
目 录

简介	3
1. 准备工作	5
1.1 开发环境.....	5
1.1.1 二次开发包.....	5
1.1.2 开发工具.....	6
1.1.3 发布与调试.....	8
1.2 必须的配置.....	8
2. 开发流程	9
3. 二次开发	11
3.1 Seemap 二次开发工程目录及文件说明.....	11
3.2 二次开发基础对象与接口的介绍.....	12
3.3 接口说明.....	13
设置地图操作状态.....	13
鼠标滚轮操作.....	14
设置当前地图显示区域.....	14
获得当前地图所有图层名.....	14
获得当前地图指定图层的字段名.....	14
设置指定图层为可选状态.....	15
设置指定图层的可见状态.....	15
获取选定对象的 ID 值.....	15
获取选定对象的所在的图层名.....	16
获取选定对象的面积.....	16
获取选定对象的字段名.....	16
获取选定对象的字段属性值.....	16
清除全部选定.....	16
设置地图视野.....	16

对象选择.....	17
对象查询.....	18
获取对象 GISID.....	21
周边查询.....	21
获得地图宽度像素.....	22
获得地图高度像素.....	22
扩展外部数据集的创建.....	22
扩展外部数据集的添加.....	23
扩展外部数据集的删除.....	24
获取指定对象的类型.....	24
获取指定对象的节点数组.....	25
地图数据的插入.....	25
地图数据的更新.....	25
设置单个对象特殊显示属性.....	26
清除对象特殊显示属性.....	26
批量设置对象特殊显示属性.....	27
聚焦对象.....	27
重绘地图.....	27
3.4 二次开发常见问题说明:	27
3.4.1 JavaScript (JS) 与 applet (SeemapApplet) 的调用问题.....	27
3.4.2 二次开发中, 页面接口的定义 (用于供页面调用的函数).....	30
3.4.3 关于数组的页面接口中的处理.....	31
3.4.4 关于回调函数.....	33
3.4.5 关于查询.....	34
4. 示例讲解	39
4.1 二次开发中 java 程序的开发及对应的页面功能说明... ..	39
4.2 页面文件说明.....	57

SEEMAP FREE 二次开发文档

简介

Seemap Free 是神州地理公司开发的新一代网络地理系统生成器，使非专业用户能够轻松定制和发布专业地理信息系统。Seemap Free 拥有功能强大的地图编辑功能，内嵌自配适 Web 服务器系统，提供方便快捷的数据库连接功能，预定制多个专业的地理信息系统模板，同时开放丰富的二次开发接口满足不同用户群体的需求。

Seemap Free 平台软件包括浏览器端系统，中间件，后台服务管理程序三部分。浏览器端系统 (Seemap Free Web) 主要任务：执行解释、显示和交互操作地图；中间件 (Seemap Free Middle) 主要功能为连接池作用，并实现对数据库信息安全访问，担当客户端一部分非常规图形的操作工作。后台服务管理程序 (Seemap Free Server) 负责对全部地理数据在线维护。该系统两端均支持矢量地图。

Seemap Free 平台软件系统的数据组织设计，充分考虑到海量地理信息数据的管理和特色各异的应用，其核心模块采用空间数据管理和应用操作接口相分离的设计原则。空间数据管理模块管理数据的接收，处理，调度和发送，保证数据的完整性和一致性；操作和数据单元交换等应用程序接口设计基于空间数据管理模块抽象，应用程序接口屏蔽技术实现细节。核心层与应用层隔离，逻辑层与业务层隔离；例如城市地理数据划分为四个层次即公共图形数据，业务应用图形数据，公共企业数据，业务应用企业数据，四类数据有统一坐标系，可分布在不同服务器上；同时平台客户端解释器的大小仅为约 90k 字节，这利于窄带传输和初始化启动；整个 Seemap Free 系统两端各自独立运行并相互协调，系统分布式处理，避免系统阻塞，提高系统性能，降低系统投资成本。浏览器中 java applet 的出现改变了浏览器脚本语言交互操作能力低效的局面。Seemap Free 平台软件所拥有的功能除包含常用 GIS 软件的功能外，其自

身主要功能特色还包括：双重空间数据管理，系统采用文件系统和数据库系统同时对空间几何数据存储与管理；多元数据共享，来源不同服务器具有统一坐标系数据在客户端实现无缝拼接；数据自由交换，支持常用地理数据转换，交换精度可自定义，支持公交数据的交换；符号制作可自定义图标符号；系统采用 Seemap Free 平台软件图像压缩格式存储和传输；自定义压缩格式的地理数据存储大小是常用 GIS 软件的 1/10 到 1/20 倍；数据缩放操作，建立当前视图显示的视图比例与图幅索引关联关系；提供多种条件模糊查询与地图上准确定位；提供最优路径与线形数据在地图上可视化编辑功能；中间件模块提供对客户端与数据库服务器端两端标准访问 sql 语句接口；系统可设定用户与地理数据的访问与修改权限，限定对数据访问，维护和显示管理权限。

准备工作

对 **Seemap Free** 进行二次开发，需要如下的准备工作：

1.1 开发环境

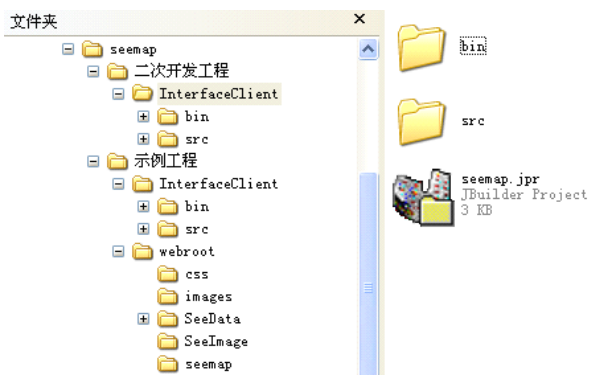
开发环境包括软、硬件环境：

软件环境：

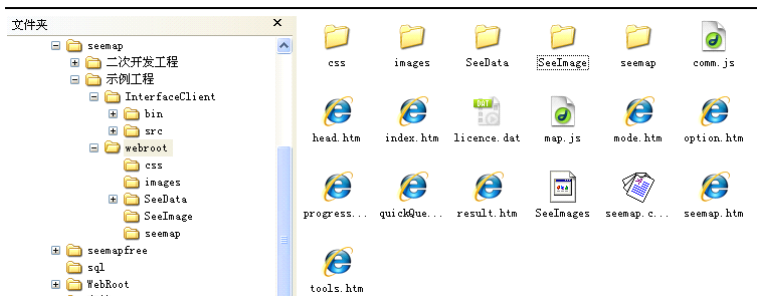
1.1.1 二次开发包

首先您需要获取 Seemap Free 二次开发包，它是使用 JAVA 编写的一个拥有 GIS 功能的程序包，能够读取通过 Seemap Free 产生的*.see 地图文件，并在 INTERNET 上显示。提供的二次开发包，主要是两个 Jbuilder 工程文件，其中包含了一个二次开发模板和一个简单示例。用户也可以将其中的内容，转移到其他平台进行开发。

提供的二次开发包中包含以下内容：



Webroot 客户端页面开发示例目录：



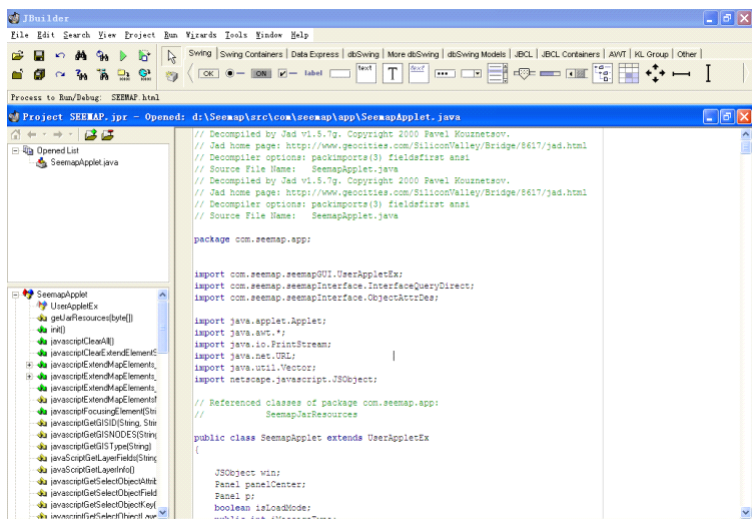
其中“二次开发工程”目录下的“InterfaceClient”，是提供的二次开发的 Jbuilder 工程目录，其下的“bin”目录中包含了提供的 Seemap Free 二次开发的客户端.class 文件，“src”目录包含了用于开发的 java 文件。在“示例工程”中，“InterfaceClient”目录结构一样，在其 src 中是开发好的一个示例，用户可以用来参考。示例工程中，还包含了一个已经开发好的页面客户端，将其拷贝到 SeemapFree 的“本地 WEB 服务器”目录下，就可以通过 <http://127.0.0.1:8840/index.htm> 访问此示例了。（具体端口与路径根据用户的选择而定）。

1.1.2 开发工具

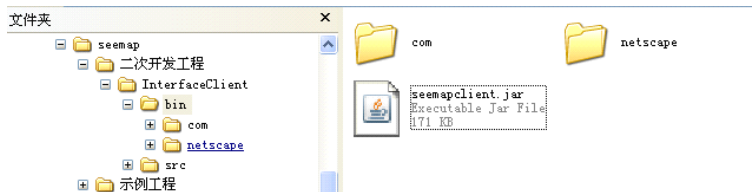
程序开发建议使用较低版本的 Jbuilder，为了获取最大程度的兼容性，在武汉神州地理软件开发有限公司网站 www.seemap.net 上可以找到 Jbuilder 的下载链接。此工具使用的是 JAVA1.3 的 JDK，这样可以兼容大部分的 java 虚拟机。用户也可以根据情况，使用其他开发工具，但为保证兼容性，请使用低版的 jdk 来编译。

Jbuilder 3 的下载与安装。在网站 www.seemap.net 中有 Jbuilder 3 的下载链接，用户也可以到网上搜索下载并安装。

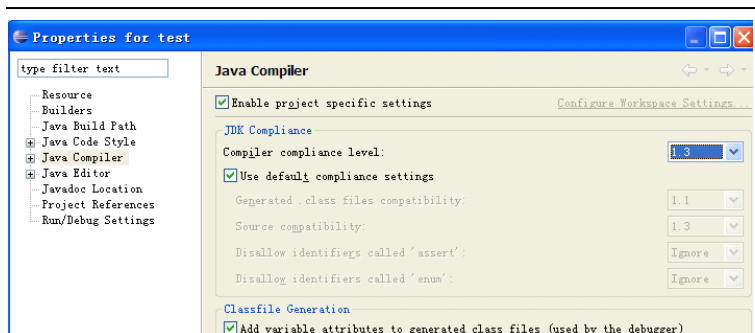
在安装好 Jbuilder 3 后，无需作进一步配置，就可以直接打开所提供的二次开发包中的 Jbuilder 工程文件 seemap.jpr。打开之后的界面如下：



使用其他开发工具，如 eclipse.exe



用户也可以新建一个 eclipse 的 java 项目，将二次开发目录中的 src 下的 java 源文件，拷贝到 eclipse 项目中的 src 目录下，并引入 bin 目录中的 seemapclient.jar 包。就可以用 eclipse 来开发了，但为了保证开发出来的程序的最大兼容性，应尽量选择低版本的 jdk，以便程序能运行在低版本的 java 虚拟机上。在 eclipse 中可以通过：菜单 Project->Properties->Java Compiler 中来设置，建议将“Compiler compliance level”设置为 1.3 即可。



1.1.3 发布与调试

发布与调试环境需要支持 APPLET 的浏览器，如 IE, Firefox 等主流浏览器，推荐使用 ie。发布则需要有 WEB 服务程序的支持，如 IIS、Tomcat 等，Seemap Free 基础地图属于静态页面；但是如果需要配置地图数据库的话，则需要支持 Servlet 的 web 容器。如 Tomcat。

硬件环境：

发布系统由于使用了分布式的处理机制，所以对服务器的要求较低，以下为最低的发布要求：

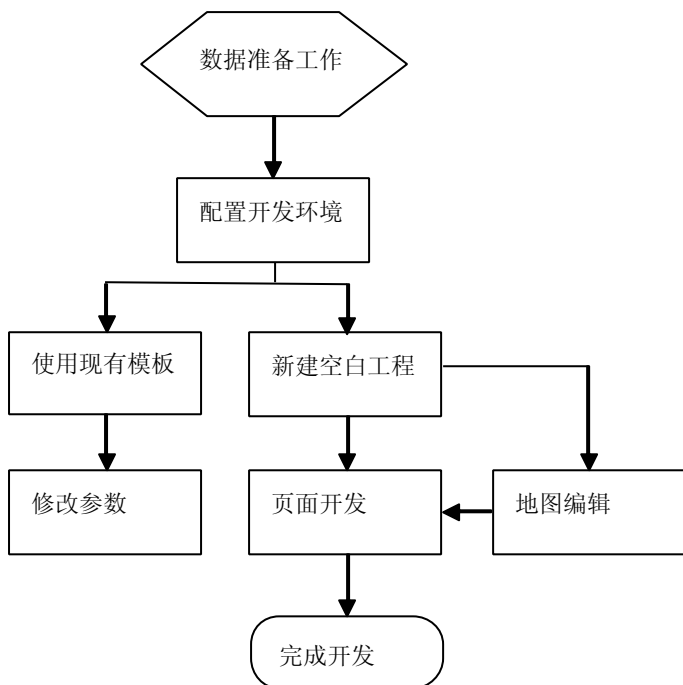
服务器端： P4 以上芯片，内存 512M 以上服务器一台

1.2 必须的配置

当满足所需的开发环境后，需要进行如下的配置工作：

- 1 java 环境配置；
- 2 开发工具配置；
- 3 发布工具配置；

2 开发流程



二次开发的流程大致如下：

- (1) 配置好开发环境，通过 Seemap Free 创建好地图数据 XXX. see 或是 XXX. see 目录下的分级数据。
- (2) 对 SeemapApplet.java 功能性开发，为页面提供功能性调用函数。为保证程序的兼容性，建设将这些函数都放在 SeemapApplet 中。比如提供一个函数，来让用户“改变当前地图视野”。

就可以在 SeemapApplet 中添加此函数，

```
public void javascriptSetMapView (float x,float y,  
int z)
```

```
{
    userMainMapControl.userSetMapView( x, y, z)
}
```

让页面中通过调用此函数，来实现“改变当前地图视野”的功能。

在做好对 java 程序的开发后，将生成的所有.class 文件包，即开发工程中的 classes 目录下的文件。以 ZIP 方式打包，比如打包成 seemap.class。

- (3) 在页面开发，让地图及其功能在网页中展现出来。将之前生成的 seemap.class 中的 Applet (即之前开发的 SeemapApplet.java)，及之前生成的地图文件 XXX.see，通过如下代码引入到页面中。

```
<applet code= "com.seemap.app.SeemapApplet.class" Archive=
"seemap.class" name= "Emap" width= "100%" height= "100%"
onmousewheel= " return Emap.javascriptWheel(wheelFlag()) "
MAYSCRIPT>
    <PARAM NAME= "APPLANGUAGE" VALUE= "cn" >
    <PARAM NAME= "APPINDEX" VALUE= "0" >
    <PARAM NAME= "APPSERVICEPORT" VALUE= "8080" >
    <PARAM NAME= "APPMAPNAME" VALUE= "XXX.see" >
    <PARAM NAME= "APPMAINMAPNAME" VALUE= "XXX.see" >
    <PARAM NAME= "APPMAPDESCRIPTION" VALUE= "SEEMAP 地图"
>
</applet>
```

如果用户想要改变当前地图视野为，以经纬度坐标 centerX, centerY 为中心，以 z 为视野高度的地图视野时，在页面中能过 JS 调用 Emap.javascriptSetMapView(centerX, centerY, z) 即可。

3 二次开发

3.1 Seemap 二次开发工程目录及文件说明

在这里，为了保证产生的程序在低版本的 JAVA 虚拟机中可以使用，默认提供的开发包使用了 JBuilder3 作为开发工具。在提供的二次开发包中，在“二次开发”接口中，有如下

工程文件夹：InterfaceClient

文件夹 InterfaceClient 包含了所有用于二次开发的 Jbuilder 开发工程。

二次工程目录结构：

<InterfaceClient>-----InterfaceClient.jpr (Jbuilder 工程文件)

```

|
|---<source>(源文件目录)---
        com. seemap. app. SeemapApplet. java (Applet
                小应用程序)
|---<classes>( . class 目录)---
|           |--<netscape> JS 与 Applet 之间调用与需
要的类
|           |--<com> Seemap Free 软件包

```

示例目录：

<InterfaceClient>-----InterfaceClient.jpr (Jbuilder 工程文件)

```

|
|---<source>(源文件目录)---
        com. seemap. app. SeemapApplet. java (Applet

```

```

小应用程序)
|---<classes>( .class 目录)---
|
|---<netscape> JS 与 Applet 之间调用与需
要的类
|
|---<com> Seemap Free 软件包
<WebRoot>-----示例的网页目录

```

3.2 二次开发基础对象与接口的介绍

二次开发集中体现在对 SeemapApplet.java 修改或扩充，现介绍个文件如下（在使用 SeeMap 软件包进行二次开发前，要求用户充分掌握该章节介绍的内容）：

SeemapApplet.java

用户 Applet 程序，要求从 userAppletEx 派生并实现 SeeMap 接口 com.seemap.seemapInterface.InterfaceQueryDirect 的成员函数；

值得注意的是，由 userAppletEx 派生的类都会继承一些重要方法和函数，

```

//成员对象
public Panel userPanelMainMap;//主画布图形
public Panel userPanelFaceMap;//鹰眼图图形
public com.seemap.seemapInterface.InterfaceQueryDirect
userMainMapControl; //实现各种功能的接口
//方法
this.userLoadMap() //装载缺省地图数据

```

要加载并显示地图，必须先调用 userLoadMap() 方法，并在 applet 中添加 userPanelMainMap 面板。这些在开发包中已经作过处理（可见 SeemapApplet.java 中的 init() 与 userinit() 方法），用户可以根据情况自行更改。

要添加功能，应该根据 userMainMapControl 中的一些接口，在

SeemapApplet.java 中来添加函数，扩充功能，这样就可以在页面中通过 applet 对象来访问了。

3.3 接口说明

Seemap Free 二次开发的接口为

com.seemap.seemapInterface. InterfaceQueryDirect,

在 SeemapApplet.java 中通过 `userMainMapControl` 来实现。其中包含的接口函数如下：

设置地图操作状态

```
public void userSetMouseMode( int i);
```

功能描述：设置地图当前的操作状态

参数描述：i int 型

i	值	状态
	0	清除
	1	放大
	2	缩小
	3	全图
	4	移动
	5	直线测距
	6	折线测距
	7	折线面积
	127	点取坐标

说明：设置为点取坐标后，当用户在地图中点击，会返当前地图点取的坐标。

回调函数为：SeemapApplet.java 中的：

```
public void userSetCenterObject(int status, float f1, float f2,
```

```
String as[])
```

其中 status 为 127 时，对应为点取坐标，f1, f2, 分别为当前点取坐标的经纬度 x, y。

鼠标滚轮操作

```
public void userSetMouseMode(int i);
```

功能描述：鼠标滚轮滚动时对应的操作

参数描述：i 鼠标控制值

0 地图放大

其它值 地图缩小

设置当前地图显示区域

```
Public void userSetExtendMapArea(Rectangle rect);
```

功能描述：设置当前地图显示区域

参数描述：

当 rect.height== -99999999 时

rect.x, rect.y 为地图中心坐标

rect.width 为显示宽度

当 rect.height!= -99999999 时

rect.x, rect.y 为地图顶点 1 坐标,

rect.width, rect.height 为地图顶点 2 坐标

获得当前地图所有图层名

```
public String[]userGetLayerInfo();
```

功能描述：获得当前地图所有图层名

返回值：返回所有图层名的 String 数组

获得当前地图指定图层的字段名

```
public String[]userGetLayerFields(String strLayerName);
```

功能描述: 获得当前地图指定图层的字段名

参数描述: strLayerName xxxx 图层名称

说明: 当 strLayerName 对应的图层不存在时返回值为 null

返回值: 返回指定图层的所有字段的 String 数组

设置指定图层为可选状态

```
public void userSetLayerFocus (String strLayerName,  
                               boolean isClearOldFocusLayer);
```

功能描述: 设置指定图层的可选状态

参数描述:

strLayerName 单个或多个图层名称,或是图层别名

格式 1: xxxx 图层名称, @@AL INAMExxxx 图层别名

格式 2: |xxxx|xxxx|@@AL INAMExxxx|@@AL INAMExxxx

isClearOldFocusLayer 是否清除已经存在的选择图层

true 表示清除, false 不清除

设置指定图层的可见状态

```
public void userSetLayerVisible(String strLayerName);
```

功能描述: 设置指定图层的可见状态

参数描述:

strLayerName 格式 1: xxxx 图层名称 ^ON xxxx 图层名称 ^OFF

@@AL INAMExxxx 图层别名 ^ON @@AL INAMExxxx 图层别名 ^OFF

格式 2 : |xxxx^ON|xxxx^FF|xxxx^ON|
|@@AL INAMExxxx^OFF|@@AL INAMExxxx^ON|

获取选定对象的 ID 值

```
public int userGetSelectObjectKey ();
```

功能描述: 获取选定对象的 ID 值

返回值: 返回 int 类型的选定对象的 ID 值

获取选定对象所在的图层名

```
public String userGetSelectObjectLayer ();
```

功能描述: 获取选定对象所在的图层名

返回值: 返回选定对象所在的图层名

获取选定对象的面积

```
public double userGetSelectObjectArea ();
```

功能描述: 获取选定对象的面积

返回值: 返回 double 类型的选定对象的面积

获取选定对象的字段名

```
public String[] userGetSelectObjectField ();
```

功能描述: 获取选定对象的字段名

返回值: 返回选定对象的字段名的 String 数组

获取选定对象的字段属性值

```
public String[] userGetSelectObjectAttrib ();
```

功能描述: 获取选定对象的字段属性值

返回值: 返回选定对象的字段属性值的 String 数组

清除全部选定

```
public void userClearAll ();
```

功能描述: 清除全部选定

设置地图视野

```
public void userSetMapView(float x, float y, int z);
```

功能描述: 设置地图当前的显示视野

参数描述:

CenterX, CenterY 地图中心坐标

$z \leq 0$ 表示不改变当前地图视野, $z > 0$ 表示视野高度

对象选择

```
int    iQueryTypeP, int    iStartIndexP, int    iPageNumP, String
strQueryWordP, boolean    iNeedReturnP, boolean
isNeedCallbackP, boolean isNeedFocusP)
```

```
    public void userExtendMapElementsKeySelect(int iResultType,
                                                String strQueryWord,
                                                int iSelectGISKey,
                                                String
                                                strSelectGISName);
```

```
    public void userExtendMapElementsKeySelect(int iResultType,
                                                String strQueryWord,
                                                int iSelectGISKey,
                                                String
                                                strSelectGISName,
                                                boolean boolFocus,
                                                boolean boolAdjust);
```

```
    public void userExtendMapElementsKeySelect(int iQueryID,
                                                String
                                                strLayer
                                                Name,
                                                String strField,
                                                int iGisKey,
                                                String strKey,
                                                int iObjectType,
                                                int iQueryMode,
                                                boolean
```

```
isNeedReturn,  
  
boolean  
isNeedFocus);
```

功能描述：对象选择

对象查询

```
public java.util.Vector userExtendMapElementsKeyQuery (  
    int iQueryID,  
    int iStartIndexP,  
    int iPageNumP,  
    String strQueryWordP,  
    boolean isNeedReturnP,  
    boolean isNeedCallbackP,  
    boolean isNeedFocusP,  
    Point[]pFilter,  
    int iFilterType);  
public java.util.Vector userExtendMapElementsKeyQuery (  
    int iQueryID,  
    int iStartIndexP,  
    int iPageNumP,  
    String strQueryWordP,  
    boolean isNeedReturnP,  
    boolean isNeedCallbackP,  
    boolean isNeedFocusP);
```

功能描述：对象查询

参数描述:

iQueryID 查询识别号 iQueryType>=20000000&&iQueryType<30000000 图层信息查询

iStartIndexP 起始-起始编号(缺省为 0)

iPageNumP 数量-需要查询的当前页数量(缺省为最大结果数)

strQueryWord 查询语句包含项目

格式:

```
layer=xxx&field=xxx&key=xxx&object=xxxx&mode=xxx&start=xxx&page=xxx
```

layer 图层-查询图层名

field 字段-查询字段(缺省为查询 0 字段)

key 值-关键字(缺省为“”)

特殊格式 key=|>100|

key=|>100|<300|

object 对象类型-缺省为 0

0 全部点线面检索

1 全部点检索

2 全部线检索

3 全部面检索

mode 模式-缺省为 1

0 包含关键字 field.indexOf(key)>=0

1 完全匹配 field.compareTo(key)==0

2 包含字符

field.indexOf("k")>=0&&field.indexOf("e")>=

0&&field.indexOf("y")>=0

start 起始-起始编号(缺省为 0)

page 数量-需要查询的当前页数量(缺省为最大结果数)

isNeedReturnP 数据返回 Vector

```

Vector[0] ObjectAttrDes("layer : :
OID", attribute[])
Vector[1] ObjectAttrDes("layer : :
OID", attribute[])

```

isNeedFocusP 数据焦点显示

isNeedCallback 是否需要调用回调函数。数据查询回调格式：回调函数
见下行

```

public boolean userSetFocusResultSet(
    int iQueryID, //查询时的 iQueryID
    int iTOTAL, //查询到的总数
    String strQueryWord, //查询时的语句
    String strName, //查询的层层名
    String[] StringHead, //查询到的字段数组
    java.util.Vector vData);
//vData 对应的数据如下
Vector[0] ObjectAttrDes("layer : :
OID", attribute[])
Vector[1] ObjectAttrDes("layer : :
OID", attribute[])
具体可查看对象 com.seemap.seemapInterface.ObjectAttrDes

```

返回 值： 数据返回 Vector

```

Vector[0] ObjectAttrDes("layer : :
OID", attribute[])
Vector[1] ObjectAttrDes("layer : :
OID", attribute[])
.....

```

返回查询到的结果

获取对象 GISID

```
public String userGetGISID (String strLayer,  
                             String strField,  
                             String strKey);
```

说 明：重要函数，根据条件获得 GISID 后，再进行其它处理

功能描述：根据指定内容查询地图对象的 GISID 值

参数描述：

strLayer 图层名

strField 字段名

strKey 值

返 回 值：返回 String 类型的对象的 GISID

周边查询

```
public String[]userExtendMapElementsNearbyQuery (String strName,  
int x,  
int y);
```

功能描述：以某个中心点, 查询其周边的对象

参数描述：

strName 查询关键字

x, y 查询中心坐标

返 回 值：返回查询到的对象的 GISID 数组

获得地图宽度像素

```
public  
int userQueryMapWidth();
```

功能描述: 获得地图宽度像素

返回值: 返回 int 型的地图宽度像素

获得地图高度像素

```
public int userQueryMapHeight();
```

功能描述: 获得地图高度像素

返回值: 返回 int 型的地图高度像素

扩展外部数据集的创建

```
public void userExtendMapElements_Create(String strName,
                                         String strElementsName
                                         [],
                                         float xs[],
                                         float ys[],
                                         boolean flag);
```

功能描述: 用于地图对外部数据的扩展, 创建外部数据, 并在地图中显示
参数描述:

strName 为扩展数据集设定的名称-Query1 等一客户端, 本参数无效
strElementsName 扩展数据集-效果描述-

"|Ellipse30|RGB255|", (圆, |Ellipse 半径|颜色|)

"|Ellipse30|RGB2555500|",

"|Piechart300|RGB255^50;12345^30;2555500^20|",

(扇形, |Piechart 半径|颜色 1^角度;颜色 2^角度 2; 颜色 3^角度 3 |)

"|Histogram20|RGB255^50;12345^30;2555500^20|"

(矩形, |Piechart 宽度|颜色 1^高度;颜色 2^高度 2; 颜色 3^高度 3 |)

"|Text 要显示的文字 |OffsetX24|OffsetY0|Font 宋体 |Size16|RGB255|&|Rect1|RGB255|FILL-65281|' "

(文字, |Text 要显示的文字|OffsetX 与点的 X 距离|OffsetY 与点的 Y 距离|字体|字体大小|字体颜色|&|边框|边框颜色|边框填充色|)

x, y 坐标数组

needAdjust 自动调整显示方式, 以显示数据集

扩展外部数据集的添加

```
public void userExtendMapElements_Add(String strName,
                                       String strElementsName,
                                       float x, float y,
                                       boolean flag);
```

```
public void userExtendMapElements_Add(String strName,
                                       String
                                       strElementsName,
                                       float x[],
                                       float y[]);
```

功能描述: 用于地图对外部数据的扩展, 为已经存在的外部数据添加数据, 如果对应的外部数据不存在, 则重新创建, 并在地图中显示。

说明:

参数描述:

strName 为扩展数据集设定的名称-Query1 等-客户端, 本参数无效

strElementsName 扩展数据集-效果描述-

"|Ellipse30|RGB255|", (圆, |Ellipse 半径|颜色|)

"|Ellipse30|RGB2555500|",

"|Piechart300|RGB255^50;12345^30;2555500^20|",

(扇形, |Piechart 半径|颜色 1^角度;颜色 2^角度 2; 颜色 3^角度 3 |)

“|Histogram20|RGB255^50;12345^30;2555500^20|”

(矩形, |Piechart 宽度|颜色 1^高度;颜色 2^高度 2;颜色 3^高度 3 |)

“|Text 要显示的文字 |OffsetX24|OffsetY0|Font 宋体 |Size16|RGB255|&|Rect1|RGB255|FILL-65281|’ ”

(文字, |Text 要显示的文字 |OffsetX 与点的 X 距离 |OffsetY 与点的 Y 距离|字体|字体大小|字体颜色|&|边框|边框颜色|边框填充色|)

x, y 增加单个对象时的坐标

xs, ys 增加多个对象时的坐标数组

needAdjust 自动调整显示方式, 以显示数据集

扩展外部数据集的删除

```
public void userExtendMapElements_Delete (
    String strName);
```

功能描述: 用于地图对外部数据的扩展, 将已经存在的外部数据删除, 并在地图清除.

参数描述:

strName 扩展数据集的名称

获取指定对象的类型

```
public byte userGetGISType(String strGISID);
```

功能描述: 查询指定对象的类型

参数描述:

strGISID 对象的 GISID

返回值: 指定对象的类型

说明: 返回值 对应类型

- 1 点
- 2 线
- 3 面
- 4 文字

获取指定对象的节点数组

```
public float []userGetGISNODES (String strGISID);
```

功能描述：获取指定对象的节点数组

参数描述：

strGISID 对象的 GISID

地图数据的插入

```
public void userInsertMapElements (String strLayerName,  
int [] idArray,  
int [] xPoints, int [] yPoints,  
String [] [] strAttrib);
```

功能描述：向地图指定图层中添加数据

参数描述：

strLayerName 图层名称

idArray 对象 id 数组

xPoints, yPoints, 对象坐标数组

strAttrib 对象属性数组

地图数据的更新

```
public void userUpdateElementAttrib (String s, String s1,  
String s2);
```

```
public void userUpdateElementAttrib (String strLayerName,  
String strField1,  
String strKey,
```

```
String strField2,  
String strAttrib);
```

功能描述: 根据条件, 更新指定对象的指定属性值

参数描述:

strLayerName 对象图层名称
strField1 条件字段
strKey 条件值
strField2 对象需要更新属性的字段
strAttrib 新的属性值

说 明: 对 strLayerName 的对象, 凡是 strField1 满足 strKey 条件, 设置 strField2 为 strAttrib

设置单个对象特殊显示属性

```
public void userSetExtendElementStyle(  
String strGISID,  
String style);
```

功能描述: 设置单个对象特殊显示属性

参数描述:

strGISID 对象的 GetGISID
style 对象需要特殊显示的风格描述

清除对象特殊显示属性

```
public void userClearExtendElementStyle();
```

功能描述: 清除对象特殊显示属性

批量设置对象特殊显示属性

```
public void userSetExtendElementStyle(  
String strLayerName,  
String strField,
```

```
String strKey,
String style);
```

功能描述：批量设置对象特殊显示属性

参数描述

strLayerName 对象图层名称
 strField1 条件字段
 strKey 条件值
 style 对象需要特殊显示的风格描述

说明：对 strLayerName 的对象，凡是 strField1 满足 strKey 条件，设置 style 显示属性

聚焦对象

```
public void userFocusingElement(String strGISID);
```

功能描述：在地图中央聚焦指定的对象

参数描述：

strGISID 对象的 GISID

重绘地图

```
public void userRePaint();
```

功能描述：重新绘制地图

3.4 二次开发常见问题说明：

3.4.1 JavaScript (JS) 与 applet (SeemapApplet) 的调用问题

(a) 在页面中调用 applet 中的函数

通过 Seemap Free 创建好地图数据 [XXX. see](#) 或是 [XXX. see](#) 目录下的分级数据。及对 SeemapApplet.java 功能性开发，为页面提供功能性调用函数。为保证程序的兼容性，建设将这些函数都放在 SeemapApplet 中。比如提供一个函数，来让用户“改变当前地图视野”。

就可以在 SeemapApplet 中添加此函数，

```
public void javascriptSetMapView (float x, float y,
                                int z)
{
    userMainMapControl.userSetMapView( x, y, z)
}
```

让页面中通过调用此函数，来实现“改变当前地图视野”的功能。

在做好对 java 程序的开发后，将生成的所有.class 文件包，即开发工程中的 classes 目录下的文件。以 ZIP 方式打包，比如打包成 **seemap.class**。

在页面开发，让地图及其功能在网页中展现出来。将之前生成的 **seemap.class** 中的 Applet (即之前开发的 SeemapApplet.java)，及之前生成的地图文件 **XXX.see**，通过如下代码引入到页面中。

```
<applet code= “com.seemap.app.SeemapApplet.class” Archive=
“seemap.class” name= “Emap” width= “100%” height= “100%”
onmousewheel= “ return Emap.javascriptWheel(wheelFlag()) ”
MAYSCRIPT>
    <PARAM NAME= “APPLANGUAGE” VALUE= “cn” >
    <PARAM NAME= “APPINDEX” VALUE= “0” >
    <PARAM NAME= “APPSERVICEPORT” VALUE= “8080” >
    <PARAM NAME= “APPMAPNAME” VALUE= “XXX.see” >
    <PARAM NAME= “APPMAINMAPNAME” VALUE= “XXX.see” >
    <PARAM NAME= “APPMAPDESCRIPTION” VALUE= “SEEMAP 地图”
>
</applet>
```

请注意 **MAYSCRIPT** 必须添加进来，这样在页面中 Emap 就对应为 applet 对象了，在 JS 中就可以通过 Emap 来访问其函数。(以下 Emap 均指 applet 对象)

如果用户想要改变当前地图视野为，以经纬度坐标

```
win.eval("appletSetCurrentPoint('" + f + "', '" + f1+
"')");
```

3.4.2 二次开发中，页面接口的定义（用于供页面调用的函数）

在二次开发过程中，对于页面接口的定义请注意以下问题：

(a) 为保证兼容性，尽量将页面接口定义在 SeemapApplet.java 中，不要直接在页面中调用 `userMainMapControl` 的函数。

比如，应将 `userMainMapControl` 中的“设置地图操作状态” `public void userSetMouseMode(int i0)`；

在 SeemapApplet.java 中封装为：

```
public void javascriptSetMouseMode(int i){
    userMainMapControl.userSetMouseMode(i);
}
```

(b) 因为页面接口是给页面 JS 来调用的，因此其参数应以基础的 String, int, float 等来为主，最好是都以 String 来转换。

比如 `userMainMapControl` 中的“改变当前地图显示区域”的函数：

```
public void userSetExtendMapArea(Rectangle rect);
```

应在 SeemapApplet.java 中封装为：

```
public void javascriptSetExtendMapArea(int
                                x, int y, int
                                width, int
                                height)
{
    Rectangle rect = new
Rectangle(x, y, width, height);

userMainMapControl.userSetExtendMapArea(rect);
```

}

3.4.3 关于数组的页面接口中的处理

(a) 传递数组参数

经多年的开发经验发现，有极少数的浏览器在调用以数组为参数的 applet 函数中，会出现类型不匹配的异常情况。为保证您写的代码的兼容性，建议您不要在页面接口中使用数组来传参数。

比如当您想向 applet 传递一个字符数组时，请不要写成如下函数

在 SeemapApplet.java 中

```
public void javascriptSendStrs( String str[]){
.....
}
```

然后在页面中，传递数组

```
var strs=new Array(2);
strs[0] = 'xx' ;
strs[1]=' yy' ;
```

Emap. javascriptSendStrs(strs);

尽管这在大多数情况下是可以的，但在有些浏览器中的 JS 数组与 applet 中的数组并不能兼容，会出现错误。因此建议采用如下方式，将要传递的多个字符串连接成一上字符串来传递，在 applet 中再把他们分开。

在 SeemapApplet.java 中

```
public void javascriptSendStrs( String strs){
    String s[ ] = strs.split(“&&”);
}
```

在页面中调用时

```
var strs=” xx” +” &&” +” yy” ;
```

Emap. javascriptSendStrs(strs);

(b) 返回数组

同样，如上所说，建设 applet 不要直接返回数组，而可以将数组转换为 Vector，这样就不会出现数组类型不兼容的问题。

比如想要获取地图全部图层名时

userMainMapControl 中对应的函数为

```
public String[] userGetLayerInfo();
```

在 applet 中应将其封装为

```
public Vector javaScriptGetLayerInfo() {
    String layers[] =
userMainMapControl.userGetLayerInfo();
    Vector res =new Vector();
    for(int i=0;i<layers.length;i++){
        res.add(layers[i]);
    }
    return res;
}
```

在页面中调用时

可以通过如下代码获取

```
<script language="JavaScript" type="text/JavaScript">
```

```
.....
```

```
var layers = Emap. javaScriptGetLayerInfo();
```

```
var length = layers.size();
```

```
for(var i = 0; i < length; i++)
```

```
{
```

```
layers.elementAt(i);
```

```
//对其进行处理
```

```
}
```

```
</script>
```

3.4.4 关于回调函数

在 applet 中有许多情况下会使用到回调函数，在此说明一下。比如用户想要通过点击从地图中获取到坐标值。

首先，因先设置地图的操作状态为点取坐标状态。

```
即调 Emap. Javascriptsetmousestatus( 127);
```

```
// public void userSetMouseMode( int i);
```

来设置地图为点取坐标状态（见接口说明中“设置地图操作状态”）；

其次，当用户用鼠标在地图中点击后，程序内部为自动调用回调函数

```
public void userSetCenterObject(int status, float f1, float f2,
String as[]
```

```
{ if(status == 127) {
```

```
    try{
```

```
        win.eval("appletSetCurrentPoint('"+ f 1 + "', '"+
```

```
f
```

```
2
```

```
;
```

```
        ")");
```

```
    }catch(Exception e) {
```

```
    }
```

```
    }
```

```
}
```

其中 status 为 127, f1, f2, 分别为当前点取坐标的经纬度 x, y。

这样在引入了地图的页面中的 appletSetCurrentPoint(x, y) 函数中就可以反返回的坐标值 x, y 进行了。

3.4.5 关于查询

在地图接口中，比较重要与复杂一点的就是查询函数了，在此以一个最为简单的查询为例，对其进行一下简单说明。

地图查询接口为：

```
public java.util.Vector userExtendMapElementsKeyQuery(  
    int iQueryID,  
    int iStartIndexP,  
    int iPageNumP,  
    String strQueryWordP,  
    boolean isNeedReturnP,  
    boolean isNeedCallbackP,  
    boolean isNeedFocusP,  
    Point[] pFilter,  
    int iFilterType);  
  
public java.util.Vector userExtendMapElementsKeyQuery (  
    int iQueryID,  
    int iStartIndexP,  
    int iPageNumP,  
    String strQueryWordP,  
    boolean isNeedReturnP,  
    boolean isNeedCallbackP,  
    boolean isNeedFocusP);
```

首先，在 applet 中定义如下函数，用户只需传入要查询的返回数据的起始值，返回的条数，和查询查询语句即可。（这里是为了简化参数，用户可以根据自己的情况来定义）

```
public void javascriptQueryMapObject(int iStartIndexP, int  
iPageNumP, String s)
```

```
{ //查询，具体参数见接口说明“地图查询”
userMainMapControl.userExtendMapElementsKeyQuery(
    0x13130ed, iStartIndexP, iPageNumP, s, false, true,
true);
}
```

然后在查询的回调函数里对其进行处理：

对于返回的数据，以如下对象存放在 applet 中

SeemapResultSet. iResultType 查询类型，

SeemapResultSet. iResultTotal 查询到的数据记录的总数

SeemapResultSet. iResultStartIndex 返回数据的起始索引值

SeemapResultSet. iResultPageCount 当前返回的数据记录个

数

SeemapResultSet. strQueryWord 查询语句；

SeemapResultSet. vRecorder SeemapRecorder 的 Vector 数

组

SeemapRecorder 对应的结构如下

SeemapRecorder

String strGISID; 对象的 GISID

String strGISName; 对象的 GIS 名称

Vector atts; 对象的属性值

Vector 数组

```
public boolean userSetFocusResultSet(int iResultType, int
    iTotal, int iStartIndex, int i, String s,
    String s1, String as[],
    Vector vector)
{
```

```
if(iResultType == 0x13130ed) { //对象查询

    seemapResultSet.iResultType = iResultType;
    seemapResultSet.iResultTotal = iTotal;
    seemapResultSet.iResultStartIndex = iStartIndex;
    seemapResultSet.iResultPageCount = i;
    seemapResultSet.strQueryWord = s;

    seemapResultSet.vRecorder.removeAllElements();

    for(int j1 = 0; j1 < vector.size(); j1++)
    {
        ObjectAttrDes objectattrdes1 =
(ObjectAttrDes)vector.elementAt(j1);

        SeemapRecorder recoder = new
SeemapRecorder(objectattrdes1.strObjectID, objectattrdes1.strOb
jectAttrib[0], objectattrdes1.strObjectAttrib);

seemapResultSet.vRecorder.addElement(recoder);

    }

    try
    {
```

```

        win.eval("appletSetResult()");//通知页面
    }
    catch(Exception exception1)
    {
        exception1.printStackTrace();
    }
}
else if(iResultType == 0x13130ee) //公交查询 {
    //.....
}
return false;
}

```

最后，在页面的 `appletSetResult()` 函数中，对回调函数中生成的数据进行处理。

<code>Emap.seemapResultSet.iResultType</code>	查询类型，此为 0x13130ed;
<code>Emap.seemapResultSet.iResultTotal</code>	查询到的数据记录的 总数
<code>Emap.seemapResultSet.iResultStartIndex</code>	返回数据的起 始索引值
<code>Emap.seemapResultSet.iResultPageCount</code>	当前返回的数据 记录个数
<code>Emap.seemapResultSet.strQueryWord</code>	查询语句；
<code>Emap.seemapResultSet.vRecorder</code>	<code>SeemapRecorder</code> 的 Vector 数组
<code>SeemapRecorder</code> 对应的结构如下	
<code>SeemapRecorder</code>	

String	strGISID;	对象的 GISID
String	strGISName;	对象的 GIS 名称
Vector	atts;	对象的属性值

Vector 数组

查询过程大致如下:

首先用户在页面提交查询:

```

Emap. javascriptQueryMapObject (0, 20,
    "layer=xxx&field=xxx&key=xxx&object=xxxx&mode=xxx&start=xxx&page=xx"
);

```

其次回调函数返回查询到的数据, 并处理, 调用页面的 `appletSetResult ()`;

最后, 用户在页面中的 `appletSetResult ()`; 中对查询到的数据

```

Emap. seemapResultSet. iResultType      查询类型, 此为
0x13130ed;

```

```

Emap. seemapResultSet. iResultTotal     查询到的数据记录的
总数

```

```

Emap. seemapResultSet. iResultStartIndex 返回数据的起
始索引值

```

```

Emap. seemapResultSet. iResultPageCount 当前返回的数据
记录个数

```

```

Emap. seemapResultSet. strQueryWord     查询语句 ;

```

```

Emap. seemapResultSet. vRecorder       SeemapRecorder 的 Vector
数组

```

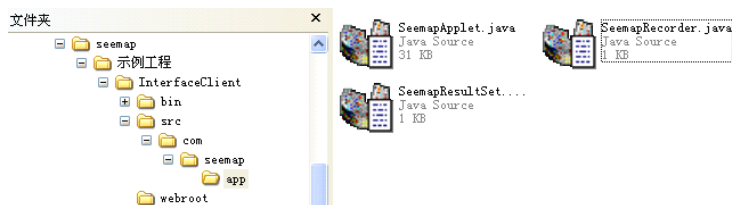
进行处理, 在页面中显示出来。

4 示例讲解

在提供的开发包中，有一个完整的示例，在此对其进行一些说明。
(对 .java 源文件的说明，以实际为准)

4.1 二次开发中 java 程序的开发及对应的页面功能说明

二次开发示例中的 java 源文件目录如下



其中 SeemapApplet.java 为地图 applet 程序，SeemapResultSet.java 与 SeemapRecorder.java 为用于处理地图查询与点选时返回的数据的对象。他们的内容如下：

(1) SeemapRecorder.java 内容如下：

```
package com.seemap.app;
import java.util.Vector;

public class SeemapRecorder {
    //查询结果的 GISID
    public String strGISID;
    //查询结果的 GIS 名称
    public String strGISName;
    //查询结果的属性值数组集
    public Vector atts;
    public SeemapRecorder(String s, String s1)
```

```
{
    strGISID = s;
    strGISName = s1;
}
public SeemapRecorder(String s, String s1,String tatts[])
{

    strGISID = s;
    strGISName = s1;
    atts = new Vector();
    for(int i = 0 ;i < tatts.length; i++)
        atts.add(tatts[i]);

}
}
```

(2) SeemapResultSet.java 内容如下:

```
package com.seemap.app;

import java.util.Vector;

public class SeemapResultSet
{
    //查询识别号 iQueryType>=20000000&&iQueryType<30000000 图
    层信息查询
    public int iResultType;
    //符合查询条件的记录总数
    public int iResultTotal;
    //当前查询页面记录数
```

```
public int iResultPageCount;
//查询到的结果在所有记录中的起始索引值
public int iResultStartIndex;
//查询语句
public String strQueryWord;
//查询结果对应的图层字段集
public Vector heads;
//查询到的结果集，在普通查询时为 SeemapRecorder 对象集，
// 图中选择对象时为点先对象的属性集
public Vector vRecorder;

public SeemapResultSet()
{
    iResultType = 1;
    iResultTotal = 0;
    iResultPageCount = 0;
    iResultStartIndex = 0;
    strQueryWord = "";
    vRecorder = new Vector();
}
}
```

(3) SeemapApplet.java 的主要内容如下:

```
package com.seemap.app;

import com.seemap.seemapGUI.UserAppletEx;
import com.seemap.seemapInterface.InterfaceQueryDirect;
```

```
import com.seemap.seemapInterface.ObjectAttrDes;

import java.applet.Applet;
import java.awt.*;
import java.io.PrintStream;
import java.net.URL;
import java.util.Vector;
import netscape.javascript.JSObject;

// Referenced classes of package com.seemap.app:
//          SeemapJarResources

public class SeemapApplet extends UserAppletEx
{

    JSObject win;
    Panel panelCenter;
    Panel p;
    boolean isLoadMode;
    public int iMessageType;
    public Vector vMessageBody;
    public SeemapResultSet seemapResultSet;
    public SeemapApplet()
    {
        seemapResultSet = new SeemapResultSet();
        isLoadMode = false;
        vMessageBody = new Vector();
    }
}
```

(a) 初始化**applet 初使化**

```
public void init()
{
    String strFunPort = getParameter("APPSERVICEPORT");
    if(strFunPort == null || strFunPort.length() <= 0)
        strFunPort = "";
    else
        strFunPort = ":" + strFunPort;
    try
    {
        userLoadMap(); //加载地图
        p = userPanelMainMap;
    }
    catch(Exception exception)
    {
        exception.printStackTrace();
    }
    try
    {
        //初使化页面对象
        win = JSObject.getWindow(this);
    }
    catch(Exception exception1) { }
    userinit();
}
```

用户初使化

```

public void userinit()
{
    try
    {
        setLayout(null);
        setBackground(new Color(221, 224, 191));
        panelCenter = new Panel(new BorderLayout());
        //加载地图面板 userPanelMainMap
        panelCenter.add(p, "Center");
        add(panelCenter);
        Rectangle rectangle = getBounds();
        reshape(rectangle.x, rectangle.y, rectangle.width,
rectangle.height);
    }
    catch(Exception exception)
    {
        exception.printStackTrace();
    }
    .....

        //调用页面初使化函数
        win.eval("appletInitMap()");
    .....
}

```

(b) 设置地图状态

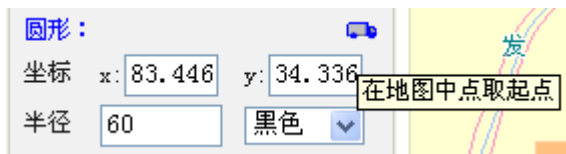


java 中对应的函数

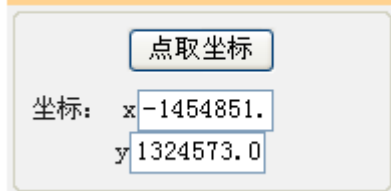
```
public void javascriptsetmousestatus(int iStatus)
{
    userMainMapControl.userSetMouseMode(iStatus);
}
```

参数描述: i int 型

i 值	状态
0	清除
1	放大
2	缩小
3	全图
4	移动
5	直线测距
6	折线测距
7	折线面积
127	点取坐标



获取坐标



说明: 设置为点取坐标后, 当用户在地图中点击, 会返当前地图点取的坐标。

回调函数为：SeemapApplet.java 中的：

```
public void userSetCenterObject(int status, float f1, float f2,
String as[])
```

其中 status 为 127 时，对应为点取坐标，f1, f2, 分别为当前点取坐标的经纬度 x, y。

//设置为点取坐标后的

```
public void userSetCenterObject(int l, float f, float f1, String as[])
{
    //获取点坐标
    if(l == 127) {
        try{
            //通过以下函数，将返回的坐标信息传递给页面
            win.eval("appletSetCurrentPoint('" + f + "', '"
+ f1+ "')");
        }catch(Exception e) {
        }
    }
}
```

(c) 地图查询

地图查询

选择图层:

匹配字段:

匹配方式:

关键字:

在页面中通过点击地图查询，会调用

`Emap.javascriptQueryMapObject(0, 15, ' layer=桥梁&field=Name&key=&object=0&mode=0'); javascriptQueryMapObject` 的定义，在 java 中如下：

//为页面查询定义的函数

```
public void javascriptQueryMapObject(int i, int j, String s)
{
    userMainMapControl.userExtendMapElementsKeyQuery(0x13130
        ed, i, j, s, false, true, true);
}
```

当执行查询 `javascriptQueryMapObject ()` 后，程序会回调以下函数：

//查询对象的回调函数

```

public boolean userSetFocusResultSet(int iResultType, int
iTotal, int iStartIndex, int i, String s, String s1, String
as[],
        Vector vector)
    {
//0x13130ed 对应为查询时所定义的值
        if(iResultType == 0x13130ed) {
//对象查询 javascriptQueryMapObject(int i, int j, String s)

                seemapResultSet.iResultType = iResultType;
                seemapResultSet.iResultTotal = iTotal;
                seemapResultSet.iResultStartIndex = iStartIndex;
                seemapResultSet.iResultPageCount = i;
                seemapResultSet.strQueryWord = s;
                seemapResultSet.vRecorder.removeAllElements();

                for(int j1 = 0; j1 < vector.size(); j1++)
                {
                        ObjectAttrDes      objectattrdes1      =
(ObjectAttrDes)vector.elementAt(j1);

                                SeemapRecorder      recorder      =      new
SeemapRecorder(objectattrdes1.strObjectID,objectattrdes1.strOb
jectAttrib[0],objectattrdes1.strObjectAttrib);

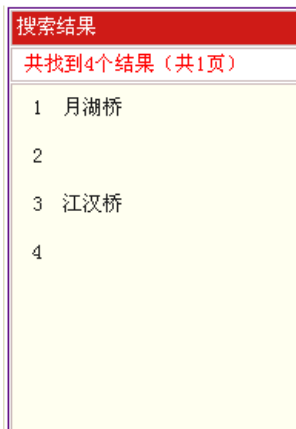
                                seemapResultSet.vRecorder.addElement(recorder);

                }

```

```
try
{
    // 通知页面, 执行查询后的处理
    win.eval("appletSetResult()");
}
catch(Exception exception1)
{
    exception1.printStackTrace();
}
}
```

在页面中对查询到的结果进行显示, 结果如下:



(D) 图层控制



获取图层名

//功能：获得当前地图全部图层名

```
public Vector javaScriptGetLayerInfo() {
    String layers[] = userMainMapControl.userGetLayerInfo();
    Vector res = new Vector();
    for (int i = 0; i < layers.length; i++) {
        res.add(layers[i]);
    }
    return res;
}
```

设置地图可见状态

```
//功能：设置指定图层为见/不可见状态
// 参数描述
//  strLayerName 格式 1： xxxx 图层名称 ^ON xxxx 图层名称 ^OFF
@@ALINAMExxxx 图层别名 ^ON @@ALINAMExxxx 图层别名 ^OFF
//
//          格式 2： |xxxx^ON|xxxx^FF|xxxx^ON|
|@@ALINAMExxxx^OFF|@@ALINAMExxxx^ON|
public void javascriptSetLayerVisible(String strLayerName) {
    userMainMapControl.userSetLayerVisible(strLayerName);
}
```

设置指定图层为可选状态

```
//功能：设置指定图层为可选/不可选状态
public void javascriptSetLayerFocus(
    String strLayerName,
    boolean isClearOldFocusLayer)
{
    userMainMapControl.userSetLayerFocus(strLayerName, isClearOldFocusLayer);
}
```


(e) 显示图形

显示图形

圆形： 

坐标 x: y:

半径 颜色

扇形： 

坐标 x: y:

角度1 颜色

角度2 颜色

矩形： 

坐标 x: y:

高度1 颜色

高度2 颜色

当点击显示图形时，调用以下函数

//功能描述：创建新的外部的数据集

```
public void javascriptExtendMapElements_Create(String
strName, String strElementsName, String xs, String ys, String
needAdjust) {
    boolean b = false;
    if (needAdjust.equals("true"))
        b = true;
    String names[] = mySplit(strElementsName, "&&");
    String xxs[] = mySplit(xs, "&&");
```

```

String yys[] = mySplit(ys, "&&");
float fx[] = new float [xxs.length];
float fy[] = new float [yys.length];
for(int i=0;i<fx.length;i++) {
    fx[i] = Float.parseFloat(xxs[i]);
}
for(int i=0;i<fy.length;i++) {
    fy[i] = Float.parseFloat(yys[i]);
}

```

```

userMainMapControl.userExtendMapElements_Create(strName, names,
fx, fy, b);
}

```

其中 userExtendMapElements_Create 的描述如下:

```

public void userExtendMapElements_Create(String strName,
String strElementsName [],
float xs[],
float ys[],
boolean flag);

```

功能描述: 用于地图对外部数据的扩展, 创建外部数据, 并在地图中显示
参数描述:

strName 为扩展数据集设定的名称-Query1 等-客户端, 本参数无效
strElementsName 扩展数据集-效果描述-

"|Ellipse30|RGB255|", (圆, |Ellipse 半径|颜色|)

"|Ellipse30|RGB2555500|",

"|Piechart300|RGB255^50;12345^30;2555500^20|",

(扇形, |Piechart 半径|颜色 1^角度;颜色 2^角度 2; 颜色

3 ^ 角度 3 |)

“|Histogram20|RGB255^50;12345^30;2555500^20|”

(矩形, |Piechart 宽度|颜色 1 ^ 高度;颜色 2 ^ 高度 2 ;颜色

3 ^ 高度 3 |)

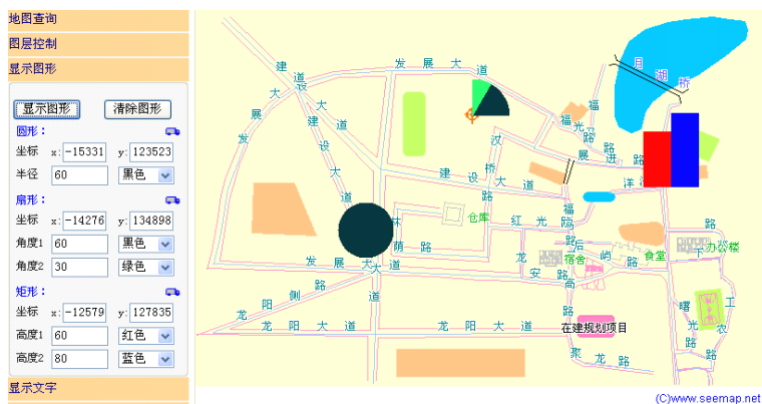
“|Text 要显示的文字 |OffsetX24|OffsetY0|Font 宋体
|Size16|RGB255|&|Rect1|RGB255|FILL-65281|’ ”

(文字, |Text 要显示的文字 |OffsetX 与点的 X 距离
|OffsetY 与点的 Y 距离|字体|字体大小|字体颜色|&|边框|边框颜色|边
框填充色|)

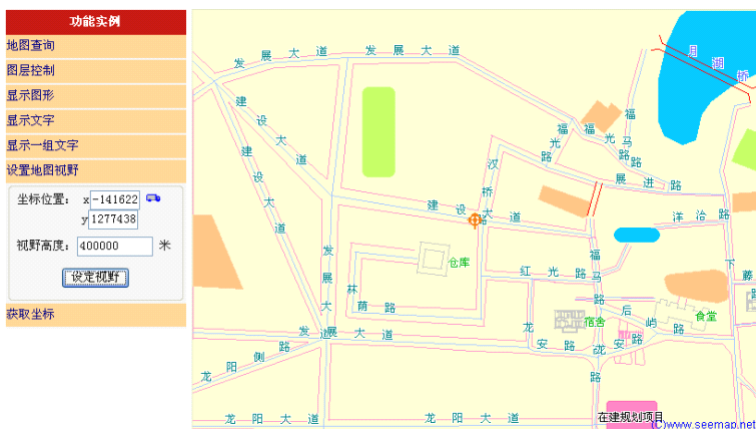
x , y 坐标数组

needAdjust 自动调整显示方式, 以显示数据集

显示效果:



(f) 设定地图视野

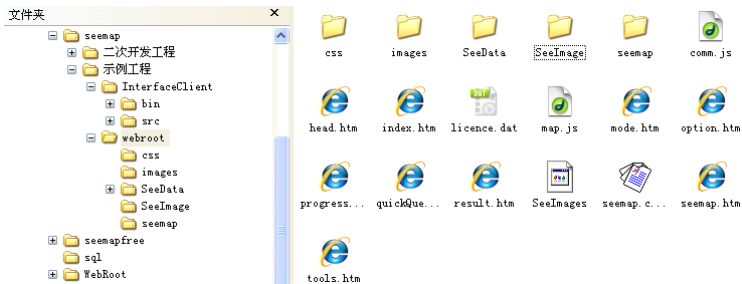


//功能：设置地图视野

```
public void javascriptSetMapView(float x, float y, int z) {  
  
    userMainMapControl.userSetMapView(x, y, z);  
  
}
```

4.2 页面文件说明

webroot 客户端页面开发示例目录如下:



重要文件说明:

地图文件: SeeData 目录中为 SeemapFree 地图文件, 其目录下的 sdf. see 目录下的 sdf. see 文件即为地图文件。

地图图标文件: SeeImages 为地图的图标文件

SeemapFree 客户端程序包: seemap.class

SeemapFree 的客户端授权文件: licence.dat

引入地图及程序包的文件: map.js

其中 map.js 的内容如下:

```
document.write("<applet
code=\"com.seemap.app.SeemapApplet.class\"
Archive=\"seemap.class\"      name=\"Emap\"      width=\"100%\"
height=\"100%\"                onmousewheel=\"return
Emap.javascriptWheel(wheelFlag())\" MAYSCRIPT>\"
+\"<PARAM NAME=\"APPLANGUAGE\" VALUE=\"cn\">\"
+\"<PARAM NAME=\"APPINDEX\" VALUE=\"0\">\"
+\"<!--PARAM                      NAME=\"APPZOOM\"
VALUE=\"|300000|160000|80000|30000|181000|81000|31000|1810
0|11000|5100|2100|1100|510|250|120|60|\"-->\"")
```

```

+“<!--PARAM                                NAME=“APPOPENID\”
VALUE=“MAPPOS=35162|623295|19800\”-->”
+“<PARAM NAME=“APPSERVICEPORT\” VALUE=“8080\”>”
+“<PARAM NAME=“APPMAPNAME\” VALUE=“sdf.see\”>”
+“<PARAM NAME=“APPMAINMAPNAME\” VALUE=“sdf.see\”>”
+“<PARAM NAME=“APPMAPDESCRIPTION\” VALUE=“SEEMAP 地
图\”>”
+“</applet>”
)
主页: index.htm
<html>
<head>
<meta http-equiv=“Content-Type” content=“text/html;
charset=gb2312”>
<title>数字地图</title>
</head>
<frameset rows=“80,28,*” cols=“*” frameborder=“NO”
border=“0” framespacing=“0” id=“mainFrameset”>
<frame src=“head.htm” name=“topFrame” scrolling=“NO”
noresize>
<frame src=“tools.htm” name=“topFrame” scrolling=“NO”
noresize >
<frameset cols=“200,*,200” frameborder=“NO” border=“0”
framespacing=“0” id=“subFrameset”>
<frame src=“option.htm” name=“queryFrame” scrolling=“NO” >
<frame src=“seemap.htm” name=“mainFrame” scrolling=“NO” >
<frame src=“result.htm” name=“leftFrame” noresize
scrolling=“auto”>

```

```
</frameset>
</frameset>
<noframes><body>
</body></noframes>
</html>
```

其他主要的页面有如下几个：

option.htm，主要展现地图的功能，对应的页面为：

功能实例

地图查询

选择图层: 桥梁

匹配字段: Name

匹配方式: 模糊匹配

关键字:

地图查询

图层控制

显示图形

显示文字

显示一组文字

设置地图视野

获取坐标

result.htm, 主要显示查询的结果等, 对应的页面为:

地图助手	搜索结果	对象属性
<ul style="list-style-type: none"> • 欢迎使用seemap平台 • 支持IE5.0以上浏览器。 • 建议使用1024*768分辨率。 • 建议使用P4以上处理器。 <p style="color: red;">这是一个简单发布向导, 可以点击以下内容获取帮助</p> <p>基本接口</p> <ul style="list-style-type: none"> ▪ 获取地图对象 ▪ 地图状态设置 ▪ 地图初始化设置 <p>图层控制</p> <ul style="list-style-type: none"> ▪ 获取所有图层信息 ▪ 获取图层字段信息 ▪ 设置地图图层显示 <p>图层查询</p> <ul style="list-style-type: none"> ▪ 根据图层字段查询 ▪ 地图中直接查询 	<p style="color: red;">共找到4个结果 (共1页)</p> <ol style="list-style-type: none"> 1 月湖桥 2 3 江汉桥 4 	<p>Name: 在建规划项目</p> <p>PopName:</p> <p>TelePhone:</p> <p>Address: 湖北省武汉市武昌区亚贸广场B座</p> <p>Update: 20090909</p> <p style="text-align: right;">返回</p>