MAPGIS VirtualEarth

二次开发接口篇

目 录

数字地球最基本的功能就是让用户能够在任何方向观察地球表面，获得感兴趣的信息。用户可以根据自己的需要定位到地球的各个角落，我们的数字地球为用户提供了这方面的功能接口，用户可以根据自己的需要编写自己的数字地球站点。

1 数字地球信息获取接口介绍

1.1 数字地球基本信息获取

接口函数：`String getInfo()`
接口说明：获取数字地球基本信息
参数说明：无
返回值： 基本信息字符串，格式为："中文名称;英文名称;版本号"
调用示例：如果 Applet 标签的 ID 为 vejapplet，使用
```
"document.getElementById('vejapplet').getSubApplet()
.getInfo()";
```
备注： 注意返回值中的标点为英文半角符号

1.2 视点获取

接口函数：`String getViewPoint()`
接口说明：获取当前视点信息，包括经度、纬度、海拔高度
参数说明：无
返回值： 视点信息字符串，格式为："纬度,经度,高度"
调用示例：如果 Applet 标签的 ID 为 vejapplet，使用
```
"document.getElementById('vejapplet').getSubApplet()
.getViewPoint()";
```
备注： 注意返回值中的标点为英文半角符号

1.3 获取 Applet 窗口宽度和高度

接口函数：`String getRectangle()`
接口说明：获取 Applet 窗口宽度和高度，以像素为单位
参数说明：无
返回值： Applet 窗口宽度和高度字符串，格式为："宽度,高度"
调用示例：如果 Applet 标签的 ID 为 vejapplet，使用
```
"document.getElementById('vejapplet').getSubApplet()
.getRectangle()";
```
备注： 注意返回值中的标点为英文半角符号

1.4 鼠标点击点的信息

1.4.1 获取鼠标点击点经纬度

接口函数： `String getLatLon()`
接口说明： 获取鼠标在数字地球上点击点的经纬度值
参数说明： 无
返回值： 鼠标在数字地球上点击点的经纬度值字符串，格式为："纬度,经度"
调用示例： 如果 Applet 标签的 ID 为 vejapplet,使用
```
"document.getElementById('vejapplet').getSubApplet()
.getLatLon()";
```
备注： 如果点击点在数字地球外，则返回null

1.4.2 获取能否得到点击点信息

接口函数： `boolean isAttainLatLonByClick()`
接口说明： 获取是否可以获取鼠标点击点信息
参数说明： 无
返回值： 可以获得则返回 true，否则返回 false
调用示例： 如果 Applet 标签的 ID 为 vejapplet,使用
```
"document.getElementById('vejapplet').getSubApplet()
.isAttainLatLonByClick()";
```
备注： 无

1.4.3 设置能否得到点击点信息

接口函数： `void setAttainLatLonByClick(boolean flag)`
接口说明： 设置是否可以获取鼠标点击点信息
参数说明： **boolean** flag：true 表示设置为可获取，false 表示设置为不可获取
返回值： 无
调用示例： 如果 Applet 标签的 ID 为 vejapplet,若要设置可获取点击点信息，使用
```
"document.getElementById('vejapplet').getSubApplet()
.setAttainLatLonByClick(true)";
```
备注： 无

1.5 计算经纬度

接口函数： `String computePositionFromScreenPoint(double x, double y)`
接口说明： 根据屏幕坐标计算经纬度坐标
参数说明： double x：Applet 窗口内点的 x 坐标

double y：Applet 窗口内点的 y 坐标

返回值： 对应所给屏幕坐标的经纬度坐标，格式为："纬度,经度"

调用示例： 如果Applet标签的ID为vejapplet,要得到屏幕坐标为[120,360]点的经纬度值，则使用"document.getElementById('vejapplet')
.getSubApplet()
.computePositionFromScreenPoint(120, 360)";

备注： x,y对应Applet窗口内屏幕坐标，单位为像素
Applet窗口左上角坐标为[0,0]，右下角坐标为[max,max]
x,y输入错误或点不在地球上则返回null
返回值中的标点为英文半角符号

2 三维地球漫游、定位控制接口介绍

2.1 定位控制

2.1.1 以默认朝向、倾斜角进行移动

接口函数：
```
boolean gotoLatLon(double lat, double lon)
```
接口说明： 当前视域以当前高度、指北方向和零倾斜角移动到指定经纬度

参数说明： double lat：纬度，范围区间[-90,90]
double Lon：经度，范围区间[-180,180]

返回值： 正常跳转返回 true，否则返回 flase

调用示例： 如果Applet标签的ID为vejapplet,如果lat=45,lon=-120，则使用"document.getElementById('vejapplet').getSubApplet()
.gotoLatLon (45, -120)";

备注： 无

2.1.2 指定朝向、倾斜角进行移动

接口函数：
```
boolean gotoLatLon(double lat, double lon, double zoom,
                   double heading, double pitch)
```
接口说明： 当前视域以指定高度、朝向、倾斜角移动到指定经纬度

参数说明： double lat：纬度，范围区间[-90,90]
double Lon：经度，范围区间[-180,180]
double zoom：视点高度，范围区间[0,20000000]
double heading：视点朝向，范围区间[-180,180]
double pitch：视点倾斜角，范围区间[0,90]

返回值： 正常跳转返回 true，否则返回 flase

备注： 无

2.2 方向控制

接口函数：`void moveForward(String direction)`
接口说明：指定地球向相对于视窗的东、南、西、北、东北、西北、东南、西南方向移动
参数说明：`String direction`：八个方向："east", "west", "south", "north", "northeast", "northwest", "southeast", "southwest"
返回值： 无
调用示例：如果Applet标签的ID为vejapplet,如果向朝东北方向移动，则使用
```
"document.getElementById('vejapplet').getSubApplet()
.moveForward('northeast')";
```
备注： 无

2.3 倾斜角和朝向控制

接口函数：`void setHeadingAndPitch(double heading, double pitch)`
接口说明：设定当前视点的倾斜角与朝向
参数说明：`double heading`：要设定的朝向值，范围区间[-180,180]
`double pitch`：要设定的倾斜角值，范围区间[0,90]
返回值： 无
调用示例：如果Applet标签的ID为vejapplet,如果heading=40.5,pitch=31.2，则使用
```
"document.getElementById('vejapplet').getSubApplet()
.setHeadingAndPitch (40.5, 31.2)";
```
备注： 无

2.4 视点高度控制

接口函数：`void setZoom(double zoom)`
接口说明：设定当前视点的海拔高度，范围区间[0,20000000]
参数说明：`double zoom`：要设定的视点的海拔高度，范围区间[0,20000000]
返回值： 无
调用示例：如果Applet标签的ID为vejapplet,如果要设置视点高度为405，则使用
```
"document.getElementById('vejapplet').getSubApplet()
.setZoom (405)";
```
备注： 无

2.5 缩放控制

接口函数：`void ctrlZoom( boolean zoom,int amount)`
接口说明：设定当前视点的海拔高度
参数说明：`boolean zoom`：缩放标识，`true` 为放大，`false` 为缩小
`int amount`：指放大缩小的速度快慢程度，推荐设定区间[-3,3]

返回值： 无
调用示例： 如果Applet标签的ID为vejapplet,如果要以amount=2放大，则使用
```
"document.getElementById('vejapplet').getSubApplet()
.ctrlZoom (true, 2)";
```
备注： 无

2.6 视点倾斜角增大与减小控制

接口函数： `void ctrlTilt(boolean tilt, int amount)`
接口说明： 控制当前视点的倾斜角的增加与减少
参数说明： **`boolean`** `tilt`：倾斜角增减标识，`true` 增大倾斜角，`false` 减小倾斜角
`int` `amount`：倾斜的速度快慢程度
返回值： 无
调用示例： 如果Applet标签的ID为vejapplet,如果要以amount=2的速度增加倾角，则使用
```
"document.getElementById('vejapplet').getSubApplet()
.ctrlTilt (true, 2)";
```
备注： 无

2.7 地球旋转控制

接口函数： `void ctrlRotate(boolean rotate, int amount)`
接口说明： 设定地球按指定方向与速度旋转
参数说明： **`boolean`** `rotate`：旋转方向，`true` 顺时针旋转，`false` 逆时针旋转
`int` `amount`：旋转速度快慢程度
返回值： 无
调用示例： 如果Applet标签的ID为vejapplet,如果要以amount=2的速度进行顺时针旋转，则使用
```
"document.getElementById('vejapplet')
.getSubApplet().ctrlRotate(true, 2)";
```
备注： 无

2.8 重置视点朝向

接口函数： `void resetHeading()`
接口说明： 重设视点朝向，使看到的地球为指北正向
参数说明： 无参数
返回值： 无
备注： 无

2.9 重置视点倾斜角

接口函数： `void resetTilt()`

接口说明：重设视点倾斜角，置倾斜角为 0
参数说明：无参数
返回值： 无
备注： 无

2.10 重置视点朝向与倾斜角

接口函数：`void resetEarth()`
接口说明：重设视点，使看到的地球为指北正向且倾斜角为 0
参数说明：无参数
返回值： 无
备注： 无

3 图层控制接口介绍

3.1 设置基本影像图层是否可见

接口函数：`void setBasicSurfaceLayerAction(boolean isActive)`
接口说明：设置基本影像图层是否可见
参数说明：`boolean isActive`：可见标识，`true` 表示可见，`false` 表示不可见
返回值： 无
调用示例：如果Applet标签的ID为vejapplet，且设置基本影像图层不可见，则使用

```
"document.getElementById('vejapplet').getSubApplet()
.setBasicSurfaceLayerAction(false)";
```

备注： 基本影像图层包括了最初所能看到地表的基本地貌影像

3.2 设置详细影像图层是否可见

接口函数：`void setParticularSurfaceLayerAction(boolean isActive)`
接口说明：设置详细影像图层是否可见
参数说明：`boolean isActive`：可见标识，`true` 表示可见，`false` 表示不可见
返回值： 无
调用示例：如果Applet标签的ID为vejapplet，且设置详细影像图层不可见，则使用

```
"document.getElementById('vejapplet').getSubApplet()
.setParticularSurfaceLayerAction(false)";
```

备注： 详细影像图层包括了到达一定海拔高度后所能看到地表的较详细的地貌影像

3.3 设置国界图层是否可见

接口函数：`void setCountryBoundariesAction(boolean isActive)`
接口说明：设置国界图层是否可见

参数说明： **boolean** isActive：可见标识，true 表示可见，false 表示不可见
返回值： 无
调用示例： 如果Applet标签的ID为vejapplet，且设置国界图层不可见，则使用
```
"document.getElementById('vejapplet').getSubApplet()
.setCountryBoundariesAction(false)";
```
备注： 无

3.4 设置地名图层是否可见

接口函数： **void** setPlaceNamesAction(**boolean** isActive)
接口说明： 设置地名图层是否可见
参数说明： **boolean** isActive：可见标识，true 表示可见，false 表示不可见
返回值： 无
调用示例： 如果Applet标签的ID为vejapplet，且设置地名图层不可见，则使用
```
"document.getElementById('vejapplet').getSubApplet()
.setPlaceNamesAction(false)";
```
备注： 地名图层包括了各个洲、海洋、国家名以及部分地名

3.5 设置鹰眼图层是否可见

接口函数： **void** setWorldMapLayerAction(**boolean** isActive)
接口说明： 设置鹰眼图层是否可见
参数说明： **boolean** isActive：可见标识，true 表示可见，false 表示不可见
返回值： 无
调用示例： 如果Applet标签的ID为vejapplet，且设置鹰眼图层不可见，则使用
```
"document.getElementById('vejapplet').getSubApplet()
.setWorldMapLayerAction(false)";
```
备注： 鹰眼指数字地球左上角的世界地图鹰眼指示

3.6 设置比例尺是否可见

接口函数： **void** setScalebarLayerAction(**boolean** isActive)
接口说明： 设置比例尺是否可见
参数说明： **boolean** isActive：可见标识，true 表示可见，false 表示不可见
返回值： 无
调用示例： 如果Applet标签的ID为vejapplet，且设置比例尺不可见，则使用
```
"document.getElementById('vejapplet').getSubApplet()
.setScalebarLayerAction(false)";
```
备注： 比例尺指数字地球右下方的比例尺指示

3.7 设置指北针是否可见

接口函数： `void setCompassLayerAction(boolean isActive)`
接口说明： 设置指北针是否可见
参数说明： **`boolean`** `isActive`：可见标识，`true` 表示可见，`false` 表示不可见
返回值： 无
调用示例： 如果`Applet`标签的`ID`为`vejapplet`，且设置指北针不可见，则使用

```
"document.getElementById('vejapplet').getSubApplet()
.setCompassLayerAction(false)";
```

备注： 指北针指数字地球右上方的视点方向指示

3.8 设置产品 Logo 是否可见

接口函数： `void setLogoLayerAction(boolean isActive)`
接口说明： 设置产品 `Logo` 是否可见
参数说明： **`boolean`** `isActive`：可见标识，`true` 表示可见，`false` 表示不可见
返回值： 无
调用示例： 如果`Applet`标签的`ID`为`vejapplet`，且设置产品`Logo`不可见，则使用

```
"document.getElementById('vejapplet').getSubApplet()
.setLogoLayerAction(false)";
```

备注： 产品`Logo`指数字地球左下方的产品`Logo`图

3.9 设置用户自定义图层是否可见

接口函数： `void setLayerAction(String layerName, boolean isActive)`
接口说明： 设置产品 `Logo` 是否可见
参数说明： `String layerName`：用户自定义图层名
`boolean` `isActive`：可见标识，`true` 表示可见，`false` 表示不可见
返回值： 无
调用示例： 如果`Applet`标签的`ID`为`vejapplet`，且设置“武汉”图层不可见，则使用

```
"document.getElementById('vejapplet').getSubApplet()
.setLayerAction('武汉', false)";
```

备注： 用户自定义的图层名称需要和MapEarth的配置文档WebConfig中配置的图层名称保持一致，否则函数调用失败。具体如何添加用户自定图层，请参看《数字地球操作手册篇》

4 标注点控制接口介绍

4.1 添加地标

4.1.1 添加默认地标

接口函数：
```
boolean addPlacemark(String placeName,
                     double latitude, double longitude)
```
接口说明：向指定位置添加使用默认图标的地标

参数说明：`String placeName`：地标名称

`double latitude`：地标添加点纬度

`double longitude`：地标添加点经度

返回值：添加成功返回`true`，否则返回`false`

调用示例：如果Applet标签的ID为`vejapplet`，添加一个名字为“武汉”，纬度为30.0度，经度为114.0度，图标为默认图标，可见高度为默认值的地标，则使用

```
"document.getElementById('vejapplet').getSubApplet()
.addPlacemark('武汉', 30.0, 114.0)";
```

备注：默认地标可见高度区间为`[0, 29000000]`，单位：米

4.1.2 添加带有描述的地标

接口函数：
```
boolean addPlacemark(String placeName, String description,
                     double latitude, double longitude)
```
接口说明：向指定位置添加使用带有简单描述信息及默认图标的地标

参数说明：`String placeName`：地标名称

`String description`：地标简单描述

`double latitude`：地标添加点纬度

`double longitude`：地标添加点经度

返回值：添加成功返回`true`，否则返回`false`

调用示例：如果Applet标签的ID为`vejapplet`，添加一个名字为“武汉”，描述信息为“湖北省会”，纬度为30.0度，经度为114.0度，图标为默认图标，可见高度为默认值的地标，则使用

```
"document.getElementById('vejapplet').getSubApplet()
.addPlacemark('武汉', '湖北省会', 30.0, 114.0)";
```

备注：当鼠标点击地标图标时，会在地球上显示一个气泡，用来展示地标的描述信息

4.1.2 添加设定可见高度区间的地标

接口函数：
```
boolean addPlacemark(String placeName, String description,
                     double latitude, double longitude,
                     double minDisplayDistance,
                     double maxDisplayDistance)
```
接口说明：向指定位置添加使用带有简单描述信息及默认图标的地标

参数说明：String placeName：地标名称
String description：地标简单描述
double latitude：地标添加点纬度
double longitude：地标添加点经度
double minDisplayDistance：地标最低可见高度
double maxDisplayDistance：地标最高可见高度

返回值： 添加成功返回 true，否则返回 false

调用示例：如果Applet标签的ID为vejapplet，添加一个名字为“武汉”，描述信息为“湖北省会”，纬度为30.0度，经度为114.0度，图标为默认图标，可见高度区间为5-3000米的地标，则使用
```
"document.getElementById('vejapplet').getSubApplet()
.addPlacemark('武汉', '湖北省会', 30.0, 114.0, 5, 3000)";
```
备注： 只有当视点高度在 5-3000 米范围内时，才可以见到该地标

4.1.3 添加自定义图标的地标

接口函数：
```
boolean addPlacemark(String placeName, String description,
                     boolean isWebImage, String iconImage,
                     double latitude, double longitude,
                     double minDisplayDistance,
                     double maxDisplayDistance)
```
接口说明：向指定位置添加使用带有简单描述信息及默认图标的地标

参数说明：String placeName：地标名称
String description：地标简单描述
boolean isWebImage：图标是否为自定义
String iconImage：图标文件路径及文件名
double latitude：地标添加点纬度
double longitude：地标添加点经度
double minDisplayDistance：地标最低可见高度
double maxDisplayDistance：地标最高可见高度

返回值： 添加成功返回 true，否则返回 false

调用示例：如果Applet标签的ID为vejapplet，添加一个名字为“武汉”，描述信息为“湖北省会”，纬度为30.0度，经度为114.0度，图标路径指定为/WebRoot/images/example.gif，可见高度区间为5-3000米的地标，则使用"document.getElementById('vejapplet').getSubApplet()
.addPlacemark('武汉', '湖北省', true,

'images/example.gif', 30.0, 114.0, 5, 3000)";

备注： 无

4.1.4 添加完全自定义的地标

接口函数：
```
boolean addPlacemark(String placeName, String description,
                     boolean isWebImage, String iconImage,
                     double latitude, double longitude,
                     double minDisplayDistance,
                     double maxDisplayDistance,
                     int fontSize,
                     int colorR, int colorG, int colorB,
                     int iconSize)
```

接口说明： 向指定位置添加使用带有简单描述信息及默认图标的地标

参数说明： String placeName：地标名称
String description：地标简单描述
boolean isWebImage：图标是否为自定义
String iconImage：图标文件路径及文件名
double latitude：地标添加点纬度
double longitude：地标添加点经度
double minDisplayDistance：地标最低可见高度
double maxDisplayDistance：地标最高可见高度
int fontSize：地标名称字体大小
int colorR：地标名称颜色的 R 分量，区间[0, 255]
int colorG：地标名称颜色的 G 分量，区间[0, 255]
int colorB：地标名称颜色的 B 分量，区间[0, 255]
int iconSize：地标图标尺寸，区间[1, 80]

返回值： 添加成功返回 true，否则返回 false

调用示例： 如果Applet标签的ID为vejapplet，添加一个名字为“武汉”，描述信息为“湖北省会”，纬度为30.0度，经度为114.0度，图标路径指定为/WebRoot/images/example.gif，可见高度区间为5-3000米，地表名称字体大小为12，颜色为[128,128,128]，图标尺寸为20的地标，则使用

```
"document.getElementById('vejapplet').getSubApplet()
.addPlacemark('武汉','湖北省',true,

'images/example.gif',30.0,114.0,5,3000,1,128,128,128,20)";
```

备注： 无

4.2 控制地标是否显示

4.2.1 控制指定地标是否显示

接口函数： `void setPlacemarkVisiable(String placeName, boolean isvisiable)`
接口说明： 通过指定地标名，控制指定的地标显示
参数说明： `String placeName`：地标名称
`boolean isvisiable`：显示标识，显示为 `true`，否则为 `false`
返回值： 无
调用示例： 如果Applet标签的ID为vejapplet，指定名字为“武汉”的地标不显示，则使用`"document.getElementById('vejapplet').getSubApplet().setPlacemarkVisiable('武汉', false)";`
备注： 无

4.2.2 控制全部地标是否显示

接口函数： `void setPlacemarkVisiable(boolean isvisiable)`
接口说明： 通过指定地标名，控制指定的地标显示
参数说明： `boolean isvisiable`：显示标识，显示为 `true`，否则为 `false`
返回值： 无
调用示例： 如果Applet标签的ID为vejapplet，指定所有地标不显示，则使用`"document.getElementById('vejapplet').getSubApplet().setPlacemarkVisiable(false)";`
备注： 无

4.3 控制地标是否可移动

接口函数： `void setPlacemarkActive(String placeName, boolean isvisiable)`
接口说明： 通过指定地标名，控制指定的地标为是否可移动
参数说明： `String placeName`：地标名称
`boolean isvisiable`：设置标识，可移动为 `true`，否则为 `false`
返回值： 无
调用示例： 如果Applet标签的ID为vejapplet，指定名字为“武汉”的地标不可移动，则使用`"document.getElementById('vejapplet').getSubApplet().setPlacemarkActive('武汉', false)";`
备注： 无

4.4 获取地标信息

接口函数：`String getPlacemarkInfo(String placeName)`
接口说明：通过指定地标名，获取指定的地标信息
参数说明：`String placeName`：地标名称
返回值： 返回地标的信息，格式为"纬度,经度,高度,朝向,倾斜角"；
如果获取失败，则返回"Error";如果没有该地标，则返回""
调用示例：如果Applet标签的ID为vejapplet，获取指定名字为“武汉”的地标信息，
则使用`"document.getElementById('vejapplet').getSubApplet()
.getPlacemarkInfo('武汉')";`
备注： 无

4.5 移除地标

4.5.1 移除指定地标

接口函数：`void removePlacemark(String placeName)`
接口说明：通过指定地标名，移除指定的地标
参数说明：`String placeName`：地标名称
返回值： 无
调用示例：如果Applet标签的ID为vejapplet，要移除指定名字为“武汉”的地标，
则使用`"document.getElementById('vejapplet').getSubApplet()
.removePlacemark('武汉')";`
备注： 无

4.5.1 移除全部地标

接口函数：`void removePlacemarkAll()`
接口说明：移除现有全部地标
参数说明：无
返回值： 无
调用示例：如果Applet标签的ID为vejapplet，要移除现有的全部的地标，则使用
`"document.getElementById('vejapplet').getSubApplet()
.removePlacemarkAll()";`
备注： 无

5 绘制接口介绍

5.1 绘制曲线

5.1.1 绘制基本曲线

接口函数：
```
void drawSurfacePolyline(String picName, int borderColorR,
                         int borderColorG, int borderColorB,
                         boolean isResponse)
```
接口说明：可以通过调用此函数后开始在数字地球表面上绘制曲线，点击鼠标左键确认曲线节点，点击鼠标右键结束绘制

参数说明：String picName：线名称
int borderColorR：曲线颜色的 R 分量，区间[0,255]
int borderColorG：曲线颜色的 G 分量，区间[0,255]
int borderColorB：曲线颜色的 B 分量，区间[0,255]
boolean isResponse：是否向指定页面发送曲线节点序列

返回值： 无

调用示例：如果Applet标签的ID为vejapplet，要开始在地球表面绘制名为"直线"，RGB为[128,128,128]的曲线，同时令获取页面得到曲线节点序列，则使用
```
"document.getElementById('vejapplet').getSubApplet()
.drawSurfacePolyline('直线', 128, 128, 128, true)";
```
然后通过鼠标点击在地球上进行绘制曲线操作

备注： 在数字地球 applet 页面中，可以通过参数指定获取数据页面，参数为 applet 标签下的 targetPage 与 targetFrame 两个参数

5.1.2 绘制指定名称颜色的基本曲线

接口函数：
```
void drawSurfacePolyline(String picName, int borderColorR,
                         int borderColorG, int borderColorB,
                         int noteColorR, int noteColorG, int noteColorB,
                         boolean isResponse)
```
接口说明：可以通过调用此函数后开始在数字地球表面绘制曲线，点击鼠标左键确认曲线节点，点击鼠标右键结束绘制

参数说明：String picName：线名称
int borderColorR：曲线颜色的 R 分量，区间[0,255]
int borderColorG：曲线颜色的 G 分量，区间[0,255]
int borderColorB：曲线颜色的 B 分量，区间[0,255]
int noteColorR：曲线名称颜色的 R 分量，区间[0,255]
int noteColorG：曲线名称颜色的 G 分量，区间[0,255]
int noteColorB：曲线名称颜色的 B 分量，区间[0,255]
boolean isResponse：是否向指定页面发送曲线节点序列

返回值： 无

调用示例： 如果Applet标签的ID为vejapplet，要开始在地球表面绘制名为"直线"，曲线和名称RGB为[128,128,128]的曲线，同时令获取页面得到曲线节点序列，则使用

```
"document.getElementById('vejapplet').getSubApplet()
.drawSurfacePolyline('直线',128,128,128,128,128,128,
          true)";
```

然后通过鼠标点击在地球上进行绘制曲线操作

备注： 无

5.1.3 通过点序列绘制基本曲线

接口函数：

```
void drawSurfacePolyline(String picName, String points,
                int borderColorR, int borderColorG,
                int borderColorB)
```

接口说明： 可以通过调用此函数在数字地球表面自动绘制曲线

参数说明： String picName：线名称

String points：曲线节点序列字符串，格式"纬度,经度;纬度,经度;..."

int borderColorR：曲线颜色的 R 分量，区间[0,255]

int borderColorG：曲线颜色的 G 分量，区间[0,255]

int borderColorB：曲线颜色的 B 分量，区间[0,255]

返回值： 无

调用示例： 如果Applet标签的ID为vejapplet，要通过点序列"114.2,39.1;114.2,39.3"在地球上绘制名为"直线"，曲线和名称RGB为[128,128,128]的曲线，则使用

```
"document.getElementById('vejapplet').getSubApplet()
.drawSurfacePolyline('直线','114.2,39.1,0;
 114.2,39.3,1', 128,128,128)";
```

备注： 无

5.1.4 通过点序列绘制指定名称颜色的基本曲线

接口函数：

```
void drawSurfacePolyline(String picName, String points,
                int borderColorR, int borderColorG,
                int borderColorB,int noteColorR,
                int noteColorG, int noteColorB)
```

接口说明： 可以通过调用此函数在数字地球表面自动绘制曲线

参数说明： String picName：线名称

String points：曲线节点序列字符串，格式"纬度,经度;纬度,经度;..."

int borderColorR：曲线颜色的 R 分量，区间[0,255]

int borderColorG：曲线颜色的 G 分量，区间[0,255]

int borderColorB：曲线颜色的 B 分量，区间[0,255]

int noteColorR：曲线名称颜色的 R 分量，区间[0,255]

int noteColorG：曲线名称颜色的 G 分量，区间[0,255]
int noteColorB：曲线名称颜色的 B 分量，区间[0,255]

返回值： 无

调用示例： 如果Applet标签的ID为vejapplet，要通过点序列"114.2,39.1;114.2,39.3"在地球表面绘制名为"直线"，曲线和名称RGB为[128,128,128]的曲线，则使用

```
"document.getElementById('vejapplet').getSubApplet()
.drawSurfacePolyline('直线','114.2,39.1,0;
 114.2,39.3,1',128,128,128,128,128,128)";
```

备注： 无

5.2 绘制多边形

5.2.1 绘制基本多边形

接口函数：

```
void drawSurfacePolygon(String picName,
                int colorR, int colorG, int colorB,
                int borderColorR, int borderColorG,
                int borderColorB, boolean isResponse)
```

接口说明： 可以通过调用此函数后开始在数字地球表面上绘制多边形，点击鼠标左键确认多边形顶点，点击鼠标右键结束绘制

参数说明： String picName：多边形名称
int colorR：多边形区域内颜色 R 分量，区间[0,255]
int colorG：多边形区域内颜色 G 分量，区间[0,255]
int colorB：多边形区域内颜色 B 分量，区间[0,255]
int borderColorR：边界线颜色的 R 分量，区间[0,255]
int borderColorG：边界线颜色的 G 分量，区间[0,255]
int borderColorB：边界线颜色的 B 分量，区间[0,255]
boolean isResponse：是否向指定页面发送曲线节点序列

返回值： 无

调用示例： 如果Applet标签的ID为vejapplet，要开始在地球表面绘制名为"三角形"，线与面的RGB都为[128,128,128]的多边形，同时令获取页面得到多边形顶点序列，则使用

```
"document.getElementById('vejapplet').getSubApplet()
.drawSurfacePolygon('三角形', 128, 128, 128, 128, 128, 128,
          true)";
```

然后通过鼠标点击在地球上进行绘制多边形操作

备注： 无

5.2.2 绘制指定名称颜色的基本多边形

接口函数：

```
void drawSurfacePolygon(String picName,
```

```
                    int colorR, int colorG, int colorB,
                    int borderColorR, int borderColorG,
                    int borderColorB,
                    int noteColorR, int noteColorG, int noteColorB,
                    boolean isResponse)
```

接口说明：可以通过调用此函数后开始在数字地球表面上绘制多边形，点击鼠标左键确认多边形顶点，点击鼠标右键结束绘制

参数说明：String picName：多边形名称

int colorR：多边形区域内颜色 R 分量，区间[0,255]

int colorG：多边形区域内颜色 G 分量，区间[0,255]

int colorB：多边形区域内颜色 B 分量，区间[0,255]

int borderColorR：边界线颜色的 R 分量，区间[0,255]

int borderColorG：边界线颜色的 G 分量，区间[0,255]

int borderColorB：边界线颜色的 B 分量，区间[0,255]

int noteColorR：曲线名称颜色的 R 分量，区间[0,255]

int noteColorG：曲线名称颜色的 G 分量，区间[0,255]

int noteColorB：曲线名称颜色的 B 分量，区间[0,255]

boolean isResponse：是否向指定页面发送曲线节点序列

返回值： 无

调用示例：如果Applet标签的ID为vejapplet，要开始在地球表面绘制名为"三角形"，线、面、多边形名称的RGB都为[128,128,128]的多边形，同时令获取页面得到多边形顶点序列，则使用

```
"document.getElementById('vejapplet').getSubApplet()
.drawSurfacePolygon('三角形', 128, 128, 128, 128, 128, 128,
          128, 128, 128, true)";
```

然后通过鼠标点击在地球上进行绘制多边形操作

备注： 无

5.2.3 通过点序列绘制基本多边形

```
接口函数：void drawSurfacePolygon(String picName, String points,
                    int colorR, int colorG, int colorB,
                    int borderColorR, int borderColorG,
                    int borderColorB)
```

接口说明：可以通过调用此函数在数字地球表面自动绘制多边形

参数说明：String picName：多边形名称

String points：多边形顶点序列字符串，格式"纬度,经度;纬度,经度;..."

int colorR：多边形区域内颜色 R 分量，区间[0,255]

int colorG：多边形区域内颜色 G 分量，区间[0,255]

int colorB：多边形区域内颜色 B 分量，区间[0,255]

int borderColorR：边界线颜色的 R 分量，区间[0,255]

int borderColorG：边界线颜色的 G 分量，区间[0,255]

int borderColorB：边界线颜色的 B 分量，区间[0,255]

返回值： 无

调用示例： 如果Applet标签的ID为vejapplet，要通过点序列"114.2,39.1;114.3,39.3;114.3,39.4"在地球上绘制名为"三角形"，边界和面的RGB都为[128,128,128]的三角形，则使用

```
"document.getElementById('vejapplet').getSubApplet()
.drawSurfacePolygon('三角形', '114.2,39.1;
 114.2,39.3;114.3,39.4', 128, 128, 128, 128, 128, 128)";
```

备注： 无

5.2.4 通过点序列绘制指定名称颜色的基本多边形

接口函数：

```
void drawSurfacePolygon(String picName, String points,
                        int colorR, int colorG, int colorB,
                        int borderColorR, int borderColorG,
                        int borderColorB,
                        int noteColorR, int noteColorG, int noteColor)
```

接口说明： 可以通过调用此函数后开始在数字地球表面上绘制多边形，点击鼠标左键确认多边形顶点，点击鼠标右键结束绘制

参数说明：

String picName：多边形名称

String points：多边形顶点序列字符串，格式"纬度,经度;纬度,经度;..."

int colorR：多边形区域内颜色 R 分量，区间[0,255]

int colorG：多边形区域内颜色 G 分量，区间[0,255]

int colorB：多边形区域内颜色 B 分量，区间[0,255]

int borderColorR：边界线颜色的 R 分量，区间[0,255]

int borderColorG：边界线颜色的 G 分量，区间[0,255]

int borderColorB：边界线颜色的 B 分量，区间[0,255]

int noteColorR：曲线名称颜色的 R 分量，区间[0,255]

int noteColorG：曲线名称颜色的 G 分量，区间[0,255]

int noteColorB：曲线名称颜色的 B 分量，区间[0,255]

返回值： 无

调用示例： 如果Applet标签的ID为vejapplet，要通过点序列"114.2,39.1;114.3,39.3;114.3,39.4"在地球上绘制名为"三角形"，边界和面的RGB都为[128,128,128]的三角形，则使用

```
"document.getElementById('vejapplet').getSubApplet()
.drawSurfacePolygon('三角形', '114.2,39.1;
 114.2,39.3;114.3,39.4',
 128,128,128,128,128,128,128,128,128)";
```

备注： 无

5.3 清除图形绘制

5.3.1 清除指定的图形

接口函数： `void clearShape(String picName)`
接口说明： 可以通过调用此函数清除指定名称的数字地球表面上绘制的图形
参数说明： `String picName`：要清除的图形的名称
返回值： 无
调用示例： 如果Applet标签的ID为vejapplet，要清除图形名为"三角形"的图形，则使用"document.getElementById('vejapplet').getSubApplet().clearShape('三角形')";
备注： 无

5.3.2 清除最后绘制的图形

接口函数： `void clearLastShape()`
接口说明： 可以通过调用此函数清除最后在地球上绘制的图形
参数说明： 无
返回值： 无
调用示例： 如果Applet标签的ID为vejapplet，要清除最后绘制的图形，则使用"document.getElementById('vejapplet').getSubApplet().clearLastShape()";
备注： 无

5.3.3 清除最后绘制的图形

接口函数： `void clearAllShapes()`
接口说明： 可以通过调用此函数清除全部在地球上绘制的图形
参数说明： 无
返回值： 无
调用示例： 如果Applet标签的ID为vejapplet，要清除全部绘制的图形，则使用"document.getElementById('vejapplet').getSubApplet().clearAllShapes()";
备注： 无

5.4 画线并测距

接口函数：
```
public void computeDistance(int colorR, int colorG,
                    int colorB, int borderColorR,
                    int borderColorG, int borderColorB)
```

接口说明：可以通过调用此函数在数字地球表面上绘制曲线并得到该曲线的长度
参数说明：int colorR：距离标识的颜色 R 分量，区间[0,255]
int colorG：距离标识的颜色 G 分量，区间[0,255]
int colorB：距离标识的颜色 B 分量，区间[0,255]
int borderColorR：曲线的颜色 R 分量，区间[0,255]
int borderColorG：曲线的颜色 G 分量，区间[0,255]
int borderColorB：曲线的颜色 B 分量，区间[0,255]
返回值： 无
调用示例：如果Applet标签的ID为vejapplet，要开始在地球上画线并测距，所有颜色设置为[128,128,128]，则使用
```
"document.getElementById('vejapplet').getSubApplet()
.computeDistance(128, 128, 128, 128, 128, 128)";
```
备注： 调用函数后，鼠标左键点击开始画线的节点，右击结束

5.5 绘制状态接口

5.5.1 检验绘制状态

接口函数：`boolean isDrawShape(int op)`
接口说明：检验当前绘制状态
参数说明：int op：绘图状态码 1：画线、2：画多边形、3：画地表线、4：画区、5：画地表圆、6：画地表点、7：贴图、8：测距
返回值： 无
调用示例：如果Applet标签的ID为vejapplet，要检查是否为绘制地表曲线，则使用
```
"document.getElementById('vejapplet').getSubApplet()
.isDrawShape(3)";
```
备注： 无

6 其他接口介绍

6.1 获取地球半径

接口函数：`double getEarthRadius()`
接口说明：可以通过调用此函数得到地球的半径
参数说明：无
返回值： 地球半径
调用示例：如果Applet标签的ID为vejapplet，要得到地球的半径，则使用
```
"document.getElementById('vejapplet').getSubApplet()
.getEarthRadius()";
```
备注： 无

6.2 计算两点的球面距离

接口函数：`double computeSphereDistance(double lat1, double lon1, double lat2, double lon2)`

接口说明：可以通过调用此函数得到地球的半径

参数说明：`double lat1`：第一个点的纬度
`double lon1`:第一个点的经度
`double lat2`:第二个点的纬度
`double lon2`:第二个点的经度

返回值： 地球半径

调用示例：如果Applet标签的ID为vejapplet，要得到"39.25,114.12"和"40.37,114.86"两点间的球面距离，则使用

```
"document.getElementById('vejapplet').getSubApplet()
.computeSphereDistance(39.25, 114.1, 40.37, 114.86)";
```

备注： 无