 基于像素的视觉剔除算法



剔除的原理就是首先设置一个阀值R（以像素为单位），然后计算出物体的像素大小r，如果r<R则将该物体从渲染列表中剔除出去，从而减少需要渲染的物体的个数以达到加速渲染的目的。

计算物体像素大小的方法如下：

图中标出的各个变量在三维场景中是可以很方便的拿到的，那么物体的像素大小便可以用如下公式计算出来：