Oneplusone Scale Interface Software

V2.9

[Operation Manual](javascript:;)

|  |
| --- |
| **Xiamen Oneplusone Intelligent Technology Co.,Ltd.** |

Address: 5th Floor, No.8, Gaoqi South 12th Road, Huli District, Xiamen City, Fujian Province.

Phone: 0592-5550806

Fax:0592-5885992

Website: [www.oneplusone.cc](http://www.oneplusone.cc)

|  |
| --- |
|  |

**1.Description**

Xiamen Oneplusone Intelligent Technology Co.,Ltd. is a privately operated high-tech enterprise specialized in commercial weighing apparatus, intelligent POS terminal and other commercial equipment. The company integrates product design, development, production, sales and service. It has a professional product research and development team, comprehensive lean production management and perfect sales service system, which can quickly respond and meet the different needs of customers for products. The dynamic library **SyncSDK.dll** supports the electronic scale products produced by Oneplusone. This document describes the interfaces of the SDK, and how to use these interfaces to interact with electronic scales, helping users and developers to use electronic scales more easily. .

1. **Interface Description**

**2.1 Interface Statement**

The programming language used in the following interface statement is Pascal.

Interface list:

function SDK\_Initialize: Boolean; stdcall;

procedure SDK\_Finalize; stdcall;

function SDK\_GetDeviceInfo(Addr: Cardinal): TSDKDeviceInfo; stdcall;

function SDK\_GetNetworkSectionDevicesInfo(Addr: Cardinal; DeviceInfos: Pointer;

Count: Cardinal): Integer; stdcall;

function SDK\_ExecTaskA(Addr, ProcType, DataType: Cardinal; FileName: PAnsiChar;

OnProgress: TSDKOnProgressEvent; UserData: Pointer): TSDKTaskHandle; stdcall;

function SDK\_ExecTask(Addr, ProcType, DataType: Cardinal; FileName: PWideChar;

OnProgress: TSDKOnProgressEvent; UserData: Pointer): TSDKTaskHandle; stdcall;

function SDK\_ExecTaskW(Addr, ProcType, DataType: Cardinal; FileName: PWideChar;

OnProgress: TSDKOnProgressEvent; UserData: Pointer): TSDKTaskHandle; stdcall;

function SDK\_GetLastTaskError: Integer; stdcall;

procedure SDK\_StopTask(TaskHandle: THandle = 0); stdcall;

procedure SDK\_WaitForTask(TaskHandle: THandle); stdcall;

* + 1. **Dynamic library initialization**

function SDK\_Initialize: Boolean; stdcall;

Function:dynamic library initialization

Parameter: no

Returned value: Bool type. True signifies the success, False signifies the failure.

* + 1. **Release Dynamic Library**

procedure SDK\_Finalize; stdcall;

Function: release dynamic library

Parameter: no

Returned value: no

* + 1. **Obtain Device Information**

function SDK\_GetDeviceInfo(Addr: Cardinal): TSDKDeviceInfo; stdcall;

Function: obtain device information

Parameter: Addr: the device IP address must be converted to Cardinal.

Returned value: device information. You should get the definition of TSDKDeviceInfo by references of 2.2.1 Device Information.

* + 1. **Search for Devices in the Network Segment**

function SDK\_GetNetworkSectionDevicesInfo(Addr: Cardinal; DeviceInfos: Pointer;

Count: Cardinal): Integer; stdcall;

Function: search for devices in the network segment

Parameter:

Addr: the device IP address must be converted to Cardinal.

DeviceInfos: Pointer. Device information Array, array. of TSDKDeviceInfo, you need to apply for array space.

Count: DeviceInfos array size.

Returned value : the quantity of searched devices

**2.1.5 Perform the Task Based on the Parameters**

The following three interfaces have the same function, the only difference is the format of the parameter FileName.

function SDK\_ExecTaskA(Addr, ProcType, DataType: Cardinal; FileName: PAnsiChar;

OnProgress: TSDKOnProgressEvent; UserData: Pointer): TSDKTaskHandle; stdcall;

function SDK\_ExecTask(Addr, ProcType, DataType: Cardinal; FileName: PWideChar;

OnProgress: TSDKOnProgressEvent; UserData: Pointer): TSDKTaskHandle; stdcall;

function SDK\_ExecTaskW(Addr, ProcType, DataType: Cardinal; FileName: PWideChar;

OnProgress: TSDKOnProgressEvent; UserData: Pointer): TSDKTaskHandle; stdcall;

Function: perform the task based on the parameters

Parameter:

Addr: the device IP address must be converted to Cardinal.

ProcType: operation type. You should see 2.2.2 operation type for more details.

DataType: data type. You should see 2.2.3 data type for more details.

FileName: file name. SDK\_ExecTaskA is declared as PAnsiChar. SDK\_ExecTask and SDK\_ExecTaskW are declared as PWideChar.

OnProgress: callback function on progress. The format is seen at 2.xxxxx.

UserData: user pointer. For users to play freely.

Returned value: task pointer.

**2.1.6 Gain the Last Error Number**

function SDK\_GetLastTaskError: Integer; stdcall;

Function: gain the last error number

Parameter: no

Returned value: error number

**2.1.7 Stop the Task**

function SDK\_GetLastTaskError: Integer; stdcall;

Function: stop the task

Parameter: TaskHandle: SDK ExecTask Return task pointer. TaskHandle-0 signifies stopping all ongoing tasks.

Returned value: no

**2.1.8 Wait for the Accomplishment of Task**

procedure SDK\_WaitForTask(TaskHandle: THandle); stdcall;

Function: wait for the accomplishment of task

Parameter: TaskHandle:SDK ExecTask Return task pointer.

Returned value: no

* 1. **The Definition of Interface Parameter**

The programming language for the following type declarations is Pascal.

* + 1. **Device Information**

The device information is 256 bytes in total.

TSDKDeviceInfo = packed Record

Addr: UInt32;

Port: UInt32;

ProtocolType: UInt32;

DeviceNo: Array [0 .. 15] of Byte;

Version: UInt32;

LanguageID: UInt8;

KeyID: UInt8;

PLUStorage: UInt16;

Note1Storage: UInt16;

Note2Storage: UInt16;

Note3Storage: UInt16;

Note4Storage: UInt16;

PrinterKm: Double;

PrinterPaperCount: UInt32;

Reserve: Array [0 .. 199] of Byte;

End;

Device Information Statement:

Addr: Device IP address

Port: port

ProtocolType：protocol type. If the returned value is zero, it signifies the failure of gaining the device information.

DeviceNo: device number

Version: version number

LanguageID: language type

KeyID: keyboard type

PLUStorage: PLU storable quantity

Note1Storage：Note1 storable quantity

Note2Storage：Note2 storable quantity

Note3Storage：Note3 storable quantity

Not4eStorage：Note4 storable quantity

PrinterKm：printer kilometers

PrinterPaperCount: printer paper quantity

Reserve: reserve

* + 1. **Operation Type**

|  |  |
| --- | --- |
| Illustration | Value |
| Download (to write data to the scale) | 0 |
| Upload (means to read data on the electronic scale). | 1 |
| Delete (Some data types cannot be deleted) | 2 |

* + 1. **Data Type**

D signifies download, U signifies upload, C signifies delete.

Red marked rows are required tables.

|  |  |  |
| --- | --- | --- |
| Illustration | Value | Operation Support |
| item | 0x0000 | DUC |
| department | 0x0001 | DU |
| unit | 0x0002 | D |
| hot key | 0x0003 | DU |
| Custom bar code | 0x0004 | DU |
| Information 1 | 0x0005 | DUC |
| Information 2 | 0x0006 | DUC |
| Information 3 | 0x0007 | DUC |
| Information 4 | 0x0008 | DUC |
| Water | 0x0011 | U |
| System parameters. | 0x000C | DU |
| Time | 0x000D | D |
| Label element | 0x2000 | D |
| Label back | 0x2001 | D |
| Label document | 0x2002 | DU |
| Advanced Barcode | 0x0021 | DU |
| Delete designated items from the scale | 0x0022 | D |
| Currency Conversion | 0x002A | DUC |

1. **Format of the Document**

See the document example in the Demo for the specific format of the document.

**3.1 Single Product Data Document Format**

Red marked rows are required tables.

|  |  |  |  |
| --- | --- | --- | --- |
| Field name | Type | Range | Illustration |
| LFCode | Int | 0~999999 | Single product fresh code |
| ItemCode | Str | 16 | Item number, up to 16 digits |
| Department | Int | 1~99 | Department number, associated with the ID in the Department table |
| Name1 | Str | 40 | Name 1 |
| Name2 | Str | 40 | Name 2 |
| Name3 | Str | 40 | Name 3 |
| Label1 | Int | 0~32 | Label 1 Number. 0 indicates that the default label is printed. If there is no default label, it is not printed. |
| Label2 | Int | 0~32 | Label 2 number. 0 indicates no printing. . |
| BarcodeType1 | Int | 0~150 | Barcode type 1 |
| BarcodeType2 | Int | 0~150 | Barcode type 2 |
| UnitPrice | Float |  | Unit price |
| WeightUnit | Int |  | Unit number. Associate the ID in the Unit table. |
| TareWeight | Int |  | Tare weight in units of g |
| ProducedDateTime | Str |  | Production time. Based on system time format. Such as:YYYY/MM/DD hh:mm:ss |
| PackageDate | Int | 0~99 | Packing days |
| PackageTime | Int | 0~99 | Packing hours |
| ValidDays | Int | 0~999 | Warranty days (hours). 0: non-use warranty days (small hours). 1 to 999: Warranty days (hours). |
| FreshDays | Int | 0~999 | Fresh-keeping days. 0: No use of fresh days. 1-999: fresh-keeping days |
| ValidDateCountF | Int | 0~1 | Shelf life calculation mark. 0: The shelf life is calculated by day, 1: but the quality period is calculated by hour |
| ProducedDateF | Int | 0~1 | Production date basis. 0: current date, 1: specified date. |
| PackageDateF | Int | 0~1 | Packaging date basis. 0: current date,1: production date. |
| ValidDateF | Int | 0~3 | Warranty date basis. 0: current date, 1: production date2: packaging date, 3: do not count |
| FreshDateF | Int | 0~3 | Freshness date standard. 0: current date, 1: production date,2: packaging date, 3: do not count |
| DiscountFlag | Int | 0~2 | Sale sign. 0: not enabled, 1: discount at the same time every day, 2: discount over a period of time |
| DiscountUnitPrice | Float |  | Discount price |
| DiscountStartDateTime | Str |  | Discount start time. Based on system time format |
| DiscountEndDateTime | Str |  | Discount stop time. Based on system time format |
| VipPrice | Float |  | Vip Price |

**3.2 Department Document Format**

|  |  |  |  |
| --- | --- | --- | --- |
| Field name | Type | Range | Illustration |
| ID | Int | 1~99 | Department number |
| Name | Str |  | Department name |

**3.3 Weight Unit Document Format**

|  |  |  |  |
| --- | --- | --- | --- |
| Field name | Type | Range | Illustration |
| ID | Int |  | Weight unit ID |
| DefaultName | Str |  | The default name. Cannot be changed |
| CustomName | Str |  | Custom weight unit name. |

Weight unit ID and DefaultName Default values such as:

|  |  |
| --- | --- |
| ID | DefaultName |
| 0 | Kg |
| 1 | g |
| 2 | 50g |
| 3 | 100g |
| 4 | 500g |
| 5 | 600g |
| 6 | oz |
| 7 | Lb |
| 8 | pcs |
| 9 | Currency(local currency) |

**3.4 Information Document Format**

Information 1· is a required form.

The document format of information 1, information 2, information 3, and information 4 is the same.

|  |  |  |  |
| --- | --- | --- | --- |
| Field name | type | range | illustration |
| LFcode | Int | 1~999999 | Fresh code |
| Value | Str |  | Value of information |

### 

### 3.5 **Currency Conversion Document Format**

|  |  |  |  |
| --- | --- | --- | --- |
| Field name | type | range | illustration |
| id | Int | 0-8 | currency ID |
| symbol | String | Maximum not exceeding 15 bytes | Custom currency symbols, such as US dollars:$ |
| exchangeRate | Uint64 | Maximum not exceeding 8 bytes | exchange rate  The exchange rate value is expressed in millions of times proportion.  For example, 1230000 represents an actual exchange rate of 1.23 (i.e. 1 unit of local currency=1.23 units of foreign currency). |
| accuracy | Int | 0-5 | Exchange rate conversion accuracy  0: Y.00  1:Y.Y0  2:Y.YY  3:Y.Y  4:Y  5:Y00.00 |
| operate | Int | 0-4 | Exchange rate conversion operation  0:round off  1:discard  2:All in  3: 0,0,0,5,5,5,5,5,10,10  4: 0,5,5,5,5,5,10,10,10,10 |

**4.Precautions**

Due to data format limitations, the following characters must be replaced when passed into the string. Otherwise, the data will not be recognized.

Original character Post-substitution character

0x09 {$09}

0x0A {$0A}

0x0D {$0D}