**[MapServer的安装和使用](http://blog.csdn.net/iwillsw/article/details/6037161" \o "MapServer的安装和使用)**

**http://blog.csdn.net/iwillsw/article/details/6037161**

**MapServer简介**

    在开源WebGIS领域，MapServer的历史和名气都超过GeoServer（20世纪90年代中期，由明尼苏达大学研制），很早就被列入OSGeo项目组。

与GeoServer不同，MapServer用C语言编写，采用传统的CGI架构，融合了Pro4j、GDAL等开源项目。对比J2EE架构的 GeoServer，MapServer显得非常精巧，5.6版本的核心部分只有33KB，源代码只有2.2M，包括Apache、Php、Pro4j、 GDAL、MapScript在内的完整安装包也只有35M。

**安装过程**

    和Java一样，C具有良好的跨平台特性，MapServer支持Windows、Linux、Mac OS X等操作系统。

    官方网站对Unix和Windows平台下的源代码安装步骤给出了详细说明。根据说明下载需要的第三方库，然后在Shell命令行模式下编译源代码，对于缺少Unix Shell或DOS操作经验的用户，这是一件比较头疼的事情。好在MapServer已经考虑到不习惯命令行操作的Windows用户，给出了Windows下的已编译安装包ms4w，目前最新版本是3.0beta11。

    ms4w的安装过程很简单，下载ms4w，解压缩到硬盘任意目录（目录名最好不要包含中文字符），打开命令提示符窗口，切换到ms4w所在目录，键入 apache-install即可。（提示：如果机器上安装有Apache或IIS，运行apache-install之前，请将它们关闭。ms4w安装后不会和原有的Apache冲突。）

看到如下信息，说明安装成功。

Installing the Apache MS4W Web Server service

The Apache MS4W Web Server service is successfully installed.

Testing httpd.conf....

Errors reported here must be corrected before the service

can be started.

The Apache MS4W Web Server service is starting.

The Apache MS4W Web Server service was started successfully.

安装完成后，使用浏览器访问 http://127.0.0.1/cgi-bin/mapserv.exe，MapServer返回如下提示信息

No query information to decode. QUERY\_STRING is set, but empty.

**发布地图数据**

    类似于Arcgis Server和GeoServer，MapServer采用零代码编写的配置文件方式管理地图发布，配置文件被称为Mapfile，后缀名为map。

    Mapfile有三种编写方式：

    1、参考官方文档Mapfile章节，手工编写

    这种方式要求开发人员对Mapfile的编写规则非常熟悉，否则很容易出错。

    2、类似Arcgis Server和GeoServer提供的Web图形化配置界面，MapServer有一个对应的开源项目MapLab，提供基于Web的图形化配置界面

    不过个人感觉，MapLab的易用性远逊于Arcgis Server和GeoServer。

    3、使用开源桌面平台QGIS，自动生成Mapfile文件

    对于新手，推荐使用第三种方式。

以下是测试用的Mapfile文件，MapServer对Mapfile文件的存放位置没有特殊要求，可以存放在不同的计算机上。

# Mapfile文件必须以MAP开头

MAP

# Map的名字 test

NAME test

STATUS ON

# 地图大小

SIZE 800 600

# Projection definition

# Projections are not currenlty supported. If desired, add your own # projection information based on Mapserver documentation.

# Map的坐标系

PROJECTION "init=epsg:4326"

END

# Map的全图范围

EXTENT 75 15 140 55

# Map的坐标单位 DD表示经纬度

UNITS DD

# Map的背景颜色 白色

IMAGECOLOR 255 255 255

# 生成的图片类型，常用gif或png

IMAGETYPE gif

# shp文件的路径，请使用绝对路径

SHAPEPATH "I://cn\_data"

#

# Start of web interface definition. Only the TEMPLATE parameter # must be specified to display a map. See Mapserver documentation

# MapServer内置了地图客户端功能，此处是Web客户端相关设置。

# 由于我们使用Openlayers作为地图客户端，MapServer做为地图服务器，此处不做设置

WEB

# HEADER

# TEMPLATE

# FOOTER

# 最小、最大比例尺等级

MINSCALE 1

MAXSCALE 13

# 设置IMAGEPATH，默认路径如下，请使用绝对路径

IMAGEPATH 'I:/ms4w/Apache/htdocs/tmp'

# 设置IMAGEURL，默认路径如下

IMAGEURL '/tmp/'

END

# 设置图层，一个Map下可以包含多个LAYER

LAYER NAME polyline  # 图层名称，MapServer使用该名称

TYPE LINE # 几何类型

STATUS ON

DATA "roa\_4m" # shp文件名，不需要带扩展名，路径在前面的SHAPEPATH项中指定

CLASS NAME "roa\_4m" # 类名

# TEMPLATE

COLOR 112 0 0 # 颜色

END

END

END # Map File

**客户端测试**

    接下来测试刚才建立的Mapfile，在浏览器地址栏中输入 http://127.0.0.1/cgi-bin/mapserv.exe?LAYERS=polyline&MAP=I:/ms4w/Apache/htdocs/my.map&FORMAT=gif&SERVICE=WMS&VERSION=1.1.1&REQUEST=GetMap&STYLES=&EXCEPTIONS=application%2Fvnd.ogc.se\_inimage&SRS=EPSG%3A4326&BBOX=75,15,140,55&WIDTH=800&HEIGHT=600

大约0.5秒后（初次加载），浏览器显示出中国1：400万道路网，地图发布成功！熟悉OGC标准的朋友可能已经发现，上面的URL是WMS地图请求。

直接通过CGI请求，得到的是一幅静态地图，为了实现地图的缩放、漫游，我们选用Openlayers地图客户端。通过OpenLayers.Layer.WMS可以直接调用MapServer地图服务，测试代码如下：

var ms\_layer = new OpenLayers.Layer.WMS(

   "polyline", //OpenLayers中的图层名

   "/cgi-bin/mapserv.exe", //MapServer地图服务器的路径

   { layers: 'polyline', //Mapfile中定义的图层名

     map: 'I:/cn\_data/my.map', //Mapfile文件的绝对路径，注意必须使用'/'而不是'/'，使用相对路径无法显示地图

     format: 'gif' },

   { reproject: false,

     'numZoomLevels': 20,

     gutter: 15,

     buffer: 0 }

);

滚动鼠标滚轮，地图的刷新速度保持在0.5秒以内。MapServer继承了C语言的特点，地图渲染效率很高。

**写在最后**

    MapServer提供两种工作方式，CGI方式（适用于CGI、AJAX、FLEX开发人员）和MapScript方式（适用于Php、Java、 C#、Python开发人员）。以原生CGI方式效率最高，配合TileCache，可以快速生成大范围的地图瓦片数据。比较基于.Net和J2EE的商业或开源平台，MapServer更适合高负荷的大型互联网地图应用。

[**MapServer使用笔记（一）**](http://blog.csdn.net/iwillsw/article/details/6037163)

**连接Oracle Spatial**

前面的测试中我们使用shape文件作为地图数据源，下面我们将从Oracle Spatial空间数据库读取地理数据。

mapserver提供两种方式连接oracle空间数据库，native oracle spatial和ogr。安装包提供了支持native oracle spatial的二进制文件，ogr方式需要自行编译源代码。

借用以往项目的Oracle 10.2.0.3数据库，MapServer采用ms4w\_2.3.1编译包，地图服务器安装Oracle10.2客户端，进行本地网络配置，测试连接正常。

关闭Apache MS4W Web Server服务，进入/ms4w/Apache/cgi-bin/ignored-libmap/oracle11g目录，将libmap.dll复制粘贴到/ms4w/Apache/cgi-bin目录，替换原有文件，启动Apache MS4W Web Server服务。

新建一个Mapfile文件，LAYER部分如下

LAYER

    NAME "road"

    TYPE LINE

    CONNECTION "geouser/123456@geodb" #数据库连接字符串

    CONNECTIONTYPE oraclespatial      #连接类型为oracle spatial

    DATA "GEOLOC FROM NAV\_ROAD1 USING SRID 4326"  #此处GEOLOC为GEOMETRY字段名，NAV\_ROAD1为表名

    PROJECTION

       "init=epsg:4326" #必须指定SRS编码

    END

    DUMP TRUE

    CLASS

        STYLE

            COLOR 0 128 128

        END

    END

END

DATA标签使用如下格式

"[geom\_column]

FROM

[table]| [(SELECT [...]

FROM [table]|[Spatial Operator]

[WHERE condition] )]

[USING [UNIQUE column]| [SRID #srid]| [FUNCTION]| [VERSION #version]

]"

注意：不要在FROM之前加入属性字段名，属性字段可添加在FROM后的子查询中，如下所示

GEOLOC FROM (SELECT MI\_PRINX AS OID, NAME AS ONAME, COLOR AS COLOR, GEOLOC AS GEOLOC FROM NAV\_ROAD1) USING SRID 4326

LABEL或STYLE用到的属性字段必须包含在SELECT子查询中。

修改openlayers客户端代码

var ms\_layer = new OpenLayers.Layer.WMS(

"Test Map",

"/cgi-bin/mapserv.exe",

                {

                    layers: 'road',

                    map: 'I:/cn\_data/c2.map',

                    format: 'png'

                },

                {

                    reproject: false,

                    'numZoomLevels': 20,

                    gutter: 15,

                    buffer: 0

                }

            );

打开浏览器，查看效果。

**为图层添加标注（Label）**

在地图上标注文字，必然涉及到字体，MapServer采用与操作系统无关的设计，不能自动获知当前操作系统下安装了哪些字体，必须手工创建字体集FONTSET文件。

该文件的格式非常简单，每行配置一个字体，左边是字体名（可自由命名），右边是字体文件的路径（请使用绝对路径），如下所示：

arial   C:/WINDOWS/Fonts/arial.ttf

sans    C:/WINDOWS/Fonts/SIMSUN.TTC

将以上内容保存为文本文件，文件名和扩展名不限，例如保存为fonts.txt

在Mapfile的Map标签下添加一行

FONTSET fonts.txt #指明字体集文件，可以包含路径

在Mapfile的Layer标签中添加一行

LABELITEM "NAME"  #指明标注所用的字段名

在Layer下的CLASS标签中添加以下内容

LABEL

    COLOR  0 0 0

    OUTLINECOLOR 255 255 255

    FONT "sans"  #此处sans对应宋体SIMSUN.TTC

    TYPE truetype

    SIZE 6

    POSITION AUTO

    PARTIALS FALSE

    ENCODING GBK #此处为文字编码，如果不设置该值，中文将不能正确显示。对于中文Windows系统，操作系统的默认编码为GBK。

END

注意：ms4w\_3.0beta11属于开发版本，支持shape文件数据源配置label标签，不支持oracle spatial数据源配置label标签。

为了读取oracle空间表中的属性生成标签，我们换用ms4w\_2.3.1正式版，系统工作正常。

**使用动态样式（STYLE）**

实际应用中对地图的渲染有很多要求，MapServer支持动态设置STYLE，修改Mapfile文件LAYER的STYLE，例如：

STYLE

    COLOR [COLOR] #COLOR必须使用[r] [g] [b]十进制格式，MapServer不能识别Web开发中常用的RGB十六进制格式

END

每条道路的颜色由道路表的属性字段COLOR决定，与之类似，可以从数据库取值填充SYMBOL,SIZE,ANGLE等标签。

[**MapServer使用笔记（二）**](http://blog.csdn.net/iwillsw/article/details/6037643)

**为MapServer添加瓦片缓存服务**

国际上有两个比较流行的开源地图瓦片缓存服务器：geowebcache和tilecache。

geowebcache基于J2EE架构，具有完整的Web图形管理界面，支持多进程并发切图，可以随时查看切图进度和剩余时间，搭配geoserver非常合适。

titlecache采用python编写，CGI工作模式，非常小巧，2.11版本只有令人惊讶的68K!(这一点很像MapServer)。

titlecache采用SHELL或DOS命令行方式运行切图程序，实时输出切图进度信息。

mapserver选择titlecache作为地图切片缓存服务器，在ms4w\_3.0beta10中包含了tilecache插件，由于beta版本存在问题，

我们在ms4w\_2.3.1正式版中手工加入tilecache服务器，具体步骤如下：

1、从tilecache官方网站下载安装包tilecache-2.11.tar.gz，该安装包适用于所有操作系统，我们使用Windows XP。

2、解压安装包，将其中的tilecache-2.11目录整体复制到ms4w的apps目录下。

3、进入ms4w的http.d目录，新建一个名为httpd\_tilecache.conf的文本文件，内容如下：

Alias /tilecache/ "/ms4w/apps/tilecache-2.11/"

<Directory "/ms4w/apps/tilecache-2.11/">

  AllowOverride None

  Options ExecCGI

  Order allow,deny

  Allow from all

</Directory>

addHandler cgi-script .cgi .py

4、进入ms4w/apps/tilecache-2.11目录，打开tilecache.cgi文件，修改第一行的python编译器路径，例如

#!E:/Python25/python.exe

5、打开该目录下的tilecache.cfg文件，修改cache标签，指定本地磁盘缓存目录，例如

[cache]

type=Disk

base=F:/tmp/tilecache

添加一个road标签，指定地图服务类型，服务地址，mapfile文件路径，图层名称，栅格格式，空间参考系统编码，例如

[road]

type=WMS

url=http://127.0.0.1/cgi-bin/mapserv.exe?map=I:/cn\_data/c2.map&transparent=true&

layers=road

extension=png

srs=EPSG:4326

6、重新启动Apache

7、打开tilecache下的index.html文件，根据实际情况修改openlayers.js的路径，例如

<script src="/openlayers/OpenLayers.js"></script>

修改layer参数，例如

layer = new OpenLayers.Layer.WMS( "road", "tilecache.cgi?", {layers: 'road', format: 'image/png' } );

使用浏览器访问http://127.0.0.1/tilecache，Web页面上将显示经过tilecache处理的地图，客户端浏览过的地图切片文件被保存在tilecache.cfg文件指定的临时目录下，这里是F:/tmp/tilecache目录，如果其中包含客户端正在请求的地图，这些地图将不再通过MapServer动态生成，而是直接从缓存目录读取静态图片文件。对于大用户量、高并发的地图访问，瓦片缓存机制可以非常有效的提高客户端读取、显示地图的速度。

通过执行tilecache\_seed.py脚本，可以手动生成地图切片，下面的命令给road图层发布0到6级的地图切片文件。

python "F:/ms4w/apps/tilecache-2.11/tilecache\_seed.py" road 0 6

等待执行完毕，进入缓存目录可以找到刚才发布的地图切片文件。

[**MapServer使用笔记（三）**](http://blog.csdn.net/iwillsw/article/details/6048104)

**发布** **WFS** **矢量地图**

OGC WFS协议定义了地图客户端查询和传送矢量数据的方法，在查询、分析、动态绘图等实际应用中非常有用。MapServer支持OGC WFS 1.0.0和1.1.0规范。

重新定义一个Mapfile文件，命名为ext2.map，内容如下：

MAP

  NAME "vector"

    SHAPEPATH "I://cn\_data"

    FONTSET fonts.txt

    IMAGECOLOR 255 255 255

    IMAGETYPE agg

    SIZE 800 600

    STATUS ON

    UNITS DD

    EXTENT 115.275 39.2204 117.475 40.9462

    OUTPUTFORMAT

        NAME agg

        DRIVER AGG/PNG

        IMAGEMODE RGB

        FORMATOPTION "INTERLACE=false"

        MIMETYPE "image/png"

    END

    PROJECTION

        "init=epsg:4326"

    END

    WEB

        METADATA

            "wms\_title"                       "road wms"

            "wfs\_title"                       "road wfs"

            "wms\_onlineresource"              "http://192.98.151.23/cgi-bin/mapserv.exe?"  #mapserver服务器的url

            "wms\_srs"                         "EPSG:4326"   #地图坐标系

        END

    END

    LAYER

        NAME "road"

        METADATA

            "wms\_title"                       "road"

            "wfs\_title"                       "road"

            "wms\_srs"                         "EPSG:4326"  #图层坐标系

            "gml\_include\_items"             "all"

            "gml\_featureid"                 "id" #必须指定id

        END

        STATUS ON

        DATA "roa\_4m.shp"

        TYPE line

        DUMP TRUE

        CLASS

            STYLE

                COLOR "#00FF00"

            END

        END

    END

END

该文件中有两组METADATA标签，是WFS（WMS） Server所需要的，分别是MAP的METADATA标签和LAYER的METADATA标签。

前者是针对整个地图的全局定义，后者是每个图层的元数据定义。

保存Mapfile文件，打开浏览器进行测试

http://192.98.151.23/cgi-bin/mapserv.exe?map=I:/cn\_data/ext2.map&SERVICE=WFS&VERSION=1.0.0&REQUEST=GetCapabilities

正常情况下，可以看到以下返回信息

<WFS\_Capabilities version="1.0.0" updateSequence="0" xsi:schemaLocation="http://www.opengis.net/wfs http://schemas.opengis.net/wfs/1.0.0/WFS-capabilities.xsd">

<!--

 MapServer version 5.2.1 OUTPUT=GIF OUTPUT=PNG OUTPUT=JPEG OUTPUT=WBMP OUTPUT=PDF OUTPUT=SWF OUTPUT=SVG SUPPORTS=PROJ SUPPORTS=AGG SUPPORTS=FREETYPE SUPPORTS=ICONV SUPPORTS=FRIBIDI SUPPORTS=WMS\_SERVER SUPPORTS=WMS\_CLIENT SUPPORTS=WFS\_SERVER SUPPORTS=WFS\_CLIENT SUPPORTS=WCS\_SERVER SUPPORTS=SOS\_SERVER SUPPORTS=FASTCGI SUPPORTS=THREADS SUPPORTS=GEOS SUPPORTS=RGBA\_PNG INPUT=JPEG INPUT=POSTGIS INPUT=ORACLESPATIAL INPUT=OGR INPUT=GDAL INPUT=SHAPEFILE

-->

<Service>

<Name>MapServer WFS</Name>

<Title>road wfs</Title>

<OnlineResource>

http://192.98.151.23/cgi-bin/mapserv.exe?map=I:/cn\_data/ext2.map&

</OnlineResource>

</Service>

<Capability>

<Request>

<GetCapabilities>

<DCPType>

<HTTP>

<Get onlineResource="http://192.98.151.23/cgi-bin/mapserv.exe?map=I:/cn\_data/ext2.map&"/>

</HTTP>

</DCPType>

<DCPType>

<HTTP>

<Post onlineResource="http://192.98.151.23/cgi-bin/mapserv.exe?map=I:/cn\_data/ext2.map&"/>

</HTTP>

</DCPType>

</GetCapabilities>

<DescribeFeatureType>

<SchemaDescriptionLanguage>

<XMLSCHEMA/>

</SchemaDescriptionLanguage>

<DCPType>

<HTTP>

<Get onlineResource="http://192.98.151.23/cgi-bin/mapserv.exe?map=I:/cn\_data/ext2.map&"/>

</HTTP>

</DCPType>

<DCPType>

<HTTP>

<Post onlineResource="http://192.98.151.23/cgi-bin/mapserv.exe?map=I:/cn\_data/ext2.map&"/>

</HTTP>

</DCPType>

</DescribeFeatureType>

<GetFeature>

<ResultFormat>

<GML2/>

</ResultFormat>

<DCPType>

<HTTP>

<Get onlineResource="http://192.98.151.23/cgi-bin/mapserv.exe?map=I:/cn\_data/ext2.map&"/>

</HTTP>

</DCPType>

<DCPType>

<HTTP>

<Post onlineResource="http://192.98.151.23/cgi-bin/mapserv.exe?map=I:/cn\_data/ext2.map&"/>

</HTTP>

</DCPType>

</GetFeature>

</Request>

</Capability>

<FeatureTypeList>

<Operations>

<Query/>

</Operations>

<FeatureType>

<Name>road</Name>

<Title>road</Title>

<SRS>EPSG:4326</SRS>

<LatLongBoundingBox minx="80.3869" miny="18.2823" maxx="132.515" maxy="49.6272"/>

</FeatureType>

</FeatureTypeList>

<ogc:Filter\_Capabilities>

<ogc:Spatial\_Capabilities>

<ogc:Spatial\_Operators>

<ogc:Equals/>

<ogc:Disjoint/>

<ogc:Touches/>

<ogc:Within/>

<ogc:Overlaps/>

<ogc:Crosses/>

<ogc:Intersect/>

<ogc:Contains/>

<ogc:DWithin/>

<ogc:BBOX/>

</ogc:Spatial\_Operators>

</ogc:Spatial\_Capabilities>

<ogc:Scalar\_Capabilities>

<ogc:Logical\_Operators/>

<ogc:Comparison\_Operators>

<ogc:Simple\_Comparisons/>

<ogc:Like/>

<ogc:Between/>

</ogc:Comparison\_Operators>

</ogc:Scalar\_Capabilities>

</ogc:Filter\_Capabilities>

</WFS\_Capabilities>

**在OpenLayers中加载WFS图层**

OpenLayers是一个功能强大的Web地图客户端引擎。

接下来，我们利用OpenLayers的OpenLayers.Protocol.WFS类，显示刚才发布的WFS图层，代码如下

       var protocol = OpenLayers.Protocol.WFS({

                    url: "http://192.98.151.23/cgi-bin/mapserv.exe?map=I:/cn\_data/ext2.map&", //mapserver地图服务器的url，加上mapfi了文件的路径

                    featureType: "road", //layer的名称

                    srsName: "EPSG:4326", //layer的坐标系

                    geometryName: "msGeometry", //geometry字段的名称

                    featurePrefix: "ms"

                });

以下是全部javascript代码：

        window.onload = function() {

                var map = new OpenLayers.Map('mapdiv', {

                    projection: "EPSG:4326",

                    units: "degree"

                });

                var roads = new OpenLayers.Layer.WMS("road", "http://192.98.151.23/cgi-bin/mapserv.exe?map=I:/cn\_data/ext2.map&", {

                    layers: 'road',

                    transparent: 'true',

                    format: 'image/png'

                }, {

                    isBaseLayer: false,

                    visibility: true,

                    buffer: 0

                });

                var empty = new OpenLayers.Layer("empty", {

                    isBaseLayer: true

                });

                var select = new OpenLayers.Layer.Vector("Selection", {

                    styleMap: new OpenLayers.Style(OpenLayers.Feature.Vector.style["select"]),

                    displayInLayerSwitcher: false

                });

                var hover = new OpenLayers.Layer.Vector("Hover", {

                    displayInLayerSwitcher: false

                });

                map.addLayers([roads, hover, select, empty]);

                var protocol = OpenLayers.Protocol.WFS({

                    url: "http://192.98.151.23/cgi-bin/mapserv.exe?map=I:/cn\_data/ext2.map&",

                    featureType: "road",

                    srsName: "EPSG:4326",

                    geometryName: "msGeometry",

                    featurePrefix: "ms"

                });

                var control = new OpenLayers.Control.GetFeature({

                    protocol: protocol,

                    box: true,

                    hover: true,

                    multipleKey: "shiftKey",

                    toggleKey: "ctrlKey"

                });

                control.events.register("featureselected", this, function(e){

                    select.addFeatures([e.feature]);

                });

                control.events.register("featureunselected", this, function(e){

                    select.removeFeatures([e.feature]);

                });

                control.events.register("hoverfeature", this, function(e){

                    hover.addFeatures([e.feature]);

                });

                control.events.register("outfeature", this, function(e){

                    hover.removeFeatures([e.feature]);

                });

                map.addControl(control);

                control.activate();

                map.addControl(new OpenLayers.Control.LayerSwitcher());

                map.addControl(new OpenLayers.Control.MousePosition());

                if (!map.getCenter()) {

                    map.zoomToMaxExtent();

                }

            }

在浏览器中运行html页面，国道数据将以矢量方式传送到客户端，由openlayers动态绘制，鼠标移动到道路上或者选中道路，道路将实时改变颜色，在Web上实现与传统桌面GIS应用类似的体验效果。

说明：如果将shape数据源换成native oracle spatial数据源，openlayers客户端将无法获取数据，这是mapserver的一个bug，该问题在5.6.4版本得到修正。