**电子印章服务器API手册**

***Version 3.2.4***

版权声明：本文档的版权属于中国金融认证中心，任何人或组织未经许可，不得擅自修改、拷贝或以其它方式使用本文档中的内容。

目 录

[1 引言 3](#_Toc459103001)

[1.1 概述 3](#_Toc459103002)

[1.2 开发平台及编程语言 3](#_Toc459103003)

[1.3 名词解释 3](#_Toc459103004)

[1.4 接口概述 4](#_Toc459103005)

[2 制章接口说明 4](#_Toc459103006)

[2.1 普通制章 4](#_Toc459103007)

[2.2 制人名章(个人章) 4](#_Toc459103008)

[2.3 更新普通印章 5](#_Toc459103009)

[2.4 更新人名章 5](#_Toc459103010)

[3 自动化制章接口说明 6](#_Toc459103011)

[3.1 自动化普通制章 6](#_Toc459103012)

[3.2 自动化制人名章(个人章) 6](#_Toc459103013)

[3.3 自动化更新普通印章 7](#_Toc459103014)

[3.4 自动化更新人名章 7](#_Toc459103015)

[4 签章接口说明 8](#_Toc459103016)

[4.1 PDF自动化签章接口 8](#_Toc459103017)

[4.2 PDF签章功能接口 8](#_Toc459103018)

[4.3 PDF自动化批量签章接口 9](#_Toc459103019)

[4.4 PDF自动化批量签章接口(不检查) 9](#_Toc459103020)

[4.5 PDF列表自动化批量签章接口 10](#_Toc459103021)

[4.6 PDF签章功能Base64接口 10](#_Toc459103022)

[4.7 PDF自动化骑缝章签章接口 11](#_Toc459103023)

[4.8 PDF添加水印接口 11](#_Toc459103024)

[4.9 WEB签章接口 11](#_Toc459103025)

[4.10 自动化计算Pdf签章Hash值接口 12](#_Toc459103026)

[4.11 Pdf合成外部签名接口 13](#_Toc459103027)

[5 验章接口说明 13](#_Toc459103028)

[5.1 PDF验章接口 13](#_Toc459103029)

[5.2 Web验章接口 14](#_Toc459103030)

[6 业务接口说明 14](#_Toc459103031)

[6.1 PDF业务数据自动化合成接口 14](#_Toc459103032)

[6.2 Html/Word转换Pdf功能 15](#_Toc459103033)

[6.3 合并PDF接口 16](#_Toc459103034)

[6.4 合并PDF列表接口 16](#_Toc459103035)

[6.5 PDF添加空白签名域接口 17](#_Toc459103036)

[6.6 获取印章申请信息接口 17](#_Toc459103037)

[7 辅助接口说明 18](#_Toc459103038)

[7.1 P1签名接口 18](#_Toc459103039)

[7.2 P7分离式签名接口 18](#_Toc459103040)

[7.3 P7分离式验签接口 19](#_Toc459103041)

[7.4 获取印章信息接口 19](#_Toc459103042)

[7.5 客户端制章并绑定接口 20](#_Toc459103043)

[7.6 客户端印章绑定接口 20](#_Toc459103044)

[7.7 获取绑定印章信息接口 21](#_Toc459103045)

[7.8 自助生成图片接口 22](#_Toc459103046)

[7.9 获取关键字在PDF中的坐标 22](#_Toc459103047)

[7.10 获取PDF页数总和 23](#_Toc459103048)

[7.11 撤销PDF印章接口 23](#_Toc459103049)

[8 附件 23](#_Toc459103050)

# 引言

## 概述

此文档是对电子印章服务器接口的使用说明。

## 开发平台及编程语言

* 开发平台

Windows 7

* 编程语言

JAVA

## 名词解释

* 数字证书

数字证书是各类实体（持卡人/个人、商户/企业、网关/银行等），在网上进行信息交流及商务活动的身份证明，在电子交易的各个环节，交易的各方都需验证对方证书的有效性，从而解决相互间的信任问题。简单的说，数字证书是一段包含用户身份信息、用户公钥信息以及身份验证机构数字签名的数据。身份验证机构的数字签名可以确保证书信息的真实性，证书格式及证书内容遵循X.509标准。

从证书的一般用途来看，数字证书可分为签名证书和加密证书。签名证书主要用于对用户信息进行签名，以保证信息的不可否认性；加密证书主要用于对用户传送信息进行加密，以保证信息的真实性和完整性。

* 电子印章

电子印章是电子签名一种表现形式，利用图像处理技术将电子签名操作转

化为与纸质文件盖章操作相同的可视效果，同时利用电子签名技术保障电子信息的真实性和完整性以及签名人的不可否认性。

* 数字签名

在进行数字签名过程中，发信者使用自己的私钥，通过非对称密码算法，对待发数据的数字摘要（哈希值）进行加密，从而得到一段信息称为数字签名（Sign）。

* 证书吊销列表

证书吊销列表CRL是一种包含吊销的证书列表的签名数据结构。CRL是证书吊销状态的公布形式，CRL就像信用卡的黑名单，它通知其他证书用户某些电子证书不再有效。

## 接口概述

工具包的接口包括的功能如下：

分为制章、pdf签章和验章、web签章及验章以及一些辅助接口等等

# 制章接口说明

## 普通制章

public String **makeSeal**（byte[] pfx, String pfxPassword, byte[] image, String sealInfoXML）throws Exception

描述：可以制作企业章和个人章，印模从接口传入

**入参：**

pfx：软证书文件流

pfxPassword：软证书文件口令

image：印章印模（为空时可以根据印章策略文件中的印模编码来制章）

sealInfoXML：印章策略文件，[见附件](#_附件)

**返回：**（Code不等于200时为异常）

<Result>

<Code>200</Code>

<Message>successfully!</Message>

</Result>

**抛出：**

Exception

## 制人名章(个人章)

public String **makeNamedSeal**(byte[] pfx, String pfxPassword, String namedSealInfoXML) throws Exception

描述：人名章默认为个人章，印模图片由服务器根据人名生成图片

**入参：**

pfx：软证书文件流

pfxPassword：软证书口令

namedSealInfoXML：印章策略文件，见附件

**返回：**（Code不等于200时为异常）

<Result>

<Code>200</Code>

<Message>successfully!</Message>

</Result>

**抛出：**

Exception

## 更新普通印章

public String **updateSeal**(byte[] pfx, String pfxPassword, byte[] image, String sealInfoXML) throws Exception

**入参：**

pfx：软证书文件流

pfxPassword：软证书口令

image：印章印模（可以为空）

sealInfoXML：印章策略文件，见附件

**返回：**（Code不等于200时为异常）

<Result>

<Code>200</Code>

<Message>successfully!</Message>

</Result>

**抛出：**

Exception

## 更新人名章

public String **updateNamedSeal**(byte[] pfx,String pfxPassword, String namedSealInfoXML) throws Exception

**入参：**

pfx：软证书文件流

pfxPassword：软证书口令

namedSealInfoXML：印章策略文件，见附件

**返回：**（Code不等于200时为异常）

<Result>

<Code>200</Code>

<Message>successfully!</Message>

</Result>

**抛出：**

Exception

# 自动化制章接口说明

## 自动化普通制章

public String **makeSealAutomation**（String certApplyXML, byte[] image, String sealInfoXML）throws Exception

描述：可以制作企业章和个人章，印模从接口传入

**入参：**

certApplyXML:

<Request>

<CustomerType>1:个人；2：企业</CustomerType>//非空

<KeyAlg>RSA/SM2</KeyAlg>//非空

<KeyLength>1024/2048/256</KeyLength>//非空

<UserName>用户名</UserName>//非空

<IdentificationType>证件类型</IdentificationType>//非空

<IdentificationNo>证件号码</IdentificationNo>//非空

</Request>

image：印章印模（为空时可以根据印章策略文件中的印模编码来制章）

sealInfoXML：印章策略文件，见附件

注：每次申请时“印章编码”和“证件类型-证件号-印章标识”必须唯一

**返回：**（Code不等于200时为异常）

<Result>

<Code>200</Code>

<Message>successfully!</Message>

</Result>

**抛出：**

Exception

## 自动化制人名章(个人章)

public String **makeNamedSealAutomation**(String certApplyXML,String namedSealInfoXML) throws Exception

描述：人名章默认为个人章，印模图片由服务器根据人名生成图片

**入参：**

certApplyXML:

<Request>

<KeyAlg>RSA/SM2</KeyAlg>

<KeyLength>1024/2048/256</KeyLength>

<UserName>用户名</UserName>

<IdentificationType>证件类型</IdentificationType>

<IdentificationNo>证件号码</IdentificationNo>

</Request>

namedSealInfoXML：印章策略文件，见附件

**返回：**（Code不等于200时为异常）

<Result>

<Code>200</Code>

<Message>successfully!</Message>

</Result>

**抛出：**

Exception

## 自动化更新普通印章

public String **updateSealAutomation**(byte[] image, String sealInfoXML) throws Exception

**入参：**

image：印章印模（可以为空，为空时印模不变）

sealInfoXML：印章策略文件，见附件

**返回：**（Code不等于200时为异常）

<Result>

<Code>200</Code>

<Message>successfully!</Message>

</Result>

**抛出：**

Exception

## 自动化更新人名章

public String **updateNamedSealAutomation**(String namedSealInfoXML) throws Exception

**入参：**

namedSealInfoXML：印章策略文件，见附件

**返回：**（Code不等于200时为异常）

<Result>

<Code>200</Code>

<Message>successfully!</Message>

</Result>

**抛出：**

Exception

# 签章接口说明

## PDF自动化签章接口

public byte[] **sealAutoPdf**(byte[] pdf, String sealStrategyXML) throws Exception

**入参：**

pdf：pdf文件流

sealStrategyXML：签章策略文件，见附件

**正常返回：**

盖章后的pdf文件

**异常返回：**

<Error>

<ErrorCode>errorCode</ ErrorCode >

<ErrorMessage>errorMessage</ErrorMessage>

</Error>

**抛出：**

Exception

## PDF签章功能接口

public byte[] **sealPdfFunction**(byte[] pdf, byte[] pfx, String pfxPassword, byte[] image, String sealFunctionStrategyXML) throws Exception

**入参：**

pdf：pdf文件流

pfx：pfx字节流

pfxPassword：pfx密码

image：印模数据

sealFunctionStrategyXML：签章功能策略文件，见附件

**正常返回：**

盖章后的pdf文件

**异常返回：**

<Error>

<ErrorCode>errorCode</ ErrorCode >

<ErrorMessage>errorMessage</ErrorMessage>

</Error>

**抛出：**

Exception

## PDF自动化批量签章接口

public byte[] **batchSealAutoPdf**(byte[] pdf, String batchSealStrategyXML) throws Exception

**入参：**

pdf：pdf文件流，pdf为空时，取第一个sealStrategyXML中的PdfURL

batchSealStrategyXML：批量签章策略文件，格式如下:

<List> sealStrategyXML1,sealStrategyXML2….</List>

**正常返回：**

盖章后的pdf文件

**异常返回：**

<Error>

<ErrorCode>errorCode</ ErrorCode >

<ErrorMessage>errorMessage</ErrorMessage>

</Error>

**抛出：**

Exception

## PDF自动化批量签章接口(不检查)

public byte[] **batchSealAutoUncheckedPdf**(byte[] pdf, String batchSealStrategyXML) throws Exception

*不检查是否存在空白域和关键字*

**入参：**

pdf：pdf文件流，pdf为空时，取第一个sealStrategyXML中的PdfURL

batchSealStrategyXML：批量签章策略文件，格式如下:

<List> sealStrategyXML1,sealStrategyXML2….</List>

**正常返回：**

盖章后的pdf文件

**异常返回：**

<Error>

<ErrorCode>errorCode</ ErrorCode >

<ErrorMessage>errorMessage</ErrorMessage>

</Error>

**抛出：**

Exception

## PDF列表自动化批量签章接口

public String **batchSealAutoPdfList**(String pdfListXML, String batchSealStrategyXML) throws Exception

**入参：**

pdfListXML：格式如下：

<PdfList><Pdf>pdf base64编码</Pdf><Pdf></Pdf>…..</PdfList>

batchSealStrategyXML：批量签章策略文件，格式如下:

<List> sealStrategyXML1,sealStrategyXML2….</List>

**正常返回：**（Code不等于200时为异常）

<Result>

<Code>200</Code >

<Message> successfully!</Message>

<Pdf>pdfBase64</Pdf>

<Pdf>pdfBase64</Pdf>

</ Result >

**抛出：**

Exception

## PDF签章功能Base64接口

public String **sealBase64PdfFunction** (String pdfBase64, String pfxBase64, String pfxPassword, String imageBase64, String sealFunctionStrategyXML) throws Exception

**入参：**

pdfBase64：pdf的Base64编码，如果此参数为空用策略文件的PdfURL参数

pfxBase64：pfx的Base64编码

pfxPassword：pfx密码的Base64编码

imageBase64：印模Base64编码

sealFunctionStrategyXML：签章功能策略文件，见附件

**返回：**

<Result>

<Code>code</Code >

<Message>message</Message>

<Pdf>pdfBase64</Pdf>

</ Result >

**抛出：**

Exception

## PDF自动化骑缝章签章接口

public byte[] **sealAutoCrossPdf**(byte[] pdf, String crossSealStrategyXML) throws Exception

**入参：**

pdf：pdf文件流

crossSealStrategyXML：骑缝签章策略文件，见附件

**正常返回：**

盖章后的pdf文件

**异常返回：**

<Error>

<ErrorCode>errorCode</ErrorCode>

<ErrorMessage>errorMessage</ErrorMessage>

</Error>

**抛出：**

Exception

## PDF添加水印接口

public byte[] **addWaterMark2Pdf**(byte[] pdf, String waterMarkStrategyXML) throws Exception

**入参：**

pdf：pdf文件流

waterMarkStrategyXML：添加水印策略文件，见附件

**正常返回：**

添加水印后的pdf文件

**异常返回：**

<Error>

<ErrorCode>errorCode</ErrorCode>

<ErrorMessage>errorMessage</ErrorMessage>

</Error>

**抛出：**

Exception

## WEB签章接口

public byte[] **signWebSeal** (String sourceBase64, String sealStrategyXml) throws Exception

**入参：**

sourceBase64：原文Base64编码

sealStrategyXML：Web签章策略文件，见附件

**正常返回：签章结果**

**异常返回：**

<Error>

<ErrorCode>errorCode</ErrorCode>

<ErrorMessage>errorMessage</ErrorMessage>

</Error>

**抛出：**

Exception

## 自动化计算Pdf签章Hash值接口

public String **autoCalculatePdfHash**(byte[] pdf, byte[] x509Cert, String strategyXML) throws Exception

**入参：**

pdf：pdf文件

x509Cert：公钥证书，可以为空，为空时外部只能传入Pkcs7。

strategyXML：策略文件，格式如下：

<Request>

<ClientSealCode></ClientSealCode>如果为空可以取ImageBase64

<ImageBase64></ ImageBase64>

<Type></Type>

<Page></Page>

<LX></LX>

<LY></LY>

<Keyword></Keyword>//只能支持文档中有一个关键字

<LocationStyle></LocationStyle>

<OffsetX></OffsetX>

<OffsetY></OffsetY>

<HashAlg>sha1/sha256 </HashAlg>

</Request>

**正常返回：**（Code不等于200时为异常）

<Result>

<Code>200</Code>

<Message>successfully!</Message>

<PdfSealHash>hash值Base64编码字符串</ PdfSealHash>

<PdfSealSource>原文Base64编码</ PdfSealHash>

<PdfHash>hashBase64做原文签名用</PdfHash>

<Id>保留值标识</ Id>

</Result>

**抛出：**

Exception

## Pdf合成外部签名接口

public byte[] **synthesizeOuterSignature**(byte[] signature, String id) throws Exception

**入参：**

signature：p1签名值或者p7签名值，计算hash时如果不带证书，此处必须是p7格式

id：保留值标识

**正常返回：**

Pdf文件

**异常返回：**

<Error>

<ErrorCode>errorCode</ ErrorCode >

<ErrorMessage>errorMessage</ErrorMessage>

</Error>

**抛出：**

Exception

# 验章接口说明

## PDF验章接口

public String **verifyPdfSeal**(byte[] sealedPdf, String verifyStrategyXML) throws Exception

**入参：**

sealedPdf：pdf文件

verifyStrategyXML：验章策略文件，见附件

**返回：**（Code不等于200时为异常）

<Result>

<Code>200</Code>

<Message>successfully!</Message>

<CertDN>dn1 |+| dn2</CertDN>

<List>

<Bean>

<CertDN></CertDN>

<SignatureTime> yyyymmddhhmmss </SignatureTime>

</Bean>

</List>

</Result>

**抛出：**

Exception

## Web验章接口

public String verifyWebSeal(String webSealSource, String sourceBase64, String verifyStrategyXML) throws Exception

**入参：**

webSealSource：web签章结果

sourceBase64：原文的Base64编码

verifyStrategyXML：验章策略文件，见附件

**返回：**（Code不等于200时为异常）

<Result>

<Code>200</Code>

<Message>successfully!</Message>

<CertDN></CertDN>

</Result>

**抛出：**

Exception

# 业务接口说明

## PDF业务数据自动化合成接口

public byte[] **synthesizeAutoBusinessPdf**(byte[] pdf, String fieldListXML，String imageListXML) throws Exception

**入参：**

pdf：pdf文件流

fieldListXML: 需要合成到pdf文本域的业务数据，可为空，格式如下：

<FieldList hashAlg="sha1">

<Field>

<FieldID></FieldID>必填

<FieldName></ FieldName>字段名称，**可为空**

<FieldValue></FieldValue>必填

<FieldType></ FieldType> FieldType：text,checkbox

<FieldValueHash></FieldValueHash>可为空

</Field>

<FieldList>

imageListXML: 需要合成到pdf的图片数据，可为空，格式如下：

<ImageList hashAlg="sha1">

<Image>可以没有该节点

< ImageValue ></ ImageValue >图片base64编码，必填

<ImageWidth></ ImageWidth >图片宽度，必填

<ImageHeight></ ImageHeight >图片高度，必填

<ImageHash></ ImageHash >可为空

<Type></Type>类型，2：坐标；3：关键字，必填

<PageNo>1</PageNo> 页码，必填

<LX>100</LX> 必填

<LY>100</LY> 必填

<Keyword></ Keyword>关键字，必填

< LocationStyle></ LocationStyle>关键字风格，U、D、L、R, 必填

< OffsetX></ OffsetX>X轴偏移量，可为空

< OffsetY></ OffsetY> X轴偏移量，可为空

</Image>

<QRCodeXML>**可为空，也可以是多个**

<QRCodeContent></ QRCodeContent>二维码内容，必填

<Width></ Width>二维码宽度默认，必填

<Height></ Height>二维码高度默认，必填

<PageNo>1</PageNo> 页码，必填

<LX></LX>二维码存放在pdf中的位置，必填

<LY></LY>二维码存放在pdf中的位置，必填

</QRCodeXML>

</ImageList >

**正常返回：**

合成后的pdf文件

**异常返回：**

<Error>

<ErrorCode>errorCode</ErrorCode>

<ErrorMessage>errorMessage</ErrorMessage>

</Error>

**抛出：**

Exception

## Html/Word转换Pdf功能

public byte[] **transformToPdf**(byte[] source，String fileType) throws Exception

**入参：**

source：待转换的内容的字节UTF-8编码

fileType：可以取值html、word、html2

**正常返回：转换后的pdf文件**

**异常返回：**

<Error>

<ErrorCode>errorCode</ErrorCode>

<ErrorMessage>errorMessage</ErrorMessage>

</Error>

**抛出：**

Exception

## 合并PDF接口

public byte[] **autoConcatPdfs**(byte[] pdf1, byte[] pdf2) throws Exception

**入参：**

pdf1：第一个pdf

pdf1：第二个pdf

**正常返回：**

合并后的pdf

**异常返回：**

<Error>

<ErrorCode>errorCode</ ErrorCode >

<ErrorMessage>errorMessage</ErrorMessage>

</Error>

**抛出：**

Exception

## 合并PDF列表接口

public String **autoConcatPdfList**(String pdfBase64List, String outputFilePath) throws Exception

**入参：**

pdfBase64List：<List>< Pdf >pdf1Base64</ Pdf >< Pdf >pdf2Base64</ Pdf ></List>如果pdf1Base64、pdf2Base64字节长度小于500时，按文件路径处理

outputFilePath：该参数可为空，该参数不为空时，合并后的pdf放在此处

**正常返回：**（Code不等于200时为异常）

<Result>

<Code>200</Code>

<Message>successfully!</Message>

<Pdf>>pdfBase64</Pdf>

</Result>

**抛出：**

Exception

## PDF添加空白签名域接口

public byte[] **addBlankSignature2Pdf**(byte[] pdf, String blankSignatureStrategyXML) throws Exception

**入参：**

pdf：pdf文件流

blankSignatureStrategyXML：添加签名域策略文件，如下：

<List>

<BlankSignature>

<FieldName>fieldname</ FieldName>

<Page>1</Page>

<LX>100</LX>//左下角为原点

<LY>100</LY>//左下角为原点

<Width>100</Width>

<Height>100</Height>

</BlankSignature>

<List>

**正常返回：**

添加签名域的pdf文件

**异常返回：**

<Error>

<ErrorCode>errorCode</ErrorCode>

<ErrorMessage>errorMessage</ErrorMessage>

</Error>

**抛出：**

Exception

## 获取印章申请信息接口

public String **getSealApplyInfo**(String requestXML) throws Exception

**入参：**

requestXML：申请XML

<RequestXML>

<UserName>用户名称</UserName>

<IdentificationType>证件类型</IdentificationType >

<IdentificationNo>证件号码</IdentificationNo>

<SealFlag></SealFlag>

</RequestXML>

**正常返回：**（Code不等于200时为异常）

<Result>

<Code>200</Code>

<Message>successfully!</Message>

<SealApplyInfo>

<SealCode><SealCode>印章编码

<CertDN><CertDN> 证书DN

<CertSN><CertSN> 证书序列号

<CertNotBefore>yyyyMMddHHmmss<CertNotBefore>证书有效期开始

<CertNotAfter> yyyyMMddHHmmss <CertNotAfter> 证书有效期结束

</SealApplyInfo>

</Result>

**抛出：**

Exception

# 辅助接口说明

## P1签名接口

public String **p1Sign**(byte[] source, String signStrategyXML) throws Exception

**入参：**

source：待签名原文

signStrategyXML：签名策略文件，见附件

**返回：**（Code不等于200时为异常）

<Result>

<Code>200</Code>

<Message>successfully!</Message>

<Signature>签名结果（Base64格式）</ Signature>

</Result>

**抛出：**

Exception

## P7分离式签名接口

public String **p7SignDetached**(byte[] source, String signStrategyXML) throws Exception

**入参：**

source：待签名原文

signStrategyXML：签名策略文件，见附件

**返回：**（Code不等于200时为异常）

<Result>

<Code>200</Code>

<Message>successfully!</Message>

<Signature>签名结果</ Signature>

</Result>

**抛出：**

Exception

## P7分离式验签接口

public String **p7VerifyDetached**(String signatureBase64, byte[] source, String verifyStrategyXML) throws Exception

**入参：**

signatureBase64：签名Base64编码

source：原文

verifyStrategyXML：验签策略文件，见附件

**返回：**（Code不等于200时为异常）

<Result>

<Code>200</Code>

<Message>successfully!</Message>

<CertDN>dn</CertDN>

<CertSN>sn</CertSN>

<P1Base64>p1Base64</P1Base64>

</Result>

**抛出：**

Exception

## 获取印章信息接口

public String **getSealInfo**(String sealCode) throws Exception

**入参：**

sealCode：印章编码

**正常返回：**（Code不等于200时为异常）

<Result>

<Code>200</Code>

<Message>successfully!</Message>

<SealInfoXML>

<SealPerson> 制章人

<SealOrg> 制章单位

<SealName> 印章名称

<SealCode> 印章编码

<Image> 印模图片Base64编码

<CertDN> 证书DN

<CertSN> 证书序列号

<CertNotBefore> 证书有效期开始

<CertNotAfter> 证书有效期结束

</SealInfoXML>

</Result>

**抛出：**

Exception

## 客户端制章并绑定接口

public String **makeAndBindClientSeal**（byte[] imageData, String clientSealXML， String userXML, String orgCode）throws Exception

描述：客户端印章制章并绑定用户，如果用户存在，更新用户信息，添加新的绑定关系。

**入参：**

clientSealXML：客户端印章策略文件，见附件

userXML：格式如下：

<User>

<CertDN></CertDN>

<CertSN></CertSN>

<Name></Name>

</RealName></ RealName >

</User>

orgCode：机构编码

**返回：**（Code不等于200时为异常）

<Result>

<Code>200</Code>

<Message>successfully!</Message>

</Result>

**抛出：**

Exception

## 客户端印章绑定接口

public String **bindClientSeal**（String clientSealCode， String userXML, String orgCode）throws Exception

描述：客户端印章绑定用户，如果用户存在，更新用户信息，添加新的绑定关系，如果绑定关系已经存在，返回绑定成功。

**入参：**

clientSealCode：客户端印章编码

userXML：格式如下：

<User>

<CertDN></CertDN>

<CertSN></CertSN>

<Name></Name>

</RealName></ RealName >

</User>

orgCode：机构编码

**返回：**（Code不等于200时为异常）

<Result>

<Code>200</Code>

<Message>successfully!</Message>

</Result>

**抛出：**

Exception

## 获取绑定印章信息接口

public String **getBindedSealInfo**(String certDN, String certSN) throws Exception

**入参：**

certDN：绑定印章的DN

certSN：绑定印章的SN

**正常返回：**（Code不等于200时为异常）

<Result>

<Code>200</Code>

<Message>successfully!</Message>

<List>

<SealInfo>

<SealPerson> 制章人

<SealOrg> 制章单位

<SealName> 印章名称

<SealCode> 印章编码

<Image> 印模图片Base64编码

</SealInfo>

</List>

</Result>

**抛出：**

Exception

## 自助生成图片接口

public String **autoGenerateImage**(String imageStrategyXML) throws Exception

**入参：**

imageStrategyXML：图片生成策略文件，见附件

**正常返回：**

图片的Base64编码

**异常返回：**

<Error>

<ErrorCode>errorCode</ ErrorCode >

<ErrorMessage>errorMessage</ErrorMessage>

</Error>

**抛出：**

Exception

## 获取关键字在PDF中的坐标

public String **getKeywordLocationsInPdf**(byte[] pdf, String keyword) throws Exception

**入参：**

pdf：pdf文件流

keyword：关键字

**正常返回：**（Code不等于200时为异常）

<Result>

<Code>200</Code>

<Message>successfully!</Message>

<List>

<Location>

<PageNo> 页码

<LX> 横坐标（以左下为参考）

<LY> 纵坐标（以左下为参考）

<Width>页面宽度

<Height>页面高度

</Location>

</List>

</Result>

**抛出：**

Exception

## 获取PDF页数总和

public String **getPdfPageCount**(byte[] pdf) throws Exception

**入参：**

pdf：pdf文件流

keyword：关键字

**正常返回：**（Code不等于200时为异常）

<Result>

<Code>200</Code>

<Message>successfully!</Message>

<PageCount>10</PageCount>

</Result>

**抛出：**

Exception

## 撤销PDF印章接口

public byte[] **revokePdfSeal**(byte[] pdf, int count) throws Exception

**入参：**

pdf：pdf文件流

count：需要撤销印章的个数，0：全部撤销，默认：1

**正常返回：**

Pdf文件

**异常返回：**

<Error>

<ErrorCode>errorCode</ ErrorCode >

<ErrorMessage>errorMessage</ErrorMessage>

</Error>

**抛出：**

Exception

# 附件

制章策略文件：

|  |  |  |
| --- | --- | --- |
| <Request> |  |  |
|  | <SealPerson> | 制章人 |
|  | <SealOrg> | 制章单位（制章人所在单位） |
|  | <SealName> | 印章名称 |
|  | <SealCode> | 印章编码 |
|  | <SealPassword> | 印章密码 |
|  | <SealType> | 印章类型，0：企业；1：个人 |
|  | <ImageCode> | 印模编码 |
|  | <OrgCode> | 机构编码 |
|  | <SealFlag> | 印章标识，客户自定义，默认0代表普通印章 |
| </Request> |  |  |

人名章制章策略文件（字体默认为宋体）

|  |  |  |
| --- | --- | --- |
| <Request> |  |  |
|  | <SealPerson> | 制章人 |
|  | <SealOrg> | 制章单位 |
|  | <SealName> | 印章名称 |
|  | <SealCode> | 印章编码 |
|  | <SealPassword> | 印章密码 |
|  | <ImageShape> | 1:方型  3：长方形（不带“章”字） |
|  | <ImageWidth> | 图片宽(默认200) |
|  | <ImageHeight> | 图片高（默认200，长方形默认100） |
|  | <Color> | 十六进制，默认FF0000 |
|  | <FontSize> | 字体大小，默认50 |
|  | <OrgCode> | 机构编码 |
|  | <SealFlag> | 印章标识，客户自定义，默认0代表普通印章 |
| </Request> |  |  |

签章策略文件：

|  |  |  |
| --- | --- | --- |
| <Request> |  |  |
|  | <Type> | 签章类型（不能为空），1=空白标签签章,2=坐标签章,3=关键字签章 |
|  | <PdfURL> | Pdf的URL地址，如果pdf为空时使用 |
|  | <SealCode> | 印章编码（不能为空） |
|  | <SealPassword> | 印章密码（不能为空） |
|  | <SealPerson> | 签章人 |
|  | <SealLocation> | 签章地点 |
|  | <SealReason> | 签章理由 |
|  | <BusinessCode> | 业务码：可以是验证码、查询码 |
|  | <BusinessCodeStyle> | font-size:10;x-ratio:0.5;y-ratio:0.5;color:FF0000; |
|  | <BlankFieldName> | 空白标签字段名，为空时代表全部 |
|  | <Page> | 页数，按坐标签章时不能为空  支持（1）单页，（1,2,3,4）多页,（1-3）  区间，（0）所有页 |
|  | <LX> | 左侧的x坐标 |
|  | <LY> | 左侧的y坐标 |
|  | <Keyword> | 关键字，按关键字签章时不能为空 |
|  | <LocationStyle> | 位置风格：上:U；下:D；左:L；右:R；中:C；默认：C |
|  | <OffsetX> | 横轴偏移，默认为0 |
|  | <OffsetY> | 纵轴偏移，默认为0 |

签章功能策略文件：

|  |  |  |
| --- | --- | --- |
| <Request> |  |  |
|  | <Type> | 签章类型（不能为空），1=空白标签签章,2=坐标签章,3=关键字签章 |
|  | <PdfURL> | Pdf的URL地址，如果pdfBase64为空时使用 |
|  | <SealPerson> | 签章人 |
|  | <SealLocation> | 签章地点 |
|  | <SealResson> | 签章理由 |
|  | <BlankFieldName> | 空白标签字段名，为空时代表全部 |
|  | <Page> | 页数，按坐标签章时不能为空  支持（1）单页，（1,2,3,4）多页,（1-3）  区间，（0）所有页 |
|  | <LX> | 左侧的x坐标 |
|  | <LY> | 左侧的y坐标 |
|  | <Keyword> | 关键字，按关键字签章时不能为空 |
|  | <LocationStyle> | 位置风格：上:U；下:D；左:L；右:R；中:C；默认：C |
|  | <OffsetX> | 横轴偏移，默认为0 |
|  | <OffsetY> | 纵轴偏移，默认为0 |

骑缝章签章策略文件：

|  |  |  |
| --- | --- | --- |
| <Request> |  |  |
|  | <SealCode> | 印章编码（不能为空） |
|  | <SealPassword> | 印章密码（不能为空） |
|  | <SealPerson> | 签章人 |
|  | <SealLocation> | 签章地点 |
|  | <SealResson> | 签章理由 |
|  | <FromPage> | 起始页 |
|  | <ToPage> | 结束页 |
|  | <CrossStyle> | 骑缝章风格  1：上下各一半  2：左右各一半  3：上下骑缝  4：左右骑缝  5：左骑缝  6：右骑缝 |

添加水印策略文件：

|  |  |  |
| --- | --- | --- |
| <Request> |  |  |
|  | <FromPage> | 起始页 |
|  | <ToPage> | 结束页 |
|  | <AbsoluteX> | 水印X坐标 |
|  | <AbsoluteY> | 水印Y坐标 |
|  | <Alpha> | 透明度，取值0-1 |
|  | <FitWidth> | 宽度 |
|  | <FitHeight> | 高度 |
|  | <RotationDegree> | 旋转度 |
|  | <WaterMarkImage> | 水印图片Base64编码 |
|  | <WaterMarkText> | 水印文本 |
|  | <TextFontSize> | 默认12 |
|  | <TextColor> | 默认0000FF |

Web签章策略文件：

|  |  |  |
| --- | --- | --- |
| <Request> |  |  |
|  | <SealCode> | 印章编码（不能为空） |
|  | <SealPassword> | 印章密码（不能为空） |
|  | <SealPerson> | 签章人 |
|  | <SignLocation> | 签章地点 |
|  | <SignReason> | 签章理由 |
| </Request> |  |  |

签名策略文件：

|  |  |  |
| --- | --- | --- |
| <Request> |  |  |
|  | <SealCode> | 印章编码（不能为空） |
|  | <SealPassword> | 印章密码（不能为空） |
|  | <SealFileName> | 签章文件名 |
|  | <SealPerson> | 签章人 |
|  | <SignLocation> | 签章地点 |
|  | <SignReason> | 签章理由 |
| </Request> |  |  |

验章（签）策略文件：

|  |  |  |
| --- | --- | --- |
| <Request> |  |  |
|  | <CertVerifyType > | 0-验章（签），对原文的base64签名或者hash的base64签名均能验过，  1-验章（签），证书是否有效（根据签章时间戳验证签章时间是否有效），  2-验章（签），证书有效期，证书链，  3-验章（签），证书有效期，证书链，CRL |
| </Request> |  |  |

客户端制章策略文件:

|  |  |  |
| --- | --- | --- |
| <Request> |  |  |
|  | <ImageCode > | 印模编号，当印模数据(imageData)为空时使用 |
|  | <SealPerson> | 制章人 |
|  | <SealOrg> | 制章单位 |
|  | <SealName> | 印章名称 |
|  | <SealCode> | 印章编码 |
| </Request> |  |  |

图片生成策略文件：

|  |  |  |
| --- | --- | --- |
| <Request> |  |  |
|  | <ImageName> | 长方形换行字符“##” |
|  | <ImageName2> | 可为空，圆形章底部正文 |
|  | <ImageWidth> | 图片宽 |
|  | <ImageHeight> | 图片高 |
|  | <ImageShape> | 1:方型；11：方形（带框）  2：圆形  3：长方形；31：长方形（带框） |
|  | <Color> | 十六进制，默认FF0000 |
|  | <FontSize> | 字体大小，默认100 |
| <Request> |  |  |