|  |
| --- |
|  |
| YimaEnc HTML5 SDK 开发手册 |
|  |
|  |
|  |
|  |

目录

[YimaEnc HTML5 SDK 开发手册 2](#_Toc421894516)

[一、 更新记录 2](#_Toc421894517)

[二、SDK简介 3](#_Toc421894518)

[2.1 SDK文件介绍 3](#_Toc421894519)

[2.2 SDK术语 3](#_Toc421894520)

[三、SDK使用说明 4](#_Toc421894521)

[3.1 如何使用SDK 4](#_Toc421894522)

[四、SDK主要模块实现 6](#_Toc421894523)

[4.1 添加船舶 6](#_Toc421894524)

[4.2 船舶轨迹回放 6](#_Toc421894525)

[4.3 动态绘制物标 8](#_Toc421894526)

[4.4 添加物标 8](#_Toc421894527)

[4.5 添加台风 9](#_Toc421894528)

[4.6 测量 10](#_Toc421894529)

[五、SDK接口说明 11](#_Toc421894530)

[5.1 船舶管理功能接口 11](#_Toc421894531)

[5.2 物标管理功能接口 21](#_Toc421894532)

[5.3 台风管理功能接口 28](#_Toc421894533)

[5.4 计算功能接口 30](#_Toc421894534)

[5.5 海图管理接口 31](#_Toc421894535)

[5.6 其它接口 36](#_Toc421894536)

[5.7 特殊接口 37](#_Toc421894537)

[六、SDK对象数据结构 39](#_Toc421894538)

[6.1 船舶相关的对象数据结构 39](#_Toc421894539)

[6.2 物标相关的对象数据结构体 41](#_Toc421894540)

[6.3 台风相关的对象数据结构体 43](#_Toc421894541)

# YimaEnc HTML5 SDK 开发手册

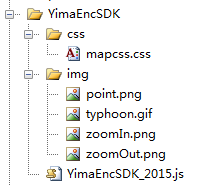
## 更新记录

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **编号** | **版本** | **更改人** | **内容** | **时间** | **审核** |
| 1 | 3.0.0 | 周才壮 | 创建 | 2015/06/12 |  |
| 2 | 3.0.1 | 周才壮 | 1. 新增接口设置点物标旋转接口：API\_SetPointObjRotation； 2. 新增设置海图图片地址接口: API\_SetMapImagesUrl; 3. 给物标结构体ObjectInfo添加了旋转角度属性rotationAngle，该属性只是对点物标有效； 4. 给物标样式结构体ObjectStyle添加了点过滤绘制判距iCheckDrawMinNearOtherLen。 | 2015/08/05 |  |
| 3 | 3.0.2 | 周才壮 | 1. 添加了是否显示工具条接口API\_SetShowToolBarOrNot； 2. 修正了IE9不能滚轮缩放问题； | 2015/08/12 |  |
| 3 | 3.0.3 | 周才壮 | 1. 添加可设置海图背景样式功能，新增接口：API\_SetMapBackGroundStyle | 2015/08/08 |  |
| 4 | 3.0.4 | 周才壮 | 1. 选中的船舶显示的样式可以不随着比例尺的变小而变化,新增接口API\_SetFocusShipShowStyleByMinScale 2. 修正了物标图层无法指定显示的比例尺范围bug； 3. 修正无法在IE7、IE8进行绘制功能。目前已经测试通过的浏览器：IE7以上的浏览器版本、火狐Firefox浏览器、谷歌Chrome浏览器、360浏览器、欧朋Opera浏览器、 | 2015/09/21 |  |
| 5 | 3.0.5 | 周才壮 | 1. 修改不能删除自定义物标； 2. 添加接口：根据船舶状态值获取该状态的所有船舶的id或者Pos值API\_GetShipIdOrPosByState； 3. 添加接口：设置图层的信息API\_SetLayerInfoByPos 4. 添加接口：设置物标的信息API\_SetObjectInfoByPos | 2015/10/19 |  |
| 6 | 3.0.5 | 周才壮 | 1. SDK返回海图右键事件方法ReturnOnMouseRightDown； | 2015/11/2 |  |
| 7 | 3.0.6 | 周才壮 | 1. 新增接口：屏幕坐标获取绘制物标信息接口API\_SelectCurScrnShowObjectInfoByScrnPo； 2. 新增接口：设置根据ID设置物标高亮API\_SetCurHighLightObjectById； 3. 新增接口：设置高亮物标的样式API\_SetCurHighLightObjectStyle； 4. 新加功能：点物标可以使用矢量符号来绘制的功能； | 2015/11/16 |  |
| 8 | 3.0.7 | 周才壮 | 1. 新增接口：获取当前比例尺API\_GetCurMapScale； 2. 新增接口：获取当前中心点API\_GetCurMapCenterLonLatPo； 3. 修正物标文字样式不起效果问题。 | 2015/12/3 |  |
| 9 | 3.0.8 | 周才壮 | 1. 新增接口：设置比例尺级数API\_SetMapMinMaxScale； 2. 新增接口：获取测距时候，测距点的信息API\_GetCurMeasureArrInfo； 3. 新增接口：测距时候，鼠标左击会激发方法ReturnCurMeasurePoInfoByMouseDown； 4. 修改绘制面物标时候，假如是绘制名称，面就被填充名称颜色问题。 | 2015/12/15 |  |
| 10 | 3.0.9 | 周才壮 | 1. 新增接口：设置动态绘制物标的样式API\_SetDynamicObjStyle； 2. 新增接口：设置动态绘制物标的坐标并刷新API\_SetCurDrawDynamicObjGeoPo； 3. 新增接口：设置显示船名方框样式API\_SetShipShowNameFrameStyle； | 2015/12/24 |  |

## 二、SDK简介

### SDK文件介绍

YimaEnc HTML5 SDK是由一些js文件、css文件、png图片等文件，这些文件全部放置在“YimaEncSDK”文件夹中，这些文件不能缺少任何一个文件，也不能随意变换文件间的相对路径，否则可能会导致SDK不能正常使用，文件结构组成如下图所示：



“css/mapcss.css” 这个css样式是SDK里面使用的样式

“img/point.png” 这个是绘制点物标时，使用默认点图片

“img/typhoon.gif” 显示台风使用的图片

“img/zoomIn.png” 缩放工具条的放大按钮图片

“img/zoomOut.png” 缩放工具条的缩小按钮图片

“YimaEncSDK\_2015.js” SDK引擎主要接口文件

### SDK术语

#### ID和Pos

ID：是对象的唯一标示，同一类的对象的ID是唯一的，不能有重复，类似于数据库中表记录的ID一样。对象的ID假如不手动设置的话，不管SDK做什么操作，该对象的ID是永远不会改变的。

Pos：是对象在SDK引擎中的索引，即这个对象在SDK中同类对象数组中的索引，假如同类对象数组的元素添加或者删除后，该对象的索引即可能会改变。

提示：一般情况下，所有对对象进行操作的接口都是通过索引Pos来实现的，所以在使用操作接口前，就根据id通过接口来获取索引pos后。

#### 当前轨迹

即当前船舶短时间内行驶的轨迹，一般添加船舶那一刻起，每更新一次信息，就是该船舶的最新当前轨迹点，SDK内部为每个船舶保留30个当前轨迹点，假如要显示船舶当前轨迹的话，当前轨迹就是这30个轨迹点。

#### 轨迹回放

轨迹回放包括船舶回放、区域回放两种。

船舶回放：即回放某个时间段内的船舶行驶的轨迹，当然显示行驶轨迹是根据时间的先后顺序进行显示的。

区域回放：即回放某个时间段内，在某些区域内的船舶行驶轨迹，即在区域外的船舶是不会显示的。

#### 动态绘制标绘

即使用鼠标在海图上进行操作绘制出图形的相关功能，包括：绘制物标(绘制点、线、面、矩形、圆)、测距、测面积、电子方位线等功能。

#### 扩展字段

即是对象的一个字符串数组的成员变量，为了让对象扩展出更多的功能，我们给某些对象添加了这个扩展字段属性，比如船舶、物标、图层、台风等等对象。假如对象不需要扩展字段的话，在接口中的对象参数直接输入null即可。

## 三、SDK使用说明

### 如何使用SDK

任何使用html脚本代码的界面都可以使用YimaEnc HTML5 SDK来实现海图功能，比如html、htm、jsp、asp、aspx、php等格式。目前通过测试的浏览器有：IE7以上的IE浏览器版本、火狐Firefox浏览器、谷歌Chrome浏览器、360浏览器、欧朋Opera浏览器、苹果Safari浏览器等。

首先把SDK包拷贝到页面文件所在的目录路径，然后只要在页面上引用SDK包中的YimaEncSDK.js文件，接着在页面上创建一个div标签，这个标签用于显示海图的容器，当然要设置这个div的位置、尺寸大小等样式，其中尺寸大小也就是显示海图视图的大小。因为SDK内部还调用了一写外部方法(即)，所以还需要创建一下这些特殊接口，见下面演示的mapView.js文件中。

例如这个是一个显示海图的html页面的脚本代码：

<html xmlns="http://www.w3.org/1999/xhtml">

<head>

<meta charset="gbk" />

<title>YimaEncMapDemo</title>

<script src="YimaEncSDK/YimaEncSDK\_2015.js" type="text/javascript"></script>

<script src="mapView.js" type="text/javascript"></script>

</head>

<body onload="init()" >

<div id="map" style="width:100%; height:100%;overflow:hidden;position:absolute; left:0px; top:0px;z-index:1;" oncontextmenu="return false;">

</div>

</body>

</html>

html页面上引入了SDK文件” YimaEncSDK/YimaEncSDK\_2015.js”以及mapView.js文件，其中mapView.js文件是海图初始化init()方法和声明特殊接口方法，(特殊接口方法见5.7)，mapView.js文件代码如下：

function init() {

var objMapInfo = [];

objMapInfo.div = "map"; //海图容器div的id，这个对应html页面上的div

objMapInfo.model = "pc"; //使用平台：pc=电脑，android=手机

API\_InitYimaMap(objMapInfo); //初始化海图

API\_SetMapImagesUrl("http://www.yimasoft.net/YimaChartImages/"); //设置图片地址 API\_SetMapViewCenter(123.5, 33.5, 10240000); //设置海图显示中心点和比例尺信息 API\_SetZoomBtnPosition(true, 80, 50); //显示缩放按钮

API\_SetMousePoInfoDivPosition(true, 20, 20); //显示鼠标位置

API\_SetScaleInfoDivPosition(true, 20, 20); //显示比例尺位置

API\_SetScaleLenInfoPosition(true, 20, 55); //显示比例尺长度

}

//------------------------以下方法是sdk内部调用的--------------------------------

//鼠标移动时，选中对象的信息（手机版本这个方法不会被调用）

function ReturnSelectObjByMouseMove(objInfo) {}

//鼠标左击时查询到的对象信息

function ReturnSelectObjByMouseLeftDown(objInfo) {}

//动态绘制物标时，选中点之后返回的坐标

function ReturnDrawDynamicObjNewInfo(objDynamicInfo) {}

//返回测距时的距离

function ReturnCurMeasureDist(CurDis, allMeasureDist, CurDegrees) {}

//返回测面积时的面积大小

function ReturnCurMeasureAreaSize(curAreaSize) { }

当然这个只是简单的显示出海图的功能，假如需要扩展船舶功能、物标功能、台风功能、气象、港口、测量等等功能的话，请参照“四、SDK主要模块实现”并结合我们的综合Demo的代码。

## 四、SDK主要模块实现

### 添加船舶

添加船舶之前，首先要添加船舶的状态样式, 即只要是显示船舶的，就必须先添加船舶状态。船舶的显示是根据船舶状态样式来显示效果的，假如没有船舶状态样式，船舶是不会被绘制显示出来的。添加船舶的具体步骤应该是：

添加船舶状态样式 🡪 添加船舶 🡪 绘制船舶

添加船舶状态样式(船舶样式可以添加多种，只要状态值不一样即可)，船舶状态只要添加一次即可，一般是在海图初始化后就进行添加：

var iStateValue = 1;//状态值,船舶状态=1时会使用这个样式

var iStatePos = API\_AddNewShipState(iStateValue);//添加一种船舶状态

var shipStateStyle = ShipStateStyle;//这个值可参照“6.1.3 ShipStateStyle”结构体，

//一个状态可以有多个状态样式，SDK会根据当前比例尺来筛选选择哪个状态样式来显示船舶

var bResult = API\_AddShipStateStyleByPos(iStatePos, shipStateStyle)

其中API\_SetShipStyleInSmallScale接口是统一设置所有船舶显示小圆点的比例尺

添加一艘船舶：

var shipInfo = curShipInfo;//这个值可参照“6.1.1 ShipInfo”船舶信息结构体

var arrExpAttrValue = { "远洋集团" , "远洋渔船" }; //扩展字段数组

var iShipPos = API\_AddOneShip(shipInfo, arrExpAttrValue);//添加船舶

绘制船舶：

API\_ReDrawShips();//重绘船舶

### 船舶轨迹回放

船舶轨迹回放可以按照船舶回放，也可以按照区域回放。

按照船舶回放：添加的船舶根据时间间隔进行回放，可以同时回放多艘船舶。

按照区域回放：根据时间间隔回放在指定的区域内的船舶，可以同时回放多个区域。

进行回放船舶轨迹前，也是要先添加船舶状态，即只要是显示船舶的，就必须先添加船舶状态，添加船舶状态见“4.1添加船舶”，这里就不再重复了。

回放船舶的步骤如下:

添加船舶状态 🡪 添加船舶(添加区域) 🡪 回放

添加船舶状态：这里就不再重复了，请参考“4.1添加船舶”

添加船舶：

var shipInfo = curShipInfo;//这个值可参照“6.1.1 ShipInfo”船舶信息结构体

var arrExpAttrValue = { "远洋集团" , "远洋渔船" }; //扩展字段数组

var curHistroyTrack = [];//轨迹数组，元素为“6.1.2 ShipTrackPoint”结构体

var iShipPos = API\_AddOnePlayShipInfo(shipInfo, arrExpAttrValue, curHistroyTrack);

//添加一艘回放船舶

添加回放区域：

var objInfo = []; //区域对象

objInfo.showText = "东海捕鱼区";

var arrGeoPo = []; //区域的坐标数组，元素格式例如{ x: 1212641489, y: 351263316};

var iAreaPos = API\_AddPlayArea(objInfo, arrGeoPo);//添加一个回放区域，可以添加多个

回放：

API\_SetPlayHistoryTrackTimeStep(300);//设置回放速度，5分钟/秒

API\_StartPlayShipHistoryTrack(); //进行回放

回放控制相关接口：

API\_StopPlayHistoryTrackOrNot(bStopOrNot)// true=暂停，false=继续

API\_FastPlayHistoryTrack(iFastStep)//负数=后退倍数，正数=快进倍数

API\_EndPlayHistoryTrack()//结束播放

API\_ReStartPlayHistoryTrack();//重播

### 动态绘制物标

使用鼠标在海图上进行绘制点、线、面、矩形、圆等物标。步骤：

设置绘制物标类型 🡪 鼠标在海图选择节点坐标 🡪 获取节点坐标 🡪 结束绘制

设置绘制物标类型：

API\_SetCurDrawDynamicUseType(iState);// 1=点，2=线，3=面，4=矩形，5=圆

鼠标在海图选择节点坐标：

//鼠标在海图上进行点击来选择节点坐标

获取节点坐标：

ReturnDrawDynamicObjNewInfo(objDynamicInfo);//接收动态物标信息

//动态绘制的物标信息会返回到这个方法

//具体请参考特殊接口“5.7.2 ReturnDrawDynamicObjNewInfo”

API\_GetCurDrawDynamicObjGeoPo();//获取本次动态绘制的所有坐标

结束绘制:

API\_SetCurDrawDynamicUseType(0);//0=结束绘制动态标绘

### 添加物标

添加物标前，必须要先添加物标所属的图层，以及图层的样式。图层和图层样式添加一次就可以了，一般海图初始化的时候就添加好。

所以添加物标的步骤为：

添加图层 🡪 添加图层样式 🡪 添加物标 🡪 绘制物标

添加图层：

var curLayerInfo = LayerInfo;//这个值可参照“6.2.1 LayerInfo”图层信息结构体

var arrExpAttrValue = null;//图层扩展字段(字符串数组)，没有就写null

var iLayerPos = API\_AddNewLayer(curLayerInfo, arrExpAttrValue);//添加一个图层

添加图层样式：

//一个图层可以有多种样式

设置图层样式对应不同类型的图层分别调用不同的接口

//---------------------点图层样式------------------------------

var pointStyle = PointStyle; //这个值可参照“6.2.3 PointStyle”点样式信息结构体

var stylePos = API\_AddPointLayerStyleByPos(iLayerPos, pointStyle);

//--------------------线图层样式------------------------------------

var lineStyle = LineStyle; //这个值可参照“6.2.4 LineStyle”线样式信息结构体

var stylePos = API\_ AddLineLayerStyleByPos (iLayerPos, lineStyle);

//-------------------面图层样式---------------------------------------

var faceStyle = FaceStyle; //这个值可参照“6.2.5 FaceStyle”面样式信息结构体

var stylePos = API\_AddFaceLayerStyleByPos(iLayerPos, faceStyle);

添加物标：

var curObjInfo = ObjectInfo;//这个值可参照“6.2.2 ObjectInfo”物标信息结构体

var arrObjPo = [];//坐标点数组，元素格式例如{ x: 1210000000, y: 350000000}

var arrExpAttrValue = null; //扩展字段(字符串数组)，假如没有可以传入null

var objPos = API\_AddNewObject(curObjInfo, arrObjPo, arrExpAttrValue);//添加物标

绘制物标：

API\_ReDrawLayer();//重绘物标

**提示：这个图层、物标功能可以扩展更多的功能：比如气象、港口、航线、监控区域等等功能。**

### 添加台风

添加台风前，首先要设置台风的显示样式，台风显示样式设置一次就行了，一般初始化时候进行设置。添加台风的步骤：

添加台风样式 🡪 添加台风 🡪 添加真实轨迹 🡪 显示

添加台风样式：

var objTyphoonStyle = TyphoonStyle;//该值请参考“6.3.4 TyphoonStyle”台风样式结构体

API\_SetTyphoonStyleColor(objTyphoonStyle);//设置台风样式

添加台风：

var curTyphoonInfo = TyphoonInfo;//该值请参考“6.3.1 TyphoonInfo”台风信息结构体

var arrExpAttrValue = null;//台风扩展字段(字符串数组)

var curTyphoonPos = API\_AddTyphoon(curTyphoonInfo, arrExpAttrValue);//添加台风

添加真实轨迹：

var objTrackInfo = TyphoonTrack;//该值请参考“6.3.2 TyphoonTrack”台风轨迹信息结构体

var arrPredictTracks = [];//预测轨迹点数组，

//该数组元素值请参考“6.3.3 TyphoonPredictTracks”台风预测轨迹点信息结构体

API\_AddOneTyphoonTrack(curTyphoonPos, objTrackInfo, arrPredictTracks);//添加一个真实轨迹

显示:

ReDrawTyphoon();//重绘台风

### 测量

测量包括测距、测面积，测量就是用鼠标在海图上进行点击选择测试点，然后再获取测试结果显示出来即可，步骤：

设置测量类型 🡪 鼠标点击选择测试点 🡪 获取测试结果 🡪 结束

设置测量类型：

API\_SetCurDrawDynamicUseType (iState);// 6=测距，7=测面积

鼠标点击选择测试点：

//鼠标在海图视图内点击选取测试点

获取测试结果:

ReturnCurMeasureDist(CurDis, allMeasureDist, CurDegrees);//返回测量距离

//请参考特殊接口ReturnCurMeasureDist

ReturnCurMeasureAreaSize(curAreaSize);//返回面积

//请参考特殊接口“ReturnCurMeasureAreaSize”

结束:

API\_SetCurDrawDynamicUseType(0);//0=结束绘制动态标绘

## 五、SDK接口说明

### 5.1 船舶管理功能接口

#### API\_AddNewShipState (添加船舶状态)

API\_AddNewShipState(iState)

//添加一种船舶状态

//iState:船舶的状态值(int)

//返回值：船舶状态的pos

#### API\_AddOneShip (添加船舶)

API\_AddOneShip(objShipInfo, arrExpAttrValue)

//添加一艘船舶，假如该船舶已经添加，则更新其信息(根据船舶id判断该船舶是否存在)

//objShipInfo:船舶的结构体

//arrExpAttrValue:船舶的扩展字段数组(字符串数组)，null=不设置扩展字段

//返回值：船舶的索引pos,-1=添加失败

#### API\_AddOnePlayShipInfo

API\_AddOnePlayShipInfo(objShipInfo,arrExpAttrValue, arrHistroyTracks)

//添加一个轨迹回放船舶

//objShipInfo:船舶数据结构体

//arrExpAttrValue:船舶的扩展字段数组(字符串数组)，null=不设置扩展字段

//arrHistroyTracks:船舶轨迹点数据数组(轨迹点结构体数组)

//返回值：true=添加成功，false=添加失败

#### API\_AddPlayArea (添加回放区域)

API\_AddPlayArea(objInfo, arrGeoPoints)

//添加一个回放的区域（多边形区域）

//objInfo:区域的信息对象，海图上显示的名称，格式{showText:showTextValue};

//arrGeoPoints:区域的坐标数组，数组每个对象的格式{x:1210000000,y:310000000}

//返回值：区域的pos,-1=添加失败

#### API\_AddShipStateStyleByPos (添加状态样式)

API\_AddShipStateStyleByPos(iStatePos, objStateStyle)

//添加一种船舶状态的样式(一个船舶状态可以有多个样式(即不同比例尺下显示不同的样式))

//iStatePos:船舶状态值(int)

//objStateStyle:状态样式(船舶状态样式结构体)

#### API\_ClearPlayHistoryTrackInfo(清空轨迹回放数据)

API\_ClearPlayHistoryTrackInfo()

//清空轨迹回放数据

//退出轨迹回放的时候，务必要调用这个接口来清除数据

//返回值：true=设置成功，false=设置失败

#### API\_DelAllShips

API\_DelAllShips()

//删除所有的船舶

//返回值:true=删除成功，false=删除失败

#### API\_DelOneShipByPos

API\_DelOneShipByPos(iShipPos)

//根据船舶的pos删除船舶

//iShipPos:船舶的pos(int)

//返回值:true=删除成功，false=删除失败

#### API\_EndPlayHistoryTrack

API\_EndPlayHistoryTrack()

//结束轨迹回放

//即立刻播放到最后时间

#### API\_FastPlayHistoryTrack

API\_FastPlayHistoryTrack(iFastStep)

//快进:即以当前的播放速度前进几倍

//iFastStep:快进的速度(int),负数则为后退

#### API\_FollowShipByPos

API\_FollowShipByPos(iShipPos)

//设置跟踪船舶

//iShipPos:船舶的pos，-1=取消跟踪

//返回值：true=成功，false=失败

#### API\_FollowPlayShipByPos

API\_FollowPlayShipByPos(iShipPos)

//设置跟踪轨迹回放的船舶

//iShipPos:船舶的pos，-1=取消跟踪

//返回值：true=成功，false=失败

#### API\_GetCurFollowShipInfo

API\_GetCurFollowShipInfo()

//获取当前跟踪的船舶信息

//返回船舶的结构体,null=没有跟踪的船舶

#### API\_GetCurSelectShipInfo

API\_GetCurSelectShipInfo()

//获取当前选中的船舶信息

//返回船舶的结构体,null=没有跟踪的船舶

#### API\_GetPlayHistroyTrackInfoByPos

API\_GetPlayHistroyTrackInfoByPos(iShipPos, iTrackPos)

//根据pos获取历史轨迹回放的船舶轨迹的信息

//iShipPos:船舶的pos

//iTrackPos:轨迹点的pos

//返回值：轨迹的信息结构体，null=没有该轨迹点

#### API\_GetPlayShipInfoByPos

API\_GetPlayShipInfoByPos(iShipPos)

//根据轨迹回放船舶索引pos获取船舶信息

//iShipPos:船舶的pos

//返回值：null=找不到船舶，船舶结构体

#### API\_GetPlayShipPosById

API\_GetPlayShipPosById(iShipId)

//根据轨迹回放的船舶id获取其pos

//iShipId:船舶的id(int)

//返回值：-1=没有该船舶（int）

#### API\_GetShipAllExpAttrByPos

API\_GetShipAllExpAttrByPos(iShipPos)

//获取船舶的所有扩展字段信息

//iShipPos:船舶的pos

//返回值：扩展字段的数组值，null=没有改扩展字段

#### API\_GetShipCountByState

API\_GetShipCountByState(iShipState)

//根据船舶状态获取船舶的数量

//iShipState:船舶的状态值，-1=获取所有船舶(int)

//返回值：船舶的数量

#### API\_GetShipIdOrPosByState

function API\_GetShipIdOrPosByState(iShipState,bGetIdOrPos)

//根据船舶状态获取船舶的ID或者Pos

//iShipState:船舶的状态值(int)

// bGetIdOrPos:true=获取船舶ID,false=获取船舶的Pos

//返回值：船舶的ID或者Pos数组，null=没有船舶

#### API\_GetShipInfoByPos

API\_GetShipInfoByPos(iShipPos)

//根据船舶索引pos获取船舶信息

//iShipPos:船舶的pos

//返回值：null=找不到船舶，船舶结构体

#### API\_GetShipInfoByScrnPo

API\_GetShipInfoByScrnPo(scrnPo, checkScrnLen)

//根据屏幕的坐标查询船舶(该接口只查询当前绘制在屏幕内的船舶)

//scrnPo:屏幕坐标,例如{x:200,y:100};

//checkScrnLen：判距(px),例如10

//返回值：船舶结构体，null=没有船舶的信息

#### API\_GetShipOneExpAttrByPos

API\_GetShipOneExpAttrByPos(iShipPos, iExpAttrPos)

//获取船舶的所有扩展字段信息

//iShipPos:船舶的pos

//iExpAttrPos:扩展字段的索引pos

//返回值：扩展字段值，null=没有改扩展字段

#### API\_GetShipPosById

API\_GetShipPosById(iShipId)

//根据船舶的id获取索引pos

//iShipId:船舶的id(int)

//返回值：-1=没有改船舶（int）

#### API\_GetShipsCount

API\_GetShipsCount()

//获取船舶的数量

//返回值：船舶的数量(int)

#### API\_ReDrawPlayShip

API\_ReDrawPlayShip()

//重绘回放船舶

#### API\_ReDrawShips

API\_ReDrawShips()

//重绘船舶

#### API\_ReStartPlayHistoryTrack

API\_ReStartPlayHistoryTrack()

//历史轨迹重放

#### API\_SelectShipByCondition

API\_SelectShipByCondition(strShipNameOrMmsi)

//根据关键字查询船舶

//strShipNameOrMmsi:船舶名称或者MMSi号关键字

//返回符合条件的船舶结构体数组，null=没有符合船舶

#### API\_SetAllShipsShowOrNot

API\_SetAllShipsShowOrNot(bShowOrNot)

//设置是否显示船舶

//bShowOrNot:设置是否显示船舶，true=显示，false=不显示

//返回值：true=成功，false=失败

#### API\_SetIsShowOfflineShipOrNot

API\_SetIsShowOfflineShipOrNot(bShowOrNot)

//设置是否显示离线船舶

//bShowOrNot:true=显示离线船舶，false=不显示离线船舶

//返回值：true=成功，false=失败

#### API\_SetFocusShipShowStyleByMinScale

API\_SetFocusShipShowStyleByMinScale(iScale)

//设置选中船舶的显示的样式至少是最小比例尺样式

//即选择的船舶不会随着比例尺的变小而变化，比如设置为640000，那么在10240000比例尺下，选中的船舶还是显示640000比例尺下的样式

//iScale:比例尺

//返回值：true=设置成功，false=设置失败

#### API\_SetOffLineShipStyle

API\_SetOffLineShipStyle(bGetStyle, objStyle)

//设置离线船舶的样式

//bGetStyle:是否启用这个样式，true=启用，false=不启用

//objStyle:个性样式结构体，包括填充颜色、线粗细，线颜色，格式{strFillColor:"#FF0000",iBorderSize:2,strBorderColor:"#FFFFFF"}

#### API\_SetOneShipStyleByPos

API\_SetOneShipStyleByPos(iShipPos, bGetOwnStyle, objStyle)

//设置一个船舶的个性样式

//iShipPos:船舶的pos

//bGetOwnStyle:是否启用自己样式，true=使用自己个性样式，false=使用状态样式,假如这个为false的时候，第三个参数objStyle可以传入null

//objStyle:个性样式结构体，包括填充颜色、线粗细，线颜色，格式{strFillColor:"#FF0000",iBorderSize:2,strBorderColor:"#FFFFFF"}

//返回值：true=设置成功，false=设置失败

#### API\_SetPlayHistoryTrackTimeStep

API\_SetPlayHistoryTrackTimeStep(iTimeStep)

//设置轨迹回放的时间速度

//iTimeStep:秒为单位

//返回值：true=设置成功，false=设置失败

#### API\_SetPlayShipToMapViewCenterByPos

API\_SetPlayShipToMapViewCenterByPos(iShipPos)

//居中轨迹回放船舶到海图界面中心

//iShipPos:船舶的pos

//返回值：true=成功，false=失败

#### API\_SetSaveTrackPoCount

API\_SetSaveTrackPoCount(iSaveTrackPoCount)

//设置保存当前轨迹点的数量

//iSaveTrackPoCount:数量(int)

//返回值：true=成功，false=失败

#### API\_SetSelectPlayShipByPos

API\_SetSelectPlayShipByPos(iShipPos)

//设置选中轨迹回放的船舶

//iShipId:船舶的id，假如取消选中可以输入-1

//返回值，true=成功，false=失败

#### API\_SetSelectShipByPos

API\_SetSelectShipByPos(iShipPos)

//设置选中船舶，即显示船舶的选中边框

//iShipId:船舶的id，假如取消选中可以输入-1

//返回值，true=成功，false=失败

#### API\_SetShipAllExpAttrByPos

API\_SetShipAllExpAttrByPos(iShipPos, arrAllExpAttr)

//获取船舶的所有扩展字段信息

//iShipPos:船舶的pos

//arrAllExpAttr:扩展字段值数组

//返回值：true=设置成功，false=失败

#### API\_SetShipHeadLineSize

API\_SetShipHeadLineSize(iLineSize)

//设置船首向线的长度

//iLineSize:长度(px),假如不显示可以设置为0

//返回值：true=成功，false=失败

#### API\_SetShipOneExpAttrByPos

API\_SetShipOneExpAttrByPos(iShipPos, iExpAttrPos, strExpAttr)

//获取船舶的所有扩展字段信息

//iShipPos:船舶的pos

//iExpAttrPos:扩展字段pos

//strExpAttr:扩展字段值

//返回值：true=设置成功，false=失败

#### SetShipShowNameFrameStyle

API\_SetShipShowNameFrameStyle(iStartScale,iWidth,iHeight,iBorder, strColor, iOpacity)

//设置船舶名称方框样式

//iStartScale:开始显示名称方框比例尺,例如10000

//iWidth：方框的长度，例如100

//iHeight：方框的高度。例如20

//iBorder：方框边粗细，例如1

//strColor：方框的颜色，例如"#FF0000"

//iOpacity：透明度，例如80

#### API\_SetShipsShowOrNotByShipState

API\_SetShipsShowOrNotByShipState(iShipState, bShowOrNot)

//根据船舶的状态设置该状态的船舶是否显示

//iShipState:船舶的状态

//bShowOrNot:是否显示，true=显示，false=不显示

#### API\_SetShipStyleInSmallScale

API\_SetShipStyleInSmallScale(iSmallScale, strColor, iCircleShipR)

//设置船舶在小比例尺下的样式，是绘制圆

//iSmallScale:绘制圆的起始比例尺(int)

//strColor:颜色，例如:"#FF0000"

//iCircleShipR:半径，单位是px,(int)

#### API\_SetShipToMapViewCenterByPos

API\_SetShipToMapViewCenterByPos(iShipPos)

//居中船舶到海图界面中心

//iShipPos:船舶的pos

//返回值：true=成功，false=失败

#### API\_SetShowPlayShipTrackOrNotByPos

API\_SetShowPlayShipTrackOrNotByPos(iShipPos, bShowOrNot)

//显示轨迹回放船舶轨迹

//iShipPos:船舶的pos(int)

//bShowOrNot:是否显示轨迹，true=显示，false=不显示

//返回值：true=设置成功，false=设置失败

#### API\_SetShowShipInfo

API\_SetShowShipInfo(bShowInfo, iStartShowScale, bShowNameOrMmsi, bNameToNullMmsi)

//设置显示船舶信息，

//bShowInfo:是否显示船舶信息，true=显示船舶信息，false=不显示船舶信息

//iStartShowScale:开始显示的比例尺，例如640000即1:1~1:640000之间的比例尺会显示船舶信息

//bShowNameOrMmsi:显示船名还是mmsi,true=船名，false=mmsi

//bNameToNullMmsi:当显示mmsi，假如mmsi非法值，是否用船名来显示，true=是，false=不显示

#### API\_SetShowShipInfoStyle

API\_SetShowShipInfoStyle(strFont, strColor, iOpacity)

//设置显示船舶信息的样式

//strFont:字体大小，例如"13px 宋体"；

//strColor:字体颜色，例如"#FF0000";

//iOpacity:透明度，0~100，100为不透明

#### API\_SetShowShipTrackOrNotByPos

API\_SetShowShipTrackOrNotByPos(iShipPos, bShowOrNot)

//是否显示船舶的当前轨迹

//iShipPos:船舶的pos(int)

//bShowOrNot:是否显示轨迹，true=显示，false=不显示

//返回值：true=设置成功，false=设置失败

#### API\_SetStartShowShipScale

API\_SetStartShowShipScale(iShowShipScale)

//设置是否显示船舶

//iShowShipScale:比例尺，比如640000,即1:1~1:640000比例尺之间才显示船舶(640000也显示)

//返回值：true=成功，false=失败

#### API\_StartPlayShipHistoryTrack

API\_StartPlayShipHistoryTrack()

//开始播放轨迹

//添加完船舶和区域后可以调用该接口进行回放

#### API\_StopPlayHistoryTrackOrNot

API\_StopPlayHistoryTrackOrNot(bStopOrNot)

//是否暂停播放轨迹

//bStopOrNot:true=暂停，false=继续

//返回值：true=成功，false=失败

#### API\_UpdateOneShipDynamicInfoByPos

API\_UpdateOneShipDynamicInfoByPos(iShipPos, objDynamicInfo)

//更新一个船舶的动态信息

//iShipPos:船舶的pos

//objDynamicInfo:动态信息结构体

//返回值：true=成功，false=失败

### 5.2 物标管理功能接口

#### API\_AddFaceLayerStyleByPos

API\_AddFaceLayerStyleByPos(iLayerPos, objFaceStyleInfo)

//给面图层添加一种样式

//iLayerPos:图层的pos

//objFaceStyleInfo:面图层样式对象结构体

//返回值：样式的pos,-1=添加失败

#### API\_AddLineLayerStyleByPos

API\_AddLineLayerStyleByPos(iLayerPos, objLineStyleInfo)

//给线图层添加一种样式

//iLayerPos:图层的pos

//objLineStyleInfo:线图层样式对象结构体

//返回值：样式的pos,-1=添加失败

#### API\_AddNewLayer

API\_AddNewLayer(objLayerInfo, arrExpAttrValue)

//创建一个图层

//objLayerInfo:图层的结构体

//arrExpAttrValue:扩展字段数组，没有可以传入null

//返回值：图层的pos，-1=添加失败

#### API\_AddNewObject

API\_AddNewObject(objInfo, arrGeoPoints, arrExpAttrValue)

//添加一个物标

//objInfo:物标对象结构体

//arrGeoPoints:物标的坐标数组，元素格式例如{ x: 1210000000, y: 350000000}

//arrExpAttrValue:扩展字段数组，没有可以传入null

//返回物标的pos，-1=添加失败

#### API\_AddPointLayerStyleByPos

API\_AddPointLayerStyleByPos(iLayerPos, objPointStyleInfo)

//给点图层添加一种样式

//iLayerPos:图层的pos

//objPointStyleInfo:点图层样式对象结构体

//返回值：样式的pos,-1=添加失败

#### API\_DelLayerByPos

API\_DelLayerByPos(iLayerPos)

//删除一个图层

//假如该图层有物标，那么物标也被删除

//iLayerPos:图层的pos

//返回值：true=删除成功，false=删除失败

#### API\_DelObjectByPos

API\_DelObjectByPos(iLayerPos, iObjPos)

//删除一个物标

//iLayerPos:物标所在的图层pos

//iObjPos:物标的pos

//返回值：true=删除成功，false=删除失败

#### API\_GetCurDrawDynamicObjGeoPo

API\_GetCurDrawDynamicObjGeoPo()

//获取动态绘制的对象坐标

//返回值：坐标数组：{(x,y),(x,y)}，null=没有数据

//例如{(x:121000000,y:31000000),(x:121200000,y:31200000)...}

#### API\_GetLayerAllExpAttrByPos

API\_GetLayerAllExpAttrByPos(iLayerPos)

//获取图层的所有扩展字段值

//iLayerPos:图层的pos

//返回值：数组，null=没有扩展字段

#### API\_GetLayerCount

API\_GetLayerCount()

//获取图层的总数

//返回值：当前图层的总数(int)

#### API\_GetLayerInfoByPos

API\_GetLayerInfoByPos(iLayerPos)

//根据图层的pos获取图层的信息

//iLayerPos:图层的pos

//返回值：图层的结构体，null=没有改图层

#### API\_GetLayerObjectCountByPos

API\_GetLayerObjectCountByPos(iLayerPos)

//根据图层的pos获取该图层的物标数量

//iLayerPos:图层的pos

//返回值：图层的物标数量(int)

#### API\_GetLayerOneExpAttrByPos

API\_GetLayerOneExpAttrByPos(iLayerPos, iExpAttrPos)

//获取图层的一个扩展字段值

//iLayerPos:图层的pos

//iExpAttrPos:扩展字段的pos

//返回值：扩展字段值，null=没有该扩展字段

#### API\_GetLayerPosById

API\_GetLayerPosById(iLayerId)

//根据图层的id获取图层的pos

//返回值：图层的pos,-1=没有该图层

#### API\_GetObjectAllExpAttrByPos

API\_GetObjectAllExpAttrByPos(iLayerPos, iObjPos)

//获取物标的所有扩展字段

//iLayerPos:图层的pos

//iObjPos:物标的pos

//返回值：扩展字段的数组，null=找不到扩展字段

#### API\_GetObjectGeoInfoByPos

API\_GetObjectGeoInfoByPos(iLayerPos, iObjPos)

//根据物标的pos获取物标坐标

//返回坐标数组，数组元素格式例如：{x:1210000000,y:31000000}

#### API\_GetObjectInfoByPos

API\_GetObjectInfoByPos(iLayerPos, iObjPos)

//根据物标的pos获取物标信息

//iLayerPos:图层的pos

//iObjPos:物标的pos

//返回值：物标的结构体，null=没有该物标

#### API\_GetObjectOneExpAttrByPos

API\_GetObjectOneExpAttrByPos(iLayerPos, iObjPos, iExpAttrPos)

//设置物标的所有扩展字段

//iLayerPos:图层的pos

//iObjPos:物标的pos

//iExpAttrPos:扩展字段的pos

//返回值：扩展字段值，null=没有该扩展字段

#### API\_GetObjectPosById

API\_GetObjectPosById(iObjId, iLayerPos)

//根据物标的id获取物标的pos(只返回第一个符合条件的物标)

//iObjId:物标的id

//iLayerPos:指定在该Pos图层中查找，假如在所有图层中查找可以传入null

//返回值：该物标的pos以及所在图层的pos,格式例如：{iLayerPos:1,iObjPos:2},null=找不到

#### API\_ReDrawLayer

API\_ReDrawLayer()

//重绘图层物标

//一般删除或者添加物标之后调用

#### API\_SelectCurScrnShowObjectInfoByScrnPo

API\_SelectCurScrnShowObjectInfoByScrnPo(scrnPo,bGetAllSelObj)

//根据屏幕坐标查询当前屏幕内绘制的物标对象

//scrnPo:屏幕坐标，格式{x,y}

//bGetAllSelObj:是否获取所有选中物标，true=所有物标,false=只返回第一个物标

//返回值：选中的物标对象信息数组，数组元素格式{layerId,objId}，即返回物标所属图层的ID以及物标ID

#### API\_SetCurDrawDynamicUseType

API\_SetCurDrawDynamicUseType(iState)

//设置当前绘制动态图形的用途类型

//iState状态值：0=无操作，1=绘制点，2=绘制线，3=绘制面

#### API\_SetCurDrawDynamicObjGeoPo

API\_SetCurDrawDynamicObjGeoPo(arrGeoPo)

//修改动态绘制的物标坐标

//arrGeoPo:坐标数组,格式{(x:1210000000,y:310000000),...}

//返回值：true=修改成功，false=失败

#### API\_SetDynamicObjStyle

API\_SetDynamicObjStyle(objStyle)

//设置动态绘制物标样式

//objStyle:物标样式结构体，格式如下：{borderWith:1, borderColor:”#FF0000”, fillColor : ”#FF0000”, textColor: ”#FF0000”, fontSize:”12px 宋体”, iOpacity:80}

//borderWith:画笔粗细

//borderColor：画笔颜色

//fillColor：填充颜色

//textColor：字体颜色(主要是测距、测面积使用)

//fontSize：字体大小(主要是测距、测面积使用)

//iOpacity：透明度

#### API\_SetCurHighLightObjectById

API\_SetCurHighLightObjectById(layerId, objId)

//根据物标ID设置当前高亮的物标

//layerId:物标所属图层的ID

//objId:物标ID

//返回值：true=成功，false=失败

#### API\_SetCurHighLightObjectStyle

API\_SetCurHighLightObjectStyle(objStyle)

//设置高亮物标的样式

//objStyle:物标高亮样式对象，格式{borderWith,borderColor,iOpacity,fillColor}即：{画笔粗细,画笔颜色(#FF0000),透明度(0~100),填充颜色(#FF0000)}

#### API\_SetFaceObjStyleByPos

API\_SetFaceObjStyleByPos(iLayerPos, iObjPos, bShowOwnStyle, objStyleInfo)

//设置线物标的个性样式

//iLayerPos:线物标所属的图层pos

//iObjPos:物标的pos

//bShowOwnStyle:是否启用个性样式

//objStyleInfo:个性样式对象，格式{线粗细,线颜色,是否填充,填充颜色,透明度}，例如{borderWith:2,borderColor:"#FF0000",bFilled:true,fillColor:"#FF000000",iOpacity:80},

#### API\_SetIsDrawLayerObjOrNot

API\_SetIsDrawLayerObjOrNot(bShowOrNot)

//设置是否绘制图层标注

//假如设置不绘制，那么所有的图层物标都不会显示，不管该图层是否设置了显示或者不显示

//bShowOrNot:是否绘制，true=绘制，false=不绘制

//返回值：true=设置成功，false=设置失败

#### API\_SetLayerAllExpAttrByPos

API\_SetLayerAllExpAttrByPos(iLayerPos, arrAllExpAttr)

//设置图层的扩展字段

//iLayerPos:图层的pos

//arrAllExpAttr:扩展字段数组值

//返回值：true=设置成功，false=设置失败

#### API\_SetLayerInfoByPos

API\_SetLayerInfoByPos(iLayerPos, iLayerId, name, bShow, minShowScale, maxShowScale, bShowTextOrNot, iStartShowTextScale, arrExpAttrValue)

//通过索引设置图层的信息

//iLayerPos:图层的索引

//iLayerId:图层id

//name:图层名称

//bShow:是否显示

//minShowScale:显示最大比例尺(int),例如5000

//maxShowScalea:显示的最小比例尺（int）,例如1260000

//bShowTextOrNot:图层的物标是否显示名称(bool),true=显示，false=不显示

//iStartShowTextScale:显示名称的最大比例尺（int）,例如320000

//arrExpAttrValue:扩展字段，数组

//返回值：true=成功，false=失败

#### API\_SetLayerObjShowTextOrNot

API\_SetLayerObjShowTextOrNot(iLayerPos, bShowTextOrNot, iStartShowTextScale)

//设置图层的物标是否显示备注信息

//iLayerPos:图层的pos

//bShowTextOrNot:是否显示，true=显示，false=不显示

//iStartShowTextScale:开始显示备注的比例尺，比如1280000

//返回值：true=成功，false=失败

#### API\_SetLayerOneExpAttrByPos

API\_SetLayerOneExpAttrByPos(iLayerPos, iExpAttrPos, strExpAttrValue)

//设置图层的一个扩展字段值

//iLayerPos:图层的pos

//iExpAttrPos:扩展字段的pos

//strExpAttrValue:扩展字段值

//返回值：true=成功，false=失败

#### API\_SetLayerShowMinMaxScale

API\_SetLayerShowMinMaxScale(iLayerPos, iMinScale, iMaxScale)

//设置图层显示的最小最大比例尺

//iLayerPos:图层的pos

//iMinScale:最小比例尺，例如20000

//iMaxScale:最大比例尺，例如10240000即（1:20000~1:10240000显示，包括边界）

//返回值：true=成功，false=失败

#### API\_SetLayerShowOrNot

API\_SetLayerShowOrNot(iLayerPos, bShowOrNot)

//设置图层是否显示

//iLayerPos:图层的pos

//bShowOrNot:是否显示，true=显示，false=不显示

//返回值：true=设置成功，false=设置失败

#### API\_SetLineObjStyleByPos

API\_SetLineObjStyleByPos(iLayerPos, iObjPos, bShowOwnStyle, objStyleInfo)

//设置线物标的个性样式

//iLayerPos:线物标所属的图层pos

//iObjPos:物标的pos

//bShowOwnStyle:是否启用个性样式

//objStyleInfo:个性样式对象，格式{线粗细,线颜色,透明度}，

//例如{borderWith:2,borderColor:"#FF0000",iOpacity:80},

#### API\_SetObjectAllExpAttrByPos

API\_SetObjectAllExpAttrByPos(iLayerPos, iObjPos, arrExpAttrValue)

//设置物标的所有扩展字段

//iLayerPos:图层的pos

//iObjPos:物标的pos

//arrExpAttrValue:扩展字段数组，null=清空扩展字段

//返回值：true=成功，false=失败

#### API\_SetObjectInfoByPos

API\_SetObjectInfoByPos(iLayerPos, iObjPos, objId, name, showText, layerStylePos, arrCurGeoPo, arrExpAttrValue)

//根据索引设置物标的信息

//iLayerPos:物标所在的图层索引Pos

//iObjPos:物标的索引Pos

//objId:物标的id，null=不设置

//name:物标的名称，null=不设置

//showText:物标在海图上显示的内容，null=不设置

//layerStylePos:物标所使用的图层样式ID，null=不设置

//arrCurGeoPo:物标的坐标，null=不设置，坐标数组，元素格式{1210000000,310000000}

//arrCurExpAttr:物标的扩展字段，null=不设置,数组

//返回值：true=成功，false=失败

#### API\_SetObjectOneExpAttrByPos

API\_SetObjectOneExpAttrByPos(iLayerPos, iObjPos, iExpAttrPos, strExpAttrValue)

//设置物标的一个扩展字段

//iLayerPos:图层的pos

//iObjPos:物标的pos

//iExpAttrPos:扩展字段的pos

//strExpAttrValue:扩展字段值

//返回值：true=成功，false=失败

#### API\_SetPointObjRotation(iLayerPos, iObjPos, fAngle)

//设置点物标的旋转角度，(正上北0度，顺时针递增)

//iLayerPos:所属图层索引

//iObjPos:物标在图层中的索引

//fAngle:旋转角度

#### API\_SetPointObjStyleByPos

API\_SetPointObjStyleByPos(iLayerPos, iObjPos, bShowOwnStyle, objStyleInfo)

//设置点物标的个性样式

//iLayerPos:点物标所属的图层pos

//iObjPos:物标的pos

//bShowOwnStyle:是否启用个性样式

//objStyleInfo:个性样式对象，格式{图片地址,宽度,高度}，

//例如{strImgSrc:"img/img.png",iImgWidth:20,iImgHeight:20},

### 5.3 台风管理功能接口

#### API\_AddOneTyphoonTrack

API\_AddOneTyphoonTrack(iTyphoonPos, objTrackInfo, arrPredictTracks)

//给台风添加一个轨迹

//iTyphoonPos:台风的pos

//objTrackInfo:轨迹信息结构体

//arrPredictTracks:预测轨迹结构体数组

//返回值：轨迹点的pos,-1=添加失败

#### API\_AddTyphoon

API\_AddTyphoon(objTyphoonInfo, arrExpAttrValue)

//添加一个台风

//objTyphoonInfo:台风信息结构体

//返回台风的pos

#### API\_GetTyphoonInfoByPos

API\_GetTyphoonInfoByPos(iTyphoonPos)

//根据台风的pos获取台风信息

//iTyphoonPos:台风的pos

//返回值：台风的数据结构体，null=没有该台风

#### API\_GetTyphoonPosById

API\_GetTyphoonPosById(typhoonId)

//根据id获取台风pos

//typhoonId:台风的id

//返回值：台风的pos,-1=没有改台风

#### API\_GetTyphoonTrackInfoByPos

API\_GetTyphoonTrackInfoByPos(typhoonPos, iTruePos, iPredictPos)

//根据pos获取台风轨迹

//typhoonPos:台风的pos

//iTruePos:真实轨迹点pos

//iPredictPos:预测轨迹点pos,假如是获取真实轨迹点，则个值要传入null

//返回值：台风轨迹点结构体（真实轨迹点/预测轨迹点信息结构体）

#### ReDrawTyphoon

ReDrawTyphoon()

//重绘台风

#### API\_SetTyphoonShowOrNotByPos

API\_SetTyphoonShowOrNotByPos(iTyphoonPos, bShowOrNot)

//设置台风是否显示

//iTyphoonPos:台风的pos

//bShowOrNot:是否显示

//返回值：true=成功，false=失败

#### API\_SetTyphoonStyleColor

API\_SetTyphoonStyleColor(objTyphoonStyle)

//设置台风显示样式

//objTyphoonStyle:台风样式结构体

//返回值：true=成功，false=失败

#### API\_SetTyphoonToMapViewCenterByPos

API\_SetTyphoonToMapViewCenterByPos(iTyphoonPos)

//设置台风海图居中

//iTyphoonPos:台风的pos

//返回值：true=成功，false=失败

### 5.4 计算功能接口

#### API\_GetAreaOfGeoRegion

API\_GetAreaOfGeoRegion(arrGeoPo)

//计算多边形面积(单位平方米)

//arrGeoPo:多边形顶点坐标数组，例如{(x:1210000000,y:310000000),(x,y)...}

//返回值：多边形面积(平方米)

#### API\_GetDegreesBetwTwoPoint

API\_GetDegreesBetwTwoPoint(iGeoPoX1, iGeoPoY1, iGeoPoX2, iGeoPoY2)

//计算两点的角度，返回值单位度

//iGeoPoX1：第一个点的经度，格式：1210000000

//iGeoPoY1：第一个点的纬度，格式：310000000

//iGeoPoX2：第二个点的经度，格式：1220000000

//iGeoPoY2：第二个点的纬度，格式：320000000

//返回值：角度(度)，正北方向为0度，顺时针旋转

#### API\_GetDistBetwTwoPoint

API\_GetDistBetwTwoPoint(iGeoPoX1, iGeoPoY1, iGeoPoX2, iGeoPoY2)

//计算两点的距离，返回值单位km

//iGeoPoX1：第一个点的经度，格式：1210000000

//iGeoPoY1：第一个点的纬度，格式：310000000

//iGeoPoX2：第二个点的经度，格式：1220000000

//iGeoPoY2：第二个点的纬度，格式：320000000

//返回值：距离(km)

#### API\_GetLonLatPoByPlanePo

API\_GetLonLatPoByPlanePo(iPlanePoX, iPlanePoY)

//坐标转换:大地坐标转换经纬度坐标

//iPlanePoX:平面坐标X，单位（米），例如：89996

//iPlanePoY:平面坐标Y，单位（米），例如：89996

//返回值：经纬度坐标，单位（度），例如{x:121.4,y:31.9}

#### API\_GetLonLatPoByScrnPo

API\_GetLonLatPoByScrnPo(iScrnPoX, iScrnPoY, scale)

//坐标转换:屏幕坐标转换经纬度坐标

//iScrnPoX:屏幕坐标（int）

//iScrnPoY:屏幕坐标(int)

//scale:比例尺，假如是当前比例尺，可以传null

//返回值:经纬度坐标（度），格式例如：{x:121.02,y:31.2}

#### API\_GetPlanePoByLonLatPo

API\_GetPlanePoByLonLatPo(lon, lat)

//坐标转换:经纬度坐标转换大地坐标

//lon:经度（度），例如：121.21

//lat:纬度（度），例如：31.4

//返回值：平面坐标，单位(米)，例如{x：33332,y:34333}

#### API\_GetScrnPoByLonLatPo

API\_GetScrnPoByLonLatPo(lon, lat, scale)

//坐标转换:经纬度坐标转换屏幕坐标

//lon:经度(度)，例如：121.22

//lat:纬度(度)，例如：31.09

//scale:比例尺，假如是当前比例尺，可以传null

//返回值:屏幕坐标（px），格式例如：{x:200,y:100}

### 5.5 海图管理接口

#### API\_CutMapViewToImg

API\_CutMapViewToImg(bDrawCenterPo, bDrawScale, imgDiv, showImgSize, strDrawText)

//进行截屏（截屏时间需要2秒左右，截屏完之后会把图片显示在imgDiv中）

//bDrawCenterPo:是否绘制中心点,true=显示，false=不显示

//bDrawScale是否绘制比例尺,true=显示，false=不显示

//imgDiv:用于显示截屏后的img标签id

//showImgSize:显示截屏后的img标签的尺寸，格式{w:800,h:500};

//strDrawText:备注内容，null=不加备注

#### API\_GetCurLngLatRange

API\_GetCurLngLatRange()

//获取海图视图区域经纬度范围

//返回值:格式例如：{minLon:121.01,,maxLon:122.2,minLat:30.5,maxLat:32}

#### API\_GetCurMapCenterLonLatPo

API\_GetCurMapCenterLonLatPo()

//获取当前屏幕中心点

//返回值：当前屏幕中心点经纬度(度)，格式{x:121.21,y:31.45}

#### API\_GetCurMapScale

API\_GetCurMapScale()

//获取当前比例尺

//返回值：当前比例尺，例如1280000

#### API\_GetCurMeasureArrInfo

API\_GetCurMeasureArrInfo()

//获取测距的点信息数组

//返回值：数组对象，数组元素格式{po,curDis,allMeasureDist,curDegrees}

//po:点经纬度坐标，例如{x:1210000000,y:310000000}

//curDis:当前点与上一个点的距离(km)

//allMeasureDist:累加距离(km),即第一个点到当前点的距离

//curDegrees:当前点与上一个点的角度(度)

#### API\_GetPageSize

API\_GetPageSize()

//获取海图窗口的大小

//返回值：窗口的大小（px）,例如{x:1200,y:800};

#### API\_GotoMyLactionByGeoPo

API\_GotoMyLactionByGeoPo(iGeoX, iGeoY)

//定位到我的位置

//iGeoX：经度位置，例如1210000000

//iGeoY：纬度位置，例如310000000

#### API\_InitYimaMap

API\_InitYimaMap(obj)

//初始化海图

//obj:海图初始化值对象，格式例如：{div:map,type:android}，

//其中div值即是显示海图容器的div的id,type是sdk使用的平台，pc=电脑，android=手机

//返回值：true=成功，false=失败

#### API\_MoveMapViewByMoveSize

API\_MoveMapViewByMoveSize(iSizeScrnX, iSizeScrnY, bGetAutoMove)

//移动海图一段距离

//iSizeScrnX:移动X轴上的距离，正数=向左移动，负数=向右移动

//iSizeScrnY:移动Y轴上的距离，正数=向上移动，负数=向下移动

//bGetAutoMove:是否启用惯性效果，惯性效果，true=启用，false=不启用

#### API\_SetMapBackGroundStyle

API\_SetMapBackGroundStyle(strBackGroundImgPath, strGroundColor)

//设置海图视图的背景

//strBackGroundImgPath:海图背景图片的路径

//strGroundColor:海图背景颜色，例如"#FF0000"

#### API\_SetMapImagesUrl (strImgUrl)

API\_SetMapImagesUrl (strImgUrl)

//设置海图显示的图片路径

//strImgUrl:设置海图图片的地址

//例如：切出来的图片具体路径如："http://www.yimasoft.net/YimaChartImages/640000/0\_0/640000\_\_0\_0\_0\_0.png"

//那么strImgUrl为<http://www.yimasoft.net/YimaChartImages/>

#### API\_SetMapMinMaxScale

API\_SetMapMinMaxScale(iMinValue,iMaxValue)

//设置海图的最大最小比例尺(比例尺会以最大比例尺数开始倍增)

//这个接口必须在海图初始化前调用，即放在API\_InitYimaMap接口前

//iMinValue：最大比例，例如5000

//iMaxValue：最小比例尺，比例如20480000

#### API\_SetMapViewCenter

API\_SetMapViewCenter(lon,lat,scale)

//设置海图中心信息，设置之后会自动刷新海图

//lon:海图中心点经度，例如"121.21"，假如不设置可以传入null

//lat:海图中心点纬度，例如"31.12"，假如不设置可以传入null

//scale:海图的比例尺，例如"320000"，假如不设置可以传入null

#### API\_SetMousePoInfoDivPosition

API\_SetMousePoInfoDivPosition(bShowOrNot, iDivToRightLen, iDivToBottomLen)

//设置显示鼠标位置div位置

//bShowOrNot是否显示

//iDivToRightLen div显示离右边距离

//iDivToBottomLen div显示离底部距离

//返回值：true=成功，false=失败

#### API\_SetScaleInfoDivPosition

API\_SetScaleInfoDivPosition(bShowOrNot, iDivToLeftLen, iDivToBottomLen)

//设置显示比例尺div位置

//bShowOrNot是否显示

//iDivToLeftLen //比例尺显示离左边距离

//iDivToBottomLen //比例尺显示离底部距离

//返回值：true=成功，false=失败

#### API\_SetScaleLenInfoPosition

API\_SetScaleLenInfoPosition(bShowOrNot, iDivToLeftLen, iDivToBottomLen)

//设置显示比例尺条码

//bShowOrNot是否显示

//iDivToLeftLen //比例尺显示离左边距离

//iDivToBottomLen //比例尺显示离底部距离

//返回值：true=成功，false=失败

#### API\_SetShowToolBarOrNot

API\_SetShowToolBarOrNot(bShowOrNot, iDivToRightLen, iDivToTopLen)

//设置是否显示工具条

//bShowOrNot是否显示

//iDivToRightLen div显示离右边距离

//iDivToTopLen div显示离顶部距离

//返回值：true=成功，false=失败

#### API\_SetZoomBtnPosition

API\_SetZoomBtnPosition(bShowOrNot, iDivToRightLen, iDivToTopLen)

//设置显示缩放按钮位置

//bShowOrNot是否显示

//iDivToRightLen div显示离右边距离

//iDivToTopLen div显示离顶部距离

//返回值：true=成功，false=失败

#### API\_ReSizeMapView

API\_ReSizeMapView()

//海图窗口大小变化时，调用该接口进行重初始化窗口

//海图视图上的标签进行重新部署位置

#### API\_ZoomMap

API\_ZoomMap(zoomType, zoomScrnPo)

//海图的缩放

//zoomType:缩放类型，1=放大，-1=缩小

//zoomScrnPo:缩放的中心点，null=以海图中心点进行缩放，格式例如{x:200,y:100}

### 5.6 其它接口

#### API\_GetMousePosition

API\_GetMousePosition(event, obj)

//获取鼠标相对于obj对象的位置

//event:鼠标事件

//obj:物标对象

//返回值:{x:200,y:100}

#### API\_GetMyUesrId

API\_GetMyUesrId()

//获取用户的id

//返回值：用户的id,是一个8位整数

#### API\_LonLatToString

API\_LonLatToString(lonLat, bIsLon)

//小数表示的经纬度转换成分秒制表示的经纬度

//lonLat：经纬度值，例如121.023

//bIsLon:是否经度，true=经度，false=纬度

//返回值：例如 E 121°31.32'

#### API\_SetMyLicenceKey

API\_SetMyLicenceKey(strMyLicenceKey)

//设置LicenceKey，设置成功后才能正常使用,

//从API\_GetMyUesrId得到用户的id，然后根据用户id去dll组件获取许可strMyLicenceKey

//strMyLicenceKey：key值

//返回值：true=注册成功，false注册失败

### 5.7 特殊接口

特殊接口为SDK组件内部引用的外部方法，即这些方法在SDK外声明即可。

#### ReturnCurMeasureDist

function ReturnCurMeasureDist(CurDis, allMeasureDist, CurDegrees)

//返回测距时的距离，测距时候SKD会把测距结果返回到该方法

//测距状态时，鼠标移动即激发该方法

//CurDis：当前段距离（km)

//allMeasureDist：累加距离（km）

//CurDegrees：当前方位（度）

#### ReturnCurMeasurePoInfoByMouseDown

function ReturnCurMeasurePoInfoByMouseDown(curGeoPo, curDis, allMeasureDist, CurDegrees)

//测距时候，鼠标点击激发该方法

//curGeoPo:鼠标当前点击的经纬度坐标，格式{1210000000,310000000}

//curDis:当前点击点与上一个点的距离（km)

//allMeasureDist:累加的距离（km）

//CurDegrees:当前点与上一个点的角度（度）

#### ReturnCurMeasureAreaSize

function ReturnCurMeasureAreaSize(curAreaSize)

//返回测面积时的面积大小

//CurDis：当前段距离（平方米)

#### ReturnDrawDynamicObjNewInfo

function ReturnDrawDynamicObjNewInfo(objDynamicInfo)

//动态绘制物标时，选中点之后返回的坐标

//绘制标绘时候，假如是点、线、面类型，那么在鼠标点击时候会激发该方法；假如是绘制矩形、圆的时候，选择第一个点会激发，之后鼠标移动也会激发。

//objDynamicInfo:动态标绘的信息结构体

//objDynamicInfo:格式例如{type:1,po:{x:1210000000,y:10000000},w:20,h:80,r:10};

//type是标绘的类型：1=点、2=线、3=面、4=矩形、5=圆

//po是坐标点，例如{x:1210000000,y:10000000}

//w:是矩形的宽度，单位km，这个属性只有绘制矩形才有

//h:是矩形的高度，单位km，这个属性只有绘制矩形才有

//r:圆的半径，单位km,这个属性只有绘制圆才有

#### ReturnSelectObjByMouseLeftDown

function ReturnSelectObjByMouseLeftDown(objInfo)

//鼠标左击时查询到的对象信息

//鼠标左击时激发该方法

//objInfo:对象的信息结构体，null=没有查询到对象

//objInfo格式例如：

//选中船舶信息{objType:1,po:{x:200,y:300},bSelPlayTrackShip:false}

//选中物标信息{objType:2,po:{x:200,y:300},layerId:1,objId:2}

//objType:对象信息类型，1=船舶信息,2=物标信息

//po:当前鼠标所在的相对海图的屏幕位置，例如{x:200,y:300}

// bSelPlayTrackShip：选择的船舶信息是否为回放轨迹船舶，true=是，false=否，因为当前船舶与轨迹回放船舶的管理接口不一样，所以会有这个属性来判断要哪些接口来获取船舶信息。

// layerId:选中图层的id

// objId:选中物标的id

#### ReturnSelectObjByMouseMove

function ReturnSelectObjByMouseMove(objInfo)

//鼠标移动时，选中对象的信息

//鼠标移动时激发该方法

//objInfo:对象的信息结构体，null=没有查询到对象

//objInfo格式例如：

//选中船舶信息{objType:1,po:{x:200,y:300},bSelPlayTrackShip:false,iTrackPos:1}

//选中物标信息{objType:2,po:{x:200,y:300},layerId:1,objId:2}

//选中台风信息{objType:2,po:{x:200,y:300},typhoonId:1,iTruePos:2,iPredictPos:3}

//objType:对象信息类型，1=船舶信息,2=物标信息

//po:当前鼠标所在的相对海图的屏幕位置，例如{x:200,y:300}

// bSelPlayTrackShip：选择的船舶信息是否为回放轨迹船舶，true=是，false=否，因为当前船舶与轨迹回放船舶的管理接口不一样，所以会有这个属性来判断要哪些接口来获取船舶信息。

//iTrackPos:是否选中轨迹点信息，null=不是，其它值=选中轨迹点的pos，该值只有轨迹回放的时候才有

// layerId:选中图层的id

// objId:选中物标的id

//typhoonId:选中台风的id

//iTruePos:选中台风的真实轨迹点信息的pos

//iPredictPos:选中预测轨迹点的pos,null=不是选中预测轨迹点，而是选中真实轨迹点iTruePos，其它值=选中预测轨迹点

#### ReturnOnMouseRightDown

function ReturnOnMouseRightDown(scrnPo)

//鼠标右键事件

//scrnPo:鼠标在海图上的屏幕位置

## 六、SDK对象数据结构

### 船舶相关的对象数据结构

#### ShipInfo船舶信息结构体

function ShipInfo() {

this.shipId; //船舶的id

this.shipMMSI; //mmsi

this.shipName; //船名名称

this.shipGeoPoX; //位置X，例如1210000000

this.shipGeoPoY; //位置Y，例如310000000

this.shipWidth; //宽度(数字)

this.shipLength; //长度(数字)

this.shipSpeed; //速度(数字)

this.shipCourse; //航向(数字)

this.shipTime; //时间(字符串)，格式:2015/6/6 12:5:18

this.iShipState; //状态(数字)

this.bOnlineOrNot; //是否在线(布尔)，true=在线，false=离线

this.bShowTrack; //是否显示轨迹(布尔)，该值在结构体中只读，设置无效

this. bFollow; //是否为跟踪船舶(布尔)，该值在结构体中只读，设置无效

}

#### ShipTrackPoint船舶轨迹点结构体

function ShipTrackPoint() {

this.trackGeoPoX = 0; //轨迹点位置X，例如1210000000

this.trackGeoPoY = 0; //轨迹点位置Y，例如310000000

this.trackCourse = 0; //船向(数字)

this.trackSpeed = 0; //速度(数字)

this.trackTime = ""; //时间(字符串)，格式:2015/6/6 12:5:18

}

#### ShipStateStyle船舶状态样式结构体

function ShipStateStyle()

{

this.arrSymbolPo; //船舶形状的坐标数组，

//格式{{ x: 0, y: -11 },{ x: -6, y: 10 },{ x: 6, y: 10 }}，三角形船舶，

//(0,0)是船舶中心点，Y轴的负方向是航向0度

this.borderSize; //线的粗细(数字)

this.fillColor; //填充颜色(字符串)，例如"#ffff66"

this.borderColor; //边颜色(字符串) ，例如"#ffff66"

this.iOpacity; //透明度(数字)，0~100，100为不透明

this.minShowScale; //起始显示比例尺，

//从minShowScale比例尺开始绘制船舶，包括minShowScale比例尺

this.maxShowScale; //开始隐藏比例尺，

//从maxShowScale比例尺开始不绘制船舶，包括maxShowScale比例尺

}

船舶形状的坐标数组可以自己定义，只要就可以实现多种形状的船舶，比如：

//三边形船舶形状

var arrThreeSymbolPo = [];

arrThreeSymbolPo.push({ x: 0, y: -11 });

arrThreeSymbolPo.push({ x: -6, y: 10 });

arrThreeSymbolPo.push({ x: 6, y: 10 });

//五变形船舶形状

var arrFiveSymbolPo = [];

arrFiveSymbolPo.push({ x: 0, y: -15 });

arrFiveSymbolPo.push({ x: 5, y: -1 });

arrFiveSymbolPo.push({ x: 5, y: 15 });

arrFiveSymbolPo.push({ x: -5, y: 15 });

arrFiveSymbolPo.push({ x: -5, y: -1 });

//六边形船舶形状

var arrSixSymbolPo = [];

arrSixSymbolPo.push({ x: -8, y: 25 });

arrSixSymbolPo.push({ x: 8, y: 25 });

arrSixSymbolPo.push({ x: 8, y: -15 });

arrSixSymbolPo.push({ x: 2, y: -25 });

arrSixSymbolPo.push({ x: -2, y: -25 });

arrSixSymbolPo.push({ x: -8, y: -15 });

### 物标相关的对象数据结构体

#### LayerInfo图层信息对象结构体

function LayerInfo() {

this.type; //图层类型，1=点图层，2=线图层，3=面图层

this.id; //图层的id(数字)

this.name; //图层名称(字符串)

this.bShow; //是否显示（布尔），true=显示，false=不显示

this.minShowScale; //显示最小比例尺(数字)，例如20000

this.maxShowScale; //显示最大比例尺(数字)，例如640000

this.bShowTextOrNot; //是否显示备注（布尔），true=显示，false=不显示

this.iStartShowTextScale; //开始显示备注的比例尺(数字)，例如640000

}

#### ObjectInfo物标信息对象结构体

function ObjectInfo() {

this.id; //物标的id(数字)

this.objType; //物标类型(数字)：1=点、2=线、3=面、4=矩形、5=圆

this.name; //物标名称(字符串)

this.strShowText; //海图上显示的内容（字符串）

this.bShow; //是否显示，true=显示，false=不显示

this.rectW; //矩形的宽度(km) (数字)，只有矩形的物标才有

this.rectH; //矩形的高度(km) (数字)，只有矩形的物标才有

this.circleR; //圆半径(数字)，只有圆形的物标才有

this.layerStylePos; //使用图层样式的索引pos（数字），只有使用图层样式，该值才有效

this.bGetOwnStyle; //是否启用自己样式（布尔），true=自己样式，false=图层样式

this.rotationAngle = 0; //点物标的旋转角度

}

#### PointStyle点图层样式对象结构体

点符号样式有三种：图片、矢量符号、圆，其中:

当 **strImgSrc**有效且**bShowImg**=true时候才使用图片显示；

否则

**arrSymbolPo**有效时候，使用矢量符号表示；

**arrSymbolPo**==null,且**iCircleScrnR**>0时候就使用圆表示；

function PointStyle() {

this.strImgSrc; //图片地址，例如"img/point.png"

this.iImgWidth; //图片宽度(数字)

this.iImgHeight; //图片高度(数字)

this.textColor; //物标名称字体颜色(字符串)，例如"#000000"

this.fontSize; //物标名称字体(字符串)，格式例如"12px 宋体"

this.bShowImg; //是否用图片显示

this.bShowText; //是否显示信息(布尔)，true=显示，false=不显示

this.iTextOpacity; //文本透明度(数字)，0~100，100为不透明

this.arrSymbolPo; //矢量符号坐标

this.iCircleScrnR; //使用圆表示时候，圆的半径

this.iCheckDrawMinNearOtherLen = 5;//过滤绘制点物标的判距(防止点物标绘制过密)

}

#### LineStyle线图层样式对象结构体

function LineStyle() {

this.iOpacity; //填充透明度(数字)，0~100，100为不透明

this.borderWith; //线粗细(数字)

this.borderColor; //线颜色(字符串)，例如"#000000"

this.textColor; //物标名称字体颜色(字符串)，例如"#000000"

this.fontSize; //物标名称字体(字符串)，格式例如"12px 宋体"

this.bShowText; //是否显示信息(布尔)，true=显示，false=不显示

this.iTextOpacity; //文本透明度(数字)，0~100，100为不透明

}

#### FaceStyle面图层样式对象结构体

function FaceStyle() {

this.bFilled; //是否填充(布尔)，true=填充，false=不填充

this.fillColor; //填充颜色(字符串)，例如"#000000"

this.iOpacity; //填充透明度(数字)，0~100，100为不透明

this.borderWith; //线粗细(数字)

this.borderColor; //线颜色(字符串)，例如"#000000"

this.textColor; //物标名称字体颜色(字符串)，例如"#000000"

this.fontSize; //物标名称字体(字符串)，格式例如"12px 宋体"

this.bShowText; //是否显示信息(布尔)，true=显示，false=不显示

this.iTextOpacity; //文本透明度(数字)，0~100，100为不透明

}

### 台风相关的对象数据结构体

#### TyphoonInfo台风结构体

function TyphoonInfo() {

this.id; //台风id(数字)

this.name; //台风名称(字符串)

this.startTime; //开始时间(字符串)，格式：2015/6/6 12:9:23

this.endTime; //结束时间

this.bShow; //是否显示

this.iCurShowTrackPos; //当前选中的真实轨迹点pos（数字）

}

#### TyphoonTrack台风真实轨迹点对象结构体

function TyphoonTrack() {

this.time; //时间(字符串)，格式例如：2015/6/6 12:9:30

this.po; //坐标(point),格式例如{x:1210000000,y:310000000}

this.windPower; //风力

this.windSpeed; //风速

this.airPressure; //气压(数字)

this.moveDirection; //移向(字符串)

this.moveSpeed ; //移速（数字）

this.sevenRadius; //7级半径(km)（数字）

this.tenRadius; //10级半径(km)（数字）

}

#### TyphoonPredictTracks台风预测轨迹点对象结构体

function TyphoonPredictTracks() {

this.time; //时间(字符串)，格式例如：2015/6/6 12:9:30

this.po; //坐标(point),格式例如{x:1210000000,y:310000000}

this.windPower; //风力（数字）

this.windSpeed; //风速（数字）

this.airPressure; //气压（数字）

this.strReportStation; //预报台(字符串)

}

#### TyphoonStyle台风样式信息结构体

function TyphoonStyle

{

this.strTropicalDepressionColor; //热带低压颜色(字符串)，例如"#FF000"

this.strTropicalStormColor; //热带风暴颜色(字符串)，例如"#FF000"

this.strSevereTropicalStormColor; //强热带风暴颜色(字符串)，例如"#FF000"

this.strTyphoonColor; //台风颜色(字符串)，例如"#FF000"

this.strSevereTyphoonColor; //强台风颜色(字符串)，例如"#FF000"

this.strSuperTyphoonColor; //超强台风颜色(字符串)，例如"#FF000"

this.sevenCircleColor; //7级圆样式颜色(字符串)，例如"#FF000"

this.tenCircleColor; //10级圆样式颜色(字符串)，例如"#FF000"

}