The JavaScript API

THIS CHAPTER DESCRIBES HOW TO USE THE JAVASCRIPT API.

The JavaScript API, and an included JavaScript-based sample application, are provided as part of the U.are.U for Windows SDK 3.0 and above.

The API provides web-based capture of fingerprints on the Windows platform through the following browsers: Internet Explorer, Microsoft Edge, Google Chrome or Mozilla Firefox. Note that it provides capture only, and does not provide authentication or identification.

This chapter provides a complete description of the methods, events and enumerations of the JavaScript API. For details on installation, setting up the JavaScript development environment, and using the sample application, consult the U.are.U SDK Windows Platform Guide (version 3.0 or later).

Overview

The JavaScript API allows web applications to conveniently and securely acquire fingerprint data from a supported fingerprint reader connected to the user's device. The API allows software developers to:

- Enumerate fingerprint readers
- Select a fingerprint reader to be used with fingerprint capture
- Get the characteristics of a fingerprint reader
- Start fingerprint capture using a selected fingerprint reader
- Stop fingerprint capture
- Receive captured fingerprints in the following formats: PNG image, WSQ, Intermediate and Raw.
- Receive activity notifications from the fingerprint reader
- Receive an indication of the quality of the fingerprint capture
- Monitor device connection and disconnection

Using the fingerprint library

The Fingerprint Library consists of two JavaScript files, referenced through use of the HTML <script> tag. In the example below, the ECMAScript 6 shim is also included.

```
<script src="es6-shim.js"></script>
<script src=" websdk.client.bundle.min.js"></script>
<script src=" fingerprint.sdk.min.js"></script>
```

All the functionality of the Fingerprint Library is accessed with the Fingerprint.WebApi object. You create an instance of Fingerprint.WebApi as shown below.

```
    this.sdk = new Fingerprint.WebApi;
```

Fingerprint.WebApi methods

The table below lists all of the Fingerpirnt.WebApi methods.

Method	Description
`enumerateDevices()`	Returns the list of fingerprint reader identifiers.
`getDeviceInfo()`	Returns information about a fingerprint reader.
`startAcquisition()`	Starts fingerprint capture.
`stopAcquisition`	Stops fingerprint capture.

Each method returns an ECMAScript 6 Promise object. The fulfillment handler receives the result of the method, if any, and the rejection handler receives an instance of the standard Error object.

enumerateDevices()

The enumerateDevices() method returns the list of identifiers for all fingerprint readers connected to the computer.

TypeScript declaration

```
enumerateDevices(): Promise<string[]>;
```

Parameters - None

Returns

Returns an ES6 promise, which gets fulfilled if the list of the readers is received successfully, or rejected otherwise. The fulfillment handler receives an array of strings, each string contains the unique identifier of each reader.

Code Snippet - The following code will print out device UIDs for all connected fingerprint readers.

```
var FingerprintSdkTest = (function () {
    function FingerprintSdkTest() {
        this.sdk = new Fingerprint.WebApi;
    }
FingerprintSdkTest.prototype.getDeviceList = function () {
        return this.sdk.enumerateDevices();
    };
    return FingerprintSdkTest;
})();

window.onload = function () {
    test = new FingerprintSdkTest();
    var allReaders = test.getDeviceList();
      allReaders.then(function (sucessObj) {
        for (i=0;i<sucessObj.length;i++){
            console.log(sucessObj[i]);
        }
    }, function (error){
        console.log(error.message);
    });
}
```

getDeviceInfo()

The getDeviceInfo() method returns information about a connected fingerprint reader.

TypeScript declaration

```
getDeviceInfo(deviceUid: string): Promise<DeviceInfo>;
```

Parameter

Parameter	Description
`deviceUid`	Required. The unique identifier of the fingerprint reader.

Returns

Returns an ES6 promise, which gets fulfilled if the information about the reader is received successfully, or rejected otherwise. The fulfillment handler receives an instance of the DeviceInfo object, which will be described below.

DeviceInfo

The DeviceInfo object contains information about the fingerprint reader.

Parameter	Description
deviceID	A string containing the unique identifier of the fingerprint reader.
eUidType	One of the DeviceUidType enumeration values that specifies the type of the unique identifier of the fingerprint reader.
eDeviceModality	One of the DeviceModality enumeration values that specifies the capture process used by the fingerprint reader.
eDeviceTech	One of the DeviceTechnology enumeration values that specifies the fingerprint reader technology.

Code **Snippet**

The following code will print out device information for all connected fingerprint readers (enumerated devices).

```
var FingerprintSdkTest = (function () {
    function FingerprintSdkTest() {
        this.sdk = new Fingerprint.WebApi;
    }
    FingerprintSdkTest.prototype.getDeviceList = function () {
        return this.sdk.enumerateDevices();
    };
    FingerprintSdkTest.prototype.getDeviceInfoWithID = function (uid) {
        var _instance = this;
        return  this.sdk.getDeviceInfo(uid);
    };
    return FingerprintSdkTest;
})();

window.onload = function () {
    test = new FingerprintSdkTest();
    var allReaders = test.getDeviceList();
      allReaders.then(function (sucessObj) {
        for (i=0;i<sucessObj.length;i++){
            printDeviceInfo(sucessObj[i]);
        }
    }, function (error){
        console.log(error.message);
    });
}

function printDeviceInfo(uid){
    var myDeviceVal = test.getDeviceInfoWithID(uid);
    myDeviceVal.then(function (sucessObj) {
            console.log(sucessObj.DeviceID); //A string containing the unique
identifier of the fingerprint reader.
            console.log(Fingerprint.DeviceTechnology[sucessObj.eDeviceTech]); // One of
the DeviceUidType enumeration values that specifies the type of the unique identifier
of the fingerprint reader.
```

```
            console.log(Fingerprint.DeviceModality[sucessObj.eDeviceModality]); // One
of the DeviceModality enumeration values that specifies the capture process used by the
fingerprint reader.
            console.log(Fingerprint.DeviceUidType[sucessObj.eUidType]); // One of the
DeviceTechnology enumeration values that specifies the fingerprint reader technology.

        }, function (error){
            console.log(error.message);
        });
}
```

StartAcquisition()

The startAcquisition() method starts fingerprint capture on the client computer.

TypeScript declaration

```
    startAcquisition(sampleFormat: SampleFormat, deviceUid?: string): Promise<void>;
```

Parameter	Description
`sampleFormat`	Required. One of the SampleFormat enumeration values that specifies the format of fingerprint data to be returned.
`deviceUid`	Optional. The unique identifier of the fingerprint reader. If not specified, the capture is performed on all available fingerprint readers.

Returns

Returns an ES6 promise, which gets fulfilled if the fingerprint capture operation is started successfully, or rejected otherwise.

Code Snippet - The following code will start capture mode.

```
var FingerprintSdkTest = (function () {
    function FingerprintSdkTest() {
        this.sdk = new Fingerprint.WebApi;
    }

   FingerprintSdkTest.prototype.startCapture = function () {
        this.sdk.startAcquisition(Fingerprint.SampleFormat.PngImage).then(function () {
            console.log("You can start capturing !!!");
        }, function (error) {
            console.log(error.message);
        });
    };

    return FingerprintSdkTest;
})();

window.onload = function () {
    test = new FingerprintSdkTest();
    test.startCapture();
}
```

stopAcquisition

The stopAcquisition() method stops the previously started fingerprint capture on the client computer.

TypeScript declaration

```
    stopAcquisition(deviceUid?: string): Promise<void>;
```

Parameter	Description
deviceUid	Optional. The unique identifier of the fingerprint reader. If not specified, the capture on all available fingerprint readers will be stopped.

Returns an ES6 promise, which gets fulfilled if the fingerprint capture operation is stopped successfully, or rejected otherwise.

Code Snippet - The following code will stop capture mode.

```
var FingerprintSdkTest = (function () {
    function FingerprintSdkTest() {
        this.sdk = new Fingerprint.WebApi;
    }

 FingerprintSdkTest.prototype.stopCapture = function () {
        this.sdk.stopAcquisition().then(function () {
            console.log("Capturing stopped !!!");
        }, function (error) {
            showMessage(error.message);
        });
    };


    return FingerprintSdkTest;
})();

window.onload = function () {
    test = new FingerprintSdkTest();
    test.stopCapture();
}
```

Fingeprint.WebApi events

The table below lists the Fingerprint.WebApi events.

Parameter	Description
DeviceConnected	A fingerprint reader is connected to the client computer.
DeviceDisconnected	A fingerprint reader is disconnected from the client computer.
SamplesAcquired	Fingerprint data has been captured.
QualityReported	Fingerprint image quality data has been reported.
ErrorOccurred	An error occurred during fingerprint capture.
CommunicationFailed	An error occurred during communication with the U.are.U SDK

DeviceConnected

The DeviceConnected event is fired when a fingerprint reader is connected to the client computer.

Property	Description
`deviceUid`	A string containing the unique identifier of the fingerprint reader.

DeviceDisconnected

The DeviceDisconnected event is fired when a fingerprint reader is disconnected from the client computer.

Property	Description
`deviceUid`	A string containing the unique identifier of the fingerprint reader.

SamplesAcquired

The SamplesAcquired event is fired when fingerprint data is captured.

Property	Description
`deviceUid`	A string containing the unique identifier of the fingerprint reader.
`SampleFormat`	One of the SampleFormat enumeration values that specifies the format of the returned fingerprint data.
`samples`	A string containing a serialized JSON array of fingerprint samples. This string can be base64url encoded and passed directly to the Altus Web server components (aka Altus Confirm) authentication service as the fingerprint credential.

QualityReported

The QualityReported event is fired when fingerprint quality advice is reported.

Property	Description
`deviceUid`	A string containing the unique identifier of the fingerprint reader.
`quality`	One of the QualityCode enumeration values that specifies advice regarding the quality of the scanned fingerprint image.

ErrorOccured

The ErrorOccured event is fired when an error occurred during fingerprint capture.

Property	Description
`deviceUid`	A string containing the unique identifier of the fingerprint reader.
`error`	A number containing the HRESULT error code.

CommunicationFailed

The CommunicationFailed event is fired when an error occurred during communication with the U.are.U SDK.

Properties - None.

Remarks

The CommunicationFailed event usually indicates that the required U.are.U SDK components are not running on the client computer. Typically, the end-user of your application should be given instructions on installing the required software.

Enumerations

DeviceUidType

The DeviceUidType enumeration lists types of unique identifiers for fingerprint readers.

```
enum DeviceUidType {
    Persistent = 0,
    Volatile = 1,
}
```

DeviceModality

The DeviceModality enumeration lists possible types of fingerprint readers relating to their capture process.

```
enum DeviceModality {
    Unknown = 0,
    Swipe = 1,
    Area = 2,
    AreaMultifinger = 3,
}
```

DeviceTechnology

The DeviceTechnology enumeration lists types of fingerprint reader technologies.

```
enum DeviceTechnology {
    Unknown = 0,
    Optical = 1,
    Capacitive = 2,
    Thermal = 3,
    Pressure = 4,
}
```

SampleFormat

The SampleFormat enumeration lists formats of fingerprint data supported by the SDK. See page *40* for descriptions and JSON representation of each format.

```
enum SampleFormat {
    Raw = 1,
    Intermediate = 2,
    Compressed = 3,
    PngImage = 5,
}
```

QualityCode

The QualityCode enumeration lists possible responses relating to the quality of the scanned fingerprint image.

```
enum QualityCode {
    Good = 0,
    NoImage = 1,
    TooLight = 2,
    TooDark = 3,
    TooNoisy = 4,
    LowContrast = 5,
    NotEnoughFeatures = 6,
    NotCentered = 7,
    NotAFinger = 8,
    TooHigh = 9,
    TooLow = 10,
    TooLeft = 11,
    TooRight = 12,
    TooStrange = 13,
    TooFast = 14,
    TooSkewed = 15,
    TooShort = 16,
    TooSlow = 17,
    ReverseMotion = 18,
    PressureTooHard = 19,
    PressureTooLight = 20,
    WetFinger = 21,
    FakeFinger = 22,
    TooSmall = 23,
    RotatedTooMuch = 24,
}
```

Sample Format details

Fingerprint data can be exported in one of the following formats.

- **Raw** - Data format is a raw (unprocessed) biometric sample, also referred to as a Fingerprint Image in most biometric documentation.
- **Intermediate** - Data format is a partially processed biometric sample, also referred to as a Feature Set in most biometric documentation.
- **Compressed** - Data format is a fully processed and compressed (WSQ) biometric sample, also referred to as a Fingerprint Template in most biometric documentation.
- **PNGImage** - Data format is a .PNG image file.

JSON representations of each of the supported export formats are shown below.

Raw

```
{
deviceUid:"30323030-3661-6533-3764-393600000000"
sampleFormat:1
samples:"[{
    "Data":"{
        "Compression":0,
        "Data":"8PDw88PDw8PDw8PDw8PDw8PA",
        // Base64url encoded image
        "Format":{
            "iHeight":403,
            "iWidth":200,
            "iXdpi":508,
            "iYdpi":508,
            "uBPP":8,
            "uDataType":1,
            "uImageType":2,
            "uPadding":2,
            "uPlanes":1,
            "uPolarity":2,
            "uRGBcolorRepresentation":0,
            "uSignificantBpp":8
            },
        "Header":{
            "DeviceId":1407938095516745728,
            "DeviceType":49264417346420736,
            "iDataAcquisitionProgress":100,
            "uDataType":1
            },
        "Version":1
        }",
    "Header":{
    "Encryption":0,
    "Factor":8,
    "Format":{
        "FormatID":0,
        "FormatOwner":51
        },
    "Purpose":0,
    "Quality":-1,
    "Type":1},
    "Version":1
    }]"
type:"SamplesAcquired"
}
```

Intermediate

```
{
deviceUid:"30323030-3661-6533-3764-393600000000"
sampleFormat:2
samples:"[{
    "Data":"eyJ0eXAiOiJKV1QiLA0KICJhbGciOiJIUzI1NiJ9",
    //Base64url encoded Feature Set
    "Header":{
        "Encryption":0,
        "Factor":8,
        "Format":{
          "FormatID":0,
          "FormatOwner":51
          },
        "Purpose":0,
        //0 - DP_PURPOSE_ANY, meaning "Data" field is
        //equivalent to DP_PRE_REG/DP_VERIFICATION FMD and
        //can be used for both fingerprint enrollment or
        //fingerprint verification/identification purposes.
        "Quality":-1,
        "Type":2
        },
    "Version":1
    }]"
type:"SamplesAcquired"
}
```

Compressed (WSQ)

```
{
deviceUid:"30323030-3661-6533-3764-393600000000"
sampleFormat:3
samples:"[{
    "Data":"{
        "Compression":2,
        "Data":"_6DqAB6TklTPFHxflOj0_6E",
        // Base64url encoded image
        "Format":{
            "iHeight":403,
            "iWidth":200,
            "iXdpi":508,
            "iYdpi":508,
            "uBPP":8,
            "uDataType":1,
            "uImageType":2,
            "uPadding":2,
            "uPlanes":1,
            "uPolarity":2,
            "uRGBcolorRepresentation":0,
            "uSignificantBpp":8},
            "Header":{
                "DeviceId":1407938095516745728,
                "DeviceType":4926441734620736,
                "iDataAcquisitionProgress":100,
                "uDataType":1
                },
            "Version":1
        }",
    "Header":{
        "Encryption":0,
        "Factor":8,
        "Format":{
            "FormatID":0,
            "FormatOwner":51
            },
        "Purpose":0,
        "Quality":-1,
        "Type":1
        },
    "Version":1
    }]"
type:"SamplesAcquired"
```