电子签批服务API

主题	Description of electronic signing and approving service
版本	V1.0.1.0
内容	Electronic signing service UOS
创建时间	February 21, 2020
创建人	李昂
更新时间	June 24, 2022

文档变更纪录

更改人	日期	变更内容

文档主要评审意见

产品组

评审人员	日期	意见

QA 组

评审人员	日期	意见

开发平台

开发环境和工具

工具	作用
Java Script	Integration and device service-driven communication

插件环境支持

system	Tongxin UOS, Yinhe Qilin, etc.
browser	Chrome, Firefox, etc
Port number	12349
communication mode	WebSocket

webSocket　通信命令说明

example

---1---

- Establish a webSocket connection

```
/*********************************
* Establish a connection with the signing service
* Parameters: Plug-in service address
* wsUri： “ws://127.0.0.1: 12349/” +flag
* Flag: 1. The PC logo is the connection of the PC-side page. 2. The 2.sign logo is the
  connection of the signing preview page of the signing equipment.
* Return value:
***********************************/
new WebSocket(wsUri);
-----------------------------------------------------------------
```

---2---

- Open the signature preview page.

```
/*********************************
* Open the signature page
* Parameters: messagestr: ″{\″ cmdtype \ ″:\″ openext screen \ ″,\″ message \ ″:{\″ URL \
″:\″ Preview page address of the approved content \ ″}″
* Cmdtype: request type
   Url: preview page address
***********************************/
websocket.send(messageStr);
```

Start signing　　(It is recommended to use the link on the preview page to send the message of opening the signature form.)

```
/*********************************
* Start signing
* Parameters: messagestr: {\ ″cmdtype \″: \ ″getsign \″, \ ″message \″: {\ ″penwidth \″:
8, \ ″pos _ x \″: 250, \ ″pos _ y \″: 200, \.
* Cmdtype: request type
   Penwidth: pen width
   Pos_x,pos_y: coordinate position of signature form.
   Width,height: width and height of signature form.
***********************************/
```

---3---

- Close the signature page

```
/*********************************
* Close the signature preview page.
* Parameters: messagestr: ″{\″ cmdtype \ ″:\″ closeextscreen \ ″,\″ message \ ″:{\″ URL \
″:\″ signature preview page \ ″}}
* Cmdtype: request type
***********************************/
```

The following messages are sent in the same way as opening/closing the signature page, and the messages returned by the service are in JSON format.

Message sent by client: Get serial number.

Serial number: {\"cmdtype\": \"getSerialnum\"}

Fingerprint suppression

Opening/closing the preview page is the same as signing.

Open the fingerprint collection window: {\ "cmdtype \": \ "getfinger \", \ "message \": {\ "pos _ x \": 350, \ "pos _ y \": 250, \ "width \": 256, \.

Close the fingerprint window: {\ "cmdtype \": \ "closefinger \", \ "message \": {\ "URL \": \ "vvv \"}}

Signature+fingerprint

Opening/closing the preview page is the same as signing.

Call the signature function first, then call the fingerprint collection when the signature result is received, and call the signature fingerprint merging command to merge the signature fingerprints after the fingerprint data is received: {\ "cmdtype \": \ "mixsignfinger \", \ "message \": {\ "URL \": \ "vvv \"}}

take a picture

Turn on the camera: {\ "cmdtype \": \ "Preview camera \", \ "Message \": {\ "URL \": \ "vvv \"}}

Taking photos: {\ "cmdtype \": \ "get camera \", \ "message \": {\ "URL \": \ "vvv \"}}

Turn off the camera: {\ "cmdtype \": \ "stop camera \", \ "message \": {\ "URL \": \ "vvv \"}}

Read id card

Start reading: {\ "cmdtype \": \ "getcard \", \ "message \": {\ "URL \": \ "vvv \"}}

Messages received by the client:

Preview the page trigger button (sign link), and get the button ID to trigger the button event according to the returned coordinates.

Button coordinates: {"response": "btninfo", "message": "x: y"}

Serial number:

Serial number: {"response": "serial num", "message": "123456"}

Signature:

Cancel signature: {"response": "sign cancel"}

Signature completion message: {"response": "signbase64", "message": "signature result base64 stream"}

Fingerprint:

Fingerprint collection timeout: automatically close the collection page after 5 seconds.

Signature completion message: {"response": "finger base64", "message": "fingerprint result base64 stream"}

Signature+fingerprint:

Signature/fingerprint as above.

Signature fingerprint merging completed: {"response": "miximgbase64", "message": "signature fingerprint merging result base64 stream"}

Take pictures:

Open the camera to preview the data: {"response": "camera base64", "message": "preview base64 stream"}

Photographing results: {"response": "pic base64", "message": "Photographing results base64 stream"}

Id card:

Id card information: {″response″: ″id card info″, ″message″: {

Address: address, backimgBase64:
back picture, bareheadBase64: ID
photo, birthday: date of birth,
efficient date: effective start
date, expireDate: effective end
date, frontier.
"idnum": "number",
"issue": "issuing authority",
"name": "name"
"nation": "nation"}

The information returned after reading the certificate overtime is empty (put the certificate first and then read it).

Abnormal:

Service exception message: {″response″: ″non-0″, ″message″: ″exception message″}