iSignature HTML5

V2.0.0.390

产 品 技 术 白 皮 书

江西金格科技股份有限公司

地址： 江西省南昌市高新区火炬大街 579 号绿悦科技大厦 15 楼
邮编： 330096
网址： http://www.kinggrid.com
电话： 0791-82221588
服务： *400-6776-800*

目录

1 iSignature HTML5

1.1. 概述

iSignature HTML5 签章*(*简称 *H5* 签章*)*是适应互联网模式，面向大众的 *WEB* 电子签章，无需加载任何客户端组件实现在 *WEB* 端完成电子签章操作，实现移动和 *PC* 端统一签章接口，在吸收旧版本的经验的基础上，提供更简单，易用的开发接口。

1.2. 集成步骤

H5 签章支持 *2* 种模式：客户端模式和签章云模式；

客户端模式：*H5* 签章调用客户端读取密钥盘中的电子签章和证书完成电子签章。

签章云模式：*H5* 签章调用签章云服务完成电子签章。

移动端：*H5* 电子签章的移动端显示方式。 *mobile.html*

PC 端：*H5* 电子签章的电脑端的显示方式。*server.html*、*client.html*

根据不同的需求：请先安装对应的客户端或者签章云服务。

部件	安装要求	说明
iSignature HTML5 组件	IE8 及以上,Chrome,Firefox	集成参考 1.2.2 H5 签章集成。
HTML5 客户端模式	Windows 7/8/10	安装对应的<iSignature 签章软件>。
HTML5 签章云模式	JDK1.6 及以上	部署 iSignatureHTML5.war 到 tomcat，并修改 \WEB-INF\classes \system.txt 内的签章服务器地址。

1.2.1. H5 签章集成

除了 *H5* 签章核心的 *js* 和 *css* 之外，需要根据手机端和 *PC* 端不用的场景引入对应的 *js* 和 *css* 文件。

***H5*签章核心：**

```
<meta http-equiv="X-UA-Compatible" content="IE=edge,chrome=1">
<link rel="stylesheet"
href="../kinggrid/dialog/artDialog/ui-dialog.css">
<link rel="stylesheet"
href="../kinggrid/core/kinggrid.plus.css">
<script type="text/javascript"
src="../kinggrid/dialog/artDialog/dialog.js"></script>

<script type="text/javascript"
src="../kinggrid/core/kinggrid.min.js"></script>
<script type="text/javascript"
src="../kinggrid/core/kinggrid.plus.min.js"></script>

<script type="text/javascript"
src="../kinggrid/signature.min.js"></script>
```

PC端功能：

```
<script type="text/javascript"
src="kinggrid/signature.pc.min.js"></script>
```

移动端：

```
<link rel="stylesheet"
href="../kinggrid/core/kinggrid.plus.mobile.css">

<script type="text/javascript"
src="../kinggrid/signature.mobile.min.js"></script>
```

注意：依赖jquery ，最低要求版本为1.7+,如果不需要兼容IE8，可以使用jQuery 3

2 模块说明

2.1 Signature 全局类

静态方法：

属性方法	描述
init(Ojbect options)	设置自定义参数项。
loadSignatures(Ojbect signatures)	加载多个电子签章。
loadSignature(String signatureid , String signatureData)	加载一个电子签章。
verify()	验证所有已加载的电子签章，并显示电子签章
show ()	显示所有已加载的电子签章
hide ()	隐藏所有已加载的电子签章
create(Ojbect options)	创建一个盖章对象。
list	获取已加载的所有电子签章对象。
showAndHideBySignatureId	通过 *signatureid*，显示、隐藏指定的电子签章
resetSignaturePos	重新定位盖章显示的位置
clearRPW()	清除浏览器记住的密码。
verifyData()	签章自定义验证方法

2.1.1 init

初始化电子签章配置项。默认配置为客户端读取硬件密钥盘模式。

参数列表：

参数名	类型	描述
options	*Object*	可配置属性参考：*options* 属性。
~~*callback*~~	~~*function*~~	~~*Init* 成功回调函数~~

参数属性说明：

1) options 属性：

属性名	类型	默认值·	描述
documentid	*String*		文档 *ID*，必须设置
documentname	*String*		文档名称
moveable	*Boolean*	*true*	签章是否可以移动
signable	*Boolean*	*true*	是否含有数字签名功能
sealType	*String*	*client* 可选	设置印章读取来源

		【*server/client*】	
certType	*String*	*client* 可选 【*server/client*】	设置证书读取来源
clientUrl	*String*	*http:///127.0.0.1:9581*	金格组件服务地址
serverUrl	*String*		*1*、 *sealType* 和 *certType* 设置为 *server* 需要。 *2*、 需要进行 *H5* 签章数据库交互操作时。 *3*、 *H5* 兼容 *H2* 数据时。
clientConfig	*Object*		设置客户端配置参数，参考“*clientConfig* 属性说明”
imgtag	*Int*	*0*	设置签章类型（*0*：无; *1*:公章; *2*:私章;）
delCallBack	*function*		删除签章回调函数
showSealsDlg	*Boolean/String*	*True*	是否显示选择签章界面。 *false*：不弹窗 *‘auto’*：自动，如果符合条件的只有一个印章则不弹签章选择框。如果有多个就弹签章选择框【*V1.0.0.338*】
password	*String*		签章密码设置。
signdate	*Object*	*{ ischeck:true, fontFormat:'yyyy-M-d', fontFamily:'宋体', fontSize:12, fontColor:' #000000', position:'居中'}*	设置日期在签章的位置和字体格式，详情请参照章节 *2.4* 日期设置
showNoPW	*Boolean*	*false*	设置选择签章对话框去掉密码输入显示
timestamp	*Boolean*	*false*	设置签章获取时间。 *sealType:server* 获取签章服务器时间， *sealType:client* 客户端获取时间戳时间信息，时间戳设置在签章服务器。
verifySignatureInfo	*Object*	是否被修改 *ismodify* 被篡改数据 *modifiedItems* 签章数据 *signatureData*	验证执行回调函数，执行该方法的同时，页面点击签章图片将不进行验证操作。 【*V1.0.0.284*】
appcode	*String*	授权码	客户端访问服务器授权控制 【*V1.0.0.288*】

isResponseErr	*Boolean*	*True*：不需要弹框 *False*：弹框（默认）	组件未安装时是否弹框
icon_move	*Boolean*	*True*（*PC* 端有效）	移动签章按钮隐藏显示，默认显示
icon_remove	*Boolean*	*True*（*PC* 端的撤销小图标、移动端撤销按钮）	撤销签章按钮隐藏显示，默认显示
icon_sign	*Boolean*	*True*（*PC* 端的数字签名小图标、移动端的证书信息）	数字签名按钮隐藏显示，默认显示
icon_signverify	*Boolean*	*True*（*PC* 端有效）	签名验证按钮隐藏显示，默认显示
icon_sealinfo	*Boolean*	*True*（*PC* 端有效）	签章验证按钮隐藏显示，默认显示
keysn	*String*	*sealType:server* 时有效	初期化页面 *keysn*
usercode	*String*	*sealType:server* 时有效	同 *keysn* 功能一样
pw_timeout	*String*	*s1800*（半小时后失效）	签章密码保存失效时长，*s*：秒；*h:*小时；*d:*天
scaleImage	*Float*	*1.0*	签章缩放显示，默认 *100%*
cache_path	*String*	*sealType:server* 时有效	签章保存在缓存文件有效，设置缓存地址。
pw_enc_save	*Boolean*	*false*	密码是否加密保存，默认不加密。【*V1.0.0.318*】
pw_server_timeout	*Boolean*	*False* 注：该属性云端有效	密码保存时间长度是否从签章服务器读取【*V1.0.0.318*】
https	*String*	客户端为 *https* 服务使用 注：该属性 *client* 端有效	客户端 *https* 服务路径【*V1.0.0.320*】
imageAndDate	*boolean*	*false*	日期是否和图片组合。【*V1.0.0.324*】
sealTag	*String*	默认为空，获取图通印章。*'GM'*：代表国密印章，云端有效	是否获取国密印章。【*V1.0.0.328*】
errorCall	*function*	默认金格错误弹窗方式	用户自处理错误信息。错误信息详见 *2.5*【*V1.0.0.332*】
signSize	*int*	默认为 *5*，控制 *H2* 的签章数据显示数量	控制页面最多能显示的签章个数，默认最多显示 *5* 个。
okCall	*function*		初始化成功回调函数【*V1.0.0.336*】
verifyResType	*String*	*click*（移动端有效）	*verifyResType*：*'click touchend'*，在特殊情况下（如

			IOS、webview）环境下第一次点击不响应的问题【V1.0.0.342】
quickClose	boolean	True(移动端有效)	解决签章撤销时，撤销签章闪退问题。【V1.0.0.342】
icon_move_donot	boolean	True（PC 端有效）	允许或者禁止移动按钮隐藏显示，缺省显示【V1.0.0.362】只有当前用户加盖的印章才可以更改该状态。客户端需要插 key。
showProtectedBtn	boolean	默认为 false	设置是否显示保护项按钮【V1.0.0.362】
showSignExpirePrompt	boolean	默认为 false	设置是否显示签章过期提示【V1.0.0.358】
fjrsIsphone	String	默认为空	设置是否福建瑞术扫码盖章（1：福建瑞术手机盾，其他：非福建瑞术手机盾）【V2.0.0.378】
clientCode	String	默认为空	客户端绿色版 code 码。通用客户端该参数省略。【V2.0.0.384】

clientConfig 属性说明：

属性名	类型	默认值·	描述
SOFTTYPE	String	“1”,可选【0/1】	客户端类型：0：标准版；1：网络版/CA 版/平台版
LANGUAGE	String	空	zh_CN 中文 en_US 英文 zh_TW 中文繁体。【V1.0.0.332】

代码：

移动端：

```
Signature.init({//初始化属性
    keysn:'1A10374905170616',
    delCallBack: delCB,
    timestamp: true,
    certType : 'server',//设置证书在签章服务器
    sealType : 'server',//设置印章从签章服务器取
    serverUrl : 'http://127.0.0.1:8080/iSignatureH5',
    documentid:'KG2016093001',//设置文档ID
    documentname:'测试文档KG2016093001',//设置文档名称
    pw_timeout:'s1800', //s：秒；h:小时；d:天
```

```
    scaleImage: 0.5 //签章图片的缩放比例
})
```

Server 端：

```
Signature.init({//初始化属性
    keysn : '1A10374905170616',
    delCallBack : delCB,
    password : '123456',
    signdate : {
        ischeck : true,
        fontFormat : 'yyyy-MM-dd hh:mm:ss',
        fontFamily : '楷体',
        fontSize : 12,
        fontColor : '#00CC00',
        position : '居右下(章内)'
    },
    timestamp : true,
    moveable : true,
    valid : false, //签章和证书有效期判断， 缺省不做判断
    icon_move : true, //移动签章按钮隐藏显示，缺省显示
    icon_remove : true, //撤销签章按钮隐藏显示，缺省显示
    icon_sign : true, //数字签名按钮隐藏显示，缺省显示
    icon_signverify : true, //签名验证按钮隐藏显示，缺省显示
    icon_sealinfo : true, //签章验证按钮隐藏显示，缺省显示
    icon_move_donot: true,//允许或者禁止按钮隐藏显示，缺省隐藏
    certType : 'server',//设置证书在签章服务器
    sealType : 'server',//设置印章从签章服务器取
    serverUrl : 'http://192.168.0.140:8080/iSignatureH5',//
    documentid : 'KG2016093001',//设置文档ID
    documentname : '测试文档KG2016093001',//设置文档名称
    pw_timeout : 's1800'//s：秒；h:小时；d:天
})
```

***Client* 端：**

```
Signature.init({//初始化属性
    clientConfig:{//初始化客户端参数
        'LANGUAGE':'zh_TW',//zh_CN 中文 en_US 英文 zh_TW 中文繁体
        'SOFTTYPE':'0'//0为：标准版， 1：网络版
    },
    delCallBack: delCB,
    valid : false,    //签章和证书有效期判断， 缺省不做判断
    icon_move : true, //移动签章按钮隐藏显示，缺省显示
    icon_remove : true, //撤销签章按钮隐藏显示，缺省显示
    icon_sign : true, //数字签名按钮隐藏显示，缺省显示
    icon_signverify : true, //签名验证按钮隐藏显示，缺省显示
    icon_sealinfo : true, //签章验证按钮隐藏显示，缺省显示
```

```
    certType : 'client',//设置证书在签章服务器
    sealType : 'client',//设置印章从签章服务器取
    documentid:'KG2016093001',//设置文档ID
    documentname:'测试文档KG2016093001',//设置文档名称
    pw_timeout:'s1800' //s：秒；h:小时；d:天
})
```

2.1.2 loadSignatures

加载多个签章数据。

参数列表：

参数名	**类型**	**描述**
signatures	*Object* 或 *Array*	支持签章数据对象数组和签章 *ID* 和签章数据组成的键值对。
~~*callfunction*~~	~~*Function*~~	~~签章自己验证方法回调。~~
callback	*Function*	签章加载成功后的回调方法，可以不设置。【*V1.0.0.340*】

Signatures 参数说明：

属性名	**类型**	**默认值·**	**描述**
signatureid	*String*		签章 *ID*
signatureData	*String*		签章数据
extra	*Object*	扩展对象	
extra .icon_move	*Function*	移动端有效	移动端签章初始化 *load* 时，不予许该章移动。

代码：

```
Signature.loadSignatures([{
    signatureId: '1467962754911688946',
    signatureData: 'eyJhcHBuYW1…'
}])//Array类型

Signature.loadSignatures({
    '1467962754911688946':'eyJhcHBuYW1…',
    '1467962779895960965':'eyJhcHBuYW1…',
    '1467962882048690651':'eyJhcHBuYW1…'
})//Object 类型
```

2.1.3 loadSignature

加载单个签章数据。

参数列表：

参数名	类型	描述
signatureid	*String*	签章 *ID*。
signatureData	*String*	签章数据。
~~*callfunction*~~	~~*Function*~~	~~签章自己验证方法回调。~~
callback	*Function*	签章加载成功后的回调方法，可以不设置。【*V1.0.0.340*】

返回值：

（无）

代码：

```
Signature.loadSignature(
    '1467962754911688946','eyJhcHBuYW1…'
);
```

2.1.4 verify

验证所有已加载的电子签章。

参数列表：

（无）

返回值：

类型	描述
Array[Signature]	返回验证失效的电子签章列表

代码：

```
var invalidSignatureArray = Signature.verify();//返回无效签章
if (invalidSignatureArray.length > 0) {
    for (var i = 0; i < invalidSignatureArray.length; i++) {
        var signature = invalidSignatureArray[i];
        console.log(signature.modifiedItems);//获取篡改的保护项
    }
}
```

2.1.5 show

显示所有加载的电子签章。

参数列表：

（无）

返回值：

（无）

代码：

```
Signature.show();
```

2.1.6 hide

隐藏所有加载的电子签章。

参数列表：

（无）

返回值：

（无）

代码：

```
Signature.hide();
```

2.1.7 create

创建一个盖章对象

参数列表：

根据 *init* 方法设置 *sealType* 设置为‘*server*’，需要指定密钥盘的密钥盘序号或者用户编码。

属性名	类型	默认值	描述
keysn	*String*		根据密钥盘序号获取服务器的印章。
usercode	*String*		根据用户编码获取服务器的印章。

sealType 为‘*client*’时，参数无。

返回值：

signatureCreator

代码：

```
var signatureCreator = Signature.create();
var protectedItems = [ 'item1', 'item2', 'item3' ]//设置保护DOM的id，自动查找ID，自动获取保护DOM的kg-desc属性作为保护项描述，value属性为保护数据。
signatureCreator.run({
    protectedItems: protectedItems,
    position: ‘posid’,//设置盖章定位dom的ID
    autoCert:true,//数字签名
    okCall: function() {//点击确定后的回调方法
        console.log("'" + this.getSignatureid() + "':'" +
        this.getSignatureData() + "'");
    },
    cancelCall : function() {//点击取消后的回调方法
```

```
        console.log("取消！ ")
    }
});
```

2.1.8 list

对象属性，获取所有已经加载的签章对象。
代码：

```
var signature = Signature.list['signatureid'];
```

2.1.9 showAndHideBySignatureId

通过 *signatureid*，显示、隐藏指定的电子签章。
参数列表：

属性名	类型	默认值	描述
sid	*String*		指定签章的 *signatureid*
bShow	*Boolean*		*ture:* 显示指定签章 *false:* 隐藏指定签章

返回值：
（无）
代码：

```
Signature.showAndHideBySignatureId(
this.getSignatureid(), false);
```

2.1.10 resetSignaturePos

重新定位盖章显示的位置。
注意：该方法必须要放在 loadSignatures 方法前有效。
参数列表：

参数名	类型	描述
signatures	*Object* 或 *Array*	支持对已经保存的签章显示时重新定位。

返回值：
（无）
代码：

```
var posArray = new Array();
var mapPos = {};
mapPos.signatureid = jsonList[i]["signatureId"];
```

```
mapPos.extra = {
    position: 'pos0',
    scopePosition: 'kg-pos0',
    offsetX:50,
    offsetY:100
};
posArray.push(mapPos);
Signature.resetSignaturePos(posArray);
```

2.1.11 clearRPW

清除用户保存的密码。
demo 默认每次刷新页面执行。
返回值：
（无）
代码：

```
    function onunload_handler() {
        Signature.clearRPW();
    }
```

2.1.12 verifyData

用户自定义验证。【*V1.0.0.352*】
当用户自定义保护项时，需要调用该方法自定义验证保护项数据，保证验证的有效性。
参数列表：

类型	描述
Array	当前印章的保护项

返回值：

类型	描述
Array	当前印章的保护项待验证数据

代码：

```
Signature.verifyData = function(data){
    console.log(data);
    return data;
}
```

2.1.13 verifySignByList

验证指定签章，返回验证签章信息。【*V1.0.0.340*】

参数列表：

类型	描述
List	当前页面加载的签章 *list*

返回值：

类型	描述
Array[Signature]	返回待验证的签章的签章信息。【*V1.0.0.340*】

代码：

```
function verifySignatureByList() {
    var list = Signature.list;
    var list2 = [];
    var strarr = ['155627555338199417','155650753606110295'];
    for(var key in list){
        var signature = list[key];
        if(strarr.indexOf(key)>-1){
            list2.push(signature);
        }
    }
    var signatureData = Signature.verifySignByList(list2);//返回
签章信息
    for (var i = 0; i < signatureData.length; i++) {
        var signature = signatureData[i];
        console.log(signature);//获取修改的保护项
        console.log(signature.modifiedItems);//获取修改的保护项
    }
}
```

2.2 SignatureCreator 对象

获取密钥盘内的印章或者服务端的印章执行盖章操作对象。

方法列表：

属性方法名	描述
run(Object config)	执行盖章操作。
handWriteDlg(Object config, function callback)	执行手写签名操作。（*PC* 端手机端可用）
barCodeDlg(Object config, function callback)	二维条码签名操作（*PC* 端手机端可用）
scanBCDlg (Object config, function callback)	扫码签章签名操作（*PC* 端手机端可用）
runHW(Object params,Object config)	设置签章图片执行盖章操作。
fingerPrintsDevice(Object config, function callback)	执行指纹签名操作（*PC* 端）
handWriteDevice(handWriteDeviceObject	执行手写设备签名操作（*PC* 端）

config, function callback)	
getBase64Image(Signatureid, SignatureData,elemid,callback)	将签章数据转化为 base64 图片数据。
toBase64Img(seal)	解析签章对象成标准 base64 图片数据
loadSeals(function, options)	搜索签章加载方法
setSeals(data)	搜索签章时设置获取的印章信息

2.2.1 run

执行盖章操作

参数列表：

参数名	类型	描述
options	Object	可配置属性参考：options 属性。

参数属性说明

options 属性说明：

属性名	类型	默认值·	描述
protectedItems	Array		设置定位页面 DOM 的 id，自动查找 ID，自动获取保护 DOM 的 kg-desc 属性作为保护项描述，value 属性为保护数据。不设置，表示不保护数据，签章永远有效。 自定义保护项【V1.0.0.352】： [{field:'item1',desc:'保护项 1',value:'同'},{field:'item2',desc:'保护项 2',value:'金格'}]
protectedItemsType	String		1:用户自定义保护项【V1.0.0.352】
position	String		定位元素 ID 批量签章中有这个属性的时候可以不写 【V2.0.0.372】
offsetX	Int		印章 X 轴偏移量(与 offsetY 成对出现)
offsetY	Int		印章 Y 轴偏移量(与 offsetX 成对出现)
batchSignature	Object 属性： documentid， documentname ， position		批量签章接口 [{documentid: 'KG20180307', documentname: '测试文档 KG20180307'} {documentid: 'KG20180308', documentname: '测试文档 KG20180308'}]
scopePosition	String		定位盖章区域 dom 的 ID 或者 Jquery

			选择器对象。
autoCert	*Boolean*	*false*	是否做数字签名
okCall	*Function*		点击确定后的回调方法，*this* 为签章对象。调用 *getSignatureid* 获取签章 *ID*，调用 *getSignatureData* 获取签章数据，必须设置。
cancelCall	*Function*		点击取消后的回调方法。
changeOffsetXY	*Function*		点击确定后，获取选择签章的宽高，重新计算签章位置偏移量。
errorCall	*Function*		错误信息回调方法，默认为空，为空时默认走 *init* 中的 *errorCall* 回调
protectedItemsCheckedAll	*Boolean*	*true*	控制保护项数据全选还是全不选，不传时默认为全选。【*V2.0.0.374*】
moveableRange	*String*		印章可移动范围的 *html* 标签的 *id*【*V2.0.0.388*】

***okCall* 函数说明：**

返回值列表：

序号	参数名	类型	描述
1	*fn*	*Function* 不写该方法：不记录日志，不显示签章 *fn(false)*：只记录日志，不显示签章 *fn(true)*:记录日志，并显示签章	控制是否需要日志，是否显示签章
2	*documentid*	*String*	批量签章时，第二个参数为 *documentid*
2	*img*	*Object*	当为单个签章时，第二个参数为签章图片对象

***batchSignature* 签章参数说明：（云模式数字签名模式无效）**

属性名	类型	默认值·	描述
documentid	*String*		文档批量签章的 *documentid*
documentname	*String*		文档批量签章的 *documentname*
position	*String*		批量签章的显示位置。
protectedItems	*Array*		签章保护项，建议自定义，如果需要 *H5* 抓取数据，该数据必须在本页面存在【*V2.0.0.380*】

返回值：

（无）

代码：

```
signatureCreator.run({
```

```
    protectedItems:[ 'item1', 'item2', 'item3' , 'item4' ],
    position: $(that).attr('pos'),
    autoCert:true,
    changeOffsetXY:function(img){
        var ret = {offsetX:20,offsetY:img.height*(-1)/2};
        return ret;
    },
    okCall: function() {
        console.log("'" + this.getSignatureid() + "':'" +
this.getSignatureData() + "'");
    },
    cancelCall : function() {//点击取消后的回调方法
        console.log("取消！")
    }
});
```

2.2.2 handWriteDlg

执行手写签名操作。

参数列表：

参数名	类型	描述
config	*Object*	可配置属性参考：*config* 属性。
callback	*Function*	手写签名窗口点击 *OK* 后，执行成功回调函数

参数属性说明

config 属性说明：

属性名	类型	默认值·	描述
image_height	*String*	*5.00/3.00*	设置手写签章高度，不可为空，单位 *CM*
image_width	*String*	*6.70/7.00*	设置手写签章宽度，不可为空，单位 *CM*
canvas_width	*String*	*670*	自定义设置画板宽度，单位像素【*V2.0.0.380*】
canvas_height	*String*	*500*	自定义设置画板高度，单位像素【*V2.0.0.380*】

返回值：

参照 *2.2.5* 章节参数名 *params*

2.2.3 barCodeDlg

二维条码签名操作

参数列表：

参数名	类型	描述
config	*Object*	可配置属性参考：*config* 属性。
callback	*Function*	二维条码回调函数

参数属性说明

config 属性说明：

属性名	类型	默认值·	描述
image_height	*String*		设置二维码签章高度，不可为空，单位 *CM*
image_width	*String*		设置二维码签章宽度，不可为空，单位 *CM*
content	*String*		设置二维码内容

返回值：

参照 *2.2.5* 章节参数名 *params*

2.2.4 scanBCDlg

扫码签章签名操作。

参数列表：

参数名	类型	描述
config	*Object*	可配置属性参考：*config* 属性。
callback	*Function*	扫描枪签名回调函数

参数属性说明

config 属性说明：

属性名	类型	默认值·	描述
image_height	*String*		设置扫码签章高度，不可为空，单位 *CM*
image_width	*String*		设置扫码签章宽度，不可为空，单位 *CM*

返回值：

参照 *2.2.5* 章节参数名 *params*

2.2.5 runHW

设置签章图片执行盖章操作

参数列表：

参数名	类型	描述
params	*Object*	可配置属性参考：*params* 属性。
options	*Object*	可配置属性参考：*2.2.1 章节的 options 属性。*

参数属性说明

params 属性说明：

属性名	类型	默认值·	描述
name	*String*		自定义设置签章名称
width	*String*		自定义设置签章宽度，不可为空，单位 *CM*
height	*String*		自定义设置签章高度，不可为空，单位 *CM*
imageData	*String*		自定义设置签章图片，不可为空，标准为 *base64* 标准串。

返回值：

无

代码：

```
var signatureCreator = Signature.create();
var that = this;
var param = {name:"金格测试", width:"4", height:"4",
imageData:"R0lGODlhlwCXAPcAAAAAAAAAMwAAZgAAmQAAzAAA/wArAAArM……
"};
signatureCreator.runHW(param, {
    protectedItems:[ 'item1', 'item2', 'item3'],//设置定位页面DOM的id，自动查找ID，自动获取保护DOM的kg-desc属性作为保护项描述，value属性为保护数据。不设置，表示不保护数据，签章永远有效。
    position: $(that).attr('pos'),//设置盖章定位dom的ID，必须设置
    okCall: function(fn) {//点击确定后的回调方法，this为签章对象 ，签章数据撤销时，将回调此方法，需要实现签章数据持久化（保存数据到后台数据库）,保存成功后必须回调fn(true/false)渲染签章到页面上
        console.log("盖章ID: "+this.getSignatureid());
        console.log("盖章数据: "+this.getSignatureData());
        fn(true);
    },
```

```
    cancelCall : function() {//点击取消后的回调方法
        console.log("取消！ ")
    }
});
```

2.2.6 fingerPrintsDevice

获取指纹设备中图片执行盖章操作

参数列表：

参数名	类型	描述
config	*Object*	可配置属性参考：*config* 属性。
callback	*Function*	指纹设备签名回调函数

参数属性说明

config 属性说明：

属性名	类型	默认值·	描述
device_type	*Int*	*0*	设备类型 默认为 *0*（*iWebFingerPrints* 组件设备*(*指昂 *ZWY-060)*）
image_type	*String*	*gif*	获取指纹图片的格式；当前支持 *BMP*、*JPG*、*GIF* 格式
image_height	*String*	从仪器中获取高度	设置指纹签章高度，不可为空，单位 *CM*
image_width	*String*	从仪器中获取宽度	设置指纹签章宽度，不可为空，单位 *CM*
errorCall	*Function*		错误信息回调方法，默认为空，为空时默认走 *init* 中的 *errorCall* 回调

返回值：

参照 *2.2.5* 章节参数名 *params*

2.2.7 handWriteDevice

获取手写设备中图片执行盖章操作

参数列表：

参数名	类型	描述

config	*Object*	可配置属性参考：*config* 属性。
callback	*Function*	手写设备签名回调函数

参数属性说明

config 属性说明：

属性名	类型	默认值·	描述
device_type	*Int*	*0*	设备类型 默认为 *0*（*iPadControls* 组件设备 *Wacom STU-530/430/500B* 专用）
copy_right	*String*		设备授权码
image_height	*String*	从仪器中获取高度	设置手写签章高度，不可为空，单位 *CM*
image_width	*String*	从仪器中获取宽度	设置手写签章宽度，不可为空，单位 *CM*
errorCall	*Function*		错误信息回调方法，默认为空，为空时默认走 *init* 中的 *errorCall* 回调

返回值：

参照 *2.2.5* 章节参数名 *params*

2.2.8 runPwd

先执行密码输入框后获取签章方法（服务端有效）【*V1.0.0.290*】

参数列表：

请参照 *2.2.1* 章节的 *options* 属性。

返回值：

（无）

代码：

```
$('.testPwd').click(function(){
    var signatureCreator = Signature.create();
    var that = this;
    signatureCreator.runPwd({
        protectedItems:[ 'item1', 'item2', 'item3'],
        position:'pos0',//设置盖章定位dom的ID，必须设置
        autoCert:true,
        okCall: function(fn) {
            console.log("盖章ID: "+this.getSignatureid());
```

```
            console.log("盖章数据: "+this.getSignatureData());
            fn(true);
        },
        cancelCall : function() {//点击取消后的回调方法
            console.log("取消！ ")
        }
    });
})
```

2.2.9 saveSignature

保存签章函数

参数列表：

参数名	类型	描述
domid	*String*	文档 *ID*
signid	*String*	签章 *ID*
signdata	*String*	签章数据
callback	*Function*	保存签章回调函数

返回值：

保存签章结果 *data*

参数名	类型	描述
result	*Boolean*	返回保存签章结果

代码：

```
$('.testSave').click(function(){
    var signatureCreator = Signature.create();
    var that = this;
    var list = Signature.list;
    for (var key in list) {
        var tt = list[key];
        signatureCreator.saveSignature("KG2016093001", key,
list[key].getSignatureData(),function(ret){
            alert("保存" + ret.result);
        });
    }
});
```

2.2.10 getSaveSignatures

获取签章函数

参数列表：

参数名	类型	描述
domid	*String*	文档 *ID* 或者 *IDs*，多个文档中间用【,】号相隔
callback	*Function*	获取签章成功回调函数
errcallback	*Function*	获取签章失败回调函数

返回值：

保存签章结果 *data*

参数名	类型	描述
result	*Boolean*	获取的签章数据

代码：

```
var signatureCreator = Signature.create();
signatureCreator.getSaveSignatures("KG2016093001,KG2016093002",
function(signs) {
    var signdata = new Array();
    var jsonList = eval("(" + signs + ")");
    for (var i = 0; i < jsonList.length; i++) {
        var map = {};
        map.signatureid = jsonList[i]["signatureId"];
        map.signatureData = jsonList[i]["signature"];
        signdata.push(map);
    }
    Signature.loadSignatures(signdata,null,function(){
        var e = $("#pos0_div").find('.kg-img-div');
        console.log(e);
    });
},function(ret){
    Console.log(ret);
});
```

2.2.11 removeSignature

删除签章函数

参数列表：

参数名	类型	描述
domid	*String*	文档 *ID*
signatureid	*String*	签章 *id*
callback	*Function*	删除签章回调函数

返回值：

保存签章结果 *data*

参数名	类型	描述

result	*Boolean*	删除签章数据是否正常

代码：

```
signatureCreator.removeSignature(this.documentid,
this.getSignatureid());
```

2.2.12 sealService

签章操作服务对象

1、verifyPwd

验证密码接口

参数列表：

参数名	类型	描述
pwd	*String*	签章密码
keysn	*String*	签章 *Keysn*
successCall	*Function*	验证密码成功回调，返回值参照 *ret*
errorCall	*Function*	验证密码失败回调。返回值无

返回值：

成功回调返回值 *ret*

参数名	类型	描述
result	*Boolean*	返回保存签章结果，*true*：成功 *false*：失败
errcode	*String*	验证失败错误代号
errmsg	*String*	验证失败错误信息

代码：

```
$('.testverifyPwd').click(function(){
    //点击盖章操作后
    //再此处填写密码进行校验
    var pwd = '123456';
    var that = this;
    var signatureCreator = Signature.create();
    signatureCreator.sealService.verifyPwd({
        pwd:pwd,
        keysn:signatureCreator.keysn,
        successCall:function(ret){
            //正常返回结果
            if(ret.result){
                console.log("密码验证成功");
            }else{
                console.log("errcode:" + ret.errcode + "_errmsg:" +
```

```
ret.errmsg);
                return;
            }
        },errorCall:function(){
            //验证密码异常
            console.log("验证密码异常！ ");
            return;
        }
    });
});
```

2.2.13 toBase64Img

解析签章对象成标准 *base64* 图片数据

参数列表：

参数名	类型	描述
seal	*Object*	签章对象

返回值：

成功回调返回图片标准 *base64* 串。

代码：

2.2.14 getBase64Image

将签章 *div*，解析成 *base64* 图片数据（包含日期）

注：此方法需要引入 *html2canvas.min.js*

参数列表：

参数名	类型	描述
signatureId	*String*	签章 *Id*
SignatureData	*Object*	签章对象，成功回调函数返回对象
elemId	*String*	显示签章位置
Callback	*Function*	成功返回回调函数

返回值：

成功回调返回图片标准 *base64* 串。

代码：

```
signatureCreator.getBase64Image(
    this.getSignatureid(),
    this.getSignatureData(),
    $(that).attr('pos'),
```

```
    function(fn, imgdata,signid, sdata){
        $('#kg-img-div-postil').find('img').attr('src',imgdata);
        //console.log(imgdata);
    });
```

2.2.15 搜索签章

搜索签章加载方法

客户端搜索签章 loadSeals 方法加载时，将会调用无组件，必须插 *key* 且 *key* 有效。

使用搜索签章功能必须引入 *search.js*

```
<script type="text/javascript" src="../kinggrid/search.js"
charset="utf-8"></script>
<link rel="stylesheet" href="../kinggrid/css/search.css">
```

初始化页面需要给指定的都说标签加载对应的签章。

```
////////////绑定搜索/////////////
bindDOM("ipt", "ser_box", "bot_box", "oul", function(){
    var signatureCreator = Signature.create();
    signatureCreator.loadSeals(function(data){
        setSeals(data);
    });
});
//////////////////////////////////
```

搜索签章对应 *div* 的设置

```
<div id="ser_box">
    <input type="search" id="ipt" />
    <span><input id="sousuo" pos="pos0" value="搜索签章"
class="s_btn" type="submit"></span>
</div>
<div id="bot_box">
    <ul id="oul"></ul>
</div>
```

runSS 接口

参数列表：

参数名	类型	描述
options	*Object*	可配置属性参考：*2.2.1* 章节的 *options* 属性。
getSeal()	*Function*	设置搜索签章的签章数据，

代码：

```
$('.s_btn').click(function(){
    var signatureCreator = Signature.create();
    var that = this;
    signatureCreator.runSS({
        protectedItems:[ 'item1', 'item2', 'item3'],//设置定位页面
    DOM的id，自动查找ID，自动获取保护DOM的kg-desc属性作为保护项描述，
    value属性为保护数据。不设置，表示不保护数据，签章永远有效。
        position: 'pos0',//设置盖章定位dom的ID，必须设置
        autoCert : false,
        okCall: function(fn) {//点击确定后的回调方法，this为签章对象 ，
    签章数据撤销时，将回调此方法，需要实现签章数据持久化（保存数据到后
    台数据库）,保存成功后必须回调fn(true/false)渲染签章到页面上
            console.log("盖章ID: "+this.getSignatureid());
            console.log("盖章数据: "+this.getSignatureData());
            fn(true);
        },
        cancelCall : function() {//点击取消后的回调方法
            console.log("取消！")
        },
        beginCall: function(){
            //alert(123);
        },
        endCall: function(){
            //alert(456);
        }
    }, getSeal());
});
```

2.3 Signature 对象

签章对象，加载的电子签章对象。

属性方法列表：

属性方法名	描述
getSignatureid()	获取签章 *ID*。
getSignatureData()	获取签章数据。
verify(Object protectedItems)	验证保护数据。
show()	显示签章
hide()	印章签章

2.3.1 getSignatureid

获取签章 *id*。

参数列表：

（无）

返回值：

类型	描述
String	签章 *ID*。

代码：

```
var signature =  Signature.list['1467962754911688946'];
signature.getSignatureid();
```

2.3.2 getSignatureData

获取加密的签章数据。

参数列表：

（无）

返回值：

类型	描述
String	签章数据。

代码：

```
var signature =  Signature.list['1467962754911688946'];
var signatureDataStr = signature.getSignatureData();
```

2.3.3 verify

验证签章的保护数据。

参数列表：

参数名	类型	描述
options	*Object*	保护数据 *key-value* 值。不设置，自动获取盖章保护项的数据

返回值：

类型	描述
Boolean	返回 *true*，保护数据正常，*false* 为有保护项被修改。

代码：

```
var signature =  Signature.list['1467962754911688946'];
var isok = signature.verify({item1: '保护数据 1'});
if(!isok){
    console.log(signature.modifiedItems);
}
```

2.3.4 show

显示签章。

参数列表：

（无）

返回值：

（无）

代码：

```
var signature =  Signature.list['1467962754911688946'];
signature.show();
```

2.3.5 hide

隐藏签章。

参数列表：

（无）

返回值：

（无）

代码：

```
var signature =  Signature.list['1467962754911688946'];
signature.hide();
```

2.4 日期设置

日期自定义参数设置

```
signdate : {
  ischeck: true,//默认勾选
  fontFormat:'yyyy/MM/dd',
  fontFamily:'楷体',
  fontSize:16,
  fontColor:'#666666',
  position:'居右下(章内)'
},
```

2.4.1 选中日期选中设置（ischeck）

参数类型：布尔（*boolean*）
参数格式：
true：显示选中日期
false：显示不选中日期（默认值）

2.4.2 日期格式（fontFormat）

参数类型：字符串（*String*）
参数格式：
yyyy、*yy*：年
MM、*M*：月
dd、*d*：日
MMM ：【 *'Jan', 'Feb', 'Mar', 'Apr', 'May', 'Jun', 'Jul', 'Aug', 'Sep', 'Oct', 'Nov', 'Dec'*】
hh：小时
mm：分钟
ss：秒
特殊日期格式：
二零一七年一月一日（*yyyy* 年 *M* 月 *d* 日*）
二〇一七年一月一日（*yyyy* 年 *M* 月 *d* 日**）

注：大小写区分，月、日两个参数两位数高位补 *0*；

2.4.3 日期字体（fontFamily）

参数类型：字符串（*String*）
参数格式：无具体格式。*HTML* 支持的字体即可。

2.4.4 字体尺寸（fontSize）

参数类型：数字类型（*int*）
参数格式：建议 *8-14* 之间的字体大小。

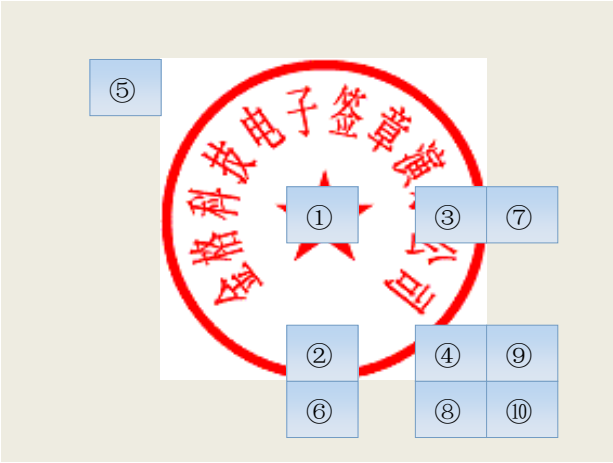
2.4.5 字体颜色（fontColor）

参数类型：字符串（*String*）
参数格式：颜色 *HTML* 代码。例如：*#FFFFFF*，仅支持#号带 *6* 位数 *0-9a-fA-F* 的代码。

2.4.6 字体位置（position）

参数类型：字符串（*String*）
参数格式：目前只支持以下 *10* 种位置

①居中
②居中下*(*章内*)*
③居右中*(*章内*)*
④居右下*(*章内*)*
⑤居左上*(*章外*)*
⑥居中下*(*章外*)*
⑦居右中*(*章外*)*
⑧居右下*(*章下*)*
⑨居右下*(*章右*)*
⑩居右下*(*章右下*)*



2.5 可移动范围控制

签章移动指定的可移动范围要求比印章本身的宽高要高。
签章显示的 *div* 最好在可移动范围的 *div* 内，否则初次移动的时候会有较大的跨越距离。

2.5.1 盖章时参数控制

执行 *run* 方法签章时，指定可移动范围的元素（*div*）的 *id*。参照参数 *moveableRange* 可移动范围的参数将放置在签章数据中保存，下次重新 *load* 该签章依旧只可以在指定范围内移动。

代码示例：

```
var protectedItems = signatureCreator.run({
          protectedItems:[ 'item1', 'item2', 'item3'],
          position: $(that).attr('pos'),
          autoCert:true,
          moveableRange:"page",//可移动DIV的id
          okCall: function(fn) {
             console.log("盖章ID: "+this.getSignatureid());
             console.log("盖章数据: "+this.getSignatureData());
             fn(true);
          },
          cancelCall : function() {//点击取消后的回调方法
             console.log("取消！")
          }
       });
   });
```

2.5.2 显示签章时，参照控制

* *

在指定显示签章的 *html* 元素上添加 *kg-mita* 属性，属性 *value* 指定可移动范围的元素（*div*）的 *id*，在签章显示或者盖章以后，在该显示位置显示的签章将只能在指定的范围内移动。

该属性不放置在签章数据中。下次 *load* 签章后，如果该 *div* 显示位置 *pos0* 没有 *kg-mita* 属性，签章可以自由移动。

代码示例：

```
<td id="pos0_div" width=200 height="100" >
   <div style="position:absolute;">
      <div id="pos0" kg-mita="pos0_div"></div>
   </div>
</td>
```

2.6 Error 定义

以下为 *iSignatureHTML5* 错误信息回调函数返回值及意义。

接口名称：客户端特定接口返回参数，默认为空

Code 码：*errorCode* 编号

描述：错误码的意义

范围：

Code 码	接口名称	描述	范围
serverErr		调用产品组件异常	（已删）
parseErr		数据格式异常	客户端、云端
responseErr		产品组件响应异常	客户端
connectionErr		产品组件连接异常	云端
timeoutErr		请求响应超时	客户端、云端
111001		该用户没有公章	客户端、云端
111002		该用户没有私章	客户端、云端
111003		该用户没有法人章	客户端、云端
111004		该用户没有法人签名印章	客户端、云端
111005		该用户没有手写签名印章	客户端、云端
11101		当前用户没有印章	云端
11102		印章错误	云端
11103		证书过期	云端
11104		印章过期	云端
11105		获取签章信息失败!	客户端、云端
11106		获取证书信息失败!	客户端、云端
11107		未搜索到您要的签章!	客户端、云端
11108		未设置签章密码!	客户端、云端
none		客户端未知异常!	客户端
interErr		获取接口异常	客户端
signErr		签名失败	客户端
-7		签名组件错误	客户端
-1		请安装客户端	客户端
-2		请插入当前签章所属的密钥盘	客户端
-12		获取证书失败	客户端
-13		签名时，加载证书失败	客户端
-15		签名数据不能为空	客户端
-16		密码不能为空	客户端
-17		解析证书失败	客户端
1		没有检测到密钥盘	客户端
2		密钥盘没有正确初始化	客户端
3		密钥盘驱动未安装	客户端
4		只能插入一个密钥盘	客户端
7		密码错误或密钥盘被锁定	客户端
8		非法授权	客户端
11		当前密钥盘没有印章	客户端
12		获取印章信息失败	客户端
35		钥匙盘未生效，不能签章！	客户端
36		钥匙盘已过期，不能签章！	客户端

37		授权信息结构非法！	客户端
38		签章证书已经过期或还未生效！	客户端
39		签章证书不受颁发机构信任！	客户端
40		签章证书已被吊销！	客户端
41		签章证书根证书未安装或无效的根证书！	客户端
42		吊销列表文件不存在	客户端
43		未检测到证书	客户端
44		根证书文件不存在	客户端
45		根证书和该证书的根证书不配套	客户端
46		吊销列表和该证书的吊销列表不配套	客户端
511		当前秘钥盘可能未发放，未启用，已过期。	云端
-101		获取密钥盘有效期异常	客户端
-103		密钥盘未生效	客户端
-104		期限许可已过期	客户端
-999		金格组件未知错误	客户端
30001@gm		验证电子印章失败：验证电子印章签名值失败！	云端（国密印章）
30002@gm		验证电子印章失败：验证电子印章制章人证书有效期失败！	云端（国密印章）
30003@gm		验证电子印章失败：验证电子印章的有效期失败！	云端（国密印章）
30004@gm		验证电子印章失败：验证签章人证书有效性失败！	云端（国密印章）
30005@gm		验证电子印章失败：验证电子签章数据失败！	云端（国密印章）
-1	*fingerPrintsDevice*	设备类型设置错误	客户端（指纹）
-2	*fingerPrintsDevice*	没有安装指纹组件	客户端（指纹）
-3	*fingerPrintsDevice*	指纹采集失败	客户端（指纹）
-1	*handWriteDevice*	设备类型设置错误	客户端（手写设备）
-2	*handWriteDevice*	没有安装手写组件	客户端（手写设备）
-3	*handWriteDevice*	授权码不正确	客户端（手写设备）
-4	*handWriteDevice*	签名取消	客户端（手写设备）
-5	*handWriteDevice*	无服务可用	客户端（手写设备）
-6	*handWriteDevice*	签名设备错误	客户端（手写设备）
-7	*handWriteDevice*	未知错误	客户端（手写设备）
1	*KGGetTSWithDigest*	获取时间戳异常	客户端，时间戳
2	*KGGetTSWithDigest*	获取时间戳失败	客户端，时间戳
3	*KGGetTSWithDigest*	响应数据为空	客户端，时间戳

1	*KGGetTSURLInfo*	时间戳未启用	客户端，时间戳
-1	*KGValidate_GM*	没有安装签名组件	客户端（国密印章）
-2	*KGValidate_GM*	没有公共模块组件	客户端（国密印章）
-3	*KGValidate_GM*	*KCL280* 验证失败，未知原因的验证失败	客户端（国密印章）
1	*KGValidate_GM*	*KCL272*_验证签章人证书失败，签章人证书已过期或未生效！	客户端（国密印章）
2	*KGValidate_GM*	*KCL274* 验证签章人证书失败，签章人证书根证书验证未通过*!*	客户端（国密印章）
3	*KGValidate_GM*	*KCL270*_验证签章人证书失败，签章人证书已被吊销！	客户端（国密印章）
4	*KGValidate_GM*	*KCL276* 验证签章人证书失败，签章人证书根证书安装未成功*!*	客户端（国密印章）
5	*KGValidate_GM*	*KCL266*_验证签章人证书失败，未检测到签章人证书吊销列表！	客户端（国密印章）
6	*KGValidate_GM*	*KCL260*_验证签章人证书失败，未检测到签章人证书！	客户端（国密印章）
7	*KGValidate_GM*	*KCL262*_验证签章人根证书失败，未检测到签章人根证书！	客户端（国密印章）
8	*KGValidate_GM*	*KCL264*_验证签章人根证书失败，签章人证书与该证书的根证书不匹配！	客户端（国密印章）
9	*KGValidate_GM*	*KCL268*_验证签章人证书失败，签章人证书与该证书吊销列表不匹配！	客户端（国密印章）
10	*KGValidate_GM*	*KCL273*_验证制章人证书失败，制章人证书已过期或未生效	客户端（国密印章）
11	*KGValidate_GM*	*KCL275* 验证制章人证书失败，制章人证书根证书验证未通过*!*	客户端（国密印章）
12	*KGValidate_GM*	*KCL271*_验证制章人证书失败，制章人证书已被吊销！	客户端（国密印章）
13	*KGValidate_GM*	*KCL277*_验证制章人证书失败，制章人证书根证书安装未成功*!*	客户端（国密印章）
14	*KGValidate_GM*	*KCL267*_验证制章人证书失败，未检测到制章人证书吊销列表！	客户端（国密印章）
15	*KGValidate_GM*	*KCL281*_验证制章人证书失败，未检测到制章人证书！	客户端（国密印章）
16	*KGValidate_GM*	*KCL263*_验证制章人根证书失败，未检测到制章人根证书！	客户端（国密印章）
17	*KGValidate_GM*	*KCL265*_验证制章人根证书失败，制章人证书与该证书的根证书不匹配！	客户端（国密印章）

18	*KGValidate_GM*	*KCL269*_验证制章人证书失败，制章人证书与该证书吊销列表不匹配！	客户端（国密印章）
19	*KGValidate_GM*	*KCL261*_验证签章人证书失败，签章人证书不在印章的签章人证书列表中！	客户端（国密印章）
20	*KGValidate_GM*	*KCL278*_验证印章制章时间失败，印章未生效！	客户端（国密印章）
21	*KGValidate_GM*	*KCL279* 验证印章制章时间失败，印章已经过期！	客户端（国密印章）

2.7 Webservice 接口说明

签名验证接口。
参数说明：
http://ip:port/iSignatureHTML5/services/ SignVerifyWS?wsdl

```
    String signVerify(@WebParam(name = "signData") String
signData,@WebParam(name = "protectedData")String protectedData);
```

signData：签名数据（H5 页面返回的签章数据）
protectedData：保护数据（json 数组字符串）

```
    [{
        field: "item1",
        desc: "保护项 1",
        value: "同创软件公司"
    },{
        field: "item2",
        desc: "保护项 2",
        value: "金格科技有限公司"
    },{
        field: "item3",
        desc: "保护项 3",
        value: "12"
    }]
```

返回值：
验证结果：*json* 字符串

```
  {
    "result": true,//验证成功或者失败，true：验证成功，false：验证失
  败
```

```
"isModified": false,//是否被修改，true 代表保护项被篡改
"signatureImgData": {
   "startdata": "",
   "signname": "金格印章 1",
   "imgext": ".gif",
   "endata": "",
   "imgdata": ".....",
   "width": "4.00",
   "imgtag": "0",
   "height": "4.00",
   "username": "金格"
},
"documentId": "KG2016093001",
"modifiedData": ["保护项 3 由:**改变为:12"]
```

3 文档声明

本文档内容改动及版本更新将不再另行通知。本文档的范例中使用的人名、公司名和数据如果没有特别指明，均属虚构。对于本文档、及本文档涉及的技术和产品，江西金格科技股份有限公司拥有其专利、商标、著作权或其它知识产权，除非得到江西金格科技股份有限公司的书面许可，本文档不授予这些专利、商标、著作权或其它知识产权的许可。

- *Kinggrid*、*iWebOffice*、*iSignature* 和 *DBPacket* 是江西金格科技股份有限公司的商标。
- 其它标牌和产品名称是其各自公司的商标或注册商标。
- 本文档最后更新时间：*2020-01-02*。

（完）