JuBiter SDK Manual for Android

V1.0

2019/9/29

FEITIAN Technologies Co., Ltd.

Website: www.FTsafe.com

Website: www.jubiterwallet.com

Table Content

1. Interface for getting device properties 2
2. Interface for getting device certificate 2
3. Interface for sending an APDU directly 2
4. Interface for checking device initialization 2
5. Interface for checking BootLoader mode 2
6. Interface for enumerating applets 3
7. Interface for enumerating supported coins 3
8. Interface for getting applet version 3
9. Interface for creating a context to operate BTC series coins - Hardware 3
10. Interface for creating a context to operate BTC series coins - Software 3
11. Interface for getting HDNode public key of a BTC series coin 4
12. Interface for getting the public key of the current BTC context 4
13. Interface for getting one address of a BTC series coin 4
14. Interface for setting a BTC quick payment address on the device 4
15. Interface for setting the unit of BTC shown on the device 5
16. Interface for signing one BTC transaction 5
17. Interface for generating the output of an USDT transaction 5
18. Interface for creating a context to operate ETH series coins - Hardware 5
19. Interface for creating a context to operate ETH series coins - Software 6
20. Interface for getting one address of a ETH series coin 6
21. Interface for getting HDNode public key of a ETH series coin 6
22. Interface for getting the main public key of the current ETH context 6
23. Interface for setting a ETH quick payment address on the device 7
24. Interface for signing one ETH transaction 7
25. Interface for generating inputs of ERC-20 token transaction 7
26. Interface for showing the nine-square PIN matrix on the JuBiter device 7
27. Interface for cancel the display of the nine-square PIN matrix 7
28. Interface for PIN verification 8
29. Interface for clearing a context 8
30. Interface for setting transaction timeout 8
31. Interface for generating mnemonic 8
32. Interface for checking mnemoric 8
33. Interface for generating seed 9
34. Interface for generating main private key 9
33. Interface for initializing the communication library of JuBiter device 9
34. Interface for enumerating JuBiter devices 9
35. Interface for stopping enumerating devices 10
36. Interface for connecting to a JuBiter device 10
37. Interface for canceling a connection 10
38. Interface for disconnecting a device 10
39. Interface for checking the connection state 10
40. Interface for getting device power 11

1. Interface for getting device properties

CommonProtos.ResultAny getDeviceInfo(int deviceID)	
Description	Get the hardware information of the current JuBiter device.
IN	deviceID: This is the device id returned from the connectDevice function. All subsequent device operations rely on this value.
OUT	Info: the data structure of the device info, includes: Label: Device label, set when JuBiter device is produced. Sn: The serial number of JuBiter device, set when JuBiter device is produced. pin_retry: Current password retries pin_max_retry: Password maximum retries ble_version: Bluetooth firmware version firmware_version: The corn firmware version/The JAVA COS version
Additional Information	Read Only

2. Interface for getting device certificate

CommonProtos.ResultString getDeviceCert(int deviceID)	
Description	Get the device certificate of one JuBiter Blade
IN	deviceID: This is the device id returned from the connectDevice function. All subsequent device operations rely on this value.
OUT	cert: This is the origin device certificate signed by Jubiter root certificate, which can be verified by the Jubiter server for authenticity of the device.
Additional Information	Device certificate format conforms to GP specification.

3. Interface for sending an APDU directly

CommonProtos.ResultString sendApdu(int deviceID, String apdu)	
Description	Send an APDU command directly
IN	deviceID: This is the device id returned from the connectDevice function. All subsequent device operations rely on this value. apdu: APDU command in Hex String
OUT	Response: response in Hex String
Additional Information	

4. Interface for checking device initialization

boolean isInitialize(int deviceID)	
Description	Check if the JuBiter device has generated mnemonics
IN	deviceID: This is the device id returned from the connectDevice function. All subsequent device operations rely on this value.
OUT	true or false
Additional Information	For security reasons, the process of generating mnemonics is done entirely on the JuBiter Blade device, and the software is not involved in all the processes of generating mnemonics. This interface is used to determine if the hardware has gone through this process. The generation of mnemonics follows the BIP39 specification.

5. Interface for checking BootLoader mode

boolean isBootLoader(int deviceID)	
Description	Check if the hardware is in BootLoader mode.

IN	deviceID: This is the device id returned from the connectDevice function. All subsequent device operations rely on this value.
OUT	true or false
Additional Information	

6. Interface for enumerating applets

`CommonProtos.ResultString enumApplets(int deviceID)`	
Description	Enumerate the applets already installed on the current JuBiter device
IN	deviceID: This is the device id returned from the connectDevice function. All subsequent device operations rely on this value.
OUT	appList: list all applet IDs separated by spaces
Additional Information	The Jubiter Blade device uses the Java card architecture, which each applet corresponds to one series of cryptocurrencies.

7. Interface for enumerating supported coins

`CommonProtos.ResultString enumSupportCoins(int deviceID)`	
Description	Enumerate all the main coins supported by the current JuBiter device, without ERC-20 tokens
IN	deviceID: This is the device id returned from the connectDevice function. All subsequent device operations rely on this value.
OUT	A list of main coins separated by spaces
Additional Information	

8. Interface for getting applet version

`CommonProtos.ResultString getAppletVersion(int deviceID, String appletID)`	
Description	Get the version of an applet
IN	deviceID: This is the device id returned from the connectDevice function. All subsequent device operations rely on this value. appID: One of the applet IDs returned from enumApplets function
OUT	version: version of applet
Additional Information	

9. Interface for creating a context to operate BTC series coins - Hardware

`CommonProtos.ResultInt createContext(BitcoinProtos.ContextCfgBTC config, int deviceID)`	
Description	Create a context for operating the BTC series coins for subsequent BTC related operations on JuBiter hardware wallet.
IN	deviceID: This is the device id returned from the connectDevice function. All subsequent device operations rely on this value. config: The configurations for creating a context
OUT	contextID: Generate a context ID
Additional Information	Recently, this interface is available for BTC, LTC, BCH, and USDT.

10.Interface for creating a context to operate BTC series coins - Software

`CommonProtos.ResultInt createContext_Software(BitcoinProtos.ContextCfgBTC config, String xPrikey)`	
Description	Create a context for operating the BTC series coins for subsequent BTC related operations on JuBiter software wallet APP.
IN	config: The configurations for creating a context

	xPrikey: The private key corresponding to the wallet to be operated
OUT	contextID: Generate a context ID
Additional Information	Recently, this interface is available for BTC, LTC, BCH, and USDT.

11. Interface for getting HDNode public key of a BTC series coin

```
CommonProtos.ResultString getHDNode(int contextID, CommonProtos.Bip32Path bip32)
```

Description	Get a HDNode public key
IN	contextID: the context ID returned from createContext or createContext_Software bip32: the standard path conforming to the bip44 specification
OUT	Xpub: a public key in XPUB format, includes chaincode, fingerprint, and so on.
Additional Information	

12. Interface for getting the public key of the current BTC context

```
CommonProtos.ResultString getMainHDNode(int contextID)
```

Description	Get the public key of the current context
IN	contextID: the context ID returned from createContext or createContext_Software
OUT	Xpub: a public key in XPUB format, includes chaincode, fingerprint, and so on.
Additional Information	Get the xpub public key of the main_path, which is specified by config of the createContext function. If the main_path is specified to the account level, the ordinary subkeys can be derived from this xpub. All the harden subkeys are generated by JuBiter device, which can effectively reduce hardware and software communication.

13. Interface for getting one address of a BTC series coin

```
CommonProtos.ResultString getAddress(int contextID, CommonProtos.Bip32Path bip32,
Boolean isShow)
```

Description	Get one address of a BTC series coin
IN	contextID: the context ID returned from createContext or createContext_Software bip32: the standard path conforming to the bip44 specification isShow: Displayed on the device screen or not
OUT	address: address of a BTC series coin in Base58 format
Additional Information	This interface is for users to confirm the payment address on a JuBiter device, in case that the smart phone shows a wrong address after the software is hacked. This interface would be blocked when the JuBiter device shows the address. The interface returns a signal after the user presses the confirmation button on the device.

14. Interface for setting a BTC quick payment address on the device

```
CommonProtos.ResultString setAddress(int contextID, CommonProtos.Bip32Path bip32)
```

Description	Set a BTC address for quick payment on the current JuBiter device
IN	contextID: the context ID returned from createContext or createContext_Software bip32: the standard path conforming to the bip44 specification
OUT	address: the set address in Base58 format
Additional Information	The Jubiter Blade device allows users to set up a quick payment address. It is safe and convenient to display the text and QR code of this quick payment address on the Jubiter Blade device without connecting to any other equipment.

	This interface requires a verification of PIN code.

15. Interface for setting the unit of BTC shown on the device

```
int setUint(int contextID, BitcoinProtos.BTC_UINT_TYPE uintType)
```

Description	Set the unit of BTC shown on the current JuBiter device during transactions
IN	contextID: the context ID returned from createContext or createContext_Software unit: the unit of Enum, check Jub_SDK.h for detailed information
OUT	Null
Additional Information	The default unit is mBTC.

16. Interface for signing one BTC transaction

```
CommonProtos.ResultString signTransaction(int contextID, BitcoinProtos.TransactionBTC txInfo)
```

Description	Sign one BTC transaction
IN	contextID: the context ID returned from createContext or createContext_Software txInfo: the detailed information of a transaction
OUT	raw: This is the signed transactions that can be used directly for broadcasting. If the user cancels the transaction, it returns an empty string.
Additional Information	This interface is blocked when the JuBiter device shows the transaction information for a user to check and confirm it. When the Jubiter device signs the transaction, it would verify whether the specified change address in the outputs is the address set in the device. If not, the device would report an error. If correct, the amount of this output will not be displayed in the transfer amount of the transaction information. It is safe, clear and correct to show the user's real transaction amount. For security reasons, the Jubiter device would only use the hash_all method for signing a transaction. This interface requires a verification of PIN code.

17. Interface for generating the output of an USDT transaction

```
CommonProtos.ResultAny buildUSDTOutput(int contextID, String usdtTo, long amount)
```

Description	Generate outputs conforming to the onmi specification
IN	contextID: the context ID returned from createContext or createContext_Software Amount: the USDT amount of the transaction
OUT	Generate outputs for signTransaction function
Additional Information	This is an auxiliary interface with no need to call the JuBiter device.

18. Interface for creating a context to operate ETH series coins - Hardware

```
CommonProtos.ResultInt createContext(EthereumProtos.ContextCfgETH config, int deviceID)
```

Description	Create a context for operating the ETH series coins for subsequent ETH related operations on JuBiter hardware device.
IN	deviceID: This is the device id returned from the connectDevice function. All subsequent device operations rely on this value. config: The configurations for creating a context
OUT	contextID: Generate a context ID

Additional Information	This interface is available for ETH and ETC.

19. Interface for creating a context to operate ETH series coins - Software

CommonProtos.ResultInt createContext_Software(EthereumProtos.ContextCfgETH config, int deviceID)	
Description	Create a context for operating the ETH series coins for subsequent ETH related operations on JuBiter software wallet APP.
IN	deviceID: This is the device id returned from the connectDevice function. All subsequent device operations rely on this value. config: The configurations for creating a context
OUT	contextID: Generate a context ID
Additional Information	This interface is available for ETH and ETC.

20. Interface for getting one address of a ETH series coin

CommonProtos.ResultString getAddress(int contextID, CommonProtos.Bip32Path bip32, boolean isShow)	
Description	Get one address of a ETH series coin
IN	contextID: the context ID returned from createContext or createContext_Software bip32: the standard path conforming to the bip44 specification isShow: Displayed on the device screen or not
OUT	address: address of a ETH series coin in Hex format, leading with 0x.
Additional Information	Similar to section 13 above.

21. Interface for getting HDNode public key of a ETH series coin

CommonProtos.ResultString getHDNode(int contextID, EthereumProtos.ENUM_PUB_FORMAT format, CommonProtos.Bip32Path bip32)	
Description	Get a HDNode public key
IN	contextID: the context ID returned from createContext or createContext_Software format: Indicates the encoding format of the public key, hex or xpub. bip32: the standard path conforming to the bip44 specification
OUT	A public key in XPUB or hex format
Additional Information	

22. Interface for getting the main public key of the current ETH context

CommonProtos.ResultString getMainHDNode(int contextID, EthereumProtos.ENUM_PUB_FORMAT format)	
Description	Get the main public key of the current ETH context
IN	contextID: the context ID returned from createContext or createContext_Software format: Indicates the encoding format of the public key, hex or xpub.
OUT	A public key in XPUB or hex format
Additional Information	

23. Interface for setting a ETH quick payment address on the device

CommonProtos.ResultString setAddress(int contextID, CommonProtos.Bip32Path bip32)	
Description	Set a ETH address for quick payment on the current JuBiter device
IN	contextID: the context ID returned from createContext or createContext_Software bip32: the standard path conforming to the bip44 specification
OUT	address: the set address in hex format leading with 0x
Additional Information	The Jubiter Blade device allows users to set up a quick payment address for ETH coin. It is safe and convenient to display the text and QR code of this quick payment address on the Jubiter Blade device without connecting to any other equipment. This interface requires a verification of PIN code.

24. Interface for signing one ETH transaction

CommonProtos.ResultString signTransaction(int contextID, EthereumProtos.TransactionETH txInfo)	
Description	Sign one ETH transaction
IN	contextID: the context ID returned from createContext or createContext_Software txInfo: the detailed information of a transaction
OUT	The signed raw tx can be used directly for broadcasting.
Additional Information	This interface is blocked when the JuBiter device shows the transaction information for a user to check and confirm it. This interface requires a verification of PIN code.

25. Interface for generating inputs of ERC-20 token transaction

CommonProtos.ResultString buildERC20Abi(int contextID, String address, String amountInWei)	
Description	Generate inputs of ERC-20 token transaction
IN	contextID: the context ID returned from createContext or createContext_Software address: the ‘to’ address of the ERC-20 transaction AmountInWei: the amount of the ERC-20 token transaction
OUT	Generate inputs for signTransaction function
Additional Information	This is an auxiliary interface with no need to call the JuBiter device.

26. Interface for showing the nine-square PIN matrix on the JuBiter device

int showVirtualPWD(int contextID)	
Description	Call the current JuBiter device to show a random nine-square PIN matrix on its screen for PIN verification
IN	contextID: the context ID returned from createContext or createContext_Software
OUT	NULL
Additional Information	Jubiter Blade device uses international standard true random number generator.

27. Interface for cancel the display of the nine-square PIN matrix

int cancelVirtualPWD(int contextID)	
Description	Call the current JuBiter device to cancel the display of the random nine-square PIN

	matrix on its screen
IN	contextID: the context ID returned from createContext or createContext_Software
OUT	NULL
Additional Information	

28.Interface for PIN verification

```
CommonProtos.ResultInt verifyPIN(int contextID, String PIN)
```

Description	Verify the user PIN
IN	contextID: the context ID returned from createContext or createContext_Software PIN: The out-of-order password entered by the user, in the order of 1 2 3 4 5 6 7 8 9
OUT	NULL
Additional Information	If the PIN is verified, the random nine-square PIN matrix on the JuBiter device would disappear, and the PIN permission will be authorized to the next context operation. If the PIN verification failed, the random nine-square PIN matrix on the JuBiter device would be reordered.

29.Interface for clearing a context

```
int clearContext(int contextID)
```

Description	Clear/Destroy one context
IN	contextID: the context ID returned from createContext or createContext_Software
OUT	Null
Additional Information	

30.Interface for setting transaction timeout

```
CommonProtos.ResultString setTimeout(int contextID, int timeout)
```

Description	Set the timeout of showing a transaction information on the JuBiter device
IN	contextID: the context ID returned from createContext or createContext_Software timeout: Timeout in seconds
OUT	Null
Additional Information	The default timeout is 120 seconds.

31.Interface for generating mnemonic

```
CommonProtos.ResultString generateMnemonic(CommonProtos.ENUM_MNEMONIC_STRENGTH strength)
```

Description	Generate the mnemonic with the specified strength
IN	strength: the strength or length or the mnemonic to be generated
OUT	The generated mnemonic
Additional Information	

32.Interface for checking mnemoric

```
int checkMnemonic(String mnemonic)
```

Description	Check the mnemonic
IN	mnemonic: mnemonic/recovery seed/recovery phrase
OUT	NULL
Additional Information	

33.Interface for generating seed

CommonProtos.ResultString generateSeed(String mnemonic, String passphrase)	
Description	Generate the only seed from the mnemonic
IN	mnemonic: mnemonic/recovery seed/recovery phrase passphrase: the passphrase set by user
OUT	The only seed
Additional Information	

34.Interface for generating main private key

CommonProtos.ResultString seedToMasterPrivateKey(String seed, CommonProtos.CURVES curve)	
Description	Generate the main private key from the seed
IN	seed: the only seed curve: the specified elliptic curve
OUT	The main private key
Additional Information	

33.Interface for initializing the communication library of JuBiter device

int initDevice()	
Description	Initialize the communication library of the current JuBiter device
IN	NULL
OUT	NULL
Additional Information	

34.Interface for enumerating JuBiter devices

void startScan(ScanResultCallback callback)	
Description	Scan with BLE and enumerate all JuBiter Blade deivces around
IN	callback: Enumeration result callback. If there are multiple devices around, the callback will be called multiple times
OUT	onScanResult: The result returned by this callback onStop: The result returned after the enumeration stops onError: The result returned after an exception occurs
Additional Information	This interface is an asynchronous operation, and the enumeration result is returned in scanCallBack function.

35. Interface for stopping enumerating devices

`int stopScan()`	
Description	Stop scanning and enumerating the devices around
IN	`NULL`
OUT	`NULL`
Additional Information	

36. Interface for connecting to a JuBiter device

`void connectDeviceAsync(String address, int timeout, ConnectionStateCallback callback)`	
Description	Connect to a specified JuBiter Blade devices
IN	`address:` the device MAC address `timeout:` the timeout of this connection
OUT	the handle of the connected JuBiter device
Additional Information	

37. Interface for canceling a connection

`int cancelConnect(String address)`	
Description	Cancel the current connecting operation when the device is in the process of connecting but has not succeeded or failed.
IN	`address:` the device MAC address
OUT	NULL
Additional Information	

38. Interface for disconnecting a device

`int disconnectDevice(int deviceID)`	
Description	disconnect a specified JuBiter Blade devices
IN	`deviceID:` the device handle
OUT	NULL
Additional Information	The disconnection state would be updated in ConnectionStateCallback function.

39. Interface for checking the connection state

`boolean isConnected(int deviceID)`	
Description	Check whether a specified JuBiter devices is being connected through BLE.
IN	`deviceID:` the device handle
OUT	`true` or `false`
Additional Information	True means connected and false means disconnected.

40.Interface for getting device power

CommonProtos.ResultInt queryBattery(int deviceID)	
Description	Get the battery power of the current JuBiter device
IN	`deviceID:` the device handle
OUT	Returns the amount of battery power displayed in decimal percent, for example 48 corresponding to battery power 48%
Additional Information	