SDK Document

SDK Document

for MTScaleKey

March 2019

METTLER TOLEDO

Contents

1 Introduction

This document is the specification of MTScaleKey APIs. MTScaleKey is a communication component for Mettler-Toledo retail scales, including i15, bTwin, 8442, RL00/RL00+, bCom, bPro, bMobile, bPlus and FreshBase, etc.

MTScaleKey contains a set of DLL files. The APIs are defined in Chapter 2. The APIs requires XML files as input and output. XML structures are defined in Chapter 3.

MTScaleKey can run on Windows 2000 and above. Microsoft Visual C++ 2010 Runtime is required. It can also run in Linux.

2 API Declarations

Some APIs have ANSI version and UNICODE version. For ANSI version, the input and output strings are encoded in ANSI, and for UNICODE version, the input and output strings are encoded in UNICODE.

2.1 ExecuteTaskInFile

Call this API to execute task, where the input and output are both files with UTF-8 format. The API declarations are:

- ANSI version

  extern "C" bool __stdcall ExecuteTaskInFile(const char* szTaskID, const char* szInputFile, const char* szOutputFile, bool bSynch)

- UNICODE version

  extern "C" bool __stdcall ExecuteTaskInFileW(const wchar_t* szTaskID, const wchar_t* szInputFile, const wchar_t* szOutputFile, bool bSynch)

Parameters of the APIs are:

szTaskID: The unique task ID;

szInputFile: The full path of the task file (above Task.XML);

szOutputFile: The full path of the result file;

bSynch: true for synchronized execution, and false for asynchronous.

The API returns a boolean value to indication if it succeeds.

2.2 QueryTask

Query the task execution status. The API declarations are:

- ANSI version:

  extern "C" char* __stdcall QueryTask(const char* szInput)

- UNICODE version:

extern "C" wchar_t* __stdcall QueryTaskW(const wchar_t* szInput)

Parameters of the APIs are:

szInput: Input XML string, where the XML structure is in section

The API returns a string of XML to represent the task's status. When querying multiple tasks in batch, the API only returns the status of existing tasks.

2.3 CancelTask

Call this API to cancel a task. The API declarations are:

- ANSI version:

extern "C" bool __stdcall CancelTask(const char* szInput)

- UNICODE version:

extern "C" bool __stdcall CancelTaskW(const wchar_t* szInput)

Parameters of the APIs are:

szInput: Task ID

The API returns a boolean value to indication if it succeeds.

2.4 QueryStatus

Call this API to query the configuration parameters of the specified components of MTScaleKey, or to query the scale list. The API declarations are:

- ANSI version:

extern "C" char* __stdcall QueryStatus(const char* szInput)

- UNICODE version:

extern "C" wchar_t* __stdcall QueryStatusW(const wchar_t* szInput)

Parameters of the APIs are:

szInput: Input XML string, where the XML structure is in section

The API returns a string of XML to represent the result.

2.5 Free

Call this API to release the returned string of above APIs. It is mandatory to invoke this API to avoid resource leakages. The API declaration is:

extern "C" void __stdcall Free(void* p)

Parameters of the APIs are:

p: Pointer to the string to be released

2.6 Release

Call this API to release the resource of DLL. This API must be invoked before unloading DLL. The API declaration is:

extern "C" void __stdcall Release()

3 XML Structures Definition

3.1 Example

Let's take writing 2 PLUs to 2 scales as an example to explain the details.

Step 1. Suppose the PLU data are

PLU Number	Name	Article Number	Unit of Measure	Unit Price	Label Number	ET Number
1	Apple	1	KGM	11.3	1	1
2	Banana	2	PCS	20	1	2

We need an XML file to represent above data, as shown in below. Suppose it is saved in the file Data.xml. (IMPORTANT: This and other files as the input to Scale Key should be in UTF-8 format.)

The detailed definition of this file structure is in section 3.6.

```
<?xml version="1.0" encoding="utf-8"?>
<Data> <!-- Contains 2 item data -->
  <Item>
    <PLU>1</PLU>   <!-- PLU number -->
    <AlternativeItemIDs>
      <AlternativeItemID>1</AlternativeItemID>     <!-- article number -->
    </AlternativeItemIDs>
    <Descriptions>
      <Description ID="0" Language="zho" Type="ItemName" Index="0">Apple</Description>   <!-- item name -->
      <Description ID="1" Language="zho" Type="ExtraText" />   <!-- ET number is 1 -->
    </Descriptions>
    <ItemPrices>
      <ItemPrice Index="0" UnitOfMeasureCode="KGM" UnitDes="" PriceOverrideFlag="false" DiscountFlag="false" Quantity="0"
Currency="CNY">11.3</ItemPrice>
    </ItemPrices>
    <LabelFormats>
      <LabelFormatID Index="0">1</LabelFormatID> <!-- Label format number is 1 -->
    </LabelFormats>
  </Item>
  <Item>
    <PLU>2</PLU>
    <AlternativeItemIDs>
      <AlternativeItemID>2</AlternativeItemID>
    </AlternativeItemIDs>
    <Descriptions>
      <Description ID="0" Language="zho" Type="ItemName" Index="0">Banana</Description>
      <Description ID="2" Language="zho" Type="ExtraText" />
    </Descriptions>
    <ItemPrices>
      <ItemPrice Index="0" UnitOfMeasureCode="PCS" UnitDes="" PriceOverrideFlag="false" DiscountFlag="false" Quantity="0"
Currency="CNY">20</ItemPrice>
    </ItemPrices>
    <LabelFormats>
      <LabelFormatID Index="0">1</LabelFormatID>
    </LabelFormats>
  </Item>
```

```
</Data>
```

Step 2. We need another XML file to describe the operation. In this example, we want to write the PLU data in Data.xml to scales. The operation maps to below XML, which is saved in below Command.xml. The detailed definition of this file structure is in section 3.4.

```
<?xml version="1.0" encoding="utf-8"?>
<Commands>
  <Command>
    <CommandText>Item</CommandText>   <!-- Command text, to indicate what type of data is transferring -->
    <CommandID>1392f2df-e76b-46bf-9ff2-46bbc8e71b93</CommandID> <!-- Any unique string -->
    <Control>Update</Control> <!-- The command that MTScaleKey will execute -->
    <ClearData>false</ClearData> <!-- Flag of clearing data before transferring -->
    <DataFile>Data.xml</DataFile> <!-- The data file being transferred -->
  </Command>
</Commands>
```

Step 3. We need define the target scales. Suppose they are

Device Number	Scale Number	Scale Type	IP	Port	Communication Type
1	1	bCom	172.30.8.82	3001	Ethernet
2	2	bPlus	172.30.8.85	2305	Ethernet

The corresponding XML file is shown as below ScaleList.xml. The detailed definition of this file structure is in section 3.3.

```
<?xml version="1.0" encoding="utf-8"?>
<Devices>
  <Scale>
    <DeviceID>1</DeviceID>     <!-- Device number is 1 -->
    <ScaleNo>1</ScaleNo>      <!-- Scale number is 1 -->
    <ScaleType>bCom</ScaleType>     <!-- Scale type is bCom -->
    <ConnectType>Network</ConnectType>
    <ConnectParams>
      <NetworkParams Type="Network" Address="172.30.8.82" Port="3001" />   <!-- Scale IP and port -->
    </ConnectParams>
    <DecimalDigits>2</DecimalDigits>
    <DataFile>Command.xml</DataFile>       <!-- Command.xml defines the actions to this scale -->
  </Scale>
  <Scale>
    <DeviceID>2</DeviceID>
    <ScaleNo>2</ScaleNo>
    <ScaleType>bPlus</ScaleType>
    <ConnectType>Network</ConnectType>
    <ConnectParams>
      <NetworkParams Type="Network" Address="172.30.8.85" Port="2305" />
    </ConnectParams>
    <DecimalDigits>2</DecimalDigits>
    <DataFile>Command.xml</DataFile>       <!-- File could be different than other scales -->
  </Scale>
</Devices>
```

Step 4. The 4th XML is to describe the task, as shown in below Task.xml. The detailed definition of this file structure is in section 3.2.

```
<?xml version="1.0" encoding="UTF-8"?>
```

```
<MTTask>
    <TaskID>4eaa9e02-2215-44ed-a2fd-d6d8fcc6ea58</TaskID>
    <TaskType>0</TaskType>
    <DataFile>ScaleList.xml</DataFile>
</MTTask>
```

Step 5. When all 4 XMLs are ready (They must be saved in the same folder), we can invoke the **ExecuteTaskInFile** API in MTScaleAPI.dll to write the 2 PLUs to the 2 scales:

ExecuteTaskInFile("UniqueID20190328132125011", "Task.xml", "Result.xml", false);

Then we can invoke the **QueryTask** API to check status of the task:

char* p =
QueryTask("<MTTask><TaskID>UniqueID20190328132125011</TaskID><TaskType>98</TaskType></MTTask>");

When the XML string pointed by *p* has *TaskStatus* as *Complete* or *Error*, the task is finished. The output file Result.xml looks like below. The detailed definition of this file structure is in section 3.5.

```
<?xml version="1.0" encoding="UTF-8"?>
<MTTaskResult>
  <TaskID>4eaa9e02-2215-44ed-a2fd-d6d8fcc6ea58</TaskID>
  <TaskType>0</TaskType>
  <StartTime>2019-04-12T08:41:51</StartTime>
  <EndTime>2019-04-12T08:41:51</EndTime>
  <TaskStatus>Complete</TaskStatus>
  <ProcessPercent>100</ProcessPercent>
  <ReturnCode>OK</ReturnCode>
  <OutputFile>C:/xmldump/4eaa9e02-2215-44ed-a2fd-d6d8fcc6ea58/Output/Result.xml</OutputFile>
  <ScaleResults>
    <ScaleResult>
      <DeviceID>1</DeviceID>
      <ScaleNo>1</ScaleNo>
      <TaskStatus>Complete</TaskStatus>
      <StartTime>2019-04-12T08:41:51</StartTime>
      <EndTime>2019-04-12T08:41:51</EndTime>
      <ProcessPercent>100</ProcessPercent>
      <ReturnCode>OK</ReturnCode>
      <ScaleType>bCom</ScaleType>
      <LocalStoreID>0</LocalStoreID>
      <CommandResults>
        <CommandResult>
          <CommandID>1392f2df-e76b-46bf-9ff2-46bbc8e71b93</CommandID>
          <CommandText>Item</CommandText>
          <Control>Update</Control>
          <TaskStatus>Complete</TaskStatus>
          <StartTime>2019-04-12T08:41:51</StartTime>
          <EndTime>2019-04-12T08:41:51</EndTime>
          <ProcessPercent>100</ProcessPercent>
          <Succeeded>1</Succeeded>
          <Failed>0</Failed>
          <Total>1</Total>
          <ReturnCode>OK</ReturnCode>
          <DataFile>Data.xml</DataFile>
```

```
        </CommandResult>
      </CommandResults>
    </ScaleResult>
    <ScaleResult>
      <DeviceID>2</DeviceID>
      <ScaleNo>2</ScaleNo>
      <TaskStatus>Complete</TaskStatus>
      <StartTime>2019-04-12T08:41:51</StartTime>
      <EndTime>2019-04-12T08:41:51</EndTime>
      <ProcessPercent>100</ProcessPercent>
      <ReturnCode>OK</ReturnCode>
      <ScaleType>bPlus</ScaleType>
      <LocalStoreID>0</LocalStoreID>
      <CommandResults>
        <CommandResult>
          <CommandID>1392f2df-e76b-46bf-9ff2-46bbc8e71b93</CommandID>
          <CommandText>Item</CommandText>
          <Control>Update</Control>
          <TaskStatus>Complete</TaskStatus>
          <StartTime>2019-04-12T08:41:51</StartTime>
          <EndTime>2019-04-12T08:41:51</EndTime>
          <ProcessPercent>100</ProcessPercent>
          <Succeeded>1</Succeeded>
          <Failed>0</Failed>
          <Total>1</Total>
          <ReturnCode>OK</ReturnCode>
          <DataFile>Data.xml</DataFile>
        </CommandResult>
      </CommandResults>
    </ScaleResult>
  </ScaleResults>
</MTTaskResult>
```

Here is a C# code example.

FileExecute.cs

Summary: The API requires an input file to describe the task. This task file contains the file name which defines the target scale list. The scale list file contains the command file names for each scale. Each command file has all command text for the scale, and each command text has a corresponding data file name. The data file describes the communication content with the scale. After execution, the API returns an output file for the result. This file nestedly describes detailed result for each communication with scales. All above mentioned files are XML.

The rest sections will describe all related XML structures. Read parts are mandatory. Blue parts are mandatory in the specific context.

Scale Manager also integrates Scale Key. You can write data to scales from Scale Manager, and

then check C:\Program Files (x86)\ScaleKey\Data to get the input and output XML files for reference.

3.2 Task Structure

This section describes the task structure, which is used in Step 4 of section 3.1.

XML node name: MTTask

XML Node	XML Node Type	Data Type	Notes
Version	Attribute	String	XML version
TaskID	Sub node	String	The **unique** task ID
TaskTime	Sub node	String	The timestamp of task creation. The format is YYYY-MM-DDTHH:mm:ss, e.g. 2014-06-09T09:12:12. **All other timestamps in this document have the same format as this.**
TaskType	Sub node	int	Type of the task, the value could be: ● 0: send data to scales ● 98: query the status of a single task.
DataFile	Sub node	String	The file name for scale list.

3.3 Scale Structure

This section describes the scale structure, which is used in Step 3 of section 3.1. The root node name is **Devices**.

XML node name: Scale

XML Node	XML Node Type	Data Type	Notes
DeviceID	Sub node	String	Unique device ID.
ScaleNo		int	Scale number
ScaleName	Sub node	String	Scale name
CWANID	Sub node	String	Customer ID for WAN
LocalStoreID	Sub node	String	Store number. It is mandatory for WAN.
ConnectType	Sub node	String	Communication type. The string could be “Network” for LAN or “Internet” for WAN.
ConnectParams	Sub node		It should contain at least one of the sub nodes NetworkParams for LAN config and InternetParams for WAN config. Both LAN and WAN can be defined, but the “ConnectType” sub node defines which works. Detailed definition is in section 3.3.1.
ScaleType	Sub node	String	Scale type. Valid strings are 8442, BlackEagle,

			bCom, bPro, Apollo, bTwin, bHigh, i15, RL00, bMobile, bDrive, Skipper 5000, Skipper 7000, bPlus, FreshBase, Rider 5000, Rider 8000, Etica, Unknown (Only used for unknown scales on auto scale scan). **All other ScaleType nodes in this document are as same as this.**
CODName	Sub node	String	COD name. Ignore it for non-COD case.
DecimalDigits	Sub node	int	Decimal place. It is expected to be the same as the setting in the scale.
DataFile	Sub node	String	It is used to store file name.
AdditionalConfig	Sub node	String	Additional configuration. It is same as that in MTTask node. If both are defined, here has higher priority.

3.3.1 NetworkParams and InterneteParams Structure

XML node name: NetworkParams

XML Node	XML Node Type	Data Type	Notes
Address	Attribute	String	IP address
Port	Attribute	Int	Port. Default is 3001.

XML node name: InternetParams

XML Node	XML Node Type	Data Type	Notes
Address	Attribute	String	IP address
Port	Attribute	Int	Port. Default is 8001.

3.4 Command Structure

This section describes the command structure, which is used in Step 2 of section 3.1.

XML node name: Command

XML Node	XML Node Type	Data Type	Notes
CommandID	Sub node	String	Command ID. It should be unique for each scale.
CommandText	Sub node	String	Command text. It means what data will be exchanged with scales, for example “item” means PLU,
Control	Sub node	String	Command control. It means what operation is to be executed. The string could be Update,

			Delete, DeleteAll, Read, ReadAll. Their meaning is same as their name. **All other Control nodes in this document are as same as this.**
ClearData	Sub node	bool	A flag to indicate if clear the corresponding type of data in scales before writing data to them. This flag is only valid when Control is Write or Update.
DataFile	Sub node	String	Data file name.
AdditionalConfig	Sub node	String	Additional configuration that in MTTask and Scale node. If any other is defined, here has higher priority.

3.5 Task Result Structure

This section describes the task result structure, which is used in Step 5 of section 3.1. It contains the status or result of the task.

XML node name: MTTaskResult

XML Node	XML Node Type	Data Type	Notes
TaskID	Sub node	String	Task ID. Same as the TaskID in section 3.2.
TaskType	Sub node	Int	Task type. Same as the TaskType in section 3.2.
StartTime	Sub node	String	Start time for the task
EndTime	Sub node	String	Finish time for the task
ProcessPercent	Sub node	Int	Task progress percentage
ReturnCode	Sub node	String	Return code of task execution. The string could be OK, ProgramError, DataFileError, TaskTypeError, TaskRepeatError, TaskNotExistedError, TransferError, ScaleDataError, DataNotFoundError, ConnectError, ServiceError, UnexpectedStopError, DataNotSupportedError, Cancelled, ReadFileError, NeedRestart, ScaleNotSupportedDataError PasswordError, ScaleSpaceFullError, ScaleTypeError, DataRepeatedError, SaveFileError. **All other ReturnCode nodes in this document are as same as this. The possible string value may be more in the future.**
TaskStatus	Sub node	String	Task status. The string could be Waiting, Executing, Output, Complete, Error or Prepare.

ErrorMessage	Sub node	String	Error message string
OtherMessage	Sub node	String	Other information
OutputFile	Sub node	String	Task result output file
ScaleResults	Sub node		Result for scales. It may contain several ScaleResult sub node (refer to section 3.5.1), mapping to each scale.

3.5.1 ScaleResult Structure

XML node name: ScaleResult

XML Node	XML Node Type	Data Type	Notes
DeviceID	Sub node	String	Device ID. Same as the DeviceID in section 3.3.
ScaleNo	Sub node	Int	Scale number. Same as the ScaleNo in section 3.3.
ScaleType	Sub node	String	Scale type. Same as the ScaleType in section 3.3.
StartTime	Sub node	String	Start time for the scale task.
EndTime	Sub node	String	Finish time for the scale task.
ReturnCode	Sub node	String	Return code of the scale task.
ErrorMessage	Sub node	String	Error message.
OtherMessage	Sub node	String	Other information.
ProcessPercent	Sub node	Int	Execution percentage of the scale task.
TaskStatus	Sub node	String	Task status. The string could be Waiting, Executing, Output, Complete, Error or Prepare.
CommandResults	Sub node		Command result. It may contain several CommandResult sub nodes (refer to section 3.5.2), mapping to each command for the scale.

3.5.2 CommandResult Structure

XML node name: CommandResult

XML Node	XML Node Type	Data Type	Notes
CommandID	Sub node	String	Command ID. Same as the CommandID in section 3.4.
CommandText	Sub node	String	Command text. Same as the CommandText in section 3.4.
Control	Sub node	String	Command control. Same as the Control in section 3.4.
StartTime	Sub node	String	Start time for the command task.

EndTime	Sub node	String	Finish time for the command task.
ReturnCode	Sub node	String	Return code of the command task.
ErrorMessage	Sub node	String	Error message.
OtherMessage	Sub node	String	Other information.
Succeeded	Sub node	Int	Count of success.
Failed	Sub node	Int	Count of failure.
Total	Sub node	Int	Total count.
DataFile	Sub node	String	File name that holds result from scale. It is valid for Read and ReadAll command control (refer to section 3.4), and it can be ignored for other controls.
ProcessPercent	Sub node	Int	Communication percentage. It is used to show progress when writing to scales.
TaskStatus	Sub node	String	Task status. The string could be Waiting, Executing, Output, Complete, Error or Prepare.

3.6 Data Content Information Structure

Data content information is the communication data of command control (refer to section 3.4). Some are only supported by part of the scale types, as listed in below table.

Data Type	CommandText	Control	Notes
PLU	Item	Update	Valid for all types of scales
		Delete	Valid for all types of scales
		DeleteAll	Valid for all types of scales except bTwin, bHigh and Etica
		Read	Valid for all types of scales except Etica
		ReadAll	Valid for all types of scales except Etica
Extra text	ExtraText	Update	Valid for 8442, BlackEagle, bCom, bPro, RL00, bMobile, bDrive, bPlus, FreshBase, Etica
		Delete	Valid for 8442, BlackEagle, bCom, bPro, RL00, bMobile, bDrive, bPlus, FreshBase, Etica
		DeleteAll	Valid for 8442, BlackEagle, bCom, bPro, RL00, bMobile, bDrive, bPlus, FreshBase, Etica
		Read	Valid for 8442, BlackEagle, bCom, bPro, RL00, bMobile, bDrive, bPlus, FreshBase, Etica
		ReadAll	Valid for 8442, BlackEagle, bCom, bPro, RL00, bMobile, bDrive, bPlus, FreshBase, Etica
Barcode	Barcode	Update	Valid for 8442, bCom, bPro, RL00, bMobile, bDrive, bPlus, FreshBase

		Delete	Valid for bCom, bPro, bMobile, bDrive, bPlus, FreshBase
		DeleteAll	Valid for bMobile, bDrive, bPlus, FreshBase
		Read	Valid for 8442, bCom, bPro, RL00, bMobile, bDrive, bPlus, FreshBase
		ReadAll	Valid for 8442, bCom, bPro, RL00, bMobile, bDrive, bPlus, FreshBase
Preset key	PresetDefinition	Update	Valid for all scale types
		Delete	Valid for bMobile, bDrive, bPlus, FreshBase
		DeleteAll	Valid for bMobile, bDrive, bPlus, FreshBase
		ReadAll	Valid for all scale types
File	File	Update	Valid for bMobile, bDrive, bPlus, FreshBase
		Delete	Valid for bMobile, bDrive, bPlus, FreshBase

3.6.1 PLU

XML node name: Item

XML Node	XML Node Type	Data Type	Notes
PLU	Sub node	Int	PLU ID
DepartmentID	Sub node	Int	Department ID. It is mandatory for bMobile, bDrive and bPlus. Default is 0.
AlternativeItemIDs	Sub node		Article number list. It may contain several AlternativeItemID sub nodes (refer to section 3.6.1.1).
Descriptions	Sub node		PLU name list. It may contain several Description sub nodes (refer to section 3.6.1.2).
Dates	Sub node		Dates. It may contain several Date sub nodes (refer to section 3.6.1.3).
ItemGroupID	Sub node	Int	PLU group. It is only valid for bMobile, bDrive and bPlus.
CategoryIDs	Sub node		PLU category. It may contain several CategoryID sub nodes (refer to section 3.6.1.4). It is only valid for bMobile, bDrive and bPlus.
Tares	Sub node		Tare list. It may contain several TareID sub nodes (refer to section 3.6.1.5) and TareWeight sub nodes (refer to section 3.6.1.6).
ItemPrices	Sub node		Price list. It may contain several ItemPrice sub nodes (refer to section 3.6.1.7).
Taxes	Sub node		Tax list. It may contain several TaxRuleID sub

			nodes (refer to section 3.6.1.8).
Ingredients	Sub node		Ingredient list. It may contain several IngredientID sub nodes (refer to section 3.6.1.9). It is only valid for bMobile, bDrive and bPlus.
LabelFormats	Sub node		Label format list. It may contain several LabelFormatID sub nodes (refer to section 3.6.1.10).
Barcodes	Sub node		Barcode list. It may contain several BarcodeID sub nodes (refer to section 3.6.1.11). It is only valid for bMobile, bDrive and bPlus.
NutritionInformation	Sub node		Nutrition list. It may contain several NutritionInformationID sub node (refer to section 3.6.1.12). It is only valid for bMobile, bDrive and bPlus.
FixedQuantity	Sub node		The fixed quantity of PLU. It may contain several FixedQuantity sub nodes (refer to section 3.6.1.13).
TraceInfoID	Sub node		Traceability ID
TraceabilityFlag	Sub node		Traceability enable flag. This flag is invalid for bPlus, bMobile and bDrive.
PriceRule	Sub node	Int	Price rule ID. It is invalid for bPlus, bMobile and bDrive.
Images	Sub node		Image ID list. It may contain several (but no more than 3) Image sub nodes (refer to section 3.6.1.14). It is only valid for bCom, bPro, bMobile, bDrive, bPlus and FreshBase.
StaggerPrices	Sub node		Stagger price list. It may contain several StaggerPrice sub nodes (refer to section 3.6.1.15). It is only valid for bMobile, bDrive, bPlus and FreshBase.

3.6.1.1 AlternativeItemID Structure

This structure is used for article number.

XML node name: AlternativeItemID

XML Node	XML Node Type	Data Type	Notes
AlternativeItemID		String	Article number. Max length is 13.

3.6.1.2 Description Structure

This structure is used for PLU description.

XML node name: Description

XML Node	XML Node Type	Data Type	Notes
Type	Attribute	String	Type of the text. The string could be “ItemName” for PLU name, “ItemShortName” for PLU 2nd name, “ExtraText” for extra text, or “ShortPinYinCode” for PinYin short code.
ID	Attribute	Int	ID of the text. It is only valid when Type is “ExtraText”, where the value is ET ID.
Language	Attribute	String	Language of the text. It is only valid for bMobile, bDrive and bPlus. The string could be “zho” for Chinese, “eng” for English, “fra” for French, or “deu” for German. **All other Language attributes in this document are as same as this.**
Text		String	Text string

3.6.1.3 Date Structure

This structure is used for date print definition.

XML node name: Dateoffset

XML Node	XML Node Type	Data Type	Notes
Type	Attribute	String	Date type. The string could be “SellBy” for sell by date, “BestBefore” for best before date, “PackedDate” for packed date, “UserDef1” for the 1st user defined date, or “UserDef2” for the 2nd user defined date.
UnitOfOffset	Attribute	String	Date unit. The string could be “day”, “hour”, “date”, “time”, where the last 3 are only valid for bMobile, bDrive, bPlus and FreshBase, and “hour” and “time” are invalid for PackedDate.
PrintFormat	Attribute	String	Print format. It is invalid for bMobile, bDrive, bPlus, and FreshBase. The string could be “YYMMDD”, “DDMMYY”, “MMDD”, “MMDDYY”, or “Days”. When Type is “PackedDate”, the format cannot be “Days”.

Value		String	Date print value. Its format is “YYYY-MM-DD” for date (e.g. 2018-06-09) and “HH:mm:ss” for time (e.g. 08:21:32).
PrintEnabled	Attribute	Bool	Flag to indicate print or not

3.6.1.4 CategoryID Structure

This structure is used for category ID.

XML node name: CategoryID

XML Node	XML Node Type	Data Type	Notes
CategoryID		Int	Category ID

3.6.1.5 TareID Structure

This structure is used for tare ID.

XML node name: TareID

XML Node	XML Node Type	Data Type	Notes
TareID		Int	Tare ID

3.6.1.6 TareWeight Structure

This structure is used for tare value.

XML node name: TareWeight

XML Node	XML Node Type	Data Type	Notes
UnitOfMeasureCode	Attribute	String	Unit of tare. The string could be “GRM” for g, “KGM” for kg, or “PCT” for percentage (it is invalid for bMobile, bDrive, bPlus and FreshBase).
Weight		Double	Tare value

3.6.1.7 ItemPrice Structure

This structure is used for price.

XML node name: ItemPrice

XML Node	XML Node Type	Data Type	Notes
Index	Attribute	Int	Sequence of unit prices, starting form 0.
ValueTypeCode	Attribute	String	Price Type. The string could be “BasePrice” for

			unit price, or “SelfService” for self service price.
UnitOfMeasureCode	Attribute	String	Unit of measure. The string could be “GRM” for /g, “KGM” for /kg, “LBR” for /lb (it is only valid for bMobile, bDrive and bPlus), “PCS” for counting, “100g” for /100g, and “500g” for /500g.
PriceOverrideFlag	Attribute	Bool	The flag to indicate if changing price in scale is allowed.
DiscountFlag	Attribute	Bool	The flag to indicate if discount in scale is allowed.
Quantity	Attribute	Int	Count of item purchased in multiple package items.
QuantityPricingFlag	Attribute	Bool	The flag to indicate if the item is priced using Quantity Package Price (e.g.: 3 for $0.89)
MarkUpFlag	Attribute	Bool	The flag to indicate if markup in scale is allowed.
UnitDes	Attribute	String	Unit of counting PLU, e.g. “bag”, “pack”.
Value		Double	Price value.
Hide	Attribute	Bool	The flag to indicate if the price is hidden in scale (cannot be selected and seen). It is only valid for bPlus, bMobile, bDrive, and FreshBase.

3.6.1.8 TaxRuleID Structure

This structure is used for tax.

XML node name: TaxRuleID

XML Node	XML Node Type	Data Type	Notes
TaxRuleID		Int	Tax ID

3.6.1.9 IngredientID Structure

This structure is used for ingredient.

XML node name: IngredientID

XML Node	XML Node Type	Data Type	Notes
IngredientID		Int	Ingredient ID

3.6.1.10 LabelFormatID Structure

This structure is used for label format.

XML node name: LabelFormatID

XML Node	XML Node Type	Data Type	Notes
Index	Attribute	Int	Sequence of label format, starting form 0. RL00 and 8442 support only one label format ID. bCom and bPro support 2. bPlus, bMobile, bDrive and FreshBase support at most 3. Other scales does not support label format.
LabelFormatID		Int	Label format ID.

3.6.1.11 BarcodeID Structure

This structure is used for barcode.

XML node name: BarcodeID

XML Node	XML Node Type	Data Type	Notes
BarcodeID		Int	Barcode ID

3.6.1.12 NutritionInformationID Structure

This structure is used for label format.

XML node name: NutritionInformationID

XML Node	XML Node Type	Data Type	Notes
NutritionInformationID		Int	Nutrition ID

3.6.1.13 FixedQuantity Structure

This structure is used for fixed quantity.

XML node name: FixedQuantity

XML Node	XML Node Type	Data Type	Notes
UnitOfMeasureCode	Attribute	String	Unit of measure. The string could be “GRM” for g, “KGM” for kg, “LBR” for lb (it is only valid for bMobile, bDrive and bPlus), “PCS” for counting, “100g” for 100g, and “500g” for 500g.
Value		Double	Fixed weight or quantity.

3.6.1.14 Image Structure

This structure is used for image.

XML node name: Image

XML Node	XML Node Type	Data Type	Notes
Index	Attribute	Int	If index is 0, the image is for PLU. If index is 1, the image is for SafeHandling. If index is 2, the image is for StoreLogo (bPro and bCom do not support this)
Type	Attribute	String	Set it as “Item” always.
ImageID		String	Image ID. It must be number for bCom and bPro.

3.6.1.15 StaggerPrice Structure

This structure is used for stagger price.

XML node name: StaggerPrice

XML Node	XML Node Type	Data Type	Notes
Index	Attribute	Int	Should be same as the index of one of the ItemPrice sub nodes (refer to section 3.6.1.7).
ValueTypeCode	Attribute	String	Price type. It should be as same as the ValueTypeCode of one of the ItemPrice sub nodes (refer to section 3.6.1.7).
Condition	Attribute	Double	Condition weight or quantity value
Price		Double	Price value

3.6.1.16 File Example

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<Data>
    <Item>
        <PLU>6</PLU>
        <DepartmentID>0</DepartmentID>
        <AlternativeItemIDs>
            <AlternativeItemID>12038664</AlternativeItemID>
        </AlternativeItemIDs>
        <Descriptions>
            <Description Type="ItemName">国产香米[kg]</Description>
            <Description Type="ItemShortName">粮食</Description>
            <Description Type="ExtraText" ID="1"/>
        </Descriptions>
        <Dates>
            <DateOffset Type="PackedDate" PrintFormat="YYMMDD" UnitOfOffset="day" PrintEnabled="true">0</DateOffset>
            <DateOffset Type="SellBy" PrintFormat="YYMMDD" UnitOfOffset="day" PrintEnabled="true">30</DateOffset>
            <DateOffset Type="BestBefore" PrintFormat="YYMMDD" UnitOfOffset="day" PrintEnabled="true">20</DateOffset>
        </Dates>
        <ItemGroupID>0</ItemGroupID>
        <Tares>
            <TareID>6</TareID>
            <TareWeight UnitOfMeasureCode="GRM">10.0</TareWeight>
        </Tares>
        <ItemPrices>
```

```
            <ItemPrice Index="0" ValueTypeCode="BasePrice" UnitOfMeasureCode="KGM" PriceOverrideFlag="false"
DiscountFlag="false" Hide="false">3.0</ItemPrice>
            <ItemPrice Index="1" ValueTypeCode="BasePrice" UnitOfMeasureCode="KGM" PriceOverrideFlag="false"
DiscountFlag="false" Hide="false">3.0</ItemPrice>
        </ItemPrices>
        <Taxes>
            <TaxRuleID>0</TaxRuleID>
        </Taxes>
        <LabelFormats>
            <LabelFormatID Index="0">2</LabelFormatID>
        </LabelFormats>
        <Barcodes>
            <BarcodeID>6</BarcodeID>
        </Barcodes>
        <Images>
            <Image Index="0" Type="Item">1</Image>
        </Images>
        <Promotions>
            <Promotion Index="0" Name="Promotion6" StartDate="2018-04-14" EndDate="2018-04-15" StartTime="13:00:00"
EndTime="17:00:00" Mode="2" Type="SpecialPrice">4.5</Promotion>
        </Promotions>
    </Item>
    <Item>
        <PLU>8</PLU>
        <DepartmentID>0</DepartmentID>
        <AlternativeItemIDs>
            <AlternativeItemID>12038435</AlternativeItemID>
        </AlternativeItemIDs>
        <Descriptions>
            <Description Type="ItemName">伊赛小牛排[kg]</Description>
            <Description Type="ItemShortName">粮食</Description>
            <Description Type="ExtraText" ID="0"/>
        </Descriptions>
        <Dates>
            <DateOffset Type="PackedDate" PrintFormat="YYMMDD" UnitOfOffset="day" PrintEnabled="true">0</DateOffset>
            <DateOffset Type="SellBy" PrintFormat="YYMMDD" UnitOfOffset="day" PrintEnabled="true">60</DateOffset>
        </Dates>
        <ItemGroupID>0</ItemGroupID>
        <CategoryIDs>
            <CategoryID>0</CategoryID>
        </CategoryIDs>
        <Tares>
            <TareID>8</TareID>
            <TareWeight UnitOfMeasureCode="GRM">30.0</TareWeight>
        </Tares>
        <ItemPrices>
            <ItemPrice Index="0" ValueTypeCode="BasePrice" UnitOfMeasureCode="KGM" PriceOverrideFlag="false"
DiscountFlag="false" Hide="false">83.05</ItemPrice>
        </ItemPrices>
        <Taxes>
            <TaxRuleID>0</TaxRuleID>
        </Taxes>
        <LabelFormats>
            <LabelFormatID Index="0">1</LabelFormatID>
        </LabelFormats>
        <Barcodes>
            <BarcodeID>5</BarcodeID>
        </Barcodes>
```

```
    </Item>
</Data>
```

3.6.2 Extra Text (ET)

XML node name: ExtraText

XML Node	XML Node Type	Data Type	Notes
ID	Sub node	Int	ET ID
Name	Sub node	String	ET name
Language	Sub node	String	Language
DepartmentID	Sub node	Int	Department ID
ETTexts	Sub node		ET content list. It may contain several ETText sub nodes (refer to section 3.6.2.1).

3.6.2.1 ETText Structure

This structure is used for ET contents.

XML node name: ETText

XML Node	XML Node Type	Data Type	Notes
Index	Attribute	Int	ET segment ID, starting from 0. 8442, BlackEagle, bCom and RL00 supports at most 4 segments. bMobile, bDrive and bPlus supports at most 8 segments.
FontSize	Attribute	Int	Font size. The value could be 1 to 29. It is invalid for bMobile, bPlus and bCom.
Text		String	ET text

3.6.2.2 File Example

```
<?xml version="1.0" encoding="utf-8"?>
<Data>
  <ExtraText>
    <ID>1</ID>
    <DepartmentID>0</DepartmentID>
    <Language>zho</Language>
    <ETTexts>
      <ETText Index="0" FontSize="21">1234567890</ETText>
      <ETText Index="1" FontSize="21">test</ETText>
      <ETText Index="2" FontSize="21">test</ETText>
      <ETText Index="3" FontSize="21">test</ETText>
```

```
    </ETTexts>
  </ExtraText>
  <ExtraText>
    <ID>2</ID>
    <DepartmentID>0</DepartmentID>
    <Language>zho</Language>
    <ETTexts>
      <ETText Index="0" FontSize="21">1234567890</ETText>
      <ETText Index="1" FontSize="21">test</ETText>
      <ETText Index="2" FontSize="21">test</ETText>
      <ETText Index="3" FontSize="21">test</ETText>
    </ETTexts>
  </ExtraText>
</Data>
```

3.6.3 Barcode

XML node name: Barcode

XML Node	XML Node Type	Data Type	Notes
Symbology	Attribute	String	Barcode type. The string could be “EAN8”, “EAN13”, “Code25N” (for Code25 NARROW), “Code25W” (for Code25 WIDE), “EAN-128”, “EAN13+5”, “UPC-A”, “Code128”, “Interleaved2Of5”, “QRCode”, “GM”, “GS1DataBarOmnidirectional”, “GS1DataBarStackedOmnidirectional”, “GS1DataBarExpanded”, “GS1DataBarExpandedStacked”. RL00 supports EAN8, EAN13, Code25N, Code25W, EAN-128, EAN13+5, Code128, QRCode and GM. 8442 supports EAN8, EAN13, Code25N, EAN-128, UPC-A and Code128. bCom supports EAN8, EAN13, Code25N, Code25W， EAN-128, EAN13+5, UPC-A, Code128, QRCode and GM. For bPlus and bMobile, they cannot have both Code25N and Code25W at the same time.
ID	Sub node	Int	Barcode ID.
Description	Sub node	String	Barcode name.
Definition	Sub node	String	The barcode format definition. For classic barcode, fill in placeholders in the string (refer to

			section 3.6.3.1). It is treated as advanced barcode if the string contains “$D”, which is not supported by 8442, bCom, bPro, RL00.
Parity	Sub node	String	Parity method. The string could be “Normal”, “Reverse” or “None”.
ItemNoShift	Sub node	Int	Shift for article number. The value is negative for left shift and positive for right shift. 8442 and RL00 do not support this. bPro and bCom only support right shift and its range is 0 ~ 8.
TotalShift	Sub node	Int	Shift for total price. The value is negative for left shift and positive for right shift. 8442 and RL00 do not support this. bPro and bCom only support right shift and its range is 0 ~ 2.
WeightShift	Sub node	Int	Shift for weight. The value is negative for left shift and positive for right shift. 8442 and RL00 do not support this. bPro and bCom only support right shift and its range is 0 ~ 2.
CountShift	Sub node	Int	Shift for count. The value is negative for left shift and positive for right shift. 8442 and RL00 do not support this. bPro and bCom only support left shift and its range is 0 ~ 3.
PriceShift	Sub node	Int	Shift for unit price. The value is negative for left shift and positive for right shift. 8442 and RL00 do not support this. bPro and bCom only support right shift and its range is 0 ~ 2.

3.6.3.1 Barcode Placeholder

Example, “26PPPPPQQQQQQBBBBBBC” means “26 + 5 digits PLU number + 6 digits weight or count + 6 digits total price + 1 digit checksum”.

Placeholder	Notes
F	Department number. Use “K” for scale types other than bPlus, bMobile and bDrive.
P	PLU number
A	Article number
G	Scale number. Only bPlus, bMobile and bDrive support it.

O	Vendor number
D	Current date
J	Current time. Use “T” for scale types other than bPlus, bMobile and bDrive.
Q	Weight or count
M	Unit price. Use “U” for scale types other than bPlus, bMobile and bDrive.
B	Total price.
C	Parity

3.6.3.2 File Example

```
<?xml version="1.0" encoding="utf-8"?>
<Data>
  <Barcode Symbology="EAN13">
    <ID>1</ID>
    <Description>Barcode1</Description>
    <Definition>2AAAAAABBBBB</Definition>
    <Parity>None</Parity>
    <ItemNoShift>2</ItemNoShift>
    <TotalShift>2</TotalShift>
    <WeightShift>2</WeightShift>
    <CountShift>2</CountShift>
    <PriceShift>2</PriceShift>
  </Barcode>
</Data>
```

3.6.4 Preset Key

XML node name: PresetDefinition

XML Node	XML Node Type	Data Type	Notes
Type	Attribute	String	Preset key type. It is mandatory for FreshBase. The string could be “BaseMode”, or “SelfService” (only FreshBase support it).
PresetPage	Sub node		The preset key pagination. It may contain several PresetPage sub nodes (refer to section 3.6.4.1).
PresetKey	Sub node		Preset key layout definition. It may contain several PresetKey sub nodes (refer to section 3.6.4.2).

3.6.4.1 PresetPage Structure

XML node name: PresetPage

XML Node	XML Node Type	Data Type	Notes

ID	Sub node	Int	Page ID. It is always 1 now.
Title	Sub node		Page title. A PresetPage may have several Title nodes. Refer to section 3.6.4.3.
Height	Sub node	Int	Number of rows of a page. It is mandatory for FreshBase.
Width	Sub node	Int	Number of columns of a page. It is mandatory for FreshBase.
KeyCount	Sub node	Int	Number of keys
Type	Sub node	String	Type is mandatory for FreshBase. Ignore it for other scale types. The string can be: Graphic: PLU page with images FunctionGraphics: Function key page VendorGraphics: Vendor page PaymentGraphics: Payment related function keypage.
Visible	Sub node	Bool	Always be true.

3.6.4.2 PresetKey Structure

XML node name: PresetKey

XML Node	XML Node Type	Data Type	Notes
PageID	Sub node	Int	Identify which page the key locates. It is always 1 now.
Type	Sub node	String	The type string can be PLU, Tare (only valid for bMobile, bDrive, bPlus, FreshBase), Event (only valid for bMobile, bDrive, bPlus, FreshBase), Page (nested page, only vaild for FreshBase)
Destination	Sub node		Key target. It may be PLU number or function key name. Refer to section 3.6.4.4.
Title	Sub node		Sub page title. It may have several Title sub nodes. Refer to section 3.6.4.3.
KeyNo	Sub node	Int	Key location. Legacy scales use KeyNo and Level to locate the target key. It is invalid for bMobile, bDrive, bPlus and FreshBase.
Column	Sub node	Int	Column number. It is only valid for bMobile, bDrive, bPlus and FreshBase. These scales use Column, Row and Level to locate the target key.
Row	Sub node	Int	Row number. It is only valid for bMobile, bDrive, bPlus and FreshBase. These scales use Column, Row and Level to locate the target key.

Level	Sub node	Int	Level can be 1 or 2.
ImageLocation	Sub node	String	Image file name in scale.

3.6.4.3 Title Structure

XML node name: Title

XML Node	XML Node Type	Data Type	Notes
Language	Attribute	String	
Text	Text	String	Title content

3.6.4.4 Destination Structure

XML node name: Title

XML Node	XML Node Type	Data Type	Notes
DepartmentID	Attribute	Int	Department ID
Text	Text	String	It can be PLU ID, tare ID, or function key name.

3.6.4.5 File Example

```
<?xml version="1.0" encoding="utf-8"?>
<Data>
  <PresetDefinition>
    <PresetPage>
      <ID>1</ID>
      <KeyCount>77</KeyCount>
    </PresetPage>
    <PresetKey>
      <PageID>1</PageID>
      <Type>PLU</Type>
      <Row>1</Row>
      <Column>1</Column>
      <Level>1</Level>
      <Destination DepartmentID="1">1</Destination>
      <Title Language="zho">AA</Title>
    </PresetKey>
    <PresetKey>
      <PageID>1</PageID>
      <Type>PLU</Type>
      <Row>1</Row>
      <Column>2</Column>
      <Level>2</Level>
      <Destination DepartmentID="1">3</Destination>
      <Title Language="zho">CC</Title>
    </PresetKey>
    <PresetKey>
      <PageID>1</PageID>
      <Type>PLU</Type>
      <Row>1</Row>
      <Column>3</Column>
      <Level>1</Level>
      <Destination DepartmentID="1">4</Destination>
```

```
      <Title Language="zho">DD</Title>
    </PresetKey>
  </PresetDefinition>
</Data>
```

3.6.5 Image File

XML node name: File

XML Node	XML Node Type	Data Type	Notes
FileType	Sub node	String	The string is "Unknown".
Name	Sub node	String	File absolute path in scale. For image, the path should start with "C:/CONFIG/BITMAPS/" following file name, e.g. "C:/CONFIG/BITMAPS/1001.BMP"
DataFile	Sub node	String	File path in PC, e.g. "C:\Temp\1001.BMP". Scale Key will read its content.

3.6.5.1 File Example

```
<?xml version="1.0" encoding="UTF-8"?>
<Data>
      <File>
        <FileType>Unknown</FileType>
        <Name>C:/CONFIG/BITMAPS/1001.BMP</Name>
        <DataFile>C:\Temp\1001.BMP</DataFile>
    </File>
</Data>
```