

Innovative Technology

POWERING TRANSACTIONS AND INTERACTIONS

ITL SDK Package Manual

Document Revision - v.9

Exported on 28/08/2025

Change History

| SDK Version | Document Version | Date | Comment |
|-------------|------------------|-------------|---|
| 3.2.6 | 9 | 20 Aug 2025 | <ul style="list-style-type: none"> Removed Global Variables and replaced with Collections Variables in Postman Added “DispenseToBezel” example to NV4000 OpenConnection request on Postman Documented serverconfig.json |
| 3.2.5 | 8 | 19 Aug 2025 | <p>Updated to CashDevice-REST-API-V1.5.2 Updated to Android-REST-API-V1.5.2.apk</p> <ul style="list-style-type: none"> Added ability to pay via bezel on N4000 <ul style="list-style-type: none"> DispenseToBezel added to OpenConnection |
| 3.2.4 | 7 | 06 Aug 2025 | <p>Updated to Android-REST-API-V1.5.1.apk based on below. Updated to CashDevice-RestAPI-V1.5.1</p> <ul style="list-style-type: none"> .NET4.8 and .NET8 supported (Windows) Added Multi Escrow Added Smart Currency Note Cashbox Levels functions added serverconfig.json added allowing configuration of <ul style="list-style-type: none"> Server Port Allowed CORS Origins Auto cashbox level clearing Auto rotation of log files /SendCustomCommand now supports raw SSP responses |
| 3.2.3 | 6 | 16 Jul 2025 | <p>Documented new Multi-Escrow endpoints Documented new Smart Currency endpoints Documented new NoteCashboxLevels endpoints</p> |
| 3.2.3 | 5 | 02 Jun 2025 | <p>REST_API_Android updated to 1.4.3, based on REST_API 1.4.4 Removed C# .NET8 / .NET4.8 - ITLDeviceTestApp</p> |
| 3.2.2 | 4 | 13 May 2025 | <p>REST_API updated to 1.4.4 ITLDeviceTestApp updated to 1.7.0 pre-release Improve logging Import latest ITLCashDevice.dll and ITLComms.dll (V1.1.2) Auto disconnect to device after resetting Fix Smart Empty and Float bugs if response is not OK</p> |
| 3.2.1 | 3 | 03 May 2025 | SDK documentation added |
| 3.2.0 | 2 | 02 Feb 2025 | <p>Updated REST to 1.4.1 (.Net8 & Android) Updated CSharp.Net4.8 to 1.6.9</p> |
| 3.1.0 | 1 | 27 Jan 2025 | .NET8 Initial Release |

Welcome to the ITL SDK Package documentation!

The below contents links allow you to browse through the ITL SDK Package documentation. We recommend and encourage you to start with the [ITL SDK Package Introduction](#) to get an initial feel and understanding of what the ITL SDK Package offers and provides.

For any questions, issues or concerns please feel free to get in touch with us at any time. Our Customer Service Team is always happy to assist wherever possible. You can use our [Helpdesk](#), send an email to support@innovative-technology.com or call one of our [offices](#).

Contents

- [ITL SDK Package Introduction](#)
- [For Windows Applications](#)
- [For Linux Applications](#)
- [For Android Applications](#)
- [Cash Device REST API](#)

ITL SDK Package Introduction

Contents

- Purpose of this document
- General Description
- Library Versions
- Supported Runtimes
- ITL SDK Package Folder Structure
 - ...\\ITL_SDK_Package_vX.y.z
 - ...\\ITL_SDK_Package_vX.y.z\\REST_API
 - ...\\ITL_SDK_Package_vX.y.z\\REST_API_Android
- API Structure
 - Application Example using the REST Interface
- Product Support
- Test Scripts
- ITL SDK Package Version Convention

Purpose of this document

This manual is intended for those who would like to implement an ITL cash handling device into a host machine application utilizing the high-level APIs from Innovative Technology Ltd. This manual provides a software engineer with the ITL REST endpoint to control a cash handling device from a host machine application via high-level API. The high-level APIs were developed by ITL to ease and speed up integrations and to reduce the time to market for our customers and partners.

It is recommended that you study this manual carefully and use it throughout the entire host software development. If you have any question or do not understand any part of this manual please contact support@innovative-technology.com.



Low level SSP source code examples are available upon request. However, implementing low level SSP makes the integration much more complex, increases the likelihood of incorrect or missing error handling and extends the development time. The available SSP source code examples are more than 10 years old and not maintained. It is up to the developer to debug and edit to their needs.

General Description

The ITL SDK Package consist of a high-level REST API that provides all the tools to develop an application on various platforms using ITL products, e.g NV4000, Spectral Payout, SMART Coin System, Banknote Validators.

Library Versions

.NET8.0 is recommended when running the REST API, and is required for Linux. The reason being is the available long-term support (LTS) and life-cycle policy from Microsoft for this version. Please refer to the links below for further information.

<https://learn.microsoft.com/en-us/lifecycle/faq/dotnet-framework>

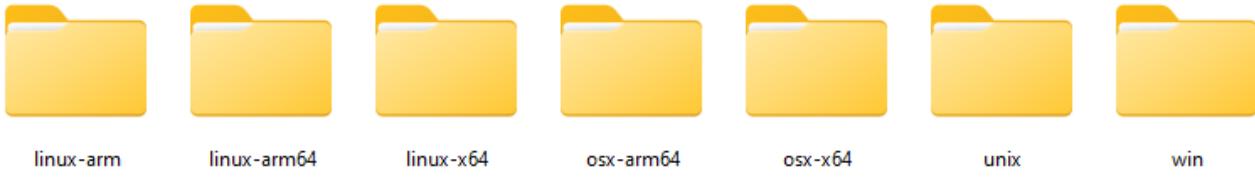
<https://dotnet.microsoft.com/en-us/platform/support/policy>

.NET Framework 4.8 is also available for Windows.

Supported Runtimes

The high-level .NET8.0 REST API currently supports the following runtimes.

...\\ITL_SDK_Package_v3.2.6\\REST_API\\CashDevice-REST-API-V1.5.2\\.NET_8.0\\runtimes



ITL SDK Package Folder Structure

...\\ITL_SDK_Package_vX.y.z

Main SDK package folder.

...\\ITL_SDK_Package_vX.y.z\\REST_API

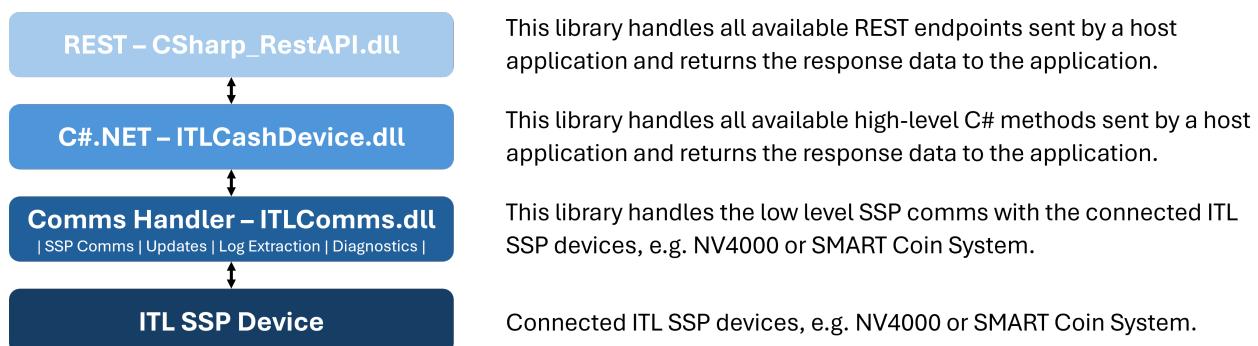
Contains the ITL REST server application and library that handles the available REST endpoints sent by a host application and returns the response data to the application. It also includes a Collection of available API endpoints with the Collections variables that can be imported into [Postman](#).

...\\ITL_SDK_Package_vX.y.z\\REST_API_Android

Contains a REST server application for Android (.apk) that handles the available REST endpoints sent by a host application and returns the response data to the application. It also includes a Collection of available API endpoints that can be imported into the [API Tester](#) app.

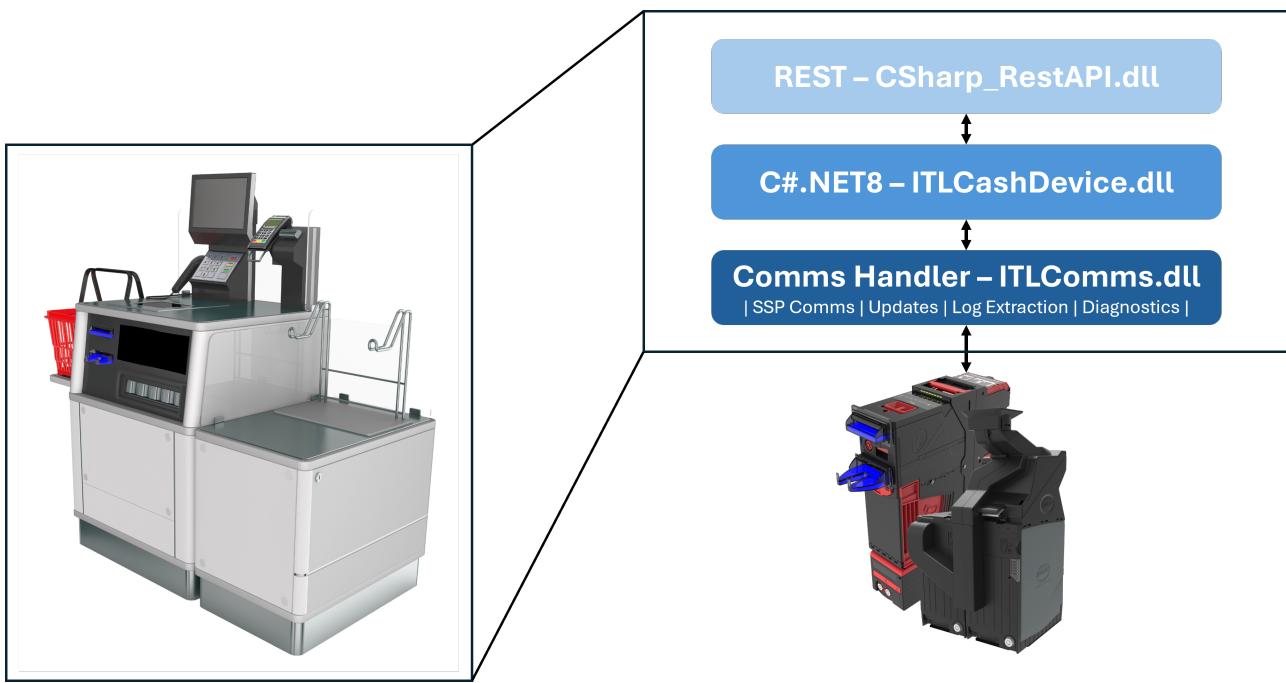
API Structure

The API structure consists of three layers as show below.



The REST library is linked to the C#.NET library that includes classes for the comms handler that does the low level SSP communication with the connected ITL SSP devices.

Application Example using the REST Interface



Product Support

Product specific information such as FAQs, user manuals, test scripts or currency datasets can be found in the [Support Hub](#) on the [ITL website](#). A [registered website account](#) is required to access the information and files within the support hub.

Test Scripts

ITL have developed generic tests scripts for various products. The test scripts are like a check list that cover basic machine design requirements in environmental conditions, power requirements, mechanical integration and software integration. The test scripts are also available in the [Support Hub](#) on the [ITL website](#).



Even for routine engineering companies ITL highly recommends to apply the test scripts to new machine developments before field trial to minimise the risk of potential field issues.

ITL SDK Package Version Convention

ITL_SDK_Package_vX.y.z

v = version

X = conceptional release

(1 = initial SSP source code examples, 2 = SSP source code examples grouped by product category, e.g. Banknote Validator or Banknote Recyclers, 3 = introduction of high-level APIs)

y = major software release

(0 = initial C#.NET6, 1 = C#.NET8, 2 = .NET8.0 & .NET Framework 4.8 for REST Only)

z = minor API feature updates or improvements

(0 = initial release)

For Windows Applications

Contents

- Description
- Installing .NET 8 if required
- Download the latest SDK
- Instructions on setting up the local server
 - Configuring the Server
 - Running the Server
- Postman Collection
 - Collection Variables
 - Authenticate Request Body
- Order of requests for initial setup

Description

This document will advise how to run the CashDevice-RestAPI.exe on your local Microsoft Windows PC.

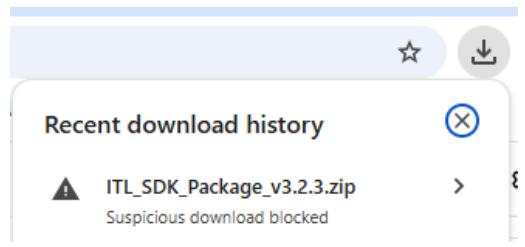
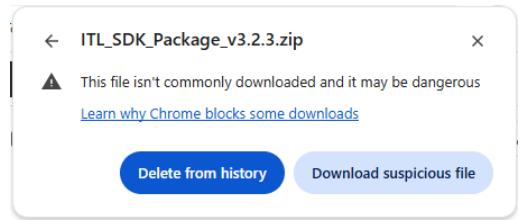
Installing .NET 8 if required

.NET Core 8 is recommended to run this API, due to Microsoft's LTS. To download, please navigate to the following link.

<https://dotnet.microsoft.com/en-us/download/dotnet/8.0>

.NET Framework 4.8 is also included.

Download the latest SDK

| | |
|---|--|
| Download the latest SDK from the following link: |  |
| If you receive a download warning, please select the item to accept the download / keep the file. |  |
| Select 'Download suspicious file' or similar depending on default web browser. |  |

Instructions on setting up the local server

Configuring the Server

- Navigate to the **CashDevice-REST-API-Vx.x.x** directory.

- Navigate into the **.NET_8.0** or **.NET_Framework_4.8** directory.
- Using a text editor, edit the **serverconfig.json** to meet your needs and save the file.
 - Port
 - AllowedOrigins (CORS)
 - AutoClearingCashboxLevels (Automatically 0 cashbox levels when the cashbox is removed and replaced)

Running the Server

- Navigate to the **CashDevice-REST-API-Vx.x.x** directory.
- Navigate into the **.NET_8.0** or **.NET_Framework_4.8** directories.
- Launch the **CashDevice-RestAPI.exe**
- The local server will boot up showing you the URL and port.

```

Logging initialised. Log file path: C:\ITL SDK package\ITL_SDK_1\REST API Logs\Cash-Device-REST-API_LOG_2025_08_20_11_39_44.log

Cash Device REST API - Version: 1.5.2.0
ITLCashDevice.dll - Version: 1.3.0.0
ITLComms.dll - Version: 1.3.0.0

info: Microsoft.Hosting.Lifetime[14]
      Now listening on: http://localhost:5000
info: Microsoft.Hosting.Lifetime[0]
      Application started. Press Ctrl+C to shut down.
info: Microsoft.Hosting.Lifetime[0]
      Hosting environment: Production
info: Microsoft.Hosting.Lifetime[0]
      Content root path: C:\ITL SDK package\ITL_SDK_Package_v3.2.5\REST_API\CashDevice-REST-API-V1.5.2\.NET_8.0
  
```

Postman Collection

- Navigate to the **Postman** folder and import the **Cash-Device-REST-API.postman_collection.json** into the software.

Collection Variables

Navigate to the Collections Variables table and complete the Current value column. Save the changes.

| Variable | Initial value | Current value |
|-------------|--------------------------------------|--|
| baseUrl | http://localhost:5000/api/CashDevice | http://localhost:5000/api/CashDevice |
| bearerToken | | eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9eyJhbGkibWVfbmPrZSI6ImFkbWluIiBmJmljoxNzU1Njg3OTUwLCJleHAiOjE... |
| ComPort | | COM10 |
| SspAddress | | 0 |
| Currency | | GBP |
| deviceID | | N14000-COM10 |

- **baseUrl** - Ensure the baseUrl uses the correct URL shown in the Terminal, in the below example <http://localhost:5000>

```

info: Microsoft.Hosting.Lifetime[14]
      Now listening on: http://localhost:5000
  
```

| Variable | Type | Initial value | Current value |
|---|---------|--------------------------------------|--------------------------------------|
| baseUrl | default | http://localhost:5000/api/CashDevice | http://localhost:5000/api/CashDevice |
| <ul style="list-style-type: none"> • bearerToken - Created and saved to the variables table automatically upon sending the Authenticate request. • ComPort - Set the correct COM port for the connected validator. COM Port can be obtained from Windows Device Manager. • SspAddress - Set the correct SSP Address for the connected device. • Currency - Set the 3 character currency code for the connected validator. | | | |
| baseURL | default | http://localhost:5000/api/CashDevice | http://localhost:5000/api/CashDevice |

- **deviceID** - Received and saved to the variables table **automatically** as part of the [OpenConnection](#) request.

 The `deviceID` provided in the response should be used as a request parameter in subsequent requests. This `deviceID` targets the specific device to receive SSP commands. For example, if your computer is connected to an NV4000 via port COM17 and a Smart Coin System on COM4, the device IDs would be `NV4000-COM17` and `SMART_COIN_SYSTEM-COM4`, respectively. To disable the acceptor on NV4000, send a request to `{{server_url}}/DisableAcceptor?deviceID=NV4000-COM17`

Authenticate Request Body

```
{  
    "Username": "admin",  
    "Password": "password"  
}
```

Order of requests for initial setup

1. [Authenticate](#)
2. [OpenConnection](#)

 This endpoint includes: **InitialiseDevice -> OpenDevice -> ConnectDevice -> StartDevice**.

For Linux Applications

Contents

- [Introduction](#)
 - [Pre-Requisites](#)
 - [Download and install required resources](#)
 - [COM Port Configurations](#)
 - [Running the ITL Rest Server](#)
 - [Making REST request using curl](#)
 - [Making REST request using Postman](#)
-

Introduction

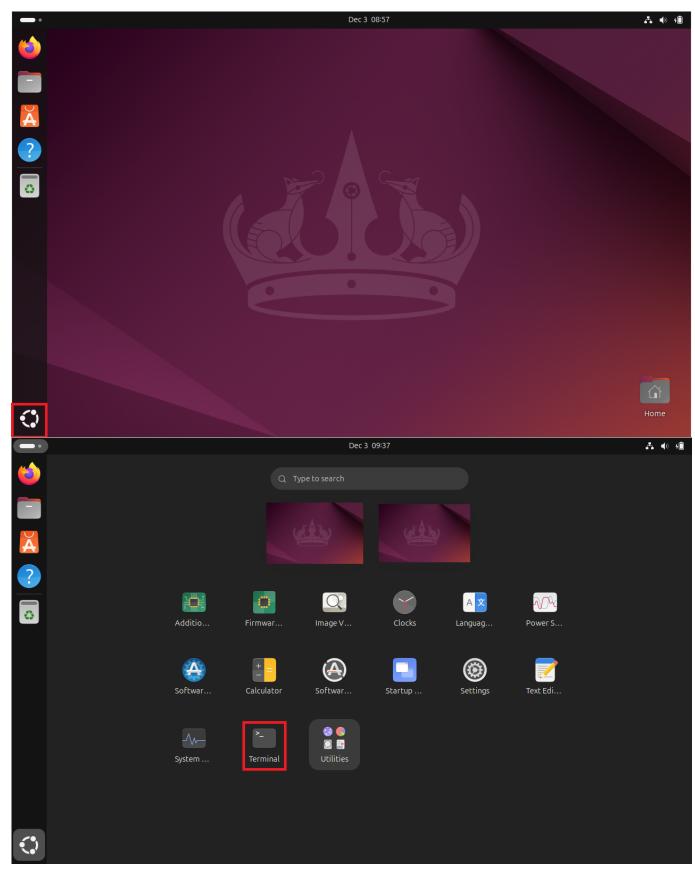
The following describes how to run the ITL REST server using the CashDevice_RestAPI.dll under Linux, make endpoint calls to it and receive response data. There are many different Linux distributions available such as Ubuntu or Debian. Hence, there are many different ways to make REST calls in Linux respectively how to get there. This documentation has been compiled using an Oracle VirtualBox running Ubuntu 24.04.1 LTS.

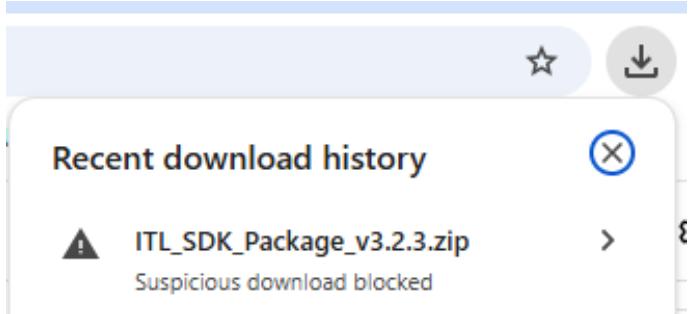
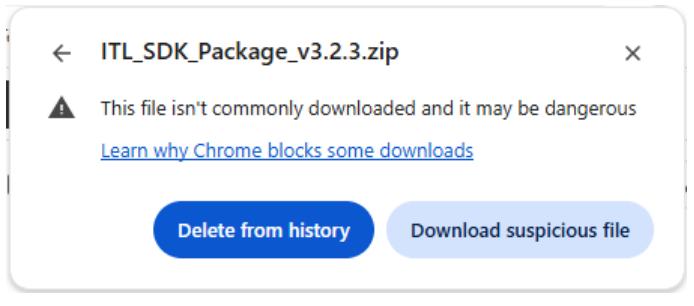
Pre-Requisites

- Linux distribution installed or running on a virtual machine
 - User account with sudo privileges
 - ITL cash handling device
-

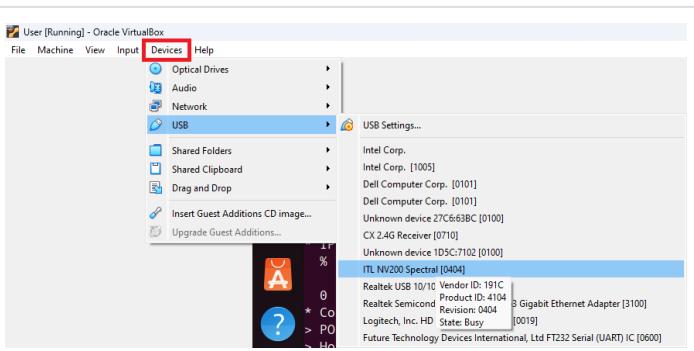
Download and install required resources

Click on “Show Apps” on the bottom left and open the “Terminal”



| | |
|---|--|
| Download and install latest distribution packages | <pre>sudo apt-get install -y upgrade</pre> |
| Download and install the .NET8 SDK | <pre>sudo apt-get install -y dotnet-sdk-8.0</pre> |
| Download and install the .NET8 runtime | <pre>sudo apt-get install -y dotnet-runtime-8.0</pre> |
| Download the latest SDK from the following link: |  |
| If you receive a download warning, please select the item to accept the download / keep the file. |  |
| Select 'Download suspicious file' or similar depending on default web browser. |  |

COM Port Configurations

| | |
|---|--|
| Connect an ITL cash handling device and add the USB connection to the virtual machine |  |
| Give user permission to use tty COM port (a restart of the Terminal may be required) | <pre>sudo usermod -a -G dialout \$USER</pre> |

Find out which COM port this has been assigned to. Direct USB usually generates a COM port at /dev/ttyACM0 and the IF17 generates COM port at /dev/ttyUSB0. The command on the right hand side will output the COM port. This will be needed later for the OpenConnection REST endpoint

```
sudo dmesg | grep tty
```

Running the ITL Rest Server

Go into the CashDevice-REST-API-Vx.x.x/.NET_8.0 directory/

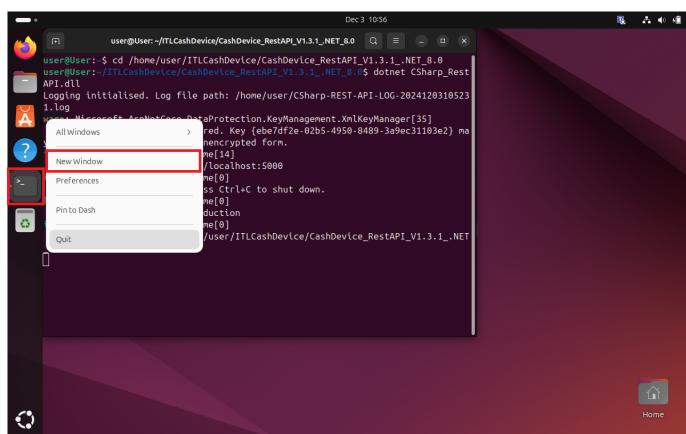
```
cd /[PATH]/CashDevice-REST-API-Vx.x.x/.NET_8.0/
```

Run the CashDevice_RestAPI.dll. This will then start the ITL REST server listening on http://localhost:5000

```
dotnet CashDevice_RestAPI.dll
```

Making REST request using curl

Open a new Terminal window



Establish a connection to the localhost on port 5000

```
curl -v localhost:5000
```

See also <https://curl.se/docs/manpage.html>

Request a Bearer Token for authorisation of upcoming REST requests and output in the Terminal window

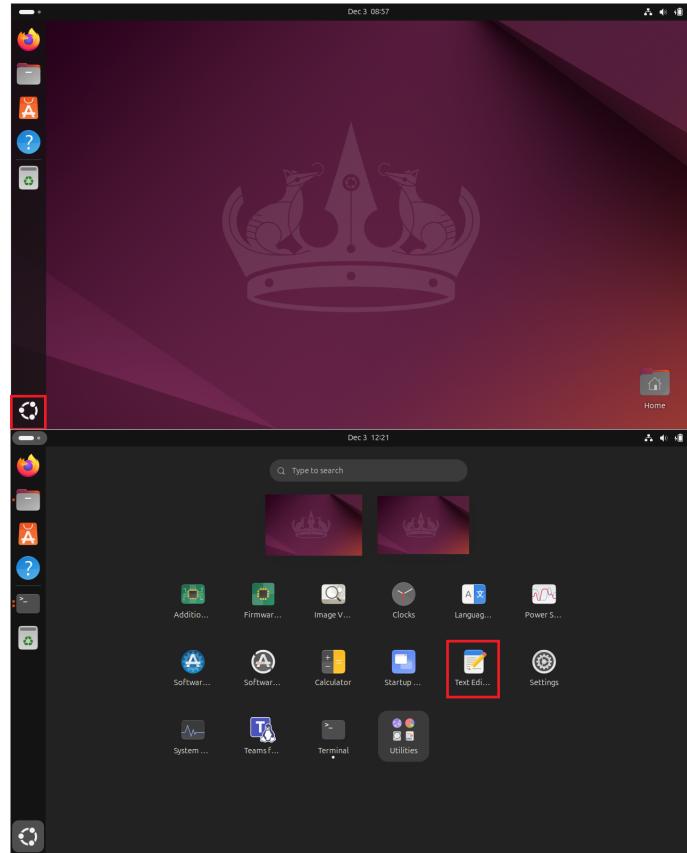
```
curl -v --json '{"Username": "admin", "Password": "password"}' localhost:5000/api/Users/Authenticate | jq
```

See also [REST API - Authenticate](#)

To output the Bearer Token into a .json file. This can then be referenced to in upcoming curl requests instead of copying and pasting the bearer token into upcoming requests

```
curl -v --json '{"Username": "admin", "Password": "password"}' localhost:5000/api/Users/Authenticate | jq .> /[PATH]/BearerToken.json
```

Instead of writing the json body into the curl request you can also reference to a .json file within the curl request. Click on “Show Apps” on the bottom left and open the Text Editor

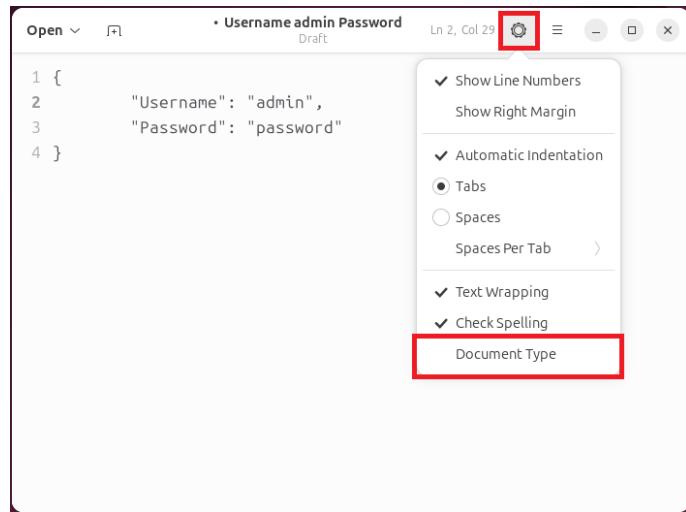


Add the json body text into the text editor

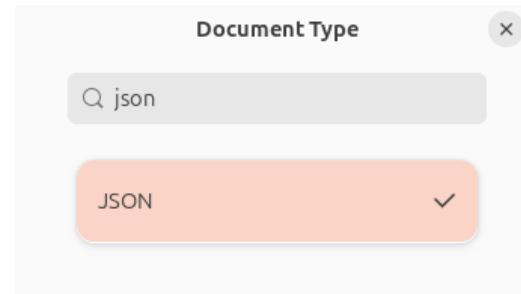
A screenshot of a text editor window. The title bar says "Open" and "Username admin Password Draft". The text area contains the following JSON code:

```
1 {
2     "Username": "admin",
3     "Password": "password"
4 }
```

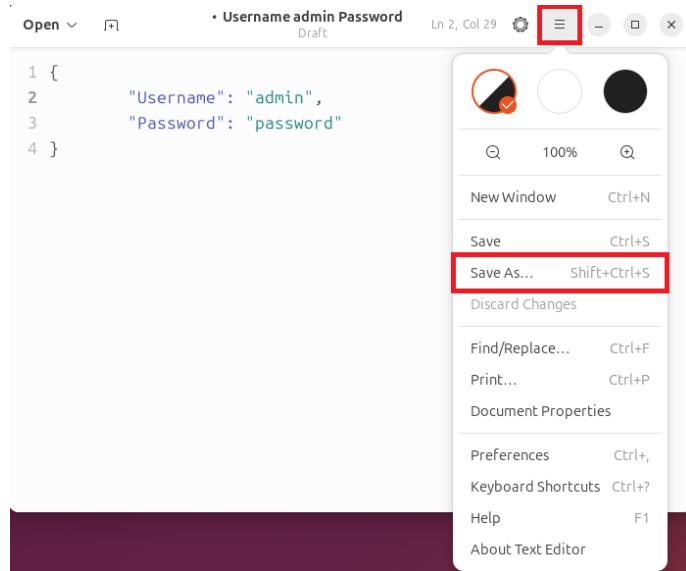
Click on the settings gear, select “Document Type”



Type json into the search bar and select “JSON”, close the Document Type



Click on the button with the 3 lines, select “Save As”, give the file a name, chose a path where to save the file and click on “Save”



Request a Bearer Token using a .json file

```
curl -v --json @/[PATH]/Authenticate.json
localhost:5000/api/Users/Authenticate | jq
```

Generate a .json file for the OpenConnection request in the same way as for the Bearer Token above or download the example below. This is an example for a Spectral Payout connected to COM port ttyACM0, SSP address 0, default encryption key (decimal), EUR5-500, all notes enabled and routed to payout, enabled acceptor and payout module with escrow mode disabled as shown on the right.



```
{
  "ComPort": "/dev/ttyACM0",
  "SspAddress": 0,
  "LogFile Path": "[PATH]/CashDevice.log",
  "EncKey": 81985526925837671,
  "SetInhibits": [
    { "Denomination": "500 EUR", "Inhibit": false },
    { "Denomination": "1000 EUR", "Inhibit": false },
    { "Denomination": "2000 EUR", "Inhibit": false },
    { "Denomination": "5000 EUR", "Inhibit": false },
    { "Denomination": "10000 EUR", "Inhibit": false },
    { "Denomination": "20000 EUR", "Inhibit": false },
    { "Denomination": "50000 EUR", "Inhibit": false }
  ],
  "SetRoutes": [
    { "Denomination": "500 EUR", "Route": 7 },
    { "Denomination": "1000 EUR", "Route": 7 },
    { "Denomination": "2000 EUR", "Route": 7 },
    { "Denomination": "5000 EUR", "Route": 7 },
    { "Denomination": "10000 EUR", "Route": 7 },
    { "Denomination": "20000 EUR", "Route": 7 },
    { "Denomination": "50000 EUR", "Route": 7 }
  ],
  "EnableAcceptor": true,
  "EnableAutoAcceptEscrow": true,
  "EnablePayout": true
}
```

See also [REST API - OpenConnection](#)

Send the OpenConnection request

```
curl -v -H "Authorization: Bearer [Bearer Token]" --json @/[PATH]/OpenConnectionSpectralPayout.json
localhost:5000/api/CashDevice/OpenConnection | jq
```

Part of the response data is the “deviceID” that is required as a parameter for further REST API requests

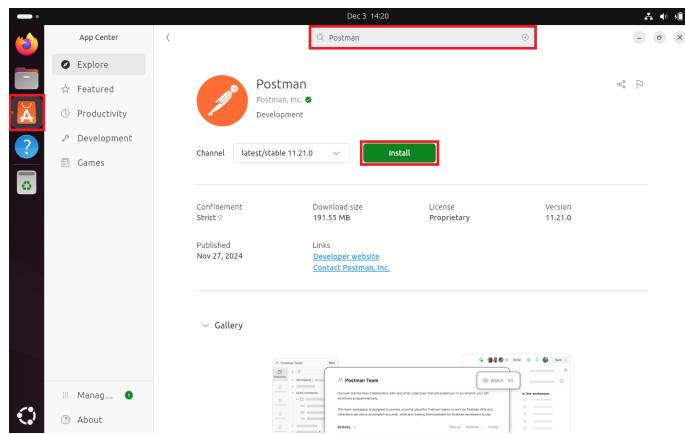
Send GetDeviceStatus with “deviceID” parameter

```
curl -v -H "Authorization: Bearer [Bearer Token]" localhost:5000/api/CashDevice/GetDeviceStatus?deviceID=SEPCTRAL_PAYOUT-/dev/ttyACM0 | jq
```

See also [REST API - GetDeviceStatus](#)

Making REST request using Postman

Postman is available in Linux Ubuntu. Click on the “App Center” icon on the left hand side, search for “Postman” and install it. You will be prompted to create an account or login to an existing account online before the app can be used



The only difference making REST request using Postman in Linux compared to [Windows](#) is the COM port reference in the [OpenConnection](#) request json body. Instead of referencing the COM port number Linux requires the path where the USB driver file is stored

Example for Windows:

```
"ComPort": "COM3"
```

Example for Linux connected via direct USB:

```
"ComPort": "/dev/ttyACM0"
```

Example for Linux connected via IF17:

```
"ComPort": "/dev/ttyUSB0"
```

For Android Applications

Contents

- [Introduction](#)
- [Pre-requisites](#)
- [Download and install the required resources](#)
- [Running the ITL REST Server](#)
- [Sending REST Requests using API Tester](#)
- [SSP Lower-Level Logging](#)

Introduction

The following describes how to run the ITL REST server using the Android_REST_API_Vx.x.x.apk and make endpoint calls to it and receive response data. There are many different Android versions available. This documentation has been compiled using Android Version 12 and version 1.3.2 of the Android REST API.

Pre-requisites

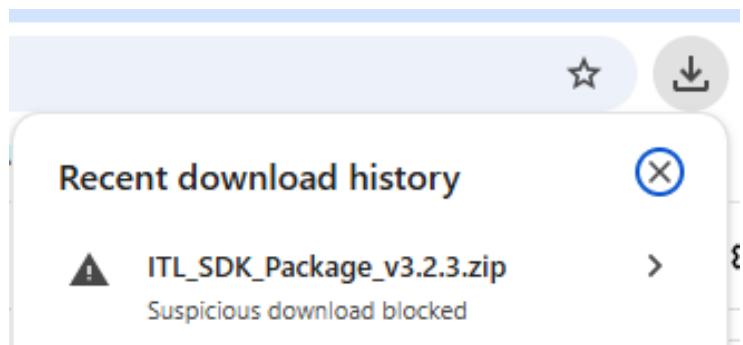
- Android Device, Android instance on a Virtual Machine or Emulator
-  Minimum Android Version 10 (API 29)
- ITL cash handling device

Download and install the required resources

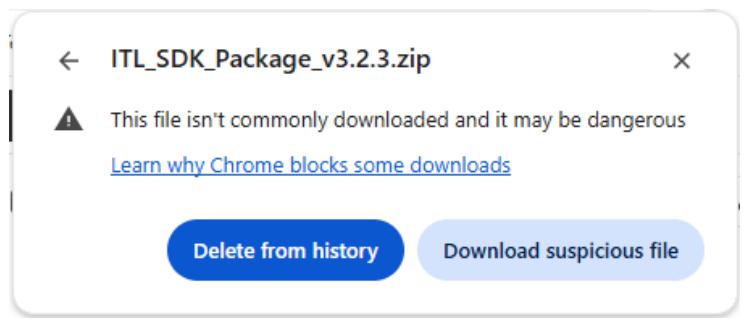
Download the latest SDK from the following link:



If you receive a download warning, please select the item to accept the download / keep the file.

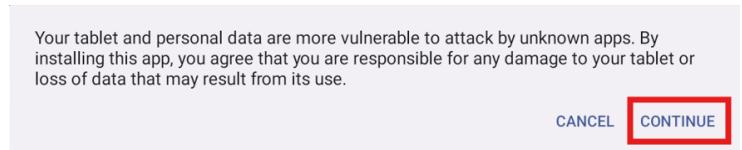
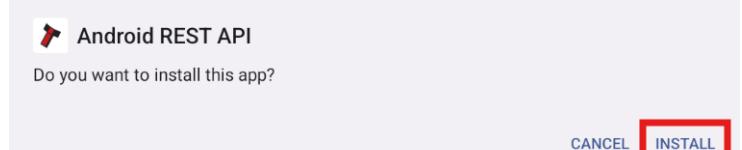
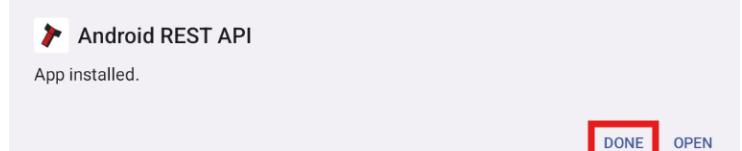
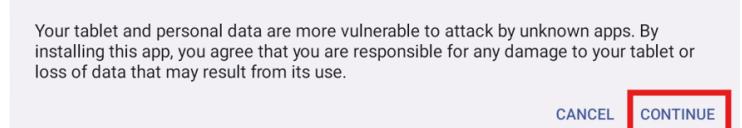
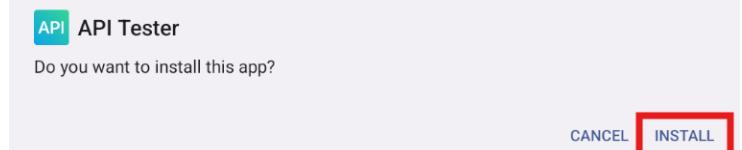


Select 'Download suspicious file' or similar depending on default web browser.

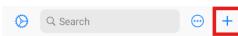


Installing the Android_REST_API_Vx.x.x.apk

Navigate to the APK download and launch the file.

| | |
|---|---|
| Select to CONTINUE on any warning messages. |  <p>Your tablet and personal data are more vulnerable to attack by unknown apps. By installing this app, you agree that you are responsible for any damage to your tablet or loss of data that may result from its use.</p> <p>CANCEL CONTINUE</p> |
| Select INSTALL to begin the process. |  <p> Android REST API Do you want to install this app?</p> <p>CANCEL INSTALL</p> |
| Select DONE once the install completes. |  <p> Android REST API App installed.</p> <p>DONE OPEN</p> |
| Download and install an API Tester like the following example: |   |
| Installing the API Tester. | |
| Navigate to the APK download and launch the file. | |
| If sideloading the APK, select to CONTINUE on any warning messages. |  <p>Your tablet and personal data are more vulnerable to attack by unknown apps. By installing this app, you agree that you are responsible for any damage to your tablet or loss of data that may result from its use.</p> <p>CANCEL CONTINUE</p> |
| Select INSTALL to begin the process. |  <p> API Tester Do you want to install this app?</p> <p>CANCEL INSTALL</p> |
| Select OPEN to launch the application. |  <p> API Tester App installed.</p> <p>DONE OPEN</p> |

Import the Collection to the API Tester.



Select the + icon.



Select any request or create one

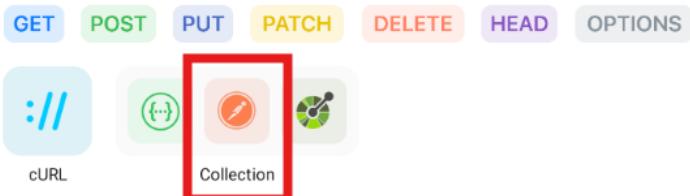
Create new request

Try some [examples](#)

Select Collection.

API Client

Create or import requests



Code

Write and run scripts on your device



Other features

Use additional formats



Select File.

Import collection via link or file

Link



Cancel

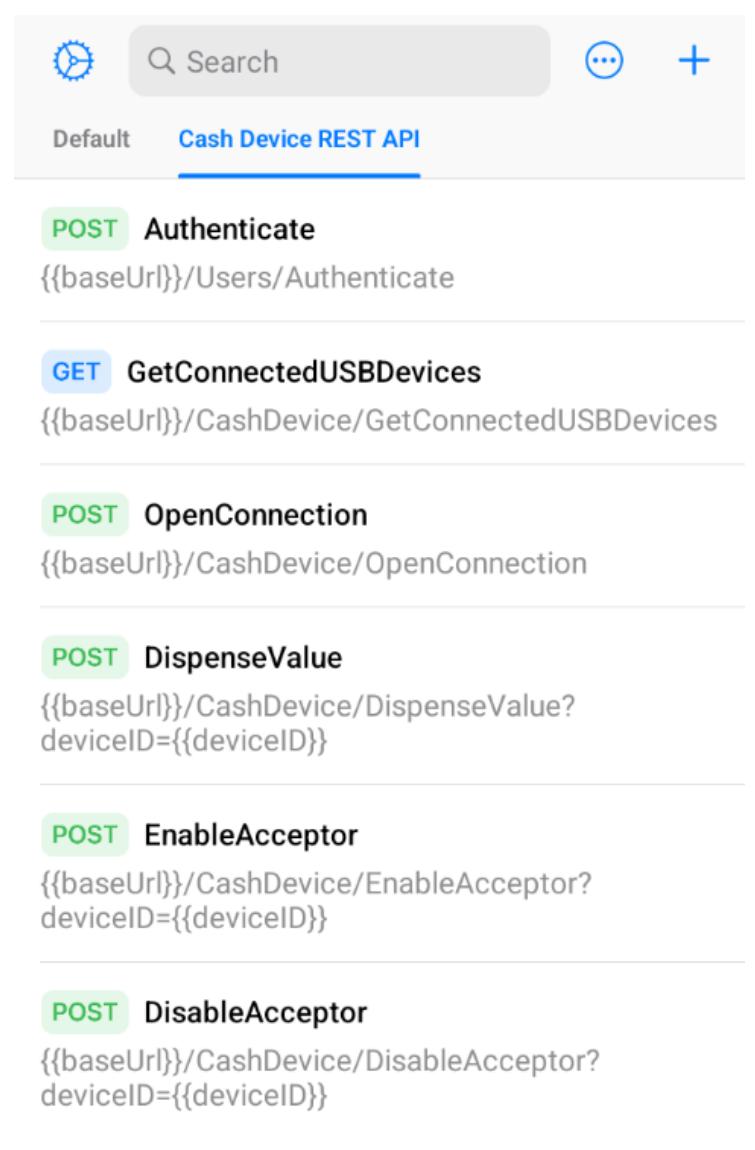
Navigate to the saved collection file `Android-API-Collection.json` and select the file.

Wait until the import completes.

 Importing...

It can take a few minutes

Once the import has completed you will find the collection list on the left of the API Tester app.



The screenshot shows the API Tester application interface. At the top, there is a search bar with a magnifying glass icon and a placeholder "Search". To the right of the search bar are three icons: a gear, a circle with three dots, and a plus sign. Below the search bar, there are two tabs: "Default" and "Cash Device REST API". The "Cash Device REST API" tab is selected and highlighted in blue. Underneath the tabs, there is a list of API endpoints:

- POST Authenticate**
{{baseUrl}}/Users/Authenticate
- GET GetConnectedUSBDevices**
{{baseUrl}}/CashDevice/GetConnectedUSBDevices
- POST OpenConnection**
{{baseUrl}}/CashDevice/OpenConnection
- POST DispenseValue**
{{baseUrl}}/CashDevice/DispenseValue?
deviceID={{deviceID}}
- POST EnableAcceptor**
{{baseUrl}}/CashDevice/EnableAcceptor?
deviceID={{deviceID}}
- POST DisableAcceptor**
{{baseUrl}}/CashDevice/DisableAcceptor?
deviceID={{deviceID}}

Running the ITL REST Server

Launch the Android REST API application



The server will launch and begin listening on
<http://127.0.0.1:5000/api/CashDevice>

- REST requests are case sensitive.
Ensure upper case letters are used where advised.

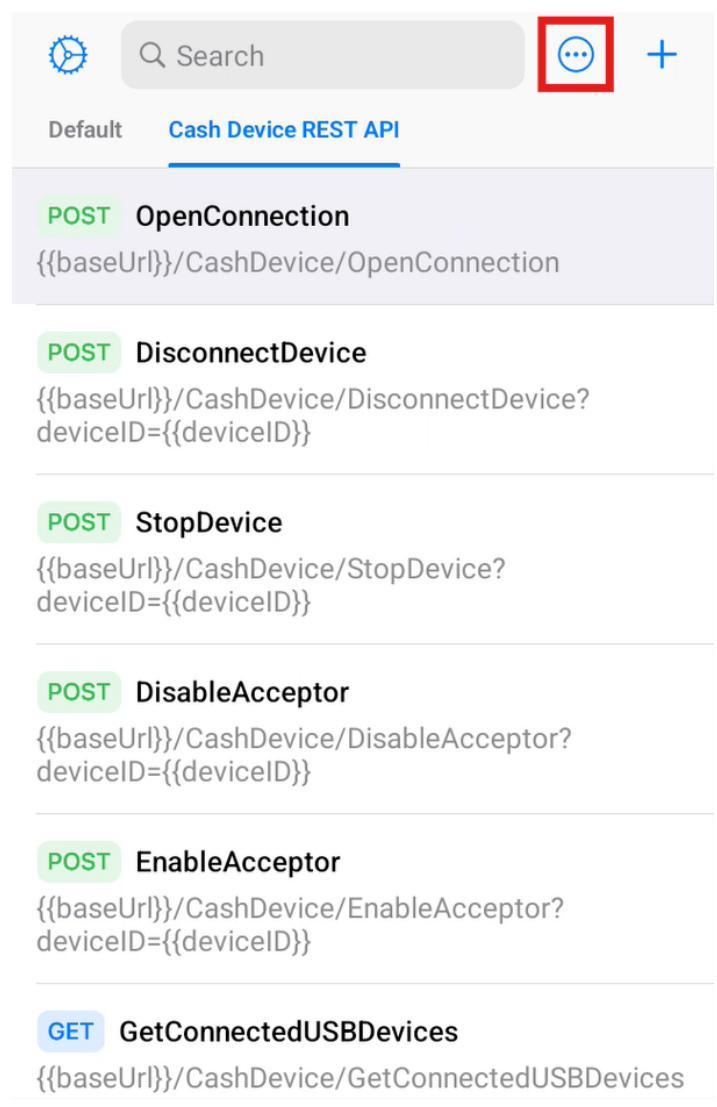


The Android REST API application must remain active at all times. Closing the application will stop the API from communicating between your software and the cash device. A fix has been implemented from version 1.3.2 to ensure the server remains active until explicitly closed, however some Android versions can pause or stop apps running in the background e.g. switching between the server app and the API Testing app. Ensure your Android version can allow apps to remain active in the background.

Sending REST Requests using API Tester

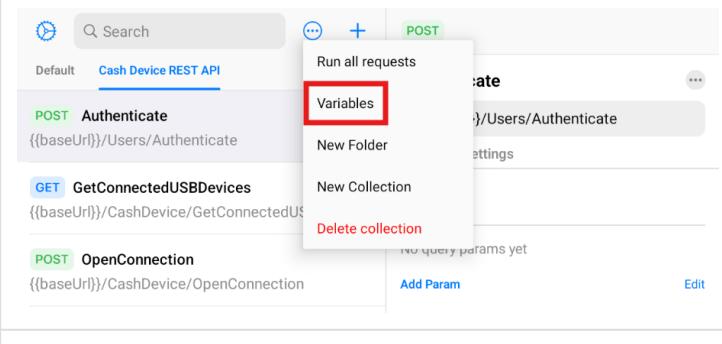
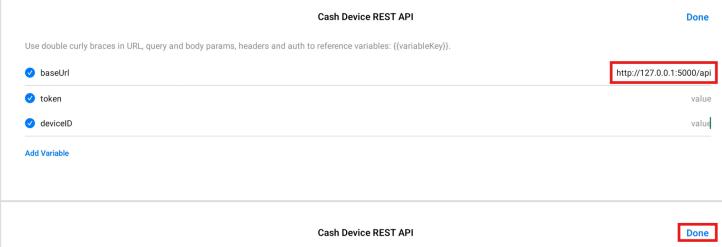
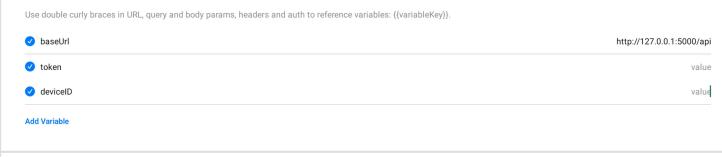
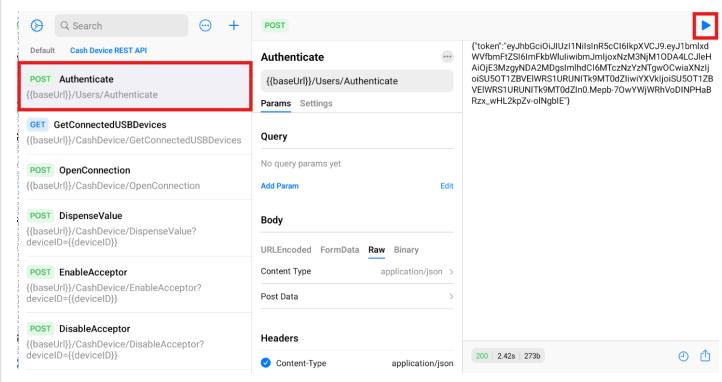
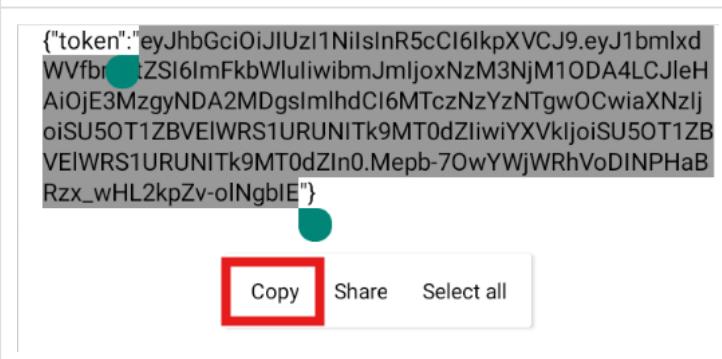
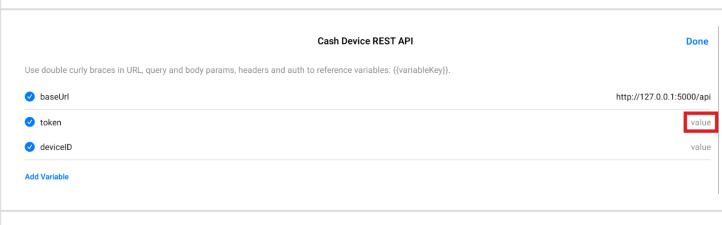
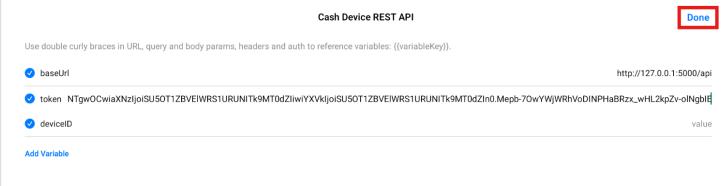
Set the baseURL in the Variables table.

To access the Variables table, select the 3 dots contained in the circle to the right of the search bar.



The screenshot shows the API Tester application interface. At the top, there's a search bar with a gear icon and a red-bordered three-dot menu icon. Below the search bar, the 'Cash Device REST API' tab is selected. The interface lists several REST endpoints:

- POST OpenConnection**
{{baseUrl}}/CashDevice/OpenConnection
- POST DisconnectDevice**
{{baseUrl}}/CashDevice/DisconnectDevice?
deviceID={{deviceID}}
- POST StopDevice**
{{baseUrl}}/CashDevice/StopDevice?
deviceID={{deviceID}}
- POST DisableAcceptor**
{{baseUrl}}/CashDevice/DisableAcceptor?
deviceID={{deviceID}}
- POST EnableAcceptor**
{{baseUrl}}/CashDevice/EnableAcceptor?
deviceID={{deviceID}}
- GET GetConnectedUSBDevices**
{{baseUrl}}/CashDevice/GetConnectedUSBDevices

| | |
|---|--|
| <p>Select Variables.</p> |  |
| <p>Ensure the correct baseUrl is set in the variables table.</p> <pre>http://127.0.0.1:5000/api</pre> |  |
| <p>Select Done to update the table and return to the request list.</p> |  |
| <p>Request a Bearer Token for authorisation of upcoming REST requests using the Authenticate request.</p> <p>Select the Authenticate request and select the play button from the top right to send the request.</p> |  |
| <p>Copy the Bearer Token to the Variables table for future requests.</p> <p>Highlight the Bearer Token and select Copy.</p> |  |
| <p>Paste the Bearer Token into the token value field of the variables table.</p> |  |
| <p>Select Done to update the table and return to the request list.</p> |  |

Send the [GetConnectedUSBDevices](#) request to obtain the correct Port.

```

{
  "Port": 8,
  "DeviceName": "NV200 Spectral",
  "VendorId": 6428,
  "ProductId": 16644
}
  
```

Navigate to the Post Data for the [OpenConnection](#) request.

No response yet
Make a request to see its response

Update the Post Data to the correct Port returned for the [GetConnectedUSBDevices](#) request.

```

{
  "Port": 8,
  "SetInhibit": 0,
  "LogFilePath": "/storage/emulated/0/Documents/ITL_SSP_logs/SSP_Log.log",
  "SetInhibit": [
    {
      "Denomination": "500 GBP",
      "Inhibit": true
    }
  ],
  "SetRoutes": [
    {
      "Denomination": "500 GBP",
      "Route": 2
    },
    {
      "Denomination": "1000 GBP",
      "Route": 3
    },
    {
      "Denomination": "2000 GBP",
      "Route": 4
    }
  ],
  "SetCashBoxPayoutLimit": [
    20,
    20,
    20,
    0
  ],
  "EnableAcceptor": true
}
  
```

Send the [OpenConnection](#) request once the Post Data has been updated. Please refer to the [OpenConnection](#) details for further information on the full Post Data body.

Update the deviceID in the variables table.

The [OpenConnection](#) request will return the correct deviceID to use on subsequent requests.

```

{
  "DeviceID": "NV4000-0",
  "isOpen": true,
  "DeviceModel": "NV4000",
  "sspProtocols": 7,
  "DeviceError": "NONE",
  "Firmware": "NVS2004301069NV4",
  "Dataset": "GBP48002",
  "MainSerialNumber": "6476329",
  "ValidatorSerialNumber": null,
  "PayoutModuleSerialNumber": null,
  "PrimaryHopperSerialNumber": null,
  "CoinFeederSerialNumber": null,
  "SecondaryHopperSerialNumber": null,
  "CoinLifterSerialNumber": null,
  "ValidatorRevision": null,
  "PayoutModuleRevision": null,
  "PrimaryHopperBuildRevision": null,
  "CoinFeederBuildRevision": null,
  "SecondaryHopperBuildRevision": null,
  "CoinLifterBuildRevision": null,
  "NV4000Fields": {
    "Recycler1SerialNumber": 6630735,
    "Recycler2SerialNumber": 6633099,
    "Recycler3SerialNumber": 6633128,
    "Recycler4SerialNumber": 0,
    "InterfaceSerialNumber": 6255817,
    "DockSerialNumber": 6216377,
    "ReplenishmentCassetteSerialNumber": 6267570,
    "ConveyorSerialNumber": 6437293,
    "RCCurrentMode": "RC_MODE_REPLENISH",
    "CurrentRC_PayoutValue": 0
  }
}
  
```

Save the deviceID to the variables table.

Cash Device REST API

Done

Use double curly braces in URL, query and body params, headers and auth to reference variables: {{variableKey}}.

baseUrl

http://127.0.0.1:5000/api

token eyJhbGciOiJIUzI1NiIsInR5cCIkVXCBJ9eyJsbmIxWVfbmFZSjlmPkblVluiwibmJmIjoxNzAxM3Ne5Mjc3LCJleHAiOjE3MzgzMjQwNzcsImhdCj6MTczOT5NyviaXNzjk

Nv4200-0

deviceID

Add Variable

Your validator will be enabled and ready to accept currency if the below parameters of the [OpenConnection](#) request are set to true.

```
"EnableAcceptor": true,  
"EnablePayout": true
```

SSP Lower-Level Logging

 Available from version 1.3.2

Android will store the Lower level SSP logs from version 1.3.2.

By default, logs are stored at the following path:

```
/storage/emulated/0/Documents/ITL_SSP_logs/{default_log_name}.log
```

If you wish to use a custom log location, simply specify it in the JSON request using the following format:

```
"LogFilePath": "{directory}/{log_file_name}.log"
```

To access the default logs on your Android device, navigate to:

Files -> Internal Storage -> Documents

Cash Device REST API

Overview

This documentation explains how to use the REST API server to establish communication between the ITL cash devices and software applications. These applications can run on various operating systems, such as Linux, and use different programming languages, such as C++, Java, etc.

API Endpoints

- [REST API - Authenticate](#)
- [Android REST API - GetConnectedUSBDevices](#)
- [REST API - UpdateCredentials](#)
- [REST API - OpenConnection](#)
- [REST API - DisconnectDevice](#)
- [REST API - StartDevice](#)
- [REST API - StopDevice](#)
- [REST API - LogRawPackets](#)
- [REST API - GetCompleteCashDevice](#)
- [REST API - GetDeviceStatus](#)
- [REST API - GetCounters](#)
- [REST API - GetAllLevels](#)
- [REST API - GetCurrencyAssignment](#)
- [REST API - SetDenominationLevel](#)
- [REST API - EnablePayout](#)
- [REST API - EnablePayoutDevice](#)
- [REST API - EnablePayoutDeviceWithByte](#)
- [REST API - DisablePayout](#)
- [REST API - EnableAcceptor](#)
- [REST API - DisableAcceptor](#)
- [REST API - SetAutoAccept](#)
- [REST API - AcceptFromEscrow](#)
- [REST API - ReturnFromEscrow](#)
- [REST API - SetMultiEscrowSize](#)
- [REST API - GetMultiEscrowSize](#)
- [REST API - GetMultiEscrowValue](#)
- [REST API - CommitMultiEscrow](#)
- [REST API - CancelMultiEscrow](#)
- [REST API - SetDenominationInhibits](#)
- [REST API - SetDenominationRoute](#)
- [REST API - GetBarcodeReaderConfiguration](#)
- [REST API - SetBarcodeReaderConfiguration](#)
- [REST API - GetBarcodeData](#)
- [REST API - DispenseValue](#)
- [REST API - Float](#)
- [REST API - SetCashboxPayoutLimit](#)
- [REST API - SmartEmpty](#)
- [REST API - SendCustomCommand](#)
- [REST API - EnableCoinMechOrFeeder](#)
- [REST API - ResetDevice](#)
- [REST API - HaltPayout](#)
- [REST API - GetRCMode](#)
- [REST API - Replenish](#)
- [REST API - RefillMode](#)
- [REST API - GetHopperOptions](#)
- [REST API - SetHopperOptions](#)
- [REST API - GetGlobalErrorCode](#)
- [REST API - GetServiceInformation](#)
- [REST API - GetServiceInformationForModule](#)

- REST API - SetServiceInformationMaintenanceReset
- REST API - SetNoPayinCount
- REST API - GetCoinAcceptance
- REST API - SetCashboxLevels
- REST API - ClearCashboxLevels
- REST API - GetCashboxLevels
- REST API - GetNoteCashboxLevels
- REST API - ClearNoteCashboxLevels
- REST API - SetSorterRoute
- REST API - GetSorterRouteAssignment
- REST API - SetPayoutLimit
- REST API - GetPayoutCount
- REST API - SetTwInMode
- REST API - comPortReadError
- REST API - GetLifterStatus
- REST API - GetLastRejectCode
- REST API - GetSmartCurrencyData
- REST API - UpdateSmartCurrencyDataset
- REST API - StartDownload
- REST API - GetDownloadStatus

REST API - Authenticate

Description

Creates a Bearer Token to be used as Authorisation on all subsequent Requests for security.

 This must be sent first to create the Bearer Token between the software and the REST Server. Only subsequent requests using the correct Bearer Token will be accepted.

Default Credentials

Username: admin

Password: password

| | |
|----------------------|--|
| Endpoint | Authenticate |
| Method | POST |
| URL | {server_url}/api/Users/Authenticate |
| Parameters | None |
| Authorisation | N/A |
| Body | <pre>{ "Username": "admin", "Password": "password" }</pre> |

Responses

| Status | Body | Notes |
|--------|---|--|
| 200 | <pre>{ "token": "eyJhb....."</pre> | OK: Provides the Bearer Token to be used for the Authorisation of subsequent requests. |
| 400 | <pre>{ "message": "Username or password is incorrect" }</pre> | Bad Request: Incorrect Credentials. |
| N/A | Error: connect ECONNREFUSED {server_url} | Unable to connect to REST Server. |

Code Examples

C#

```
var options = new RestClientOptions("http://localhost:5000")
{
    MaxTimeout = -1,
};

var client = new RestClient(options);
var request = new RestRequest("/api/Users/Authenticate", Method.Post);
request.AddHeader("Content-Type", "application/json");
var body = @"{\n    \"Username\": \"admin\",\n    \"Password\": \"password\"\n}";
request.AddStringBody(body, DataFormat.Json);
RestResponse response = await client.ExecuteAsync(request);
Console.WriteLine(response.Content);
```

NodeJS

```
var request = require('request');
var options = {
    'method': 'POST',
    'url': 'http://localhost:5000/api/Users/Authenticate',
    'headers': {
        'Content-Type': 'application/json'
    },
    body: JSON.stringify({
        "Username": "admin",
        "Password": "password"
    })
};

request(options, function (error, response) {
    if (error) throw new Error(error);
    console.log(response.body);
});
```

Python

```
import http.client
import json

conn = http.client.HTTPSConnection("localhost", 5000)
payload = json.dumps({
    "Username": "admin",
    "Password": "password"
})
headers = {
    'Content-Type': 'application/json'
}
conn.request("POST", "/api/Users/Authenticate", payload, headers)
res = conn.getresponse()
```

```
data = res.read()
print(data.decode("utf-8"))
```

Java

```
OkHttpClient client = new OkHttpClient().newBuilder()
    .build();
MediaType mediaType = MediaType.parse("application/json");
RequestBody body = RequestBody.create(mediaType, "{\r\n    \"Username\": \"admin\",\r\n    \"Password\": \"password\"\r\n}");
Request request = new Request.Builder()
    .url("http://localhost:5000/api/Users/Authenticate")
    .method("POST", body)
    .addHeader("Content-Type", "application/json")
    .build();
Response response = client.newCall(request).execute();
```

Android REST API - GetConnectedUSBDevices

Description

Returns the port address for the connected device.

 This request is used for Android only.

| | |
|----------------------|--|
| Endpoint | GetConnectedUSBDevices |
| Method | Get |
| URL | {server_url}/api/CashDevice/GetConnectedUSBDevices |
| Parameters | None |
| Authorisation | Bearer Token |
| Body | None |

Responses

| Status | Body | Notes |
|--------|--|-------------------------------|
| 200 | <pre>[{ "Port": 0, "DeviceName": "NV200 Spectral", "VendorId": 6428, "ProductId": 16644 }]</pre> | OK: Provides the port details |

Code Examples

C#

```
var options = new RestClientOptions("http://127.0.0.1:5000")
{
    MaxTimeout = -1,
};
var client = new RestClient(options);
var request = new RestRequest("/api/CashDevice/GetConnectedUSBDevices", Method.Get);
request.AddHeader("Authorization", "Bearer eyJhb.....");
RestResponse response = await client.ExecuteAsync(request);
Console.WriteLine(response.Content);
```

NodeJS

```
var request = require('request');
var options = {
  'method': 'GET',
  'url': 'http://127.0.0.1:5000/api/CashDevice/GetConnectedUSBDevices',
  'headers': {
    'Authorization': 'Bearer eyJhb.....'
  }
};
request(options, function (error, response) {
  if (error) throw new Error(error);
  console.log(response.body);
});
```

Python

```
import requests

url = "http://127.0.0.1:5000/api/CashDevice/GetConnectedUSBDevices"

payload = {}
headers = {
  'Authorization': 'Bearer eyJhb.....'
}

response = requests.request("GET", url, headers=headers, data=payload)

print(response.text)
```

Java

```
OkHttpClient client = new OkHttpClient().newBuilder()
  .build();
MediaType mediaType = MediaType.parse("text/plain");
RequestBody body = RequestBody.create(mediaType, "");
Request request = new Request.Builder()
  .url("http://127.0.0.1:5000/api/CashDevice/GetConnectedUSBDevices")
  .method("GET", body)
  .addHeader("Authorization", "Bearer eyJhb.....")
  .build();
Response response = client.newCall(request).execute();
```

REST API - UpdateCredentials

Description

Used to change the Username and Password for the [Authenticate](#) request.

 You must have Authenticated using the current credentials before updating to the new credentials.

Default Credentials

Username: admin

Password: password

| | |
|----------------------|--|
| Endpoint | Authenticate |
| Method | POST |
| URL | {server_url}/api/Users/UpdateCredentials |
| Parameters | None |
| Authorisation | Bearer Token |
| Body | <pre>{ "CurrentUsername": "admin", "CurrentPassword": "password", "NewUsername": "<new username>", "NewPassword": "<new password>" }</pre> |

Responses

| Status | Body | Notes |
|--------|---|---|
| 200 | <pre>{ "message": "Credentials updated successfully!" }</pre> | OK: Confirms the credentials have been updated. |
| 401 | Old username or password is incorrect | Unauthorised: Incorrect Credentials. |

Code Examples

C#

```
var options = new RestClientOptions("http://localhost:5000")
{
```

```

    MaxTimeout = -1,
};

var client = new RestClient(options);
var request = new RestRequest("/api/Users/UpdateCredentials", Method.Post);
request.AddHeader("Content-Type", "application/json");
request.AddHeader("Authorization", "Bearer eyJhb.....");
var body = @"{` + "\n" +
@` ""CurrentUsername"": ""admin""` + "\n" +
@` ""CurrentPassword"": ""password""` + "\n" +
@` ""NewUsername"": ""<new username>""` + "\n" +
@` ""NewPassword"": ""<new password>""` + "\n" +
@`}";
request.AddStringBody(body, DataFormat.Json);
RestResponse response = await client.ExecuteAsync(request);
Console.WriteLine(response.Content);

```

NodeJS

```

var request = require('request');
var options = {
  'method': 'POST',
  'url': 'http://localhost:5000/api/Users/UpdateCredentials',
  'headers': {
    'Content-Type': 'application/json',
    'Authorization': 'Bearer eyJhb.....'
  },
  body: JSON.stringify({
    "CurrentUsername": "admin",
    "CurrentPassword": "password",
    "NewUsername": "<new username>",
    "NewPassword": "<new password>"
  })
};

request(options, function (error, response) {
  if (error) throw new Error(error);
  console.log(response.body);
});

```

Python

```

import requests
import json

url = "http://localhost:5000/api/Users/UpdateCredentials"

payload = json.dumps({
    "CurrentUsername": "admin",
    "CurrentPassword": "password",
    "NewUsername": "<new username>",
    "NewPassword": "<new password>"
})
headers = {
    'Content-Type': 'application/json',
    'Authorization': 'Bearer eyJhb.....'
}

response = requests.request("POST", url, headers=headers, data=payload)

```

```
print(response.text)
```

Java

```
OkHttpClient client = new OkHttpClient().newBuilder()
    .build();
MediaType mediaType = MediaType.parse("application/json");
RequestBody body = RequestBody.create(mediaType, "{\"r\n  \"CurrentUsername\":\n\"admin\",\\r\\n  \"CurrentPassword\": \"password\",\\r\\n  \"NewUsername\": \"<new\nusername>\",\\r\\n  \"NewPassword\": \"<new password>\"\r\n}");
Request request = new Request.Builder()
    .url("http://localhost:5000/api/Users/UpdateCredentials")
    .method("POST", body)
    .addHeader("Content-Type", "application/json")
    .addHeader("Authorization", "Bearer eyJhb.....")
    .build();
Response response = client.newCall(request).execute();
```

REST API - OpenConnection

Description

- The **/OpenConnection** endpoint streamlines the process of initiating communication with cash-handling devices by configuring essential connection and device-specific settings. This endpoint simplifies the previous multi-step process for establishing a connection to an SSP device. Previously, the steps included: **InitialiseDevice -> OpenDevice -> ConnectDevice -> StartDevice**. With **/OpenConnection**, these operations are automated, enabling the device to be ready for cash operations such as payout and note acceptance.
- The structure of the request body and the JSON response varies based on the device model in use. For example, NV4000 requires a distinct request format and includes a different set of fields in its JSON response compared to Smart Coin System.
- To run two devices, different COM ports are required with an /OpenConnection for each. The [/GetDeviceStatus](#) must be interleaved between the two.



The `deviceID` provided in the response should be used as a request parameter in subsequent requests. This `deviceID` targets the specific device to receive SSP commands. For example, if your computer is connected to an NV4000 via port COM17 and a Smart Coin System on COM4, the device IDs would be `NV4000-COM17` and `SMART_COIN_SYSTEM-COM4`, respectively. To disable the acceptor on NV4000, send a request to `{{server_url}}/DisableAcceptor?deviceID=NV4000-COM17`

| | |
|----------------------|---|
| Endpoint | OpenConnection |
| Method | POST |
| URL | <code>{server_url}/api/CashDevice/OpenConnection</code> |
| Parameters | None |
| Authorisation | Bearer Token |

Body

Please see the example below for Spectral Payout:

```
{  
    "ComPort": "{{ComPort}}",  
    "SspAddress": {{SspAddress}},  
    "LogFilePath": "C:\\Temp\\Cash_Device_Log.log",  
    "SetInhibits": [  
        { "Denomination": "2000 {{Currency}}", "Inhibit": true },  
        { "Denomination": "5000 {{Currency}}", "Inhibit": true }  
    ],  
    "SetRoutes": [  
        { "Denomination": "500 {{Currency}}", "Route": 7 },  
        { "Denomination": "1000 {{Currency}}", "Route": 7 }  
    ],  
    "EnableAcceptor": true,  
    "EnableAutoAcceptEscrow": true,  
    "EnablePayout": true  
}
```

Please refer to the [Request Body Structure](#) below for details of the request body requirements.

Request Body Structure

| Field | Device Type | Example | Comments |
|-------------|-------------|--|--|
| ComPort | Any | "ComPort": "COM5" | |
| SspAddress | Any | "SspAddress": 0 | |
| EncKey | Any | "EncKey": 81985526925837671 | Custom eSSP Key |
| LogFilePath | Any | "LogFilePath": "C:\\Temp\\Cash_Device_Log.log" | |
| SetInhibits | Any | "SetInhibits": [{ "Denomination": "2000 GBP", "Inhibit": true }, { "Denomination": "5000 GBP", "Inhibit": false }] | |
| SetRoutes | Any | "SetRoutes": [{ "Denomination": "500 GBP", "Route": 7 }, { "Denomination": "1000 GBP", "Route": 0 }] | <ul style="list-style-type: none">• Cashbox = 0• Recycler (NV11) = 1• Recycler 1 (NV4000) = 2• Recycler 2 (NV4000) = 3• Recycler 3 (NV4000) = 4• Recycler 4 (NV4000) = 5• Replenishment Cassette (NV4000) = 6• Payout = 7 |

| Field | Device Type | Example | Comments |
|--|----------------|---|--|
| SetCoinMechInhibits | Coin Validator | "SetCoinMechInhibits": { "ValueCountryCodes": ["1 USD", "5 USD"], "Inhibit": true } | See Command 0x40 - Set Coin Mech Inhibits |
| SetHopperOptions | Coin Hopper | "SetHopperOptions": { "Reg0": 4, "Reg1": 0 } | See Command 0x50 - Set Options (Coin) |
| SetCashBoxPayoutLimit █ Not Available on NV4000 | Any | "SetCashBoxPayoutLimit": [100,0,50,30,20,5,0] | See Command 0x4E - Set Cashbox Payout Limit (Coin) or Command 0x4E - Set Cashbox Payout Limit (Note) |
| SetTwinMode | TSCS | "SetTwinMode":0 | See Sub Command 0x05 0x87 - Set Twin Mode |
| SetFeederRoutes | TSCS | "SetFeederRoutes": [{ "Denomination": "1 USD", "Route": 0 }, { "Denomination": "25 USD", "Route": 0 }, { "Denomination": "100 USD", "Route": 0 }] | See Sub Command 0x05 0x85 - Sorter Channel Set Route |
| EnableAcceptor | Any | "EnableAcceptor": true | |
| EnableAutoAcceptEscrow | Note Validator | "EnableAutoAcceptEscrow": true | Automatically accept a note held in Escrow. |
| EnablePayout | Note Validator | "EnablePayout": true | |
| DispenseToBezel | NV4000 | "DispenseToBezel": true | Dispense notes from the payin Bezel. Defaults to False if not set (Pay from tray) |

Responses

| Status | Body | Notes |
|--------|---|---|
| 200 | <pre>{ "deviceID": "SPECTRAL_DEVICE-COM18", "isOpen": true, "deviceModel": "SPECTRAL_DEVICE", "sspProtocolVersion": 7, "deviceError": "NONE", "firmware": "NVS2004301038000", "dataset": "GBP01056", "validatorSerialNumber": "4891655", "payoutModuleSerialNumber": "5175198", "validatorRevision": "7.10", "payoutModuleRevision": "1.00", "realTimeClock": "Successfully set real time clock to 28/10/2024 16:24:38.", "allLevels": [{ "countryCode": "GBP", "value": 500, "stored": 0 }, { "countryCode": "GBP", "value": 1000, "stored": 0 }, { "countryCode": "GBP", "value": 2000, "stored": 0 }, { "countryCode": "GBP", "value": 5000, "stored": 0 }], "counters": "Number of counters in set: 5\nStacked: 999\nStored: 2\nDispensed: 0\nTransferred to stack: 2\nRejected: 258\n", "inhibits": "SUCCESS: 2000 GBP INHIBITED.\nSUCCESS: 5000 GBP INHIBITED.\n", "routes": "SUCCESS: Set route for 500 GBP to PAYOUT.\nSUCCESS: Set route for 1000 GBP to PAYOUT.\n", "acceptorEnabled": true, "autoAcceptEscrowEnabled": true, "payoutEnabled": true }</pre> | OK: Successfully retrieved device information and configured essential connection and device-specific settings. |

| Status | Body | Notes |
|--------|--|---|
| | } | |
| 400 | <pre>{ "deviceID": "", "openResult": "PORT_ERROR", "error": "Failed to open device." }</pre> | Bad Request: Port Error |
| 500 | <pre>{ "DeviceID": "deviceID_value", "Error": "Error opening cash device: error_message" }</pre> | Internal Server Error: An error occurred while attempting connection. |
| N/A | Error: connect ECONNREFUSED {server_url} | Unable to connect to REST Server. |

Code Examples

C#

```
var options = new RestClientOptions("http://localhost:5000")
{
    MaxTimeout = -1,
};
var client = new RestClient(options);
var request = new RestRequest("/api/CashDevice/OpenConnection", Method.Post);
request.AddHeader("Content-Type", "application/json");
request.AddHeader("Authorization", "Bearer eyJhb.....");
var body = @"
" + "\n" +
@"  ""ComPort"": ""COM5"",
" + "\n" +
@""  ""SspAddress"": 0,
" + "\n" +
@""  ""LogFilePath"": ""C:\\Temp\\Cash_Device_Log.log"",
" + "\n" +
@""  ""EncKey"": 81985526925837671, " + "\n" +
@""  ""SetInhibits"": [
" + "\n" +
@"    { ""Denomination"": ""2000 GBP"", ""Inhibit"": true },
" + "\n" +
@"    { ""Denomination"": ""5000 GBP"", ""Inhibit"": true }
" + "\n" +
@"  ],
" + "\n" +
@""  ""SetRoutes"": [
" + "\n" +
@"    { ""Denomination"": ""500 GBP"", ""Route"": 7 },
" + "\n" +
```

```

@"      { ""Denomination"": ""1000 GBP"", ""Route"": 7 }
" + "\n" +
@"  ],
" + "\n" +
@"  ""EnableAcceptor"": true,
" + "\n" +
@"  ""EnableAutoAcceptEscrow"": true,
" + "\n" +
@"  ""EnablePayout"": true
" + "\n" +
@"}\n" +
@"";
request.AddStringBody(body, DataFormat.Json);
RestResponse response = await client.ExecuteAsync(request);
Console.WriteLine(response.Content);

```

NodeJS

```

var request = require('request');
var options = {
  'method': 'POST',
  'url': 'http://localhost:5000/api/CashDevice/OpenConnection',
  'headers': {
    'Content-Type': 'application/json',
    'Authorization': 'Bearer eyJhb.....'
  },
  body: JSON.stringify({
    "ComPort": "COM5",
    "SspAddress": 0,
    "LogFile Path": "C:\\Temp\\Cash_Device_Log.log",
    "EncKey": 81985526925837660,
    "SetInhibits": [
      {
        "Denomination": "2000 GBP",
        "Inhibit": true
      },
      {
        "Denomination": "5000 GBP",
        "Inhibit": true
      }
    ],
    "SetRoutes": [
      {
        "Denomination": "500 GBP",
        "Route": 7
      },
      {
        "Denomination": "1000 GBP",
        "Route": 7
      }
    ],
    "EnableAcceptor": true,
    "EnableAutoAcceptEscrow": true,
    "EnablePayout": true
  })
};

request(options, function (error, response) {

```

```

if (error) throw new Error(error);
console.log(response.body);
});

```

Python

```

import requests
import json

url = "http://localhost:5000/api/CashDevice/OpenConnection"

payload = json.dumps({
    "ComPort": "COM5",
    "SspAddress": 0,
    "LogFilePath": "C:\\Temp\\Cash_Device_Log.log",
    "EncKey": 81985526925837660,
    "SetInhibits": [
        {
            "Denomination": "2000 GBP",
            "Inhibit": True
        },
        {
            "Denomination": "5000 GBP",
            "Inhibit": True
        }
    ],
    "SetRoutes": [
        {
            "Denomination": "500 GBP",
            "Route": 7
        },
        {
            "Denomination": "1000 GBP",
            "Route": 7
        }
    ],
    "EnableAcceptor": True,
    "EnableAutoAcceptEscrow": True,
    "EnablePayout": True
})
headers = {
    'Content-Type': 'application/json',
    'Authorization': 'Bearer eyJhb....'
}

response = requests.request("POST", url, headers=headers, data=payload)

print(response.text)

```

Java

```

OkHttpClient client = new OkHttpClient().newBuilder()
    .build();
MediaType mediaType = MediaType.parse("application/json");
RequestBody body = RequestBody.create(mediaType, "{\r\n    \"ComPort\": \"COM5\", \r\n    \"SspAddress\": 0,\r\n    \"LogFilePath\": \"C:\\\\Temp\\\\Cash_Device_Log.log\", \r\n    \"EncKey\": 81985526925837671, \r\n    \"SetInhibits\": [\r\n        {\r\n            \"Denomination\": \"2000 GBP\", \r\n            \"Inhibit\": true\r\n        }\r\n    ]\r\n}");

```

```
\"Inhibit\": true }\\r\\n ],\\r\\n \"SetRoutes\": [\\r\\n     { \"Denomination\": \"500 GBP\", \"Route\": 7 },\\r\\n     { \"Denomination\": \"1000 GBP\", \"Route\": 7 }\\r\\n ],\\r\\n \"EnableAcceptor\": true,\\r\\n \"EnableAutoAcceptEscrow\": true,\\r\\n \"EnablePayout\": true\\r\\n}\\r\\n);\nRequest request = new Request.Builder()\n    .url("http://localhost:5000/api/CashDevice/OpenConnection")\n    .method("POST", body)\n    .addHeader("Content-Type", "application/json")\n    .addHeader("Authorization", "Bearer eyJhb.....")\n    .build();\nResponse response = client.newCall(request).execute();
```

REST API - DisconnectDevice

Description

Sets the device state to not connected and closes the communication port and threads.

| | |
|----------------------|---|
| Endpoint | DisconnectDevice |
| Method | POST |
| URL | {server_url}/api/CashDevice/DisconnectDevice |
| Parameters | <ul style="list-style-type: none">• deviceID (string): The value of "deviceID" obtained from the response to the OpenConnection request |
| Authorisation | Bearer Token |
| Body | None |

Responses

| Status | Body | Notes |
|--------|--|---|
| 200 | NV200S: Device disconnected and cleaned up successfully. | OK: If the device disconnected successfully. |
| 404 | Device with ID NV200 not found. | Not Found: Device ID Incorrect. |
| 500 | | Internal Server Error: If there is an error disconnecting the device. |

Code Examples

C#

```
var options = new RestClientOptions("http://localhost:5000")
{
    MaxTimeout = -1,
};
var client = new RestClient(options);
var request = new RestRequest("/api/CashDevice/DisconnectDevice?deviceID=NV200S",
    Method.Post);
RestResponse response = await client.ExecuteAsync(request);
Console.WriteLine(response.Content);
```

NodeJS

```
var request = require('request');
var options = {
  'method': 'POST',
  'url': 'http://localhost:5000/api/CashDevice/DisconnectDevice?deviceID=NV200S',
  'headers': {
  }
};
request(options, function (error, response) {
  if (error) throw new Error(error);
  console.log(response.body);
});
```

Python

```
import http.client

conn = http.client.HTTPSConnection("localhost", 5000)
payload = ''
headers = {}
conn.request("POST", "/api/CashDevice/DisconnectDevice?deviceID=NV200S", payload,
headers)
res = conn.getresponse()
data = res.read()
print(data.decode("utf-8"))
```

Java

```
OkHttpClient client = new OkHttpClient().newBuilder()
  .build();
MediaType mediaType = MediaType.parse("text/plain");
RequestBody body = RequestBody.create(mediaType, "");
Request request = new Request.Builder()
  .url("http://localhost:5000/api/CashDevice/DisconnectDevice?deviceID=NV200S")
  .method("POST", body)
  .build();
Response response = client.newCall(request).execute();
```

REST API - StartDevice

Description

Creates a new thread to ‘run’ with an SSP device. The thread is essentially a state machine which runs through the some initial setup of the device and retrieving information from it and then enters a normal ‘run’ routine, polling the device for any new events.



Need to send “ConnectDevice” request before sending “StartDevice”.

| | |
|----------------------|---|
| Endpoint | StartDevice |
| Method | POST |
| URL | {server_url}/api/CashDevice/StartDevice |
| Parameters | <ul style="list-style-type: none">deviceID (string): The value of "deviceID" obtained from the response to the OpenConnection request |
| Authorisation | Bearer Token |
| Body | None |

Responses

| Status | Body | Notes |
|--------|--------------------------------------|--|
| 200 | NV200S: Device started successfully. | OK: If the device started successfully. |
| 500 | | Internal Server Error: If there is an error starting the device. |
| | | |

Code Examples

C#

```
var options = new RestClientOptions("http://localhost:5000")
{
    MaxTimeout = -1,
};
var client = new RestClient(options);
var request = new RestRequest("/api/CashDevice/StartDevice?deviceID=NV200S",
    Method.Post);
request.AddHeader("Authorization", "Bearer eyJhb.....");
RestResponse response = await client.ExecuteAsync(request);
```

```
Console.WriteLine(response.Content);
```

NodeJS

```
var request = require('request');
var options = {
  'method': 'POST',
  'url': 'http://localhost:5000/api/CashDevice/StartDevice?deviceID=NV200S',
  'headers': {
    'Authorization': 'Bearer eyJhb.....'
  }
};
request(options, function (error, response) {
  if (error) throw new Error(error);
  console.log(response.body);
});
```

Python

```
import http.client

conn = http.client.HTTPSConnection("localhost", 5000)
payload = ''
headers = {
  'Authorization': 'Bearer eyJhb.....'
}
conn.request("POST", "/api/CashDevice/StartDevice?deviceID=NV200S", payload, headers)
res = conn.getresponse()
data = res.read()
print(data.decode("utf-8"))
```

Java

```
OkHttpClient client = new OkHttpClient().newBuilder()
  .build();
MediaType mediaType = MediaType.parse("text/plain");
RequestBody body = RequestBody.create(mediaType, "");
Request request = new Request.Builder()
  .url("http://localhost:5000/api/CashDevice/StartDevice?deviceID=NV200S")
  .method("POST", body)
  .addHeader("Authorization", "Bearer eyJhb.....")
  .build();
Response response = client.newCall(request).execute();
```

REST API - StopDevice

Description

Stops any running cash device threads, but does not close the COM port

| | |
|----------------------|---|
| Endpoint | StopDevice |
| Method | POST |
| URL | {server_url}/api/CashDevice/StopDevice |
| Parameters | <ul style="list-style-type: none">• deviceID (string): The value of "deviceID" obtained from the response to the OpenConnection request |
| Authorisation | Bearer Token |
| Body | None |

Responses

| Status | Body | Notes |
|--------|--------------------------------------|--|
| 200 | NV200S: Device stopped successfully. | OK: If the device stopped successfully. |
| 500 | | Internal Server Error: If there is an error stopping the device. |

Code Examples

C#

```
var options = new RestClientOptions("http://localhost:5000")
{
    MaxTimeout = -1,
};
var client = new RestClient(options);
var request = new RestRequest("/api/CashDevice/StopDevice?deviceID=NV200S",
Method.Post);
RestResponse response = await client.ExecuteAsync(request);
Console.WriteLine(response.Content);
```

NodeJS

```
var request = require('request');
var options = {
    'method': 'POST',
    'url': 'http://localhost:5000/api/CashDevice/StopDevice?deviceID=NV200S',
```

```

    'headers': {
    }
};

request(options, function (error, response) {
  if (error) throw new Error(error);
  console.log(response.body);
});

```

Python

```

import http.client

conn = http.client.HTTPSConnection("localhost", 5000)
payload = ''
headers = {}
conn.request("POST", "/api/CashDevice/StopDevice?deviceID=NV200S", payload, headers)
res = conn.getresponse()
data = res.read()
print(data.decode("utf-8"))

```

Java

```

OkHttpClient client = new OkHttpClient().newBuilder()
  .build();
MediaType mediaType = MediaType.parse("text/plain");
RequestBody body = RequestBody.create(mediaType, "");
Request request = new Request.Builder()
  .url("http://localhost:5000/api/CashDevice/StopDevice?deviceID=NV200S")
  .method("POST", body)
  .build();
Response response = client.newCall(request).execute();

```

REST API - LogRawPackets

Description

Takes rawPacketsOptionSelected by default as true and flags the SSP to log the raw packet data.

| | |
|----------------------|---|
| Endpoint | LogRawPackets |
| Method | POST |
| URL | {server_url}/api/CashDevice/LogRawPackets |
| Parameters | <ul style="list-style-type: none">• deviceID (string): The value of "deviceID" obtained from the response to the OpenConnection request |
| Authorisation | Bearer Token |
| Body | bool rawPacketsOptionSelected true |

Responses

| Status | Body | Notes |
|--------|--|---|
| 200 | NV200S: Raw packet logging has been enabled. | OK: If the raw packet logging was set successfully. |
| 400 | | Bad Request: If there is an error setting raw packet logging. |
| 404 | NV200S: Cash device not found. | Not Found: The cashDevice has not been created. |

Code Examples

C#

```
var options = new RestClientOptions("http://localhost:5000")
{
    MaxTimeout = -1,
};
var client = new RestClient(options);
var request = new RestRequest("/api/CashDevice/LogRawPackets?
deviceID=SPECTRAL_PAYOUT-COM5", Method.Post);
request.AddHeader("Content-Type", "application/json");
request.AddHeader("Authorization", "Bearer eyJhb.....");
```

```

var body = @"true";
request.AddStringBody(body, DataFormat.Json);
RestResponse response = await client.ExecuteAsync(request);
Console.WriteLine(response.Content);

```

NodeJS

```

var request = require('request');
var options = {
  'method': 'POST',
  'url': 'http://localhost:5000/api/CashDevice/LogRawPackets?
deviceID=SPECTRAL_PAYOUT-COM5',
  'headers': {
    'Content-Type': 'application/json',
    'Authorization': 'Bearer eyJhb.....'
  },
  body: JSON.stringify(true)

};

request(options, function (error, response) {
  if (error) throw new Error(error);
  console.log(response.body);
});

```

Python

```

import requests
import json

url = "http://localhost:5000/api/CashDevice/LogRawPackets?deviceID=SPECTRAL_PAYOUT-
COM5"

payload = json.dumps(True)
headers = {
  'Content-Type': 'application/json',
  'Authorization': 'Bearer eyJhb.....'
}

response = requests.request("POST", url, headers=headers, data=payload)

print(response.text)

```

Java

```

OkHttpClient client = new OkHttpClient().newBuilder()
  .build();
MediaType mediaType = MediaType.parse("application/json");
RequestBody body = RequestBody.create(mediaType, "true");
Request request = new Request.Builder()
  .url("http://localhost:5000/api/CashDevice/LogRawPackets?deviceID=SPECTRAL_PAYOUT-
COM5")
  .method("POST", body)
  .addHeader("Content-Type", "application/json")
  .addHeader("Authorization", "Bearer eyJhb.....")
  .build();

```

```
Response response = client.newCall(request).execute();
```

REST API - GetCompleteCashDevice

Description

- Returns all public read-only variables of the CashDevice object with their corresponding values.
- Refer to the following [table for all available Public Read-Only Variables of CashDevice class](#)

| | |
|----------------------|---|
| Endpoint | GetCompleteCashDevice |
| Method | GET |
| URL | {server_url}/api/CashDevice/GetCompleteCashDevice |
| Parameters | <ul style="list-style-type: none">• deviceID (string): The value of "deviceID" obtained from the response to the OpenConnection request |
| Authorisation | Bearer Token |
| Body | None |

Public Read-Only Variables of CashDevice class Table

| Variable Name | Description |
|-----------------------|--|
| comPort | The communication port name used by the device. |
| deviceModel | The model of the device. |
| deviceError | The current error status of the device. |
| isOpen | Indicates whether the device is open. |
| isDownloading | Indicates whether a download operation is in progress. |
| firmware | The firmware version of the device. |
| extendedFirmware | The extended firmware version of the device. |
| dataset | The dataset version of the device. |
| extendedDataset | The extended dataset version of the device. |
| mainSerialNumber | The main serial number of the device. |
| recycler1SerialNumber | The serial number of the first recycler. |
| recycler2SerialNumber | The serial number of the second recycler. |

| Variable Name | Description |
|--|---|
| recycler3SerialNumber | The serial number of the third recycler. |
| recycler4SerialNumber | The serial number of the fourth recycler. |
| interfaceSerialNumber | The serial number of the interface module. |
| dockSerialNumber | The serial number of the docking station. |
| replenishmentCassetteSerialNumber | The serial number of the replenishment cassette. |
| conveyorSerialNumber | The serial number of the conveyor module. |
| coinFeederSerialNumber | The serial number of the coin feeder module. |
| secondaryHopperSerialNumber | The serial number of the secondary hopper. |
| payoutModuleSerialNumber | The serial number of the payout module. |
| coinLifterSerialNumber | The serial number of the coin lifter module. |
| mainSerialNumberValid | Indicates whether the main serial number is valid. |
| recycler1SerialNumberValid | Indicates whether the serial number of the first recycler is valid. |
| recycler2SerialNumberValid | Indicates whether the serial number of the second recycler is valid. |
| recycler3SerialNumberValid | Indicates whether the serial number of the third recycler is valid. |
| recycler4SerialNumberValid | Indicates whether the serial number of the fourth recycler is valid. |
| interfaceSerialNumberValid | Indicates whether the serial number of the interface module is valid. |
| dockSerialNumberValid | Indicates whether the serial number of the docking station is valid. |
| replenishmentCassetteSerialNumberValid | Indicates whether the serial number of the replenishment cassette is valid. |
| conveyorSerialNumberValid | Indicates whether the serial number of the conveyor module is valid. |
| coinFeederSerialNumberValid | Indicates whether the serial number of the coin feeder module is valid. |
| secondaryHopperSerialNumberValid | Indicates whether the serial number of the secondary hopper is valid. |
| payoutModuleSerialNumberValid | Indicates whether the serial number of the payout module is valid. |

| Variable Name | Description |
|------------------------------------|---|
| coinLifterSerialNumberValid | Indicates whether the serial number of the coin lifter module is valid. |
| buildRevisionString | The build revision string of the device. |
| initialMainBuildRevision | The initial main build revision of the device. |
| mainBuildRevisionString | The main build revision string of the device. |
| coinFeederBuildRevisionString | The build revision string of the coin feeder. |
| secondaryHopperBuildRevisionString | The build revision string of the secondary hopper. |
| coinLifterBuildRevisionString | The build revision string of the coin lifter. |
| payoutModuleRevisionString | The build revision string of the payout module. |
| primaryHopperAsciiTypeString | The ASCII type string of the primary hopper. |
| coinFeederAsciiTypeString | The ASCII type string of the coin feeder. |
| majorBuildRevision | The major build revision number. |
| minorBuildRevision | The minor build revision number. |
| reg_0 | The value of register 0. |
| reg_1 | The value of register 1. |
| reg_0_hex_string | The hexadecimal string representation of register 0. |
| reg_1_hex_string | The hexadecimal string representation of register 1. |
| globalErrorCode_0 | The global error code 0. |
| globalErrorCode_1 | The global error code 1. |
| unit_info_retrieved | Indicates whether the unit information was retrieved. |
| lifterConnected | Indicates whether the lifter is connected. |
| lifterOptoClear | Indicates whether the lifter opto is clear. |
| lifterJammed | Indicates whether the lifter is jammed. |
| lastGetLevelsSuccessful | Indicates whether the last get levels operation was successful. |

| Variable Name | Description |
|------------------------------------|---|
| service_information_string | The service information string. |
| noPayinCount | The number of no payin counts. |
| coinAcceptance_string | The coin acceptance string. |
| counters_string | The counters string. |
| coins_payout_request | The number of coins requested for payout. |
| coins_seen_at_exit_sensor | The number of coins seen at the exit sensor. |
| real_time_clock_string | The real-time clock string. |
| cashbox_levels_string | The cashbox levels string. |
| payout_count | The payout count. |
| commandFailedDetailsString | The details string for command failures. |
| maintenanceRequiredDetailsString | The details string for maintenance required. |
| lifterEventString | The lifter event string. |
| rejectCategory | The reject category. |
| error_during_payout_details_string | The details string for errors during payout. |
| ticketStatus | The status of the barcode ticket. |
| barcode_ascii_data | The ASCII data of the barcode. |
| barCodeHardwareStatus | The hardware status of the barcode reader. |
| readersEnabled | The enabled status of the barcode readers. |
| barCodeFormat | The format of the barcode. |
| numberOfCharacters | The number of characters in the barcode. |
| barCodeInhibit | The inhibit status of the barcode reader. |
| RC_CurrentMode | The current mode of the replenishment cassette. |
| currentRC_PayoutValue | The current payout value of the replenishment cassette. |

| Variable Name | Description |
|---------------------------|--|
| RC_DenominationForPayout | The denomination for payout in the replenishment cassette. |
| unknown_stored_in_cashbox | The number of unknown items stored in the cashbox. |
| sspProtocolVersion | The SSP protocol version. |
| downloadStatus | The download status. |
| cashDeviceModules | An array of cash device modules. |
| dispenseState | The dispense transaction state. |
| noteInEscrow | Indicates whether there is a note in escrow. |
| isMultiCurrency | Indicates whether the device supports multiple currencies. |

Responses

| Status | | Body | Notes |
|--------|--|---|--|
| 200 | | <pre> SPECTRAL_PAYOUT-COM5: { "comPort": "COM5", "deviceModel": "SPECTRAL_DEVICE", "deviceError": "NONE", "isOpen": true, "isDownloading": false, "firmware": "NVS2004301038000", "extendedFirmware": null, "dataset": "GBP07056", "extendedDataset": null, "mainSerialNumber": 6331574, "recycler1SerialNumber": 0, "recycler2SerialNumber": 0, "recycler3SerialNumber": 0, "recycler4SerialNumber": 0, "interfaceSerialNumber": 0, "dockSerialNumber": 0, "replenishmentCassetteSerialNumber": 0, "conveyorSerialNumber": 0, "coinFeederSerialNumber": 0, "secondaryHopperSerialNumber": 0, "payoutModuleSerialNumber": 5785286, "coinLifterSerialNumber": 0, "mainSerialNumberValid": false, "recycler1SerialNumberValid": false, "recycler2SerialNumberValid": false, "recycler3SerialNumberValid": false, "recycler4SerialNumberValid": false, "interfaceSerialNumberValid": false, "dockSerialNumberValid": false, "replenishmentCassetteSerialNumberValid": false, "conveyorSerialNumberValid": false, "coinFeederSerialNumberValid": false, "secondaryHopperSerialNumberValid": false, } </pre> | OK: Returns the complete information of the cash device. |

| Status | | Body | Notes |
|--------|--|--|-------|
| | | <pre> "payoutModuleSerialNumberValid": true, "coinLifterSerialNumberValid": false, "buildRevisionString": "1.01", "initialMainBuildRevision": 0, "mainBuildRevisionString": "9.10", "coinFeederBuildRevisionString": null, "secondaryHopperBuildRevisionString": null, "coinLifterBuildRevisionString": null, "payoutModuleRevisionString": "1.01", "primaryHopperAsciiTypeString": null, "coinFeederAsciiTypeString": null, "majorBuildRevision": 1, "minorBuildRevision": 1, "reg_0": 0, "reg_1": 0, "reg_0_hex_string": null, "reg_1_hex_string": null, "globalErrorCode_0": 0, "globalErrorCode_1": 0, "unit_info_retrieved": true, "lifterConnected": false, "lifterOptoClear": false, "lifterJammed": false, "lastGetLevelsSuccessful": true, "service_information_string": null, "noPayinCount": 0, "coinAcceptance_string": null, "counters_string": "Number of counters in set: 5\nStacked: 261\nStored: 42\nDispensed: 30\nTransferred to stack: 11\nRejected: 26\n", "coins_payout_request": 0, "coins_seen_at_exit_sensor": 0, "real_time_clock_string": null, "cashbox_levels_string": null, "payout_count": 0, "commandFailedDetailsString": null, </pre> | |

| Status | | Body | Notes |
|--------|--|--|-------|
| | | <pre> "maintenanceRequiredDetailsString": null, "lifterEventString": null, "rejectCategory": null, "error_during_payout_details_string": null, "ticketStatus": null, "barcode_ascii_data": null, "barCodeHardwareStatus": null, "readersEnabled": null, "barCodeFormat": null, "numberOfCharacters": null, "barCodeInhibit": null, "nv22SpectralDevice": false, "enablePayoutDeviceError": null, "recyclerCountersString": null, "RC_CurrentMode": "RC_MODE_REPLENISH", "currentRC_PayoutValue": null, "RC_DenominationForPayout": null, "unknown_stored_in_cashbox": 0, "sspProtocolVersion": 7, "downloadStatus": { "State": "IDLE", "CurrentRamBlock": 0, "TotalRamBlocks": 0, "CurrentDownloadBlock": 0, "TotalDownloadBlocks": 0 }, "cashDeviceModules": null, "dispenseState": "COMPLETED", "isMultiCurrency": false, "noteInEscrow": false, "currencyAssignment": [{ "Type": "BANKNOTE", "ValueCountryCode": { "Value": 500, "CountryCode": "GBP", "Fraud_Attempt_Value": -1, "Calibration_Failed_Value": -1 }, "Value": 500, "CountryCode": "GBP", "IsInhibited": false, "IsRecyclable": true, "AcceptRoute": "PAYOUT", "Stored": 0, "StoredInCashbox": 0, "Channel": 1 }] } </pre> | |

| Status | | Body | Notes |
|--------|--|---|-------|
| | | <pre> }, { "Type": "BANKNOTE", "ValueCountryCode": { "Value": 1000, "CountryCode": "GBP", "Fraud_Attempt_Value": -1, "Calibration_Failed_Value": -1 }, "Value": 1000, "CountryCode": "GBP", "IsInhibited": false, "IsRecyclable": true, "AcceptRoute": "PAYOUT", "Stored": 0, "StoredInCashbox": 0, "Channel": 2 }, { "Type": "BANKNOTE", "ValueCountryCode": { "Value": 2000, "CountryCode": "GBP", "Fraud_Attempt_Value": -1, "Calibration_Failed_Value": -1 }, "Value": 2000, "CountryCode": "GBP", "IsInhibited": true, "IsRecyclable": true, "AcceptRoute": "CASHBOX", "Stored": 0, "StoredInCashbox": 0, "Channel": 3 }, { "Type": "BANKNOTE", "ValueCountryCode": { "Value": 5000, "CountryCode": "GBP", "Fraud_Attempt_Value": -1, "Calibration_Failed_Value": -1 }, "Value": 5000, "CountryCode": "GBP", "IsInhibited": true, "IsRecyclable": true, "AcceptRoute": "CASHBOX", "Stored": 0, "StoredInCashbox": 0, "Channel": 4 } } </pre> | |

| Status | | Body | Notes |
|--------|--|--|--|
| | | <pre>], "sorterRouteAssignment": [], "deviceState": "IDLE" }</pre> | |
| 404 | | | Not Found: If the device is not found. |
| 500 | | | Internal Server Error: If there is an error retrieving the device details. |

Code Examples

C#

```
var options = new RestClientOptions("http://localhost:5000")
{
    MaxTimeout = -1,
};
var client = new RestClient(options);
var request = new RestRequest("/api/CashDevice/GetCompleteCashDevice?
deviceID=SPECTRAL_PAYOUT-COM5", Method.Get);
request.AddHeader("Authorization", "Bearer eyJhb.....");
RestResponse response = await client.ExecuteAsync(request);
Console.WriteLine(response.Content);
```

NodeJS

```
var request = require('request');
var options = {
    'method': 'GET',
    'url': 'http://localhost:5000/api/CashDevice/GetCompleteCashDevice?
deviceID=NV200S',
    'headers': {
        'Authorization': 'Bearer eyJhb.....'
    }
};
request(options, function (error, response) {
    if (error) throw new Error(error);
    console.log(response.body);
});
```

Python

```
import http.client

conn = http.client.HTTPSConnection("localhost", 5000)
payload = ''
headers = {
    'Authorization': 'Bearer eyJhb.....'
}
```

```
conn.request("GET", "/api/CashDevice/GetCompleteCashDevice?deviceID=NV200S", payload,  
headers)  
res = conn.getresponse()  
data = res.read()  
print(data.decode("utf-8"))
```

Java

```
OkHttpClient client = new OkHttpClient().newBuilder()  
.build();  
MediaType mediaType = MediaType.parse("text/plain");  
RequestBody body = RequestBody.create(mediaType, "");  
Request request = new Request.Builder()  
.url("http://localhost:5000/api/CashDevice/GetCompleteCashDevice?deviceID=NV200S")  
.method("GET", body)  
.addHeader("Authorization", "Bearer eyJhb.....")  
.build();  
Response response = client.newCall(request).execute();
```

REST API - GetDeviceStatus

Description

Retrieve the state changes of the device when different types of events are triggered.

-  The client should implement a mechanism to continuously poll the device status from the server at a specified interval. E.g. 200 milliseconds.

| | |
|----------------------|---|
| Endpoint | GetDeviceStatus |
| Method | GET |
| URL | {server_url}/api/CashDevice/GetDeviceStatus |
| Parameters | <ul style="list-style-type: none">• deviceID (string): The value of "deviceID" obtained from the response to the OpenConnection request |
| Authorisation | Bearer Token |
| Body | None |

Response Types and Strings

| type | Strings |
|--|--|
| DeviceStatusResponse (stateAsString) | NOT_CONNECTED, CONNECTING, STARTING, STARTED, DISABLED, IDLE, ACCEPTING, DISPENSING, FLOATING, EMPTYING, REPLENISHING, JAM_RECOVERY, ERROR, DEVICE_FULL, RESET, CONNECTED, STOPPED, PURGING, PURGED, INITIALISING, CASHBOX_REMOVED, CASHBOX_REPLACE, STOPPING, PAY_IN_ACTIVE, COIN_MECH_ENABLED, COIN_MECH_DISABLED, MAINTENANCE_REQUIRED, CHANNEL_DISABLE, CALIBRATION_FAILED, FRAUD_ATTEMPT, LIFTER_EVENT, REJECTING, REJECTED, COIN_MECH_JAMMED, NOTE_HELD_IN_BEZEL, BAR_CODE_TICKET_VALIDATED, BAR_CODE_TICKET_ACKNOWLEDGE, NOTE_FLOAT_REMOVED, NOTE_FLOAT_ATTACHED, NOTE_PATH_OPEN, SCSTATUS_NO_CARD, SCSTATUS_CARD_INITIALISING, SCSTATUS_CARD_READY, SCSTATUS_NO_ACTIVE_DATASET, SCSTATUS_CHANGING_DATASET, SCSTATUS_DATASET_READY |
| DispenserTransactionEventResponse (stateAsString) | IN_PROGRESS, COMPLETED, ERROR |

| type | Strings |
|---|---|
| CashEventResponse (eventTypeAsString) | ESCROW, STACKED, STACKED_FRAUD_ATTEMPT, STORED, STORED_FRAUD_ATTEMPT, DISPENSING, DISPENSED, REJECTED, RETRIEVED, MOVED_TO_CASHBOX, NOTE_IN_BEZEL_HOLD, VALUE_ADDED, TIME_OUT, INCOMPLETE_PAYOUT, INCOMPLETE_FLOAT, FRAUD_ATTEMPT, HALTED, CALIBRATION_FAILED, JAMMED, COIN_CREDIT, ERROR_DURING_PAYOUT, NOTE_CLEARED_TO_CASHBOX, NOTE_CLEARED_FROM_FRONT, MULTIESCROW_CANCELLED, MULTIESCROW_COMMITTING, MULTIESCROW_COMMITED, MULTIESCROW_INCOMPLETE_CANCEL, MULTIESCROW_INCOMPLETE_COMMIT |
| ReplenishEventResponse (stateAsString) | REPLENISH_STORED, REPLENISH_SENT_TO_RC_TRAY, REPLENISH_CASSETTE_REMOVED, REPLENISH_CASSETTE_REPLACE, REPLENISH_CASSETTE_FULL |
| type | result |
| SetDenominationRouteFinishedEventResponse | True False |

transactionType

- 0 = PAYOUT
- 1 = FLOAT
- 2 = REPLENISH
- 3 = EMPTY
- 4 = PAYOUT_BY_DENOMINATION

Response Examples

| Status | Body | Notes |
|--------|--|--|
| 200 | <pre>[{ "type": "DeviceStatusResponse", "state": 1, "stateAsString": "CONNECTING", "isRunning": true, "message": "Device state changed: CONNECTING" }, { "type": "DeviceStatusResponse", "state": 15, "stateAsString": "CONNECTED", "isRunning": true, "message": "Device state changed: CONNECTED" }]</pre> | <p>OK: Returns the list of states of the device.</p> <p>E.g, Returning a queue of states [CONNECTING, CONNECTED]</p> |

| Status | Body | Notes |
|--------|---|--|
| 200 | <pre>[{ "type": "DeviceStatusResponse", "stateAsString": "ACCEPTING", "message": "Device state changed: ACCEPTING" }, { "type": "CashEventResponse", "eventTypeAsString": "ESCROW", "value": 500, "value2": 0, "countryCode": "GBP", "message": "Cash event: ESCROW - Value: 500 Value2: 0 - CountryCode: GBP" }, { "type": "CashEventResponse", "eventTypeAsString": "STORED", "value": 500, "value2": 0, "countryCode": "GBP", "message": "Cash event: STORED - Value: 500 Value2: 0 - CountryCode: GBP" }, { "type": "DeviceStatusResponse", "stateAsString": "IDLE", "message": "Device state changed: IDLE" }]</pre> | <p>OK: Returns the list of states of the device.</p> <p>E.g. Returning a queue of states [ACCEPTING, ESCROW VALUE 500 GBP, STORED VALUE 500 GBP, IDLE]</p> |
| 500 | | Internal Server Error: If there is an error retrieving the device states. |

Code Examples

C#

```
var options = new RestClientOptions("http://localhost:5000")
{
  MaxTimeout = -1,
};
var client = new RestClient(options);
var request = new RestRequest("/api/CashDevice/GetDeviceStatus?
deviceID=SPECTRAL_PAYOUT-COM5", Method.Get);
request.AddHeader("Authorization", "Bearer eyJhb.....");
RestResponse response = await client.ExecuteAsync(request);
Console.WriteLine(response.Content);
```

NodeJS

```
var request = require('request');
var options = {
  'method': 'GET',
  'url': 'http://localhost:5000/api/CashDevice/GetDeviceStatus?
deviceID=SPECTRAL_PAYOUT-COM5',
  'headers': {
    'Authorization': 'Bearer eyJhb.....'
  }
};
request(options, function (error, response) {
  if (error) throw new Error(error);
  console.log(response.body);
});
```

Python

```
import requests

url = "http://localhost:5000/api/CashDevice/GetDeviceStatus?deviceID=SPECTRAL_PAYOUT-
COM5"

payload = {}
headers = {
  'Authorization': 'Bearer eyJhb.....'
}

response = requests.request("GET", url, headers=headers, data=payload)

print(response.text)
```

Java

```
OkHttpClient client = new OkHttpClient().newBuilder()
  .build();
MediaType mediaType = MediaType.parse("text/plain");
RequestBody body = RequestBody.create(mediaType, "");
Request request = new Request.Builder()
  .url("http://localhost:5000/api/CashDevice/GetDeviceStatus?
deviceID=SPECTRAL_PAYOUT-COM5")
  .method("GET", body)
  .addHeader("Authorization", "Bearer eyJhb.....")
  .build();
Response response = client.newCall(request).execute();
```

REST API - GetCounters

Description

Retrieves the counter information from the device and formats the data into a readable string:

- SMART_COIN_SYSTEM, TWIN_SMART_COIN_SYSTEM, SMART_HOPPER_4: Retrieves and formats the number of counters, coins paid out, coins paid in, feeder rejects, hopper jams, feeder jams, fraud attempts, calibration fails, resets, and coins sent to cashbox.
- NV4000, SPECTRAL_DEVICE: Retrieves and formats the number of counters, stacked, stored, dispensed, transferred to stack, and rejected counts.

| | |
|----------------------|---|
| Endpoint | GetCounters |
| Method | GET |
| URL | {server_url}/api/CashDevice/GetCounters |
| Parameters | <ul style="list-style-type: none">• deviceId (string): The value of "deviceId" obtained from the response to the OpenConnection request |
| Authorisation | Bearer Token |
| Body | None |

Responses

| Status | Body | Notes |
|--------|--|---|
| 200 | <pre>SPECTRAL_PAYOUT-COM5: { CountersData = Number of counters in set: 5 Stacked: 261 Stored: 42 Dispensed: 30 Transferred to stack: 11 Rejected: 26 }</pre> | OK: If the counters data was retrieved successfully. |
| 400 | | Bad Request: If there is an error retrieving the counters data. |

Code Examples

C#

```
var options = new RestClientOptions("http://localhost:5000")
{
    MaxTimeout = -1,
};
var client = new RestClient(options);
```

```

var request = new RestRequest("/api/CashDevice/GetCounters?deviceID=SPECTRAL_PAYOUT-
COM5", Method.Get);
request.AddHeader("Authorization", "Bearer eyJhb.....");
RestResponse response = await client.ExecuteAsync(request);
Console.WriteLine(response.Content);

```

NodeJS

```

var request = require('request');
var options = {
  'method': 'GET',
  'url': 'http://localhost:5000/api/CashDevice/GetCounters?deviceID=SPECTRAL_PAYOUT-
COM5',
  'headers': {
    'Authorization': 'Bearer eyJhb.....'
  }
};
request(options, function (error, response) {
  if (error) throw new Error(error);
  console.log(response.body);
});

```

Python

```

import requests

url = "http://localhost:5000/api/CashDevice/GetCounters?deviceID=SPECTRAL_PAYOUT-
COM5"

payload = {}
headers = {
  'Authorization': 'Bearer eyJhb.....'
}

response = requests.request("GET", url, headers=headers, data=payload)

print(response.text)

```

Java

```

OkHttpClient client = new OkHttpClient().newBuilder()
  .build();
MediaType mediaType = MediaType.parse("text/plain");
RequestBody body = RequestBody.create(mediaType, "");
Request request = new Request.Builder()
  .url("http://localhost:5000/api/CashDevice/GetCounters?deviceID=SPECTRAL_PAYOUT-
COM5")
  .method("GET", body)
  .addHeader("Authorization", "Bearer eyJhb.....")
  .build();
Response response = client.newCall(request).execute();

```

REST API - GetAllLevels

Description

- Retrieves the current levels of all denominations from the device and updates the internal state with this information.
- For each level entry, it calculates the index using the denomination value and currency code.
- If the index is valid, it updates the stored level for that denomination.

| | |
|----------------------|---|
| Endpoint | GetAllLevels |
| Method | GET |
| URL | {server_url}/api/CashDevice/GetAllLevels |
| Parameters | <ul style="list-style-type: none">• deviceID (string): The value of "deviceID" obtained from the response to the OpenConnection request |
| Authorisation | Bearer Token |
| Body | None |

Responses

| Status | Body | Notes |
|--------|--|---|
| 200 | <pre>{ "levels": [{ "countryCode": "GBP", "value": 500, "stored": 0 }, { "countryCode": "GBP", "value": 1000, "stored": 0 }, { "countryCode": "GBP", "value": 2000, "stored": 0 }, { "countryCode": "GBP", "value": 5000, "stored": 0 }], "success": true, "message": "Successfully retrieved all levels." }</pre> | OK: Returns the levels information. |
| 400 | | If there is an error retrieving levels information. |

Response with 200. The response body contains a LevelsResult object. The structure of the LevelsResult is described below:

| Field | Type | Description |
|---------|-----------------|---|
| Levels | List<LevelInfo> | A list of LevelInfo objects representing the levels of different denominations. |
| Success | bool | Indicates whether the operation was successful. |
| Message | string | A message providing additional information about the operation. |

Each LevelInfo object in the Levels list has the following structure:

| Field | Type | Description |
|--------------|--------|---------------------------------------|
| Country Code | string | The country code of the denomination. |

| Field | Type | Description |
|--------|------|--|
| Value | uint | The value of the denomination. |
| Stored | uint | The number of stored units of this denomination. |

Code Examples

C#

```
var options = new RestClientOptions("http://localhost:5000")
{
    MaxTimeout = -1,
};
var client = new RestClient(options);
var request = new RestRequest("/api/CashDevice/GetAllLevels?deviceID=SPECTRAL_PAYOUT-
COM5", Method.Get);
request.AddHeader("Authorization", "Bearer eyJhb.....");
RestResponse response = await client.ExecuteAsync(request);
Console.WriteLine(response.Content);
```

NodeJS

```
var request = require('request');
var options = {
    'method': 'GET',
    'url': 'http://localhost:5000/api/CashDevice/GetAllLevels?deviceID=SPECTRAL_PAYOUT-
COM5',
    'headers': {
        'Authorization': 'Bearer eyJhb.....'
    }
};
request(options, function (error, response) {
    if (error) throw new Error(error);
    console.log(response.body);
});
```

Python

```
import requests

url = "http://localhost:5000/api/CashDevice/GetAllLevels?deviceID=SPECTRAL_PAYOUT-
COM5"

payload = {}
headers = {
    'Authorization': 'Bearer eyJhb.....'
}

response = requests.request("GET", url, headers=headers, data=payload)

print(response.text)
```

Java

```
OkHttpClient client = new OkHttpClient().newBuilder()
    .build();
MediaType mediaType = MediaType.parse("text/plain");
RequestBody body = RequestBody.create(mediaType, "");
Request request = new Request.Builder()
    .url("http://localhost:5000/api/CashDevice/GetAllLevels?deviceID=SPECTRAL_PAYOUT-
COM5")
    .method("GET", body)
    .addHeader("Authorization", "Bearer eyJhb.....")
    .build();
Response response = client.newCall(request).execute();
```

REST API - GetCurrencyAssignment

Description

Get the levels of cash currently stored for each channel in the devices dataset.

| | |
|----------------------|---|
| Endpoint | GetCurrencyAssignment |
| Method | GET |
| URL | {server_url}/api/CashDevice/GetCurrencyAssignment |
| Parameters | <ul style="list-style-type: none">• deviceID (string): The value of "deviceID" obtained from the response to the OpenConnection request |
| Authorisation | Bearer Token |
| Body | None |

Responses

| Status | Body | Notes |
|--------|---|--|
| 200 | <pre>[{ "type": 1, "valueCountryCode": { "value": 500, "countryCode": "GBP", "fraud_Attempt_Value": -1, "calibration_Failed_Value": -1 }, "value": 500, "countryCode": "GBP", "isInhibited": false, "isRecyclable": true, "acceptRoute": 7, "stored": 0, "storedInCashbox": 0, "channel": 1 }, { "type": 1, "valueCountryCode": { "value": 1000, "countryCode": "GBP", "fraud_Attempt_Value": -1, "calibration_Failed_Value": -1 }, "value": 1000, "countryCode": "GBP", "isInhibited": false, "isRecyclable": true, "acceptRoute": 7, "stored": 0, "storedInCashbox": 0, "channel": 2 }, { "type": 1, "valueCountryCode": { "value": 2000, "countryCode": "GBP", "fraud_Attempt_Value": -1, "calibration_Failed_Value": -1 }, "value": 2000, "countryCode": "GBP", "isInhibited": true, "isRecyclable": true, "acceptRoute": 0, "stored": 0, "storedInCashbox": 0, "channel": 3 }]</pre> | Ok: If the currency assignments were retrieved successfully. |

| Status | Body | Notes |
|--------|--|--|
| | <pre> "stored": 0, "storedInCashbox": 0, "channel": 3 }, { "type": 1, "valueCountryCode": { "value": 5000, "countryCode": "GBP", "fraud_Attempt_Value": -1, "calibration_Failed_Value": -1 }, "value": 5000, "countryCode": "GBP", "isInhibited": true, "isRecyclable": true, "acceptRoute": 0, "stored": 0, "storedInCashbox": 0, "channel": 4 }] </pre> | |
| 404 | | Not Found: If no currency assignments were found. |
| 400 | | Bad Request: If there is an error retrieving the currency assignments. |

Code Examples

C#

```

var options = new RestClientOptions("http://localhost:5000")
{
    MaxTimeout = -1,
};
var client = new RestClient(options);
var request = new RestRequest("/api/CashDevice/GetCurrencyAssignment?
deviceID=SPECTRAL_PAYOUT-COM5", Method.Get);
request.AddHeader("Authorization", "Bearer eyJhb....");
RestResponse response = await client.ExecuteAsync(request);
Console.WriteLine(response.Content);

```

NodeJS

```

var request = require('request');
var options = {
    'method': 'GET',

```

```

'url': 'http://localhost:5000/api/CashDevice/GetCurrencyAssignment?
deviceID=SPECTRAL_PAYOUT-COM5',
'headers': {
    'Authorization': 'Bearer eyJhb.....'
}
};

request(options, function (error, response) {
    if (error) throw new Error(error);
    console.log(response.body);
});

```

Python

```

import requests

url = "http://localhost:5000/api/CashDevice/GetCurrencyAssignment?
deviceID=SPECTRAL_PAYOUT-COM5"

payload = {}
headers = {
    'Authorization': 'Bearer eyJhb.....'
}

response = requests.request("GET", url, headers=headers, data=payload)

print(response.text)

```

Java

```

OkHttpClient client = new OkHttpClient().newBuilder()
    .build();
MediaType mediaType = MediaType.parse("text/plain");
RequestBody body = RequestBody.create(mediaType, "");
Request request = new Request.Builder()
    .url("http://localhost:5000/api/CashDevice/GetCurrencyAssignment?
deviceID=SPECTRAL_PAYOUT-COM5")
    .method("GET", body)
    .addHeader("Authorization", "Bearer eyJhb.....")
    .build();
Response response = client.newCall(request).execute();

```

REST API - SetDenominationLevel

Description

A command to increment the level of coins of the specified denomination stored in the hopper.

| | |
|-------------------|--|
| Endpoint | SetDenominationLevel |
| Method | POST |
| URL | {server_url}/api/CashDevice/SetDenominationLevel |
| Parameters | <ul style="list-style-type: none">• deviceID (string): The value of "deviceID" obtained from the response to the OpenConnection request |
| Body | <p>SetDenominationLevelRequest: The request body containing Value, CountryCode and NumCoinsToAdd.</p> <ul style="list-style-type: none">• UInt32 Value: Value of denomination to set e.g 500, 1000 etc• string CountryCode: ASCII country code of denomination e.g. "GBP", "EUR", "USD" etc• uint numCoinsToAdd: the amount of coins to add to level (0 will clear the level) <div style="border: 1px solid #ccc; padding: 10px; margin-top: 10px;"><pre>{ "Value": 100, "CountryCode": "{{Currency}}", "NumCoinsToAdd": 1 }</pre></div> |

Responses

| Status | Body | Notes |
|--------|---|---|
| 200 | SMART_COIN_SYSTEM-COM10: Denomination level set successfully. | OK: If the denomination level was set successfully. |
| 400 | | Bad Request: Failed to set denomination level. |

Code Examples

C#

```
var options = new RestClientOptions("http://localhost:5000")
{
    MaxTimeout = -1,
};
var client = new RestClient(options);
var request = new RestRequest("/api/CashDevice/SetDenominationLevel?
deviceID=SMART_COIN_SYSTEM-COM10", Method.Post);
request.AddHeader("Content-Type", "application/json");
```

```

request.AddHeader("Authorization", "Bearer eyJhb.....");
var body = @"{` + "\n" +
@`    ""Value"": 100, ` + "\n" +
@`    ""CountryCode"": ""GBP""` + "\n" +
@`    ""NumCoinsToAdd"": 1` + "\n" +
@`}` + "\n" +
@"";

request.AddStringBody(body, DataFormat.Json);
RestResponse response = await client.ExecuteAsync(request);
Console.WriteLine(response.Content);

```

NodeJS

```

var request = require('request');
var options = {
  'method': 'POST',
  'url': 'http://localhost:5000/api/CashDevice/SetDenominationLevel?
deviceID=SMART_COIN_SYSTEM-COM10',
  'headers': {
    'Content-Type': 'application/json',
    'Authorization': 'Bearer eyJhb.....'
  },
  body: JSON.stringify({
    "Value": 100,
    "CountryCode": "GBP",
    "NumCoinsToAdd": 1
  })
};

request(options, function (error, response) {
  if (error) throw new Error(error);
  console.log(response.body);
});

```

Python

```

import requests
import json

url = "http://localhost:5000/api/CashDevice/SetDenominationLevel?
deviceID=SMART_COIN_SYSTEM-COM10"

payload = json.dumps({
    "Value": 100,
    "CountryCode": "GBP",
    "NumCoinsToAdd": 1
})
headers = {
    'Content-Type': 'application/json',
    'Authorization': 'Bearer eyJhb.....'
}

response = requests.request("POST", url, headers=headers, data=payload)

print(response.text)

```

Java

```
OkHttpClient client = new OkHttpClient().newBuilder()
    .build();
MediaType mediaType = MediaType.parse("application/json");
RequestBody body = RequestBody.create(mediaType, "{\r\n    \"Value\": 100, \r\n    \"CountryCode\": \"GBP\", \r\n    \"NumCoinsToAdd\": 1\r\n}\r\n");
Request request = new Request.Builder()
    .url("http://localhost:5000/api/CashDevice/SetDenominationLevel?
deviceID=SMART_COIN_SYSTEM-COM10")
    .method("POST", body)
    .addHeader("Content-Type", "application/json")
    .addHeader("Authorization", "Bearer eyJhb.....")
    .build();
Response response = client.newCall(request).execute();
```

REST API - EnablePayout

Description

Enables a connected pay-out module for storing and payout cash.

| | |
|----------------------|---|
| Endpoint | EnablePayout |
| Method | POST |
| URL | {server_url}/api/CashDevice/EnablePayout |
| Parameters | <ul style="list-style-type: none">• deviceID (string): The value of "deviceID" obtained from the response to the OpenConnection request |
| Authorisation | Bearer Token |
| Body | None |

Responses

| Status | Body | Notes |
|--------|------------------------------|--|
| 200 | Payout enabled successfully. | OK: If the payout was enabled successfully. |
| 400 | | Bad Request: If there is an error enabling the payout. |

Code Examples

C#

```
var options = new RestClientOptions("http://localhost:5000")
{
    MaxTimeout = -1,
};
var client = new RestClient(options);
var request = new RestRequest("/api/CashDevice/EnablePayout?deviceID=SPECTRAL_PAYOUT-
COM5", Method.Post);
request.AddHeader("Authorization", "Bearer eyJhb.....");
RestResponse response = await client.ExecuteAsync(request);
Console.WriteLine(response.Content);
```

NodeJS

```
var request = require('request');
var options = {
    'method': 'POST',
```

```

'url': 'http://localhost:5000/api/CashDevice/EnablePayout?deviceID=SPECTRAL_PAYOUT-
COM5',
'headers': {
    'Authorization': 'Bearer eyJhb.....'
}
};

request(options, function (error, response) {
    if (error) throw new Error(error);
    console.log(response.body);
});

```

Python

```

import requests

url = "http://localhost:5000/api/CashDevice/EnablePayout?deviceID=SPECTRAL_PAYOUT-
COM5"

payload = {}
headers = {
    'Authorization': 'Bearer eyJhb.....'
}

response = requests.request("POST", url, headers=headers, data=payload)

print(response.text)

```

Java

```

OkHttpClient client = new OkHttpClient().newBuilder()
    .build();
MediaType mediaType = MediaType.parse("text/plain");
RequestBody body = RequestBody.create(mediaType, "");
Request request = new Request.Builder()
    .url("http://localhost:5000/api/CashDevice/EnablePayout?deviceID=SPECTRAL_PAYOUT-
COM5")
    .method("POST", body)
    .addHeader("Authorization", "Bearer eyJhb.....")
    .build();
Response response = client.newCall(request).execute();

```

REST API - EnablePayoutDevice

Description

Enables a connected pay-out module for storing and payout cash.

| | |
|----------------------|---|
| Endpoint | EnablePayoutDevice |
| Method | POST |
| URL | {server_url}/api/CashDevice/EnablePayoutDevice |
| Parameters | <ul style="list-style-type: none">• deviceID (string): The value of "deviceID" obtained from the response to the OpenConnection request |
| Authorisation | Bearer Token |
| Body | None |

Responses

| Status | Body | Notes |
|--------|---|--|
| 200 | SPECTRAL_PAYOUT-COM5: Payout device enabled successfully. | OK: If the payout was enabled successfully. |
| 400 | Cash device not found | Cash device not found |
| 500 | Failed to enable payout device. Error: {error_message} | Internal Server Error: An error occurred while enabling the payout device. |

Code Examples

C#

```
var options = new RestClientOptions("http://localhost:5000")
{
    MaxTimeout = -1,
};

var client = new RestClient(options);
var request = new RestRequest("/api/CashDevice/EnablePayoutDevice?
deviceID=SPECTRAL_PAYOUT-COM5", Method.Post);
request.AddHeader("Authorization", "Bearer eyJhb.....");
var body = @"";
request.AddParameter("text/plain", body, ParameterType.RequestBody);
```

```
RestResponse response = await client.ExecuteAsync(request);
Console.WriteLine(response.Content);
```

NodeJS

```
var request = require('request');
var options = {
  'method': 'POST',
  'url': 'http://localhost:5000/api/CashDevice/EnablePayoutDevice?
deviceID=SPECTRAL_PAYOUT-COM5',
  'headers': {
    'Authorization': 'Bearer eyJhb.....'
  }
};
request(options, function (error, response) {
  if (error) throw new Error(error);
  console.log(response.body);
});
```

Python

```
import requests

url = "http://localhost:5000/api/CashDevice/EnablePayoutDevice?
deviceID=SPECTRAL_PAYOUT-COM5"

payload = ""
headers = {
  'Authorization': 'Bearer eyJhb.....'
}

response = requests.request("POST", url, headers=headers, data=payload)

print(response.text)
```

Java

```
OkHttpClient client = new OkHttpClient().newBuilder()
  .build();
MediaType mediaType = MediaType.parse("text/plain");
RequestBody body = RequestBody.create(mediaType, "");
Request request = new Request.Builder()
  .url("http://localhost:5000/api/CashDevice/EnablePayoutDevice?
deviceID=SPECTRAL_PAYOUT-COM5")
  .method("POST", body)
  .addHeader("Authorization", "Bearer eyJhb.....")
  .build();
Response response = client.newCall(request).execute();
```

REST API - EnablePayoutDeviceWithByte

Description



These options do not persist in memory and after a reset they will go to their default values.

Enables a connected pay-out module for storing and payout cash with an additional byte for settings.

| | |
|----------------------|---|
| Endpoint | EnablePayoutDeviceWithByte |
| Method | POST |
| URL | {server_url}/api/CashDevice/EnablePayoutDeviceWithByte |
| Parameters | <ul style="list-style-type: none">• deviceID (string): The value of "deviceID" obtained from the response to the OpenConnection request |
| Authorisation | Bearer Token |
| Body | EnablePayoutDeviceByte (byte): Specifies whether to enable the payout device with specific settings. <pre>{ "EnablePayoutDeviceByte": 0 }</pre> |

Request Body Structure

Single Denomination Recycler

For NV11 / NV11S this request uses an addition data byte, a bit register allows some options to be set. When the additional byte is not sent, all the bits default to 0.

| Bit | Function |
|-----|--|
| 0 | GIVE_VALUE_ON_STORED. Set to 1 to enable the value of the note stored to be given with the Note Stored event. |
| 1 | NO_HOLD_NOTE_ON_PAYOUT. Set to 1 to enable the function of fully rejecting the dispensed banknote rather than holding it in the bezel. |
| 2 | Unused - Set to 0 |
| 3 | Enable report reverse validated value on payout (NV11S only, fw >= 1.19). If set 1, unit will report reverse validated value on dispensed event. If set 0, unit will report expected note value. |
| 4 | Unused - Set to 0 |
| 5 | Unused - Set to 0 |

| Bit | Function |
|-----|-------------------|
| 6 | Unused - Set to 0 |
| 7 | Unused - Set to 0 |

Multi Denomination Recycler

For Spectral Payout Range and NV22 this request uses an additional data byte, a bit register allows some options to be set. When the additional byte is not sent, all the bits default to 0.

| Bit | Function |
|-----|--|
| 0 | REQUIRE_FULL_STARTUP. If set to 1 the Payout will return busy until it has fully completed the startup procedure. |
| 1 | OPTIMISE_FOR_PAYIN_SPEED. If set to 1 the device will always move towards an empty slot when idle to try and ensure the shortest pay in speed possible. |
| 2 | HIGHEST_VALUE_SPLIT (NV22 only , Firmware versions ≥ 1.17). <ul style="list-style-type: none"> Set to 0: unit will split the amount to pay it quicker Set to 1: the unit will use the smallest number of notes to pay out the amount The highest value only applies to payout/float by amount. |
| 3 | On Spectral Payout only (FW versions ≥ 4.29). If bit is set to 1 unit will report dispensing event with expected note value when reverse validation fail. Dataset with reverse validation enabled must be used (EUR11, GBP11, USD11...). |
| 4 | Unused - Set to 0 |
| 5 | Unused - Set to 0 |
| 6 | Unused - Set to 0 |
| 7 | Unused - Set to 0 |

Responses

| Status | Body | Notes |
|--------|---|--|
| 200 | SPECTRAL_PAYOUT-COM5: Payout device enabled successfully. | OK: If the payout was enabled successfully. |
| 404 | Cash device not found | Not Found: The specified cash device was not initialised |

| Status | Body | Notes |
|--------|---|--|
| 500 | <pre>Failed to enable payout device. Error: {error_message}</pre> | Internal Server Error: An error occurred while enabling the payout device. |

Code Examples

C#

```
var options = new RestClientOptions("http://localhost:5000")
{
    MaxTimeout = -1,
};

var client = new RestClient(options);
var request = new RestRequest("/api/CashDevice/EnablePayoutDeviceWithByte?
deviceID=SPECTRAL_PAYOUT-COM5", Method.Post);
request.AddHeader("Content-Type", "application/json");
request.AddHeader("Authorization", "Bearer eyJhb.....");

var body = @"{" + "\n" +
@""      ""EnablePayoutDeviceByte"": 0" + "\n" +
@"}" + "\n" +
@"";

request.AddStringBody(body, DataFormat.Json);
RestResponse response = await client.ExecuteAsync(request);
Console.WriteLine(response.Content);
```

NodeJS

```
var request = require('request');
var options = {
    'method': 'POST',
    'url': 'http://localhost:5000/api/CashDevice/EnablePayoutDeviceWithByte?
deviceID=SPECTRAL_PAYOUT-COM5',
    'headers': {
        'Content-Type': 'application/json',
        'Authorization': 'Bearer eyJhb.....'
    },
    body: JSON.stringify({
        "EnablePayoutDeviceByte": 0
    })
};

request(options, function (error, response) {
    if (error) throw new Error(error);
    console.log(response.body);
});
```

Python

```
import requests
import json
```

```

url = "http://localhost:5000/api/CashDevice/EnablePayoutDeviceWithByte?
deviceID=SPECTRAL_PAYOUT-COM5"

payload = json.dumps({
    "EnablePayoutDeviceByte": 0
})
headers = {
    'Content-Type': 'application/json',
    'Authorization': 'Bearer eyJhb.....'
}

response = requests.request("POST", url, headers=headers, data=payload)

print(response.text)

```

Java

```

OkHttpClient client = new OkHttpClient().newBuilder()
    .build();
MediaType mediaType = MediaType.parse("application/json");
RequestBody body = RequestBody.create(mediaType, "{\"r\n"
    +"\"EnablePayoutDeviceByte\": 0\\r\\n}\\r\\n");
Request request = new Request.Builder()
    .url("http://localhost:5000/api/CashDevice/EnablePayoutDeviceWithByte?
deviceID=SPECTRAL_PAYOUT-COM5")
    .method("POST", body)
    .addHeader("Content-Type", "application/json")
    .addHeader("Authorization", "Bearer eyJhb.....")
    .build();
Response response = client.newCall(request).execute();

```

REST API - DisablePayout

Description

Disables a connected pay-out module.

| | |
|----------------------|---|
| Endpoint | DisablePayout |
| Method | POST |
| URL | {server_url}/api/CashDevice/DisablePayout |
| Parameters | <ul style="list-style-type: none">• deviceID (string): The value of "deviceID" obtained from the response to the OpenConnection request |
| Authorisation | Bearer Token |
| Body | None |

Responses

| Status | Body | Notes |
|--------|-------------------------------|---|
| 200 | Payout disabled successfully. | OK: If the payout was disabled successfully. |
| 400 | | Bad Request: If there is an error disabling the payout. |

Code Examples

C#

```
var options = new RestClientOptions("http://localhost:5000")
{
    MaxTimeout = -1,
};
var client = new RestClient(options);
var request = new RestRequest("/api/CashDevice/DisablePayout?
deviceID=SPECTRAL_PAYOUT-COM5", Method.Post);
request.AddHeader("Authorization", "Bearer eyJhb....");
RestResponse response = await client.ExecuteAsync(request);
Console.WriteLine(response.Content);
```

NodeJS

```
var request = require('request');
var options = {
    'method': 'POST',
    'url': 'http://localhost:5000/api/CashDevice/DisablePayout?
deviceID=SPECTRAL_PAYOUT-COM5',
```

```

'headers': {
    'Authorization': 'Bearer eyJhb.....'
}
};

request(options, function (error, response) {
    if (error) throw new Error(error);
    console.log(response.body);
});

```

Python

```

import requests

url = "http://localhost:5000/api/CashDevice/DisablePayout?deviceID=SPECTRAL_PAYOUT-
COM5"

payload = {}
headers = {
    'Authorization': 'Bearer eyJhb.....'
}

response = requests.request("POST", url, headers=headers, data=payload)

print(response.text)

```

Java

```

OkHttpClient client = new OkHttpClient().newBuilder()
    .build();
MediaType mediaType = MediaType.parse("text/plain");
RequestBody body = RequestBody.create(mediaType, "");
Request request = new Request.Builder()
    .url("http://localhost:5000/api/CashDevice/DisablePayout?deviceID=SPECTRAL_PAYOUT-
COM5")
    .method("POST", body)
    .addHeader("Authorization", "Bearer eyJhb.....")
    .build();
Response response = client.newCall(request).execute();

```

REST API - EnableAcceptor

Description

Enables the cash device to accept currency, does not enable any connected pay-out modules.

| | |
|----------------------|---|
| Endpoint | EnableAcceptor |
| Method | POST |
| URL | {server_url}/api/CashDevice/EnableAcceptor |
| Parameters | <ul style="list-style-type: none">deviceID (string): The value of "deviceID" obtained from the response to the OpenConnection request |
| Authorisation | Bearer Token |
| Body | None |

Responses

| Status | Body | Notes |
|--------|--|--|
| 200 | NV200S: Acceptor enabled successfully. | OK: If the acceptor was enabled successfully. |
| 500 | | Internal Server Error: If there is an error enabling the acceptor. |

Code Examples

C#

```
var options = new RestClientOptions("http://localhost:5000")
{
    MaxTimeout = -1,
};
var client = new RestClient(options);
var request = new RestRequest("/api/CashDevice/EnableAcceptor?deviceID=NV200S",
    Method.Post);
request.AddHeader("Authorization", "Bearer eyJhb.....");
RestResponse response = await client.ExecuteAsync(request);
Console.WriteLine(response.Content);
```

NodeJS

```
var request = require('request');
var options = {
```

```

'method': 'POST',
'url': 'http://localhost:5000/api/CashDevice/EnableAcceptor?deviceID=NV200S',
'headers': {
  'Authorization': 'Bearer eyJhb.....'
}
};

request(options, function (error, response) {
  if (error) throw new Error(error);
  console.log(response.body);
});

```

Python

```

import http.client

conn = http.client.HTTPSConnection("localhost", 5000)
payload = ''
headers = {
  'Authorization': 'Bearer eyJhb.....'
}
conn.request("POST", "/api/CashDevice/EnableAcceptor?deviceID=NV200S", payload,
headers)
res = conn.getresponse()
data = res.read()
print(data.decode("utf-8"))

```

Java

```

OkHttpClient client = new OkHttpClient().newBuilder()
  .build();
MediaType mediaType = MediaType.parse("text/plain");
RequestBody body = RequestBody.create(mediaType, "");
Request request = new Request.Builder()
  .url("http://localhost:5000/api/CashDevice/EnableAcceptor?deviceID=NV200S")
  .method("POST", body)
  .addHeader("Authorization", "Bearer eyJhb.....")
  .build();
Response response = client.newCall(request).execute();

```

REST API - DisableAcceptor

Description

Disables the acceptor to stop accepting currency, any connected and enabled pay-out modules will stay enabled and still pay-out cash

| | |
|----------------------|---|
| Endpoint | DisableAcceptor |
| Method | POST |
| URL | {server_url}/api/CashDevice/DisableAcceptor |
| Parameters | <ul style="list-style-type: none">• deviceID (string): The value of "deviceID" obtained from the response to the OpenConnection request |
| Authorisation | Bearer Token |
| Body | None |

Responses

| Status | Body | Notes |
|--------|---|---|
| 200 | SPECTRAL_PAYOUT-COM5: Acceptor disabled successfully. | OK: If the acceptor was disabled successfully. |
| 500 | | Internal Server Error: If there is an error disabling the acceptor. |

Code Examples

C#

```
var options = new RestClientOptions("http://localhost:5000")
{
    MaxTimeout = -1,
};
var client = new RestClient(options);
var request = new RestRequest("/api/CashDevice/DisableAcceptor?
deviceID=SPECTRAL_PAYOUT-COM5", Method.Post);
request.AddHeader("Authorization", "Bearer eyJhb.....");
RestResponse response = await client.ExecuteAsync(request);
Console.WriteLine(response.Content);
```

NodeJS

```
var request = require('request');
```

```

var options = {
  'method': 'POST',
  'url': 'http://localhost:5000/api/CashDevice/DisableAcceptor?
deviceID=SPECTRAL_PAYOUT-COM5',
  'headers': {
    'Authorization': 'Bearer eyJhb.....'
  }
};

request(options, function (error, response) {
  if (error) throw new Error(error);
  console.log(response.body);
});

```

Python

```

import requests

url = "http://localhost:5000/api/CashDevice/DisableAcceptor?deviceID=SPECTRAL_PAYOUT-
COM5"

payload = {}
headers = {
  'Authorization': 'Bearer eyJhb.....'
}

response = requests.request("POST", url, headers=headers, data=payload)

print(response.text)

```

Java

```

OkHttpClient client = new OkHttpClient().newBuilder()
  .build();
MediaType mediaType = MediaType.parse("text/plain");
RequestBody body = RequestBody.create(mediaType, "");
Request request = new Request.Builder()
  .url("http://localhost:5000/api/CashDevice/DisableAcceptor?
deviceID=SPECTRAL_PAYOUT-COM5")
  .method("POST", body)
  .addHeader("Authorization", "Bearer eyJhb.....")
  .build();
Response response = client.newCall(request).execute();

```

REST API - SetAutoAccept

Description

Enable or disable auto accept from escrow.

| | |
|----------------------|---|
| Endpoint | SetAutoAccept |
| Method | POST |
| URL | {server_url}/api/CashDevice/SetAutoAccept |
| Parameters | <ul style="list-style-type: none">• deviceID (string): The value of "deviceID" obtained from the response to the OpenConnection request |
| Authorisation | Bearer Token |
| Body | (boolean) true to enable and false to disable auto accept notes held in escrow. |

Responses

| Status | Body | Notes |
|--------|---|--|
| 200 | SPECTRAL_PAYOUT-COM5: Auto-accept set to True | OK: If auto accept was set correctly. |
| 500 | | Internal Server Error: If there is an error enabling the acceptor. |

Code Examples

C#

```
var options = new RestClientOptions("http://localhost:5000")
{
    MaxTimeout = -1,
};

var client = new RestClient(options);
var request = new RestRequest("/api/CashDevice/SetAutoAccept?
deviceID=SPECTRAL_PAYOUT-COM5", Method.Post);
request.AddHeader("Content-Type", "application/json");
request.AddHeader("Authorization", "Bearer eyJhb.....");
var body = @"true";
request.AddStringBody(body, DataFormat.Json);
RestResponse response = await client.ExecuteAsync(request);
Console.WriteLine(response.Content);
```

NodeJS

```
var request = require('request');
var options = {
  'method': 'POST',
  'url': 'http://localhost:5000/api/CashDevice/SetAutoAccept?',
deviceID=SPECTRAL_PAYOUT-COM5',
  'headers': {
    'Content-Type': 'application/json',
    'Authorization': 'Bearer eyJhb.....'
  },
  body: JSON.stringify(true)

};

request(options, function (error, response) {
  if (error) throw new Error(error);
  console.log(response.body);
});
```

Python

```
import requests
import json

url = "http://localhost:5000/api/CashDevice/SetAutoAccept?deviceID=SPECTRAL_PAYOUT-
COM5"

payload = json.dumps(True)
headers = {
  'Content-Type': 'application/json',
  'Authorization': 'Bearer eyJhb.....'
}

response = requests.request("POST", url, headers=headers, data=payload)

print(response.text)
```

Java

```
OkHttpClient client = new OkHttpClient().newBuilder()
  .build();
MediaType mediaType = MediaType.parse("application/json");
RequestBody body = RequestBody.create(mediaType, "true");
Request request = new Request.Builder()
  .url("http://localhost:5000/api/CashDevice/SetAutoAccept?deviceID=SPECTRAL_PAYOUT-
COM5")
  .method("POST", body)
  .addHeader("Content-Type", "application/json")
  .addHeader("Authorization", "Bearer eyJhb.....")
  .build();
Response response = client.newCall(request).execute();
```

REST API - AcceptFromEscrow

Description

Accepts a note currently being held in escrow.

| | |
|----------------------|---|
| Endpoint | AcceptFromEscrow |
| Method | POST |
| URL | {server_url}/api/CashDevice/AcceptFromEscrow |
| Parameters | <ul style="list-style-type: none">• deviceID (string): The value of "deviceID" obtained from the response to the OpenConnection request |
| Authorisation | Bearer Token |
| Body | None |

Responses

| Status | Body | Notes |
|--------|---|---|
| 200 | SPECTRAL_PAYOUT-COM5: Note can be accepted from escrow. | OK: If the note can be accepted from escrow. |
| 400 | SPECTRAL_PAYOUT-COM5: Error accepting note from escrow | Fail: No note in escrow. |
| 500 | | Internal Server Error: If there is an error accepting the note from escrow. |

Code Examples

C#

```
var options = new RestClientOptions("http://localhost:5000")
{
    MaxTimeout = -1,
};
var client = new RestClient(options);
var request = new RestRequest("/api/CashDevice/AcceptFromEscrow?
deviceID=SPECTRAL_PAYOUT-COM5", Method.Post);
request.AddHeader("Authorization", "Bearer eyJhb.....");
RestResponse response = await client.ExecuteAsync(request);
Console.WriteLine(response.Content);
```

NodeJS

```
var request = require('request');
var options = {
  'method': 'POST',
  'url': 'http://localhost:5000/api/CashDevice/AcceptFromEscrow?
deviceID=SPECTRAL_PAYOUT-COM5',
  'headers': {
    'Authorization': 'Bearer eyJhb.....'
  }
};
request(options, function (error, response) {
  if (error) throw new Error(error);
  console.log(response.body);
});
```

Python

```
import requests

url = "http://localhost:5000/api/CashDevice/AcceptFromEscrow?
deviceID=SPECTRAL_PAYOUT-COM5"

payload = {}
headers = {
  'Authorization': 'Bearer eyJhb.....'
}

response = requests.request("POST", url, headers=headers, data=payload)

print(response.text)
```

Java

```
OkHttpClient client = new OkHttpClient().newBuilder()
  .build();
MediaType mediaType = MediaType.parse("text/plain");
RequestBody body = RequestBody.create(mediaType, "");
Request request = new Request.Builder()
  .url("http://localhost:5000/api/CashDevice/AcceptFromEscrow?
deviceID=SPECTRAL_PAYOUT-COM5")
  .method("POST", body)
  .addHeader("Authorization", "Bearer eyJhb.....")
  .build();
Response response = client.newCall(request).execute();
```

REST API - ReturnFromEscrow

Description

Returns a note currently being held in escrow

| | |
|----------------------|---|
| Endpoint | ReturnFromEscrow |
| Method | POST |
| URL | {server_url}/api/CashDevice/ReturnFromEscrow |
| Parameters | <ul style="list-style-type: none">deviceID (string): The value of "deviceID" obtained from the response to the OpenConnection request |
| Authorisation | Bearer Token |
| Body | None |

Responses

| Status | Body | Notes |
|--------|---|---|
| 200 | SPECTRAL_PAYOUT-COM5: Note can be returned from escrow. | OK: If the note can be returned from escrow. |
| 400 | SPECTRAL_PAYOUT-COM5: Error returning note from escrow | Bad Request: No note in escrow. |
| 500 | | Internal Server Error: If there is an error returning the note from escrow. |

Code Examples

C#

```
var options = new RestClientOptions("http://localhost:5000")
{
    MaxTimeout = -1,
};
var client = new RestClient(options);
var request = new RestRequest("/api/CashDevice/ReturnFromEscrow?
deviceID=SPECTRAL_PAYOUT-COM5", Method.Post);
request.AddHeader("Authorization", "Bearer eyJhb.....");
RestResponse response = await client.ExecuteAsync(request);
Console.WriteLine(response.Content);
```

NodeJS

```
var request = require('request');
var options = {
  'method': 'POST',
  'url': 'http://localhost:5000/api/CashDevice/ReturnFromEscrow?
deviceID=SPECTRAL_PAYOUT-COM5',
  'headers': {
    'Authorization': 'Bearer eyJhb.....'
  }
};
request(options, function (error, response) {
  if (error) throw new Error(error);
  console.log(response.body);
});
```

Python

```
import requests

url = "http://localhost:5000/api/CashDevice/ReturnFromEscrow?
deviceID=SPECTRAL_PAYOUT-COM5"

payload = {}
headers = {
  'Authorization': 'Bearer eyJhb.....'
}

response = requests.request("POST", url, headers=headers, data=payload)

print(response.text)
```

Java

```
OkHttpClient client = new OkHttpClient().newBuilder()
  .build();
MediaType mediaType = MediaType.parse("text/plain");
RequestBody body = RequestBody.create(mediaType, "");
Request request = new Request.Builder()
  .url("http://localhost:5000/api/CashDevice/ReturnFromEscrow?
deviceID=SPECTRAL_PAYOUT-COM5")
  .method("POST", body)
  .addHeader("Authorization", "Bearer eyJhb.....")
  .build();
Response response = client.newCall(request).execute();
```

REST API - SetMultiEscrowSize

Description

Used to set the Escrow note size for multi-escrow firmware.

 Maximum multi-escrow note limit is 20.

Can only be used with the MNE protocol available on:

- Spectral Payout
- NV4000

| | |
|----------------------|---|
| Endpoint | SetMultiEscrowSize |
| Method | Post |
| URL | {server_url}/api/CashDevice/SetMultiEscrowSize |
| Parameters | <ul style="list-style-type: none">• deviceID (string): The value of "deviceID" obtained from the response to the OpenConnection request |
| Authorisation | Bearer Token |
| Body | uint EscrowSize: The maximum number of notes used for multi-escrow. e.g to set the multi-escrow note limit to 20. 20 |

Responses

| Status | Body | Notes |
|--------|--|--|
| 200 | Successfully set multi-escrow size to 20. | OK: Successfully set the multi-escrow note size. |
| 400 | Failed to set new multi-escrow size. | Bad Request: Unable to set the requested multi-escrow note size. |
| 404 | | Not Found: The specified cash device was not initialised |
| 500 | | Internal Server Error: Unable to set the multi-escrow note size. |

Code Examples

C#

```
var client = new HttpClient();
var request = new HttpRequestMessage(HttpMethod.Post, "http://localhost:5000/api/CashDevice/SetMultiEscrowSize?deviceID=SPECTRAL_PAYOUT-COM5");
request.Headers.Add("Authorization", "Bearer eyJhb.....");
var content = new StringContent("20", null, "application/json");
request.Content = content;
var response = await client.SendAsync(request);
response.EnsureSuccessStatusCode();
Console.WriteLine(await response.Content.ReadAsStringAsync());
```

NodeJS

```
var request = require('request');
var options = {
  'method': 'POST',
  'url': 'http://localhost:5000/api/CashDevice/SetMultiEscrowSize?deviceID=SPECTRAL_PAYOUT-COM5',
  'headers': {
    'Content-Type': 'application/json',
    'Authorization': 'Bearer eyJhb.....'
  },
  body: JSON.stringify(20)
};

request(options, function (error, response) {
  if (error) throw new Error(error);
  console.log(response.body);
});
```

Python

```
import requests
import json

url = "http://localhost:5000/api/CashDevice/SetMultiEscrowSize?deviceID=SPECTRAL_PAYOUT-COM5"

payload = json.dumps(20)
headers = {
  'Content-Type': 'application/json',
  'Authorization': 'Bearer eyJhb.....'
}

response = requests.request("POST", url, headers=headers, data=payload)

print(response.text)
```

Java

```
OkHttpClient client = new OkHttpClient().newBuilder()
  .build();
```

```
MediaType mediaType = MediaType.parse("application/json");
RequestBody body = RequestBody.create(mediaType, "20");
Request request = new Request.Builder()
    .url("http://localhost:5000/api/CashDevice/SetMultiEscrowSize?
deviceID=SPECTRAL_PAYOUT-COM5")
    .method("POST", body)
    .addHeader("Content-Type", "application/json")
    .addHeader("Authorization", "Bearer eyJhb.....")
    .build();
Response response = client.newCall(request).execute();
```

REST API - GetMultiEscrowSize

Description

Used to get the Escrow note size for multi-escrow firmware.

 Maximum multi-escrow note limit is 20.
Can only be used with the MNE protocol available on:

- Spectral Payout
- NV4000

| | |
|----------------------|---|
| Endpoint | GetMultiEscrowSize |
| Method | Get |
| URL | {server_url}/api/CashDevice/GetMultiEscrowSize |
| Parameters | <ul style="list-style-type: none">• deviceID (string): The value of "deviceID" obtained from the response to the OpenConnection request |
| Authorisation | Bearer Token |
| Body | None |

Responses

| Status | Body | Notes |
|--------|--|--|
| 200 | <pre>{ "multiEscrowSize": 20 }</pre> | OK: Successfully retrieved the multi-escrow note size. |
| 404 | | Not Found: The specified cash device was not initialised |
| 500 | | Internal Server Error: Unable to get the multi-escrow note size. |

Code Examples

C#

```
var client = new HttpClient();
var request = new HttpRequestMessage(HttpMethod.Get, "http://localhost:5000/api/
CashDevice/GetMultiEscrowSize?deviceID=SPECTRAL_PAYOUT-COM5");
request.Headers.Add("Authorization", "Bearer eyJhb.....");
```

```
var response = await client.SendAsync(request);
response.EnsureSuccessStatusCode();
Console.WriteLine(await response.Content.ReadAsStringAsync());
```

NodeJS

```
var request = require('request');
var options = {
  'method': 'GET',
  'url': 'http://localhost:5000/api/CashDevice/GetMultiEscrowSize?
deviceID=SPECTRAL_PAYOUT-COM5',
  'headers': {
    'Authorization': 'Bearer eyJhb.....'
  }
};
request(options, function (error, response) {
  if (error) throw new Error(error);
  console.log(response.body);
});
```

Python

```
import requests

url = "http://localhost:5000/api/CashDevice/GetMultiEscrowSize?
deviceID=SPECTRAL_PAYOUT-COM5"

payload = {}
headers = {
  'Authorization': 'Bearer eyJhb.....'
}

response = requests.request("GET", url, headers=headers, data=payload)

print(response.text)
```

Java

```
OkHttpClient client = new OkHttpClient().newBuilder()
  .build();
MediaType mediaType = MediaType.parse("text/plain");
RequestBody body = RequestBody.create(mediaType, "");
Request request = new Request.Builder()
  .url("http://localhost:5000/api/CashDevice/GetMultiEscrowSize?
deviceID=SPECTRAL_PAYOUT-COM5")
  .method("GET", body)
  .addHeader("Authorization", "Bearer eyJhb.....")
  .build();
Response response = client.newCall(request).execute();
```

REST API - GetMultiEscrowValue

Description

Used to retrieve the value of notes currently stored in Escrow for the multi-escrow firmware.

-  Can only be used with the MNE protocol available on:
- Spectral Payout
 - NV4000

| | |
|----------------------|---|
| Endpoint | GetMultiEscrowValue |
| Method | Get |
| URL | {server_url}/api/CashDevice/GetMultiEscrowValue |
| Parameters | <ul style="list-style-type: none">• deviceID (string): The value of "deviceID" obtained from the response to the OpenConnection request |
| Authorisation | Bearer Token |
| Body | None |

Responses

| Status | Body | Notes |
|--------|---|---|
| 200 | <pre>{ "multiEscrowValue": 1000, "multiEscrowCurrency": "EUR" }</pre> | OK: Successfully retrieved the value of notes currently held in Escrow. |
| 404 | | Not Found: The specified cash device was not initialised |
| 500 | | Internal Server Error: Unable to get the multi-escrow note size. |

Code Examples

C#

```
var client = new HttpClient();
var request = new HttpRequestMessage(HttpMethod.Get, "http://localhost:5000/api/
CashDevice/GetMultiEscrowValue?deviceID=SPECTRAL_PAYOUT-COM5");
request.Headers.Add("Authorization", "Bearer eyJhb.....");
var response = await client.SendAsync(request);
```

```
response.EnsureSuccessStatusCode();
Console.WriteLine(await response.Content.ReadAsStringAsync());
```

NodeJS

```
var request = require('request');
var options = {
  'method': 'GET',
  'url': 'http://localhost:5000/api/CashDevice/GetMultiEscrowValue?
deviceID=SPECTRAL_PAYOUT-COM5',
  'headers': {
    'Authorization': 'Bearer eyJhb.....'
  }
};
request(options, function (error, response) {
  if (error) throw new Error(error);
  console.log(response.body);
});
```

Python

```
import requests

url = "http://localhost:5000/api/CashDevice/GetMultiEscrowValue?
deviceID=SPECTRAL_PAYOUT-COM5"

payload = {}
headers = {
  'Authorization': 'Bearer eyJhb.....'
}

response = requests.request("GET", url, headers=headers, data=payload)

print(response.text)
```

Java

```
OkHttpClient client = new OkHttpClient().newBuilder()
  .build();
MediaType mediaType = MediaType.parse("text/plain");
RequestBody body = RequestBody.create(mediaType, "");
Request request = new Request.Builder()
  .url("http://localhost:5000/api/CashDevice/GetMultiEscrowValue?
deviceID=SPECTRAL_PAYOUT-COM5")
  .method("GET", body)
  .addHeader("Authorization", "Bearer eyJhb.....")
  .build();
Response response = client.newCall(request).execute();
```

REST API - CommitMultiEscrow

Description

Used to commit notes currently held in Escrow. Notes will remain in the payout (available space permitting).

 The routing of notes is disabled when using multi-escrow. Notes can be sent to the cashbox using the [Float](#) endpoint.

Can only be used with the MNE protocol available on:

- Spectral Payout
- NV4000

| | |
|----------------------|---|
| Endpoint | CommitMultiEscrow |
| Method | Post |
| URL | {server_url}/api/CashDevice/CommitMultiEscrow |
| Parameters | <ul style="list-style-type: none">• deviceID (string): The value of "deviceID" obtained from the response to the OpenConnection request |
| Authorisation | Bearer Token |
| Body | None |

Responses

| Status | Body | Notes |
|--------|--------------------------------------|---|
| 200 | Successfully committed multi-escrow. | OK: Successfully committed escrow notes. |
| 400 | Failed to commit multi-escrow. | Bad Request: Failed to commit notes. Check notes are waiting to be committed using the GetMultiEscrowValue endpoint. |
| 404 | | Not Found: The specified cash device was not initialised |
| 500 | | Internal Server Error: Unable to get the multi-escrow note size. |

Code Examples

C#

```
var client = new HttpClient();
```

```

var request = new HttpRequestMessage(HttpMethod.Post, "http://localhost:5000/api/CashDevice/CommitMultiEscrow?deviceID=SPECTRAL_PAYOUT-COM5");
request.Headers.Add("Authorization", "Bearer eyJhb.....");
var response = await client.SendAsync(request);
response.EnsureSuccessStatusCode();
Console.WriteLine(await response.Content.ReadAsStringAsync());

```

NodeJS

```

var request = require('request');
var options = {
  'method': 'POST',
  'url': 'http://localhost:5000/api/CashDevice/CommitMultiEscrow?',
  deviceID=SPECTRAL_PAYOUT-COM5',
  'headers': {
    'Authorization': 'Bearer eyJhb.....'
  }
};
request(options, function (error, response) {
  if (error) throw new Error(error);
  console.log(response.body);
});

```

Python

```

import requests

url = "http://localhost:5000/api/CashDevice/CommitMultiEscrow?
deviceID=SPECTRAL_PAYOUT-COM5"

payload = {}
headers = {
  'Authorization': 'Bearer eyJhb.....'
}

response = requests.request("POST", url, headers=headers, data=payload)

print(response.text)

```

Java

```

OkHttpClient client = new OkHttpClient().newBuilder()
  .build();
MediaType mediaType = MediaType.parse("text/plain");
RequestBody body = RequestBody.create(mediaType, "");
Request request = new Request.Builder()
  .url("http://localhost:5000/api/CashDevice/CommitMultiEscrow?
deviceID=SPECTRAL_PAYOUT-COM5")
  .method("POST", body)
  .addHeader("Authorization", "Bearer eyJhb.....")
  .build();
Response response = client.newCall(request).execute();

```

REST API - CancelMultiEscrow

Description

Used to return the notes currently held in Escrow. No credit given to the host.

 Can only be used with the MNE protocol available on:

- Spectral Payout
- NV4000

| | |
|----------------------|---|
| Endpoint | CancelMultiEscrow |
| Method | Post |
| URL | {server_url}/api/CashDevice/CancelMultiEscrow |
| Parameters | <ul style="list-style-type: none">• deviceID (string): The value of "deviceID" obtained from the response to the OpenConnection request |
| Authorisation | Bearer Token |
| Body | <pre>true</pre> |

Responses

| Status | Body | Notes |
|--------|---|---|
| 200 | <pre>Successfully cancelled multi-escrow.</pre> | OK: Successfully cancelled the multi-escrow notes. |
| 400 | <pre>Failed to cancel multi-escrow.</pre> | Bad Request: Failed to cancel multi-escrow notes. Check notes are waiting to be cancelled using the GetMultiEscrowValue endpoint. |
| 404 | | Not Found: The specified cash device was not initialised |
| 500 | | Internal Server Error: Unable to get the multi-escrow note size. |

Code Examples

C#

```
var client = new HttpClient();
var request = new HttpRequestMessage(HttpMethod.Post, "http://localhost:5000/api/CashDevice/CancelMultiEscrow?deviceID=SPECTRAL_PAYOUT-COM5");
request.Headers.Add("Authorization", "Bearer eyJhb.....");
var content = new StringContent("true", null, "application/json");
request.Content = content;
var response = await client.SendAsync(request);
response.EnsureSuccessStatusCode();
Console.WriteLine(await response.Content.ReadAsStringAsync());
```

NodeJS

```
var request = require('request');
var options = {
  'method': 'POST',
  'url': 'http://localhost:5000/api/CashDevice/CancelMultiEscrow?deviceID=SPECTRAL_PAYOUT-COM5',
  'headers': {
    'Content-Type': 'application/json',
    'Authorization': 'Bearer eyJhb.....'
  },
  body: JSON.stringify(true)

};

request(options, function (error, response) {
  if (error) throw new Error(error);
  console.log(response.body);
});
```

Python

```
import requests
import json

url = "http://localhost:5000/api/CashDevice/CancelMultiEscrow?deviceID=SPECTRAL_PAYOUT-COM5"

payload = json.dumps(True)
headers = {
  'Content-Type': 'application/json',
  'Authorization': 'Bearer eyJhb.....'
}

response = requests.request("POST", url, headers=headers, data=payload)

print(response.text)
```

Java

```
OkHttpClient client = new OkHttpClient().newBuilder()
  .build();
```

```
MediaType mediaType = MediaType.parse("application/json");
RequestBody body = RequestBody.create(mediaType, "true");
Request request = new Request.Builder()
    .url("http://localhost:5000/api/CashDevice/CancelMultiEscrow?
deviceID=SPECTRAL_PAYOUT-COM5")
    .method("POST", body)
    .addHeader("Content-Type", "application/json")
    .addHeader("Authorization", "Bearer eyJhb.....")
    .build();
Response response = client.newCall(request).execute();
```

REST API - SetDenominationInhibits

Description

Sets the inhibit status for a list of denomination and country code pairs.

| | |
|----------------------|--|
| Endpoint | SetDenominationInhibits |
| Method | POST |
| URL | {server_url}/api/CashDevice/SetDenominationInhibits |
| Parameters | <ul style="list-style-type: none">• deviceID (string): The value of "deviceID" obtained from the response to the OpenConnection request |
| Authorisation | Bearer Token |
| Body | InhibitsRequest: The request body containing ValueCountryCodes and Inhibit. <ul style="list-style-type: none">• List<string> ValueCountryCodes: A list of string representing the denominations and their respective country codes.• bool inhibit: Set true to inhibit the denominations, false to uninhibit <div style="border: 1px solid #ccc; padding: 10px; margin-top: 10px;"><pre>{ "ValueCountryCodes": ["2000 {{Currency}}", "5000 {{Currency}}"], "Inhibit": true }</pre></div> |

Responses

| Status | Body | Notes |
|--------|---|---|
| 200 | SPECTRAL_PAYOUT-COM5: Denomination inhibits set successfully. | OK: If denomination inhibits set successfully. |
| 400 | Failed to set denomination inhibits. | Bad Request: Failed to set denomination inhibits. |

Code Examples

C#

```
var options = new RestClientOptions("http://localhost:5000")
{
    MaxTimeout = -1,
};
var client = new RestClient(options);
```

```

var request = new RestRequest("/api/CashDevice/SetDenominationInhibits?
deviceID=SPECTRAL_PAYOUT-COM5", Method.Post);
request.AddHeader("Content-Type", "application/json");
request.AddHeader("Authorization", "Bearer eyJhb.....");
var body = @""{ + "\n" +
@""    ""ValueCountryCodes"": [""2000 GBP"", ""5000 GBP""],," + "\n" +
@""    ""Inhibit"": true" + "\n" +
@""}" + "\n" +
@"" + "\n" +
@"";
request.AddStringBody(body, DataFormat.Json);
RestResponse response = await client.ExecuteAsync(request);
Console.WriteLine(response.Content);

```

NodeJS

```

var request = require('request');
var options = {
  'method': 'POST',
  'url': 'http://localhost:5000/api/CashDevice/SetDenominationInhibits?
deviceID=SPECTRAL_PAYOUT-COM5',
  'headers': {
    'Content-Type': 'application/json',
    'Authorization': 'Bearer eyJhb.....'
  },
  body: JSON.stringify({
    "ValueCountryCodes": [
      "2000 GBP",
      "5000 GBP"
    ],
    "Inhibit": true
  })
};

request(options, function (error, response) {
  if (error) throw new Error(error);
  console.log(response.body);
});

```

Python

```

import requests
import json

url = "http://localhost:5000/api/CashDevice/SetDenominationInhibits?
deviceID=SPECTRAL_PAYOUT-COM5"

payload = json.dumps({
    "ValueCountryCodes": [
        "2000 GBP",
        "5000 GBP"
    ],
    "Inhibit": True
})
headers = {
    'Content-Type': 'application/json',
    'Authorization': 'Bearer eyJhb.....'
}

```

```
response = requests.request("POST", url, headers=headers, data=payload)

print(response.text)
```

Java

```
OkHttpClient client = new OkHttpClient().newBuilder()
    .build();
MediaType mediaType = MediaType.parse("application/json");
RequestBody body = RequestBody.create(mediaType, "{\"r\\n      \"ValueCountryCodes\":\r\n[\"2000 GBP\", \"5000 GBP\"],\\r\\n      \"Inhibit\": true\\r\\n}\\r\\n\\r\\n");
Request request = new Request.Builder()
    .url("http://localhost:5000/api/CashDevice/SetDenominationInhibits?
deviceID=SPECTRAL_PAYOUT-COM5")
    .method("POST", body)
    .addHeader("Content-Type", "application/json")
    .addHeader("Authorization", "Bearer eyJhb.....")
    .build();
Response response = client.newCall(request).execute();
```

REST API - SetDenominationRoute

Description

Configure the specified denomination to be either routed to cashbox or stored to be made available for later possible payout

| | |
|-------------------|--|
| Endpoint | SetDenominationRoute |
| Method | POST |
| URL | {server_url}/api/CashDevice/SetDenominationRoute |
| Parameters | <ul style="list-style-type: none">• deviceID (string): The unique identifier of the cash device. |
| Body | <p>SetDenomRoutesRequest: The request body containing Value, CountryCode and Route.</p> <ul style="list-style-type: none">• ValueCountryCode valueCountryCode: is an object representing the denomination and its respective country code:<ul style="list-style-type: none">• UInt32 Value: Denomination value e.g 500, 1000 etc• string CountryCode: The country code of the currency e.g. "GBP", "EUR", "USD" etc• DenominationRoute route:<ul style="list-style-type: none">• "Route": 0: DenominationRoute.CASHBOX: Routes to cashbox• "Route": 1 : DenominationRoute.RECYCLER: Routes to single recycler (Smart Pay-out) or an available recycler (NV4000)• "Route": 2 : DenominationRoute.RECYCLER_1: Routes to the first (top) recycler, NV4000 only• "Route": 3 : DenominationRoute.RECYCLER_2: Routes to the second recycler, NV4000 only• "Route": 4 : DenominationRoute.RECYCLER_3: Routes to the third recycler, NV4000 only• "Route": 5 : DenominationRoute.RECYCLER_4: Routes to the fourth (bottom) recycler, NV4000 only <div style="border: 1px solid #ccc; padding: 10px; margin-top: 10px;"><pre>{ "Value": 500, "CountryCode": "{{currency}}", "Route": 1 }</pre></div> |

Responses

| Status | Body | Notes |
|--------|---|---|
| 200 | NV200S: Set 500 EUR route to RECYCLER successfully! | Ok: If the denomination route was set successfully. |
| 400 | | Bad Request: If there is an error setting the denomination route. |

Code Examples

C#

```
var options = new RestClientOptions("http://localhost:5000")
{
    MaxTimeout = -1,
};

var client = new RestClient(options);
var request = new RestRequest("/api/CashDevice/SetDenominationRoute?deviceID=NV200S",
Method.Post);
request.AddHeader("Content-Type", "application/json");
request.AddHeader("Authorization", "Bearer eyJhb.....");

var body = @"
" + "\n" +
@""      ""Value"": 500,
" + "\n" +
@""      ""CountryCode"": ""EUR"",
" + "\n" +
@""      ""Route"": 1
" + "\n" +
@"}"
" + "\n" +
@"";

request.AddStringBody(body, DataFormat.Json);
RestResponse response = await client.ExecuteAsync(request);
Console.WriteLine(response.Content);
```

NodeJS

```
var request = require('request');
var options = {
    'method': 'POST',
    'url': 'http://localhost:5000/api/CashDevice/SetDenominationRoute?deviceID=NV200S',
    'headers': {
        'Content-Type': 'application/json',
        'Authorization': 'Bearer eyJhb.....'
    },
    body: JSON.stringify({
        "Value": 500,
        "CountryCode": "EUR",
        "Route": 1
    })
};

request(options, function (error, response) {
    if (error) throw new Error(error);
    console.log(response.body);
});
```

Python

```
import requests
import json

url = "http://localhost:5000/api/CashDevice/SetDenominationRoute?deviceID=NV200S"
```

```

payload = json.dumps({
    "Value": 500,
    "CountryCode": "EUR",
    "Route": 1
})
headers = {
    'Content-Type': 'application/json',
    'Authorization': 'Bearer eyJhb.....'
}

response = requests.request("POST", url, headers=headers, data=payload)

print(response.text)

```

Java

```

OkHttpClient client = new OkHttpClient().newBuilder()
    .build();
MediaType mediaType = MediaType.parse("application/json");
RequestBody body = RequestBody.create(mediaType, "{\"r\\n    \"Value\": 500,\\r\\n\"CountryCode\": \"EUR\",\\r\\n    \"Route\": 1\\r\\n}\\r\\n\"");
Request request = new Request.Builder()
    .url("http://localhost:5000/api/CashDevice/SetDenominationRoute?deviceID=NV200S")
    .method("POST", body)
    .addHeader("Content-Type", "application/json")
    .addHeader("Authorization", "Bearer eyJhb.....")
    .build();
Response response = client.newCall(request).execute();

```

REST API - GetBarcodeReaderConfiguration

Description

Allows the host to set-up the barcode reader(s) configuration on the device

| | |
|----------------------|---|
| Endpoint | GetBarcodeReaderConfiguration |
| Method | GET |
| URL | {server_url}/api/CashDevice/GetBarcodeReaderConfiguration |
| Parameters | <ul style="list-style-type: none">• deviceID (string): The value of "deviceID" obtained from the response to the OpenConnection request |
| Authorisation | Bearer Token |
| Body | None |

Responses

- hardwareStatus:
 - None
 - Top reader fitted
 - Bottom reader fitted
 - Both fitted
- readerEnabled:
 - None
 - Top
 - Bottom
 - Both
- barcodeFormat:
 - None
 - Interleaved 2 of 5
- numCharacters:
 - Min6, Max 24

| Status | Body | Notes |
|--------|--|--|
| 200 | <pre>{ "hardwareStatus": "Both fitted", "readersEnabled": "Both", "barcodeFormat": "Interleaved 2 of 5", "numCharacters": "18" }</pre> | OK: Successfully retrieved the barcode reader configuration. |
| 404 | <pre>Cash device not found</pre> | Not Found: The specified cash device was not initialised |

| Status | Body | Notes |
|--------|--|--|
| 500 | Unable to get barcode reader configuration. | Internal Server Error: Unable to retrieve barcode reader configuration. |
| 500 | Error during barcode reader configuration: {error_message} | Internal Server Error: An error occurred while retrieving the configuration. |

Code Examples

C#

```
var options = new RestClientOptions("http://localhost:5000")
{
    MaxTimeout = -1,
};
var client = new RestClient(options);
var request = new RestRequest("/api/CashDevice/GetBarcodeReaderConfiguration?
deviceID=SPECTRAL_PAYOUT-COM5", Method.Get);
request.AddHeader("Authorization", "Bearer eyJhb.....");
var body = @"";
request.AddParameter("text/plain", body, ParameterType.RequestBody);
RestResponse response = await client.ExecuteAsync(request);
Console.WriteLine(response.Content);
```

NodeJS

```
var request = require('request');
var options = {
    'method': 'GET',
    'url': 'http://localhost:5000/api/CashDevice/GetBarcodeReaderConfiguration?
deviceID=SPECTRAL_PAYOUT-COM5',
    'headers': {
        'Authorization': 'Bearer eyJhb.....'
    }
};
request(options, function (error, response) {
    if (error) throw new Error(error);
    console.log(response.body);
});
```

Python

```
import requests

url = "http://localhost:5000/api/CashDevice/GetBarcodeReaderConfiguration?
deviceID=SPECTRAL_PAYOUT-COM5"

payload = ""
```

```
headers = {
    'Authorization': 'Bearer eyJhb.....'
}

response = requests.request("GET", url, headers=headers, data=payload)

print(response.text)
```

Java

```
OkHttpClient client = new OkHttpClient().newBuilder()
    .build();
MediaType mediaType = MediaType.parse("text/plain");
RequestBody body = RequestBody.create(mediaType, "");
Request request = new Request.Builder()
    .url("http://localhost:5000/api/CashDevice/GetBarcodeReaderConfiguration?
deviceID=SPECTRAL_PAYOUT-COM5")
    .method("GET", body)
    .addHeader("Authorization", "Bearer eyJhb.....")
    .build();
Response response = client.newCall(request).execute();
```

REST API - SetBarcodeReaderConfiguration

Description

Allows the host to set-up the barcode reader(s) configuration on the device

| | |
|----------------------|--|
| Endpoint | SetBarcodeReaderConfiguration |
| Method | POST |
| URL | {server_url}/api/CashDevice/SetBarcodeReaderConfiguration |
| Parameters | <ul style="list-style-type: none">• deviceID (string): The value of "deviceID" obtained from the response to the OpenConnection request |
| Authorisation | Bearer Token |
| Body | <ul style="list-style-type: none">• byte EnableReaders: Enable Readers<ul style="list-style-type: none">• 0x00 = Enable none• 0x01 = Enable top• 0x02 = Enable bottom• 0x03 = Enable both• byte BarCodeFormat: Bar code format<ul style="list-style-type: none">• 0x00 = None• 0x01 = Interleaved 2 of 5• byte NumberOfCharacters: Number of characters<ul style="list-style-type: none">• Min 6 – Max 24 <div style="border: 1px solid #ccc; padding: 10px; margin-top: 10px;"><pre>{ "EnableReaders": 0, "BarCodeFormat": 0, "NumberOfCharacters": 6 }</pre></div> |

Responses

| Status | Body | Notes |
|--------|--|--|
| 200 | Barcode reader configuration updated successfully. | OK: Barcode reader configuration was successfully updated. |
| 404 | Cash device not found | Not Found: The specified cash device was not initialised |
| 500 | Unable to update barcode reader configuration. | Internal Server Error: Unable to retrieve Barcode Data. |

| Status | Body | Notes |
|--------|--|--|
| 500 | Error during barcode reader configuration: {error_message} | Internal Server Error: An error occurred while updating configuration. |

Code Examples

C#

```
var options = new RestClientOptions("http://localhost:5000")
{
    MaxTimeout = -1,
};
var client = new RestClient(options);
var request = new RestRequest("/api/CashDevice/SetBarcodeReaderConfiguration?
deviceID=SPECTRAL_PAYOUT-COM5", Method.Post);
request.AddHeader("Content-Type", "application/json");
request.AddHeader("Authorization", "Bearer eyJhb.....");
var body = @ "{" + "\n" +
@"    ""EnableReaders"": 0," + "\n" +
@"    ""BarCodeFormat"": 0," + "\n" +
@"    ""NumberOfCharacters"": 6" + "\n" +
@"}";
request.AddStringBody(body, DataFormat.Json);
RestResponse response = await client.ExecuteAsync(request);
Console.WriteLine(response.Content);
```

NodeJS

```
var request = require('request');
var options = {
    'method': 'POST',
    'url': 'http://localhost:5000/api/CashDevice/SetBarcodeReaderConfiguration?
deviceID=SPECTRAL_PAYOUT-COM5',
    'headers': {
        'Content-Type': 'application/json',
        'Authorization': 'Bearer eyJhb.....'
    },
    body: JSON.stringify({
        "EnableReaders": 0,
        "BarCodeFormat": 0,
        "NumberOfCharacters": 6
    })
};

request(options, function (error, response) {
    if (error) throw new Error(error);
    console.log(response.body);
});
```

Python

```
import requests
import json

url = "http://localhost:5000/api/CashDevice/SetBarcodeReaderConfiguration?
deviceID=SPECTRAL_PAYOUT-COM5"

payload = json.dumps({
    "EnableReaders": 0,
    "BarCodeFormat": 0,
    "NumberOfCharacters": 6
})
headers = {
    'Content-Type': 'application/json',
    'Authorization': 'Bearer eyJhb.....'
}

response = requests.request("POST", url, headers=headers, data=payload)

print(response.text)
```

Java

```
OkHttpClient client = new OkHttpClient().newBuilder()
    .build();
MediaType mediaType = MediaType.parse("application/json");
RequestBody body = RequestBody.create(mediaType, "{\"r\\n      \"EnableReaders\": 0,r\\n      \"BarCodeFormat\": 0,r\\n      \"NumberOfCharacters\": 6\\r\\n}+");
Request request = new Request.Builder()
    .url("http://localhost:5000/api/CashDevice/SetBarcodeReaderConfiguration?
deviceID=SPECTRAL_PAYOUT-COM5")
    .method("POST", body)
    .addHeader("Content-Type", "application/json")
    .addHeader("Authorization", "Bearer eyJhb.....")
    .build();
Response response = client.newCall(request).execute();
```

REST API - GetBarcodeData

Description

Obtains the last valid barcode ticket data.

| | |
|----------------------|---|
| Endpoint | GetBarcodeData |
| Method | Get |
| URL | {server_url}/api/CashDevice/GetBarcodeData |
| Parameters | <ul style="list-style-type: none">• deviceID (string): The value of "deviceID" obtained from the response to the OpenConnection request |
| Authorisation | Bearer Token |
| Body | None |

Responses

| Status | Body | Notes |
|--------|--|---|
| 200 | <pre>{ "ticketStatus": "ticket_status_value" "barcodeData": "barcode_ascii_data_value" }</pre> | OK: Successfully retrieved the barcode data and ticket status |
| 404 | Cash device not found | Not Found: The specified cash device was not initialised |
| 500 | Failed to get Barcode Data | Internal Server Error: Unable to retrieve Barcode Data. |

Code Examples

C#

```
var options = new RestClientOptions("http://localhost:5000")
{
    MaxTimeout = -1,
};
var client = new RestClient(options);
var request = new RestRequest("/api/CashDevice/GetBarcodeData?
deviceID=SPECTRAL_PAYOUT-COM5", Method.Get);
```

```

request.AddHeader("Authorization", "Bearer eyJhb.....");
var body = @"";
request.AddParameter("text/plain", body, ParameterType.RequestBody);
RestResponse response = await client.ExecuteAsync(request);
Console.WriteLine(response.Content);

```

NodeJS

```

var request = require('request');
var options = {
  'method': 'GET',
  'url': 'http://localhost:5000/api/CashDevice/GetBarcodeData?deviceID=SPECTRAL_PAYOUT-COM5',
  'headers': {
    'Authorization': 'Bearer eyJhb.....'
  }
};
request(options, function (error, response) {
  if (error) throw new Error(error);
  console.log(response.body);
});

```

Python

```

import requests

url = "http://localhost:5000/api/CashDevice/GetBarcodeData?deviceID=SPECTRAL_PAYOUT-COM5"

payload = ""
headers = {
  'Authorization': 'Bearer eyJhb.....'
}

response = requests.request("GET", url, headers=headers, data=payload)

print(response.text)

```

Java

```

OkHttpClient client = new OkHttpClient().newBuilder()
  .build();
MediaType mediaType = MediaType.parse("text/plain");
RequestBody body = RequestBody.create(mediaType, "");
Request request = new Request.Builder()
  .url("http://localhost:5000/api/CashDevice/GetBarcodeData?deviceID=SPECTRAL_PAYOUT-COM5")
  .method("GET", body)
  .addHeader("Authorization", "Bearer eyJhb.....")
  .build();
Response response = client.newCall(request).execute();

```

REST API - DispenseValue

Description

Dispenses the specified value of a specified currency.

| | |
|----------------------|---|
| Endpoint | DispenseValue |
| Method | POST |
| URL | {server_url}/api/CashDevice/DispenseValue |
| Parameters | <ul style="list-style-type: none">• deviceID (string): The value of "deviceID" obtained from the response to the OpenConnection request |
| Authorisation | Bearer Token |
| Body | <ul style="list-style-type: none">• The denomination and its respective country code.<ul style="list-style-type: none">• UInt32 Value: Denomination value to pay-out e.g 500, 1000 etc• string CountryCode: The country code of the currency that is to be paid out e.g. “GBP”, “EUR”, “USD” etc <div style="border: 1px solid #ccc; padding: 10px; margin-top: 10px;"><pre>{ "Value": 1500, "CountryCode": "{{Currency}}" }</pre></div> |

Responses

| Status | Body | Notes |
|--------|--|--|
| 200 | Dispense operation initiated successfully. | OK: Pay-out can be completed. |
| 400 | Failed to initiate dispense operation: INVALID_INPUT | Bad Request: If there is an error during payout operation. |

errorReason

| Code | Error | Description |
|------|----------------------------|--|
| 1 | Not enough value in device | The value requested exceeds the level stored in the payout device |
| 2 | Cannot pay exact amount | The value requested cannot be paid with the levels stored in the payout device |

| Code | Error | Description |
|-------------|-----------------|---|
| 3 | Device busy | The payout device cannot execute the payout request because it is busy with other tasks |
| 4 | Device disabled | The payout device is in its disabled state and hence refuses the payout request |

Code Examples

C#

```
var options = new RestClientOptions("http://localhost:5000")
{
    MaxTimeout = -1,
};
var client = new RestClient(options);
var request = new RestRequest("/api/CashDevice/DispenseValue?
deviceID=SPECTRAL_PAYOUT-COM5", Method.Post);
request.AddHeader("Content-Type", "application/json");
request.AddHeader("Authorization", "Bearer eyJhb.....");
var body = @""{"" + "\n" +
@"" ""Value"": 1500," + "\n" +
@"" ""CountryCode"": ""GBP"" + "\n" +
@""}"" + "\n" +
@""";
request.AddStringBody(body, DataFormat.Json);
RestResponse response = await client.ExecuteAsync(request);
Console.WriteLine(response.Content);
```

NodeJS

```
var request = require('request');
var options = {
    'method': 'POST',
    'url': 'http://localhost:5000/api/CashDevice/DispenseValue?
deviceID=SPECTRAL_PAYOUT-COM5',
    'headers': {
        'Content-Type': 'application/json',
        'Authorization': 'Bearer eyJhb.....'
    },
    body: JSON.stringify({
        "Value": 1500,
        "CountryCode": "GBP"
    })
};

request(options, function (error, response) {
    if (error) throw new Error(error);
    console.log(response.body);
});
```

Python

```
import requests
import json

url = "http://localhost:5000/api/CashDevice/DispenseValue?deviceID=SPECTRAL_PAYOUT-
COM5"

payload = json.dumps({
    "Value": 1500,
    "CountryCode": "GBP"
})
headers = {
    'Content-Type': 'application/json',
    'Authorization': 'Bearer eyJhb.....'
}

response = requests.request("POST", url, headers=headers, data=payload)

print(response.text)
```

Java

```
OkHttpClient client = new OkHttpClient().newBuilder()
    .build();
MediaType mediaType = MediaType.parse("application/json");
RequestBody body = RequestBody.create(mediaType, "{\"Value\": 1500,\n\"CountryCode\": \"GBP\"\n}");
Request request = new Request.Builder()
    .url("http://localhost:5000/api/CashDevice/DispenseValue?deviceID=SPECTRAL_PAYOUT-
COM5")
    .method("POST", body)
    .addHeader("Content-Type", "application/json")
    .addHeader("Authorization", "Bearer eyJhb.....")
    .build();
Response response = client.newCall(request).execute();
```

REST API - Float

Description

Leaves the specified number of notes in the pay-out and moves the excess to the cashbox to maintain a stored pay-out amount for the purpose of being able to collect excess notes whilst leaving some for pay-out purposes.

| | |
|---------------------------|--|
| Endpoint | Float |
| Method | POST |
| URL | {server_url}/api/CashDevice/Float |
| Parameters | <ul style="list-style-type: none">• deviceID (string): The value of "deviceID" obtained from the response to the OpenConnection request |
| Authorisation | Bearer Token |
| Body | <ul style="list-style-type: none">• UInt32[] noteCounts:<ul style="list-style-type: none">• An array of UInt32 of size equal to the number of channels in the dataset. Each UInt32 is the number of notes to be left for payout, its index is the channel to float• E.g. For a GBP dataset containing £5, £10, £20 and £50<ul style="list-style-type: none">• If noteCounts = [2, 2, 2, 0]<ul style="list-style-type: none">• 2 x £5, 2 x £10, 2 x £20 and 0 x £50 notes would be left in the payout. |
| <code>[2, 2, 2, 0]</code> | |

Responses

| Status | Body | Notes |
|--------|---|---|
| 200 | SPECTRAL_PAYOUT-COM5: Float operation completed successfully. | OK: If the float operation was initiated successfully. |
| 400 | Not enough value to perform float . | Bad Request: If there is an error initiating the float operation. |

Code Examples

C#

```
var options = new RestClientOptions("http://localhost:5000")
{
    MaxTimeout = -1,
};
var client = new RestClient(options);
```

```

var request = new RestRequest("/api/CashDevice/Float?deviceID=SPECTRAL_PAYOUT-COM5",
Method.Post);
request.AddHeader("Content-Type", "application/json");
request.AddHeader("Authorization", "Bearer eyJhb.....");
var body = @" [2, 2, 2, 0]" + "\n" +
@""";
request.AddStringBody(body, DataFormat.Json);
RestResponse response = await client.ExecuteAsync(request);
Console.WriteLine(response.Content);

```

NodeJS

```

var request = require('request');
var options = {
  'method': 'POST',
  'url': 'http://localhost:5000/api/CashDevice/Float?deviceID=SPECTRAL_PAYOUT-COM5',
  'headers': {
    'Content-Type': 'application/json',
    'Authorization': 'Bearer eyJhb.....'
  },
  body: JSON.stringify([
    2,
    2,
    2,
    0
  ])
};

request(options, function (error, response) {
  if (error) throw new Error(error);
  console.log(response.body);
});

```

Python

```

import requests
import json

url = "http://localhost:5000/api/CashDevice/Float?deviceID=SPECTRAL_PAYOUT-COM5"

payload = json.dumps([
  2,
  2,
  2,
  0
])
headers = {
  'Content-Type': 'application/json',
  'Authorization': 'Bearer eyJhb.....'
}

response = requests.request("POST", url, headers=headers, data=payload)

print(response.text)

```

Java

```
OkHttpClient client = new OkHttpClient().newBuilder()
    .build();
MediaType mediaType = MediaType.parse("application/json");
RequestBody body = RequestBody.create(mediaType, " [2, 2, 2, 0]\r\n");
Request request = new Request.Builder()
    .url("http://localhost:5000/api/CashDevice/Float?deviceID=SPECTRAL_PAYOUT-COM5")
    .method("POST", body)
    .addHeader("Content-Type", "application/json")
    .addHeader("Authorization", "Bearer eyJhb.....")
    .build();
Response response = client.newCall(request).execute();
```

REST API - SetCashboxPayoutLimit

Description

Sets the cashbox payout limit by specifying the maximum number of notes for each denomination.



Not available for NV4000

| | |
|------------------------|--|
| Endpoint | SetCashboxPayoutLimit |
| Method | POST |
| URL | {server_url}/api/CashDevice/SetCashboxPayoutLimit |
| Parameters | <ul style="list-style-type: none">• deviceID (string): The value of "deviceID" obtained from the response to the OpenConnection request |
| Authorisation | Bearer Token |
| Body | <ul style="list-style-type: none">• uint[] noteCounts: An array of note counts representing the maximum number of notes that can be paid out for each denomination. Each index corresponds to a specific denomination in the currency assignment array.• Allow the host to specify a maximum level of coins/notes, by denomination, to be left in the hopper/recycler. The default value after the unit is powered up/reset is 0 (no limit). Limit is enabled for the denomination only when the limit value is > 0.• When the denomination level is above the desired limit for the recycler, the note will be sent to cashbox even if routed to recycler.<ul style="list-style-type: none">• E.g. For the dataset containing £5, £10, £20, £50.• To set the payout limit to 5 for £5, 5 for £10, 5 for £20, the parameter noteCounts would be [5,5,5,0]. |
| <code>[5,5,5,0]</code> | |

Responses

| Status | Body | Notes |
|--------|---|---|
| 200 | <div style="border: 1px solid #ccc; padding: 5px; width: fit-content;">SPECTRAL_PAYOUT-COM5: Cashbox payout limit set successfully.</div> | OK: If the cashbox payout limit was set successfully. |
| 400 | | Bad Request: If there is an error setting the cashbox payout limit. |

Code Examples

C#

```
var options = new RestClientOptions("http://localhost:5000")
{
    MaxTimeout = -1,
};

var client = new RestClient(options);
var request = new RestRequest("/api/CashDevice/SetCashboxPayoutLimit?
deviceID=SPECTRAL_PAYOUT-COM5", Method.Post);
request.AddHeader("Content-Type", "application/json");
request.AddHeader("Authorization", "Bearer eyJhb.....");

var body = @"[5,5,5,0]";
request.AddStringBody(body, DataFormat.Json);
RestResponse response = await client.ExecuteAsync(request);
Console.WriteLine(response.Content);
```

NodeJS

```
var request = require('request');
var options = {
    'method': 'POST',
    'url': 'http://localhost:5000/api/CashDevice/SetCashboxPayoutLimit?
deviceID=SPECTRAL_PAYOUT-COM5',
    'headers': {
        'Content-Type': 'application/json',
        'Authorization': 'Bearer eyJhb.....'
    },
    body: JSON.stringify([
        5,
        5,
        5,
        0
    ])
};

request(options, function (error, response) {
    if (error) throw new Error(error);
    console.log(response.body);
});
```

Python

```
import requests
import json

url = "http://localhost:5000/api/CashDevice/SetCashboxPayoutLimit?
deviceID=SPECTRAL_PAYOUT-COM5"

payload = json.dumps([
    5,
    5,
    5,
    0
])
headers = {
```

```
'Content-Type': 'application/json',
'Authorization': 'Bearer eyJhb.....'
}

response = requests.request("POST", url, headers=headers, data=payload)

print(response.text)
```

Java

```
OkHttpClient client = new OkHttpClient().newBuilder()
    .build();
MediaType mediaType = MediaType.parse("application/json");
RequestBody body = RequestBody.create(mediaType, "[5,5,5,0]");
Request request = new Request.Builder()
    .url("http://localhost:5000/api/CashDevice/SetCashboxPayoutLimit?
deviceID=SPECTRAL_PAYOUT-COM5")
    .method("POST", body)
    .addHeader("Content-Type", "application/json")
    .addHeader("Authorization", "Bearer eyJhb.....")
    .build();
Response response = client.newCall(request).execute();
```

REST API - SmartEmpty

Description

Initiates a smart empty operation on the specified module number of the dispenser.

| | |
|----------------------|---|
| Endpoint | SmartEmpty |
| Method | POST |
| URL | {server_url}/api/CashDevice/SmartEmpty |
| Parameters | <ul style="list-style-type: none">• deviceID (string): The value of "deviceID" obtained from the response to the OpenConnection request |
| Authorisation | Bearer Token |
| Body | <ul style="list-style-type: none">• byte moduleNumber: the module number to empty. If NV4000 is true and moduleNumber is 0, all modules will be emptied.• bool NV4000: indicating whether the operation is for NV4000 model. Default is false. <div style="border: 1px solid #ccc; padding: 10px; margin-top: 10px;"><pre>{ "ModuleNumber": 0, "IsNV4000": false }</pre></div> |

Responses

| Status | Body | Notes |
|--------|--|---|
| 200 | <pre>{ "dispenseResult": "COMPLETED" }</pre> | OK: If the smart empty operation was completed successfully. |
| 400 | | Bad Request: If there is an error performing the smart empty operation. |

Code Examples

C#

```
var options = new RestClientOptions("http://localhost:5000")
{
    MaxTimeout = -1,
};
var client = new RestClient(options);
var request = new RestRequest("/api/CashDevice/SmartEmpty?deviceID=SPECTRAL_PAYOUT-
COM5", Method.Post);
```

```

request.AddHeader("Content-Type", "application/json");
request.AddHeader("Authorization", "Bearer eyJhb.....");
var body = @"{" + "\n" +
@"" ""ModuleNumber"": 0," + "\n" +
@"" ""IsNV4000"": false" + "\n" +
@"}";
request.AddStringBody(body, DataFormat.Json);
RestResponse response = await client.ExecuteAsync(request);
Console.WriteLine(response.Content);

```

NodeJS

```

var request = require('request');
var options = {
  'method': 'POST',
  'url': 'http://localhost:5000/api/CashDevice/SmartEmpty?deviceID=SPECTRAL_PAYOUT-
COM5',
  'headers': {
    'Content-Type': 'application/json',
    'Authorization': 'Bearer eyJhb.....'
  },
  body: JSON.stringify({
    "ModuleNumber": 0,
    "IsNV4000": false
  })
};

request(options, function (error, response) {
  if (error) throw new Error(error);
  console.log(response.body);
});

```

Python

```

import requests
import json

url = "http://localhost:5000/api/CashDevice/SmartEmpty?deviceID=SPECTRAL_PAYOUT-COM5"

payload = json.dumps({
    "ModuleNumber": 0,
    "IsNV4000": False
})
headers = {
    'Content-Type': 'application/json',
    'Authorization': 'Bearer eyJhb.....'
}

response = requests.request("POST", url, headers=headers, data=payload)

print(response.text)

```

Java

```

OkHttpClient client = new OkHttpClient().newBuilder()
    .build();

```

```
MediaType mediaType = MediaType.parse("application/json");
RequestBody body = RequestBody.create(mediaType, "{\"r\n  \"ModuleNumber\": 0,\r\n\"IsNV4000\": false\r\n}");
Request request = new Request.Builder()
  .url("http://localhost:5000/api/CashDevice/SmartEmpty?deviceID=SPECTRAL_PAYOUT-
COM5")
  .method("POST", body)
  .addHeader("Content-Type", "application/json")
  .addHeader("Authorization", "Bearer eyJhb.....")
  .build();
Response response = client.newCall(request).execute();
```

REST API - SendCustomCommand

Description

Sends a custom command to the device, constructed as an array of bytes

| | |
|----------------------|---|
| Endpoint | SendCustomCommand |
| Method | POST |
| URL | {server_url}/api/CashDevice/SendCustomCommand |
| Parameters | <ul style="list-style-type: none">• deviceID (string): The value of "deviceID" obtained from the response to the OpenConnection request |
| Authorisation | Bearer Token |
| Body | string CustomCommandDataString: A string containing the raw byte data to be sent to the device. <div style="border: 1px solid #ccc; padding: 10px; width: fit-content;"><pre>{ "CustomCommandDataString": "01" }</pre></div> |

Responses

| Status | Body | Notes |
|--------|-----------------------------------|--|
| 200 | Custom command sent successfully. | OK: If the custom command was sent successfully and acknowledged by cash device. |
| 400 | | Bad Request: If there is an error sending the custom command. |

Code Examples

C#

```
var options = new RestClientOptions("http://localhost:5000")
{
    MaxTimeout = -1,
};

var client = new RestClient(options);
var request = new RestRequest("/api/CashDevice/SendCustomCommand?
deviceID=SPECTRAL_PAYOUT-COM5", Method.Post);
request.AddHeader("Content-Type", "application/json");
request.AddHeader("Authorization", "Bearer eyJhb.....");

var body = @"{" + "\n" +
@""    ""CustomCommandDataString"": """01"""" + "\n" +
```

```

        @"}" + "\n" +
        @"";
    request.AddStringBody(body, DataFormat.Json);
    RestResponse response = await client.ExecuteAsync(request);
    Console.WriteLine(response.Content);

```

NodeJS

```

var request = require('request');
var options = {
  'method': 'POST',
  'url': 'http://localhost:5000/api/CashDevice/SendCustomCommand?
deviceID=SPECTRAL_PAYOUT-COM5',
  'headers': {
    'Content-Type': 'application/json',
    'Authorization': 'Bearer eyJhb.....'
  },
  body: JSON.stringify({
    "CustomCommandDataString": "01"
  })
};

request(options, function (error, response) {
  if (error) throw new Error(error);
  console.log(response.body);
});

```

Python

```

import requests
import json

url = "http://localhost:5000/api/CashDevice/SendCustomCommand?
deviceID=SPECTRAL_PAYOUT-COM5"

payload = json.dumps({
    "CustomCommandDataString": "01"
})
headers = {
    'Content-Type': 'application/json',
    'Authorization': 'Bearer eyJhb.....'
}

response = requests.request("POST", url, headers=headers, data=payload)

print(response.text)

```

Java

```

OkHttpClient client = new OkHttpClient().newBuilder()
    .build();
MediaType mediaType = MediaType.parse("application/json");
RequestBody body = RequestBody.create(mediaType, "{\r\n
\"CustomCommandDataString\": \"01\"\r\n}");
Request request = new Request.Builder()

```

```
.url("http://localhost:5000/api/CashDevice/SendCustomCommand?  
deviceID=SPECTRAL_PAYOUT-COM5")  
.method("POST", body)  
.addHeader("Content-Type", "application/json")  
.addHeader("Authorization", "Bearer eyJhb.....")  
.build();  
Response response = client.newCall(request).execute();
```

REST API - EnableCoinMechOrFeeder

Description

Enables the coin mechanism or feeder for specific device models

| | |
|----------------------|---|
| Endpoint | EnableCoinMechOrFeeder |
| Method | POST |
| URL | {server_url}/api/CashDevice/EnableCoinMechOrFeeder |
| Parameters | <ul style="list-style-type: none">deviceID (string): The value of "deviceID" obtained from the response to the OpenConnection request |
| Authorisation | Bearer Token |
| Body | None |

Responses

| Status | Body | Notes |
|--------|--|-------------------------------|
| 200 | Coin mechanism or feeder enabled successfully. | OK: If successfully enabled. |
| 400 | | Bad Request: If enable fails. |

Code Examples

C#

```
var options = new RestClientOptions("http://localhost:5000")
{
    MaxTimeout = -1,
};
var client = new RestClient(options);
var request = new RestRequest("/api/CashDevice/EnableCoinMechOrFeeder?
deviceID=SMART_COIN_SYSTEM-COM10", Method.Post);
request.AddHeader("Authorization", "Bearer eyJhb.....");
RestResponse response = await client.ExecuteAsync(request);
Console.WriteLine(response.Content);
```

NodeJS

```
var request = require('request');
var options = {
    'method': 'POST',
```

```

'url': 'http://localhost:5000/api/CashDevice/EnableCoinMechOrFeeder?
deviceID=SMART_COIN_SYSTEM-COM10',
'headers': {
  'Authorization': 'Bearer eyJhb.....'
}
};

request(options, function (error, response) {
  if (error) throw new Error(error);
  console.log(response.body);
});

```

Python

```

import requests

url = "http://localhost:5000/api/CashDevice/EnableCoinMechOrFeeder?
deviceID=SMART_COIN_SYSTEM-COM10"

payload = {}
headers = {
  'Authorization': 'Bearer eyJhb.....'
}

response = requests.request("POST", url, headers=headers, data=payload)

print(response.text)

```

Java

```

OkHttpClient client = new OkHttpClient().newBuilder()
  .build();
MediaType mediaType = MediaType.parse("text/plain");
RequestBody body = RequestBody.create(mediaType, "");
Request request = new Request.Builder()
  .url("http://localhost:5000/api/CashDevice/EnableCoinMechOrFeeder?
deviceID=SMART_COIN_SYSTEM-COM10")
  .method("POST", body)
  .addHeader("Authorization", "Bearer eyJhb.....")
  .build();
Response response = client.newCall(request).execute();

```

REST API - ResetDevice

Description

Resets the device to its initial state.

| | |
|----------------------|---|
| Endpoint | ResetDevice |
| Method | POST |
| URL | {server_url}/api/CashDevice/ResetDevice |
| Parameters | <ul style="list-style-type: none">deviceID (string): The value of "deviceID" obtained from the response to the OpenConnection request |
| Authorisation | Bearer Token |
| Body | None |

Responses

| Status | Body | Notes |
|--------|----------------------------|---|
| 200 | Device reset successfully. | OK: If the device was reset successfully. |
| 400 | | Bad Request: If there is an error resetting the device. |

Code Examples

C#

```
var options = new RestClientOptions("http://localhost:5000")
{
    MaxTimeout = -1,
};
var client = new RestClient(options);
var request = new RestRequest("/api/CashDevice/ResetDevice?deviceID=SPECTRAL_PAYOUT-
COM5", Method.Post);
request.AddHeader("Authorization", "Bearer eyJhb.....");
RestResponse response = await client.ExecuteAsync(request);
Console.WriteLine(response.Content);
```

NodeJS

```
var request = require('request');
var options = {
    'method': 'POST',
```

```

'url': 'http://localhost:5000/api/CashDevice/ResetDevice?deviceID=SPECTRAL_PAYOUT-
COM5',
'headers': {
    'Authorization': 'Bearer eyJhb.....'
}
};

request(options, function (error, response) {
    if (error) throw new Error(error);
    console.log(response.body);
});

```

Python

```

import requests

url = "http://localhost:5000/api/CashDevice/ResetDevice?deviceID=SPECTRAL_PAYOUT-
COM5"

payload = {}
headers = [
    'Authorization': 'Bearer eyJhb.....'
]

response = requests.request("POST", url, headers=headers, data=payload)

print(response.text)

```

Java

```

OkHttpClient client = new OkHttpClient().newBuilder()
    .build();
MediaType mediaType = MediaType.parse("text/plain");
RequestBody body = RequestBody.create(mediaType, "");
Request request = new Request.Builder()
    .url("http://localhost:5000/api/CashDevice/ResetDevice?deviceID=SPECTRAL_PAYOUT-
COM5")
    .method("POST", body)
    .addHeader("Authorization", "Bearer eyJhb.....")
    .build();
Response response = client.newCall(request).execute();

```

REST API - HaltPayout

Description

Halts the payout operation of the device.

| | |
|----------------------|---|
| Endpoint | HaltPayout |
| Method | POST |
| URL | {server_url}/api/CashDevice/HaltPayout |
| Parameters | <ul style="list-style-type: none">deviceID (string): The value of "deviceID" obtained from the response to the OpenConnection request |
| Authorisation | Bearer Token |
| Body | None |

Responses

| Status | Body | Notes |
|--------|-----------------------------|---|
| 200 | Payout halted successfully. | OK: If the device was halted successfully. |
| 400 | | Bad Request: If there is an error halting the payout. |

Code Examples

C#

```
var options = new RestClientOptions("http://localhost:5000")
{
    MaxTimeout = -1,
};
var client = new RestClient(options);
var request = new RestRequest("/api/CashDevice/HaltPayout?deviceID=SPECTRAL_PAYOUT-
COM5", Method.Post);
request.AddHeader("Authorization", "Bearer eyJhb.....");
RestResponse response = await client.ExecuteAsync(request);
Console.WriteLine(response.Content);
```

NodeJS

```
var request = require('request');
var options = {
    'method': 'POST',
```

```

'url': 'http://localhost:5000/api/CashDevice/HaltPayout?deviceID=SPECTRAL_PAYOUT-
COM5',
'headers': {
  'Authorization': 'Bearer eyJhb.....'
}
};

request(options, function (error, response) {
  if (error) throw new Error(error);
  console.log(response.body);
});

```

Python

```

import requests

url = "http://localhost:5000/api/CashDevice/HaltPayout?deviceID=SPECTRAL_PAYOUT-COM5"

payload = {}
headers = {
  'Authorization': 'Bearer eyJhb.....'
}

response = requests.request("POST", url, headers=headers, data=payload)

print(response.text)

```

Java

```

OkHttpClient client = new OkHttpClient().newBuilder()
  .build();
MediaType mediaType = MediaType.parse("text/plain");
RequestBody body = RequestBody.create(mediaType, "");
Request request = new Request.Builder()
  .url("http://localhost:5000/api/CashDevice/HaltPayout?deviceID=SPECTRAL_PAYOUT-
COM5")
  .method("POST", body)
  .addHeader("Authorization", "Bearer eyJhb.....")
  .build();
Response response = client.newCall(request).execute();

```

REST API - GetRCMode

Description

Retrieve the current mode of replenishment cassette.

- Replenishment Mode
- Payout Mode



Can only be used with the NV4000 with RC device model.

| | |
|----------------------|---|
| Endpoint | GetRCMode |
| Method | Get |
| URL | {server_url}/api/CashDevice/GetRCMode |
| Parameters | <ul style="list-style-type: none">• deviceID (string): The value of "deviceID" obtained from the response to the OpenConnection request |
| Authorisation | Bearer Token |
| Body | None |

Responses

| Status | Body | Notes |
|--------|------------------------|--|
| 200 | RC_MODE_PAYOUT | OK: Successfully retrieved the RC mode |
| 404 | Cash device not found | Not Found: The specified cash device was not initialised |
| 500 | Failed to get RC Mode. | Internal Server Error: Unable to retrieve RC Mode. |

Code Examples

C#

```
var options = new RestClientOptions("http://localhost:5000")
{
    MaxTimeout = -1,
};

var client = new RestClient(options);
```

```

var request = new RestRequest("/api/CashDevice/GetRCMode?deviceID=SPECTRAL_PAYOUT-
COM5", Method.Get);
request.AddHeader("Authorization", "Bearer eyJhb.....");
RestResponse response = await client.ExecuteAsync(request);
Console.WriteLine(response.Content);

```

NodeJS

```

var request = require('request');
var options = {
  'method': 'GET',
  'url': 'http://localhost:5000/api/CashDevice/GetRCMode?deviceID=SPECTRAL_PAYOUT-
COM5',
  'headers': {
    'Authorization': 'Bearer eyJhb.....'
  }
};
request(options, function (error, response) {
  if (error) throw new Error(error);
  console.log(response.body);
});

```

Python

```

import requests

url = "http://localhost:5000/api/CashDevice/GetRCMode?deviceID=SPECTRAL_PAYOUT-COM5"

payload = {}
headers = {
  'Authorization': 'Bearer eyJhb.....'
}

response = requests.request("GET", url, headers=headers, data=payload)

print(response.text)

```

Java

```

OkHttpClient client = new OkHttpClient().newBuilder()
  .build();
MediaType mediaType = MediaType.parse("text/plain");
RequestBody body = RequestBody.create(mediaType, "");
Request request = new Request.Builder()
  .url("http://localhost:5000/api/CashDevice/GetRCMode?deviceID=SPECTRAL_PAYOUT-
COM5")
  .method("GET", body)
  .addHeader("Authorization", "Bearer eyJhb.....")
  .build();
Response response = client.newCall(request).execute();

```

REST API - Replenish

Description

Replenishes the specified number of notes from the replenishment cassette to the recyclers.



Can only be used with the NV4000 with RC device model.

| | |
|----------------------|--|
| Endpoint | Replenish |
| Method | POST |
| URL | {server_url}/api/CashDevice/Replenish |
| Parameters | <ul style="list-style-type: none">• deviceId (string): The value of "deviceId" obtained from the response to the OpenConnection request |
| Authorisation | Bearer Token |
| Body | <ul style="list-style-type: none">• uint NumberToReplenish: The number of notes to replenish from the RC to the pay-out modules e.g. to replenish 5 notes to recyclers, set NumberToReplenish to 5. <div style="border: 1px solid #ccc; padding: 5px; text-align: center;">5</div> |

Responses

| Status | Body | Notes |
|--------|---------------------------------------|---|
| 200 | Replenishment completed successfully. | OK: Notes replenished successfully. |
| 400 | | Bad Request: If there is an error during replenishment. |

Code Examples

C#

```
var options = new RestClientOptions("http://localhost:5000")
{
    MaxTimeout = -1,
};
var client = new RestClient(options);
var request = new RestRequest("/api/CashDevice/Replenish?deviceID=NV4000-COM5",
    Method.Post);
request.AddHeader("Content-Type", "application/json");
request.AddHeader("Authorization", "Bearer eyJhb.....");
var body = @"4";
```

```

request.AddStringBody(body, DataFormat.Json);
RestResponse response = await client.ExecuteAsync(request);
Console.WriteLine(response.Content);

```

NodeJS

```

var request = require('request');
var options = {
  'method': 'POST',
  'url': 'http://localhost:5000/api/CashDevice/Replenish?deviceID=NV4000-COM5',
  'headers': {
    'Content-Type': 'application/json',
    'Authorization': 'Bearer eyJhb.....'
  },
  body: JSON.stringify(4)

};

request(options, function (error, response) {
  if (error) throw new Error(error);
  console.log(response.body);
});

```

Python

```

import requests
import json

url = "http://localhost:5000/api/CashDevice/Replenish?deviceID=NV4000-COM5"

payload = json.dumps(4)
headers = {
  'Content-Type': 'application/json',
  'Authorization': 'Bearer eyJhb.....'
}

response = requests.request("POST", url, headers=headers, data=payload)

print(response.text)

```

Java

```

OkHttpClient client = new OkHttpClient().newBuilder()
  .build();
MediaType mediaType = MediaType.parse("application/json");
RequestBody body = RequestBody.create(mediaType, "4");
Request request = new Request.Builder()
  .url("http://localhost:5000/api/CashDevice/Replenish?deviceID=NV4000-COM5")
  .method("POST", body)
  .addHeader("Content-Type", "application/json")
  .addHeader("Authorization", "Bearer eyJhb.....")
  .build();
Response response = client.newCall(request).execute();

```

REST API - RefillMode

Description

Enables or disables the refill mode of the device

| | |
|----------------------|---|
| Endpoint | RefillMode |
| Method | POST |
| URL | {server_url}/api/CashDevice/RefillMode |
| Parameters | <ul style="list-style-type: none">• deviceID (string): The value of "deviceID" obtained from the response to the OpenConnection request |
| Authorisation | Bearer Token |
| Body | <ul style="list-style-type: none">• True to enable, or false to disable. |

Responses

| Status | Body | Notes |
|--------|---|--|
| 200 | NV4000-COM5: Refill mode disabled successfully. | OK: If the refill mode was set successfully. |
| 400 | | Bad Request: If there is an error setting the refill mode. |

Code Examples

C#

```
var options = new RestClientOptions("http://localhost:5000")
{
    MaxTimeout = -1,
};
var client = new RestClient(options);
var request = new RestRequest("/api/CashDevice/RefillMode?deviceID=NV4000-COM5",
    Method.Post);
request.AddHeader("Content-Type", "application/json");
request.AddHeader("Authorization", "Bearer eyJhb.....");
var body = @"false";
request.AddStringBody(body, DataFormat.Json);
RestResponse response = await client.ExecuteAsync(request);
Console.WriteLine(response.Content);
```

NodeJS

```
var request = require('request');
var options = {
  'method': 'POST',
  'url': 'http://localhost:5000/api/CashDevice/RefillMode?deviceID=NV4000-COM5',
  'headers': {
    'Content-Type': 'application/json',
    'Authorization': 'Bearer eyJhb.....'
  },
  body: JSON.stringify(false)

};

request(options, function (error, response) {
  if (error) throw new Error(error);
  console.log(response.body);
});
```

Python

```
import requests
import json

url = "http://localhost:5000/api/CashDevice/RefillMode?deviceID=NV4000-COM5"

payload = json.dumps(False)
headers = {
  'Content-Type': 'application/json',
  'Authorization': 'Bearer eyJhb.....'
}

response = requests.request("POST", url, headers=headers, data=payload)

print(response.text)
```

Java

```
OkHttpClient client = new OkHttpClient().newBuilder()
  .build();
MediaType mediaType = MediaType.parse("application/json");
RequestBody body = RequestBody.create(mediaType, "false");
Request request = new Request.Builder()
  .url("http://localhost:5000/api/CashDevice/RefillMode?deviceID=NV4000-COM5")
  .method("POST", body)
  .addHeader("Content-Type", "application/json")
  .addHeader("Authorization", "Bearer eyJhb.....")
  .build();
Response response = client.newCall(request).execute();
```

REST API - GetHopperOptions

Description

Retrieves the current hopper options from the device and updates internal registers with this information.

| | |
|----------------------|---|
| Endpoint | GetHopperOptions |
| Method | GET |
| URL | {server_url}/api/CashDevice/GetHopperOptions |
| Parameters | <ul style="list-style-type: none">deviceID (string): The value of "deviceID" obtained from the response to the OpenConnection request |
| Authorisation | Bearer Token |
| Body | None |

Responses

| Reg_0 bits and their meaning | | |
|------------------------------|-------------|--|
| Bit | Parameter | Function |
| 0 | Pay Mode | 0x00 = Split by highest value (Default). The device will attempt to payout a requested value by starting from the highest to the lowest coins available. This mode will payout the minimum number of coins possible. 0x01 = Free pay. The device will payout a coin as it passes its discriminator system if it fits into the current payout value and will leave enough of other coins to payout the rest of the value. This may give a faster payout but could result in a large number of coins of small denominations paid out. |
| 1 | Level Check | 0x00 = Disabled. The device will not refer to the level counters when calculating if a payout value can be made. 0x01 = Enabled (Default). The device will check the level counters and accept or refuse a payout request based on levels and/or split of available levels. |
| 2 | Motor Speed | 0x00 = Low speed. Payouts run at a lower motor speed. 0x01 = High Speed (default always after reset). The motors run at max speed for payouts. |

| Reg_0 bits and their meaning | | |
|------------------------------|--------------------------------|---|
| Bit | Parameter | Function |
| 3 | Cashbox Pay Active | <p>This bit is used in conjunction with Bit 0. If bit 3 is zero, then the Pay modes will be as described in bit 0. If Bit 3 is set then coins routed to the cashbox will be used in coins paid out of the front if they can fit into the current payout request. This is shown below.</p> <p>Pay Mode Type Bit 0 Bit 3</p> <p>Free Play 1 0 Highest Split 0 1 All Route Free Pay 1 1 All Route Highest Split 0 1</p> |
| 4 | Route 0 level coins to cashbox | Set to 0x01 means that any coins detected with a level setting of 0 will be paid to the cashbox, even if it is routed to the payout. |
| 5 | High Efficiency Split | Default set to 0x01 to enable a more efficient, smarter coin payout algorithm which will tend to use coins which have higher level counts - thus speeding up the payout process. |
| 6 | Unknown to Payout | Set to 0x01 means any unknown coins will be paid out during Smart Empty (otherwise they will be routed to cashbox). |
| 7 | Value Added | 0x00 = Coin added event 0x01 = Value added event |
| Reg_1 bits and their meaning | | |
| Bit | Parameter | Function |
| 0 | Reject Events | Set to 1 gives reject event 0xBA Coin Rejected. |
| 1 | Reject Events Full | Set to 1 gives reject event 0xBA with coin value if known. |
| 2 | Empty Route | Set to 1 will route coins to payout when a Empty or Smart Empty command is received. |
| 3 | Full to Cashbox | Set to 1 will start moving coins to cashbox during payouts when full. |
| 4 | Value Coin | Set to 1 gives individual coin payout events. |
| 5 | N/A | Set to 0. |
| 6 | N/A | Set to 0. |
| 7 | N/A | Set to 0. |

Example Response

| Status | Body | Notes |
|--------|--|--|
| 200 | <pre>SMART_COIN_SYSTEM-COM10: { Reg0 = 04, Reg1 = 00 }</pre> | OK: If the hopper options were retrieved successfully. |
| 400 | | Bad Request: If there is an error retrieving the hopper options. |

Code Examples

C#

```
var options = new RestClientOptions("http://localhost:5000")
{
    MaxTimeout = -1,
};
var client = new RestClient(options);
var request = new RestRequest("/api/CashDevice/GetHopperOptions?
deviceID=SMART_COIN_SYSTEM-COM10", Method.Get);
request.AddHeader("Authorization", "Bearer eyJhbim... import requests

url = "http://localhost:5000/api/CashDevice/GetHopperOptions?
deviceID=SMART_COIN_SYSTEM-COM10"

payload = {}
headers = {
    'Authorization': 'Bearer
eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJ1bm...fbmFtZSI6ImFkbWluIiwibmJmIjoxNzM4MDU1
NjEzLCJleHAiOjE3Mzg2NjA0MTMsImlhdCI6MTczODA1NTYxMywiaXNzIjoiSU50T1ZBVElWRVRFQ0hOT0xPR
1kiLCJhdWQiOjJJTk5PVkFUSVZFVEVDSE5PTE9HWSJ9.zi55PwWV2dwGkzqXEI7Jk4R3SQnXbphnZyyif72QC
Ws'
}

response = requests.request("GET", url, headers=headers, data=payload)

print(response.text)
.....");
RestResponse response = await client.ExecuteAsync(request);
Console.WriteLine(response.Content);
```

NodeJS

```
var request = require('request');
var options = {
    'method': 'GET',
    'url': 'http://localhost:5000/api/CashDevice/GetHopperOptions?
deviceID=SMART_COIN_SYSTEM-COM10',
    'headers': {
        'Authorization': 'Bearer eyJhb....'
    }
};
request(options, function (error, response) {
```

```
if (error) throw new Error(error);
console.log(response.body);
});
```

Python

```
import requests

url = "http://localhost:5000/api/CashDevice/GetHopperOptions?
deviceID=SMART_COIN_SYSTEM-COM10"

payload = {}
headers = {
    'Authorization': 'Bearer eyJhb.....'
}

response = requests.request("GET", url, headers=headers, data=payload)

print(response.text)
```

Java

```
OkHttpClient client = new OkHttpClient().newBuilder()
    .build();
MediaType mediaType = MediaType.parse("text/plain");
RequestBody body = RequestBody.create(mediaType, "");
Request request = new Request.Builder()
    .url("http://localhost:5000/api/CashDevice/GetHopperOptions?
deviceID=SMART_COIN_SYSTEM-COM10")
    .method("GET", body)
    .addHeader("Authorization", "Bearer eyJhb.....")
    .build();
Response response = client.newCall(request).execute();
```

REST API - SetHopperOptions

Description

Sets the hopper options on the device using the provided register values.

| | |
|----------------------|--|
| Endpoint | SetHopperOptions |
| Method | POST |
| URL | {server_url}/api/CashDevice/SetHopperOptions |
| Parameters | <ul style="list-style-type: none">deviceID (string): The value of "deviceID" obtained from the response to the OpenConnection request |
| Authorisation | Bearer Token |
| Body | HopperOptionsRequest: The request body containing the hopper options Reg0 and Reg1. <ul style="list-style-type: none">byte reg_0: The value to set for the first hopper register.byte reg_1: The value to set for the second hopper register. |

Reg_0 bits and their meaning

| Bit | Parameter | Function |
|-----|--------------------|--|
| 0 | Pay Mode | 0x00 = Split by highest value (Default). The device will attempt to payout a requested value by starting from the highest to the lowest coins available. This mode will payout the minimum number of coins possible. 0x01 = Free pay. The device will payout a coin as it passes its discriminator system if it fits into the current payout value and will leave enough of other coins to payout the rest of the value. This may give a faster payout but could result in a large number of coins of small denominations paid out. |
| 1 | Level Check | 0x00 = Disabled. The device will not refer to the level counters when calculating if a payout value can be made. 0x01 = Enabled (Default). The device will check the level counters and accept or refuse a payout request based on levels and/or split of available levels. |
| 2 | Motor Speed | 0x00 = Low speed. Payouts run at a lower motor speed. 0x01 = High Speed (default). The motors run at max speed for payouts. |
| 3 | Cashbox Pay Active | This bit is used in conjunction with Bit 0. If bit 3 is zero, then the Pay modes will be as described in bit 0. If Bit 3 is set then coins routed to the cashbox will be used in coins paid out of the front if they can fit into the current payout request. This is shown below. Pay Mode Type Bit 0 Bit 3 Free Play 1 0 Highest Split 0 1 All Route Free Pay 1 1 All Route Highest Split 0 1 |

| Reg_0 bits and their meaning | | |
|------------------------------|--------------------------------|---|
| Bit | Parameter | Function |
| 4 | Route 0 level coins to cashbox | Set to 0x01 means that any coins detected with a level setting of 0 will be paid to the cashbox, even if it is routed to the payout. |
| 5 | High Efficiency Split | Default set to 0x01 to enable a more efficient, smarter coin payout algorithm which will tend to use coins which have higher level counts - thus speeding up the payout process. |
| 6 | Unknown to Payout | Set to 0x01 means any unknown coins will be paid out during Smart Empty (otherwise they will be routed to cashbox). |
| 7 | Value Added | 0x00 = Coin added event 0x01 = Value added event |
| Reg_1 bits and their meaning | | |
| Bit | Parameter | Function |
| 0 | Reject Events | Set to 1 gives reject event 0xBA Coin Rejected. |
| 1 | Reject Events Full | Set to 1 gives reject event 0xBA with coin value if known. |
| 2 | Empty Route | Set to 1 will route coins to payout when a Empty or Smart Empty command is received. |
| 3 | Full to Cashbox | Set to 1 will start moving coins to cashbox during payouts when full. |
| 4 | Value Coin | Set to 1 gives individual coin payout events. |
| 5 | N/A | Set to 0. |
| 6 | N/A | Set to 0. |
| 7 | N/A | Set to 0. |

Responses

| Status | Body | Notes |
|--------|----------------------------------|---|
| 200 | Hopper options set successfully. | OK: If the hopper options were set successfully. |
| 400 | | Bad Request: If there is an error setting the hopper options. |

Code Examples

C#

```
var options = new RestClientOptions("http://localhost:5000")
{
    MaxTimeout = -1,
};

var client = new RestClient(options);
var request = new RestRequest("/api/CashDevice/SetHopperOptions?
deviceID=SMART_COIN_SYSTEM-COM10", Method.Post);
request.AddHeader("Content-Type", "application/json");
request.AddHeader("Authorization", "Bearer eyJhb.....");

var body = @ "{" + "\n" +
@"    ""Reg0"": 4," + "\n" +
@"    ""Reg1"": 0" + "\n" +
@"}";
request.AddStringBody(body, DataFormat.Json);
RestResponse response = await client.ExecuteAsync(request);
Console.WriteLine(response.Content);
```

NodeJS

```
var request = require('request');
var options = {
    'method': 'POST',
    'url': 'http://localhost:5000/api/CashDevice/SetHopperOptions?
deviceID=SMART_COIN_SYSTEM-COM10',
    'headers': {
        'Content-Type': 'application/json',
        'Authorization': 'Bearer eyJhb.....'
    },
    body: JSON.stringify({
        "Reg0": 4,
        "Reg1": 0
    })
};

request(options, function (error, response) {
    if (error) throw new Error(error);
    console.log(response.body);
});
```

Python

```
import requests
import json

url = "http://localhost:5000/api/CashDevice/SetHopperOptions?
deviceID=SMART_COIN_SYSTEM-COM10"

payload = json.dumps({
    "Reg0": 4,
    "Reg1": 0
})
headers = {
    'Content-Type': 'application/json',
```

```

    'Authorization': 'Bearer eyJhb.....'
}

response = requests.request("POST", url, headers=headers, data=payload)

print(response.text)

```

Java

```

OkHttpClient client = new OkHttpClient().newBuilder()
    .build();
MediaType mediaType = MediaType.parse("application/json");
RequestBody body = RequestBody.create(mediaType, "{\"r\n  \"Reg0\": 4,\n  \"Reg1\":\n  0\r\n}"); 
Request request = new Request.Builder()
    .url("http://localhost:5000/api/CashDevice/SetHopperOptions?
deviceID=SMART_COIN_SYSTEM-COM10")
    .method("POST", body)
    .addHeader("Content-Type", "application/json")
    .addHeader("Authorization", "Bearer eyJhb.....")
    .build();
Response response = client.newCall(request).execute();

```

REST API - GetGlobalErrorCode

Description

Sets the globalErrorCode_0 and globalErrorCode_1 to the error code in the cash device

| | |
|----------------------|---|
| Endpoint | GetGlobalErrorCode |
| Method | GET |
| URL | {server_url}/api/CashDevice/GetGlobalErrorCode |
| Parameters | <ul style="list-style-type: none">deviceID (string): The value of "deviceID" obtained from the response to the OpenConnection request |
| Authorisation | Bearer Token |
| Body | None |

Responses

Please refer to Global Error State for detailed information on responses.

| Status | Body | Notes |
|--------|---|---|
| 200 | SMART_COIN_SYSTEM-COM10: { ErrorCode0 = 00, ErrorCode1 = 00 } | OK: If the global error code was retrieved successfully. |
| 400 | | Bad Request: If there is an error retrieving the global error code. |

Code Examples

C#

```
var options = new RestClientOptions("http://localhost:5000")
{
    MaxTimeout = -1,
};
var client = new RestClient(options);
var request = new RestRequest("/api/CashDevice/GetGlobalErrorCode?
deviceID=SMART_COIN_SYSTEM-COM10", Method.Get);
request.AddHeader("Authorization", "Bearer eyJhb.....");
RestResponse response = await client.ExecuteAsync(request);
Console.WriteLine(response.Content);
```

NodeJS

```
var request = require('request');
var options = {
  'method': 'GET',
  'url': 'http://localhost:5000/api/CashDevice/GetGlobalErrorCode?
deviceID=SMART_COIN_SYSTEM-COM10',
  'headers': {
    'Authorization': 'Bearer eyJhb.....'
  }
};
request(options, function (error, response) {
  if (error) throw new Error(error);
  console.log(response.body);
});
```

Python

```
import requests

url = "http://localhost:5000/api/CashDevice/GetGlobalErrorCode?
deviceID=SMART_COIN_SYSTEM-COM10"

payload = {}
headers = {
  'Authorization': 'Bearer eyJhb.....'
}

response = requests.request("GET", url, headers=headers, data=payload)

print(response.text)
```

Java

```
OkHttpClient client = new OkHttpClient().newBuilder()
  .build();
MediaType mediaType = MediaType.parse("text/plain");
RequestBody body = RequestBody.create(mediaType, "");
Request request = new Request.Builder()
  .url("http://localhost:5000/api/CashDevice/GetGlobalErrorCode?
deviceID=SMART_COIN_SYSTEM-COM10")
  .method("GET", body)
  .addHeader("Authorization", "Bearer eyJhb.....")
  .build();
Response response = client.newCall(request).execute();
```

REST API - GetServiceInformation

Description

- Retrieves service information from the device based on the provided sub-command and formats the information into a readable string.
- The extracted information is formatted into a readable string and stored in the variable `service_information_string`

| | |
|----------------------|---|
| Endpoint | GetServiceInformation |
| Method | GET |
| URL | <code>{server_url}/api/CashDevice/GetServiceInformation</code> |
| Parameters | <ul style="list-style-type: none">• <code>deviceID</code> (string): The value of "deviceID" obtained from the response to the OpenConnection request |
| Authorisation | Bearer Token |
| Body | <ul style="list-style-type: none">• <code>byte subCommand</code>: The sub-command to specify the type of service information to retrieve. Possible values:<ul style="list-style-type: none">• Byte: 0x00 - Service Type: Date Information - Description: Returns the current raw byte date• Byte: 0x01 - Service Type: Note Information - Description: Extracts the number of notes since various events such as maintenance reset, last download, power on, and last jam.• Byte: 0x02 - Service Type: Accept Note Information - Description: Extracts the number of accepted notes since maintenance reset, last download, and power on.• Byte: 0x03 - Service Type: Jam Information - Description: Extracts the number of jams since maintenance reset, last download, and in the last 1000 notes.• Byte: 0x04 - Service Type: Service Flags - Description: Returns current service flag status |

Responses

| Status | Body | Notes |
|--------|---|--|
| 200 | <pre>NV4000-COM5: { ServiceInformation = (Note Information) Notes Since Maintenance Reset: 11902 Notes Since Last Download: 461 Notes Since Power On: 2 Notes Since Last Jam: 2 }</pre> | OK: If the service information is retrieved and formatted. |
| 400 | | Bad Request: If fails to retrieve the service information. |

Code Examples

C#

```
var options = new RestClientOptions("http://localhost:5000")
{
    MaxTimeout = -1,
};

var client = new RestClient(options);
var request = new RestRequest("/api/CashDevice/GetServiceInformation?deviceID=NV4000-
COM5", Method.Get);
request.AddHeader("Content-Type", "application/json");
request.AddHeader("Authorization", "Bearer eyJhb.....");

var body = @"1" + "\n" +
@"" + "\n" +
@"";

request.AddStringBody(body, DataFormat.Json);
RestResponse response = await client.ExecuteAsync(request);
Console.WriteLine(response.Content);
```

NodeJS

```
var request = require('request');

var options = {
    'method': 'GET',
    'url': 'http://localhost:5000/api/CashDevice/GetServiceInformation?deviceID=NV4000-
COM5',
    'headers': {
        'Content-Type': 'application/json',
        'Authorization': 'Bearer eyJhb.....'
    },
    body: JSON.stringify(1)
};

request(options, function (error, response) {
    if (error) throw new Error(error);
    console.log(response.body);
});
```

Python

```
import requests
import json

url = "http://localhost:5000/api/CashDevice/GetServiceInformation?deviceID=NV4000-
COM5"

payload = json.dumps(1)
headers = {
    'Content-Type': 'application/json',
    'Authorization': 'Bearer eyJhb.....'
}

response = requests.request("GET", url, headers=headers, data=payload)

print(response.text)
```

Java

```
OkHttpClient client = new OkHttpClient().newBuilder()
    .build();
MediaType mediaType = MediaType.parse("application/json");
RequestBody body = RequestBody.create(mediaType, "1\r\n\r\n");
Request request = new Request.Builder()
    .url("http://localhost:5000/api/CashDevice.GetServiceInformation?deviceID=NV4000-
COM5")
    .method("GET", body)
    .addHeader("Content-Type", "application/json")
    .addHeader("Authorization", "Bearer eyJhb.....")
    .build();
Response response = client.newCall(request).execute();
```

REST API - GetServiceInformationForModule

Description

- Retrieves service information from the device based on the provided module and sub-command, and formats the information into a readable string.
- The extracted information is formatted into a readable string and stored in the variable `service_information_string`

| | |
|----------------------|--|
| Endpoint | GetServiceInformationForModule |
| Method | GET |
| URL | <code>{server_url}/api/CashDevice/GetServiceInformationForModule</code> |
| Parameters | <ul style="list-style-type: none">• <code>deviceID</code> (string): The value of "deviceID" obtained from the response to the OpenConnection request |
| Authorisation | Bearer Token |
| Body | <ul style="list-style-type: none">• <code>byte module</code>: The module identifier specifying the type of module to retrieve information from.<ul style="list-style-type: none">• 0x05: NV4000 Lifetime Counts.• 0x00, 0x01, 0x02, 0x03: Smart Coin System modules including Primary Hopper, Feeder, Secondary Hopper, and Lifter.• <code>byte subCommand</code>: The sub-command to specify the type of service information to retrieve. Possible values:<ul style="list-style-type: none">• Byte: 0x00 - Service Type: Service Status - Description: Returns the service status of the device• Byte: 0x01 - Service Type: Next Service Due - Description: Returns the next service due. Should be sent for each module• Byte: 0x02 - Service Type: Last Service Information - Description: Returns the last 3 services, 3 bytes per service, 9 bytes in total• Byte: 0x03 - Service Type: Performance Since Last Service - Description: Extracts the number of coins processed, acceptance percentage, jams, and calibration failures since the last service• Byte: 0x04 - Service Type: Service Flags - Description: Returns current service flags• Byte: 0x05 - Service Type: Lifetime Counts - Description: Get the lifetime counts of the module |

```
{  
    "Module": 5,  
    "SubCommand": 0  
}
```

Responses

| Status | Body | Notes |
|--------|---|--|
| 200 | <pre>NV4000-COM5: { ServiceInformation = (Lifetime Counts) Total cycles: 32717 Power on count: 157 Power on time: 20580 }</pre> | OK: If the service information is retrieved and formatted. |
| 400 | | Bad Request: If fails to retrieve the service information. |

Code Examples

C#

```
var options = new RestClientOptions("http://localhost:5000")
{
    MaxTimeout = -1,
};
var client = new RestClient(options);
var request = new RestRequest("/api/CashDevice.GetServiceInformationForModule?
deviceID=NV4000-COM5", Method.Get);
request.AddHeader("Content-Type", "application/json");
request.AddHeader("Authorization", "Bearer eyJhb....");
var body = @"{"
    + "\n" +
@""    ""Module"": 5," + "\n" +
@""    ""SubCommand"": 0" + "\n" +
@"}" + "\n" +
@"";
request.AddStringBody(body, DataFormat.Json);
RestResponse response = await client.ExecuteAsync(request);
Console.WriteLine(response.Content);
```

NodeJS

```
var request = require('request');
var options = {
    'method': 'GET',
    'url': 'http://localhost:5000/api/CashDevice.GetServiceInformationForModule?
deviceID=NV4000-COM5',
    'headers': {
        'Content-Type': 'application/json',
        'Authorization': 'Bearer eyJhb....'
    },
    body: JSON.stringify({
        "Module": 5,
        "SubCommand": 0
    })
};

request(options, function (error, response) {
```

```

if (error) throw new Error(error);
console.log(response.body);
});

```

Python

```

import requests
import json

url = "http://localhost:5000/api/CashDevice/GetServiceInformationForModule?
deviceID=NV4000-COM5"

payload = json.dumps({
    "Module": 5,
    "SubCommand": 0
})
headers = {
    'Content-Type': 'application/json',
    'Authorization': 'Bearer eyJhb.....'
}

response = requests.request("GET", url, headers=headers, data=payload)

print(response.text)

```

Java

```

OkHttpClient client = new OkHttpClient().newBuilder()
    .build();
MediaType mediaType = MediaType.parse("application/json");
RequestBody body = RequestBody.create(mediaType, "{\r\n      \"Module\": 5,\r\n      \"SubCommand\": 0\r\n}\r\n");
Request request = new Request.Builder()
    .url("http://localhost:5000/api/CashDevice/GetServiceInformationForModule?
deviceID=NV4000-COM5")
    .method("GET", body)
    .addHeader("Content-Type", "application/json")
    .addHeader("Authorization", "Bearer eyJhb.....")
    .build();
Response response = client.newCall(request).execute();

```

REST API - SetServiceInformationMaintenanceReset

Description

Sets the maintenance reset information for the device using the provided ASCII bytes representing the week and year of the maintenance reset.

| | |
|----------------------|--|
| Endpoint | SetServiceInformationMaintenanceReset |
| Method | POST |
| URL | {server_url}/api/CashDevice/SetServiceInformationMaintenanceReset |
| Parameters | <ul style="list-style-type: none">• deviceId (string): The value of "deviceId" obtained from the response to the OpenConnection request |
| Authorisation | Bearer Token |
| Body | <ul style="list-style-type: none">• ServiceInformationMaintenanceResetRequest: The request body containing WeekNumber1AsciiByte, WeekNumber2AsciiByte, YearNumber1AsciiByte and YearNumber2AsciiByte.<ul style="list-style-type: none">• byte weekNumber1_ascii_byte: The module identifier specifying the type of module to set the service information for.• byte weekNumber2_ascii_byte: The type of service being set.• byte yearNumber1_ascii_byte: The month of the service date.• byte yearNumber2_ascii_byte: The year of the service date. <div style="border: 1px solid #ccc; padding: 10px; margin-top: 10px;"><pre>{ "WeekNumber1AsciiByte": 0, "WeekNumber2AsciiByte": 0, "YearNumber1AsciiByte": 0, "YearNumber2AsciiByte": 0 }</pre></div> |

Responses

| Status | Body | Notes |
|--------|--|--|
| 200 | NV4000-COM5: Maintenance reset information set successfully. | OK: If the maintenance reset information was set successfully. |
| 400 | | Bad Request: If there is an error setting the maintenance reset information. |

Code Examples

C#

```
var options = new RestClientOptions("http://localhost:5000")
{
    MaxTimeout = -1,
};

var client = new RestClient(options);
var request = new RestRequest("/api/CashDevice/SetServiceInformationMaintenanceReset?
deviceID=NV4000-COM5", Method.Post);
request.AddHeader("Content-Type", "application/json");
request.AddHeader("Authorization", "Bearer eyJhb.....");

var body = @"{" + "\n" +
@"" ""WeekNumber1AsciiByte"": 0," + "\n" +
@"" ""WeekNumber2AsciiByte"": 0," + "\n" +
@"" ""YearNumber1AsciiByte"": 0," + "\n" +
@"" ""YearNumber2AsciiByte"": 0" + "\n" +
@"}";
request.AddStringBody(body, DataFormat.Json);
RestResponse response = await client.ExecuteAsync(request);
Console.WriteLine(response.Content);
```

NodeJS

```
var request = require('request');
var options = {
    'method': 'POST',
    'url': 'http://localhost:5000/api/CashDevice/SetServiceInformationMaintenanceReset?
deviceID=NV4000-COM5',
    'headers': {
        'Content-Type': 'application/json',
        'Authorization': 'Bearer eyJhb.....'
    },
    body: JSON.stringify({
        "WeekNumber1AsciiByte": 0,
        "WeekNumber2AsciiByte": 0,
        "YearNumber1AsciiByte": 0,
        "YearNumber2AsciiByte": 0
    })
};

request(options, function (error, response) {
    if (error) throw new Error(error);
    console.log(response.body);
});
```

Python

```
import requests
import json

url = "http://localhost:5000/api/CashDevice/SetServiceInformationMaintenanceReset?
deviceID=NV4000-COM5"

payload = json.dumps({
    "WeekNumber1AsciiByte": 0,
```

```

    "WeekNumber2AsciiByte": 0,
    "YearNumber1AsciiByte": 0,
    "YearNumber2AsciiByte": 0
})
headers = {
    'Content-Type': 'application/json',
    'Authorization': 'Bearer eyJhb.....'
}

response = requests.request("POST", url, headers=headers, data=payload)

print(response.text)

```

Java

```

OkHttpClient client = new OkHttpClient().newBuilder()
    .build();
MediaType mediaType = MediaType.parse("application/json");
RequestBody body = RequestBody.create(mediaType, "{\"r\n  \"WeekNumber1AsciiByte\":\n0,\r\n  \"WeekNumber2AsciiByte\": 0,\r\n  \"YearNumber1AsciiByte\": 0,\r\n  \"YearNumber2AsciiByte\": 0\r\n}");
Request request = new Request.Builder()
    .url("http://localhost:5000/api/CashDevice/SetServiceInformationMaintenanceReset?
deviceID=NV4000-COM5")
    .method("POST", body)
    .addHeader("Content-Type", "application/json")
    .addHeader("Authorization", "Bearer eyJhb.....")
    .build();
Response response = client.newCall(request).execute();

```

REST API - SetNoPayinCount

Description

Sets the no-payin count for the device, which specifies the number of transactions allowed without pay-ins.

| | |
|----------------------|---|
| Endpoint | SetNoPayinCount |
| Method | POST |
| URL | {server_url}/api/CashDevice/SetNoPayinCount |
| Parameters | <ul style="list-style-type: none">deviceID (string): The value of "deviceID" obtained from the response to the OpenConnection request |
| Authorisation | Bearer Token |
| Body | <ul style="list-style-type: none">byte count: The number of transactions allowed without pay-ins. |

Responses

| Status | Body | Notes |
|--------|---|---|
| 200 | SMART_COIN_SYSTEM-COM10: { NoPayinCount = 0 } | OK: If the no-payin count was set successfully. |
| 400 | | Bad Request: If there is an error setting the no-payin count. |

Code Examples

C#

```
var options = new RestClientOptions("http://localhost:5000")
{
    MaxTimeout = -1,
};

var client = new RestClient(options);
var request = new RestRequest("/api/CashDevice/SetNoPayinCount?
deviceID=SMART_COIN_SYSTEM-COM10", Method.Post);
request.AddHeader("Content-Type", "application/json");
request.AddHeader("Authorization", "Bearer eyJhb.....");
var body = @"0";
request.AddStringBody(body, DataFormat.Json);
RestResponse response = await client.ExecuteAsync(request);
Console.WriteLine(response.Content);
```

NodeJS

```
var request = require('request');
var options = {
  'method': 'POST',
  'url': 'http://localhost:5000/api/CashDevice/SetNoPayinCount?',
  deviceID=SMART_COIN_SYSTEM-COM10',
  'headers': {
    'Content-Type': 'application/json',
    'Authorization': 'Bearer eyJhb.....'
  },
  body: JSON.stringify(0)

};

request(options, function (error, response) {
  if (error) throw new Error(error);
  console.log(response.body);
});
```

Python

```
import requests
import json

url = "http://localhost:5000/api/CashDevice/SetNoPayinCount?
deviceID=SMART_COIN_SYSTEM-COM10"

payload = json.dumps(0)
headers = {
  'Content-Type': 'application/json',
  'Authorization': 'Bearer eyJhb.....'
}

response = requests.request("POST", url, headers=headers, data=payload)

print(response.text)
```

Java

```
OkHttpClient client = new OkHttpClient().newBuilder()
  .build();
MediaType mediaType = MediaType.parse("application/json");
RequestBody body = RequestBody.create(mediaType, "0");
Request request = new Request.Builder()
  .url("http://localhost:5000/api/CashDevice/SetNoPayinCount?
deviceID=SMART_COIN_SYSTEM-COM10")
  .method("POST", body)
  .addHeader("Content-Type", "application/json")
  .addHeader("Authorization", "Bearer eyJhb.....")
  .build();
Response response = client.newCall(request).execute();
```

REST API - GetCoinAcceptance

Description

Retrieves the coin acceptance status from the specified device and formats the data into a readable string

| | |
|----------------------|---|
| Endpoint | GetCoinAcceptance |
| Method | GET |
| URL | {server_url}/api/CashDevice/GetCoinAcceptance |
| Parameters | <ul style="list-style-type: none">• deviceID (string): The value of "deviceID" obtained from the response to the OpenConnection request |
| Authorisation | Bearer Token |
| Body | <ul style="list-style-type: none">• byte device: The byte representing the specific device component from which to retrieve the coin acceptance status |

Device

| Byte | Function | Size |
|------|--|------|
| 0 | Primary Hopper acceptance percentage | 1 |
| 1 | Feeder acceptance percentage | 1 |
| 2 | Secondary Hopper acceptance percentage | |

Responses

| Status | Body | Notes |
|--------|---|--|
| 200 | SMART_COIN_SYSTEM-COM10: { CoinAcceptanceData = 64 } | OK: If the coin acceptance data was retrieved successfully. |
| 400 | Failed to retrieve coin acceptance data. | Bad Request: If there is an error retrieving the coin acceptance data. |

Code Examples

C#

```
var options = new RestClientOptions("http://localhost:5000")
```

```

{
    MaxTimeout = -1,
};

var client = new RestClient(options);
var request = new RestRequest("/api/CashDevice/GetCoinAcceptance?
deviceID=SMART_COIN_SYSTEM-COM10", Method.Get);
request.AddHeader("Content-Type", "application/json");
request.AddHeader("Authorization", "Bearer eyJhb.....");
var body = @" 0";
request.AddStringBody(body, DataFormat.Json);
RestResponse response = await client.ExecuteAsync(request);
Console.WriteLine(response.Content);

```

NodeJS

```

var request = require('request');
var options = {
  'method': 'GET',
  'url': 'http://localhost:5000/api/CashDevice/GetCoinAcceptance?
deviceID=SMART_COIN_SYSTEM-COM10',
  'headers': {
    'Content-Type': 'application/json',
    'Authorization': 'Bearer eyJhb.....'
  },
  body: JSON.stringify(0)
};

request(options, function (error, response) {
  if (error) throw new Error(error);
  console.log(response.body);
});

```

Python

```

import requests
import json

url = "http://localhost:5000/api/CashDevice/GetCoinAcceptance?
deviceID=SMART_COIN_SYSTEM-COM10"

payload = json.dumps(0)
headers = {
  'Content-Type': 'application/json',
  'Authorization': 'Bearer eyJhb.....'
}

response = requests.request("GET", url, headers=headers, data=payload)

print(response.text)

```

Java

```

OkHttpClient client = new OkHttpClient().newBuilder()
  .build();
MediaType mediaType = MediaType.parse("application/json");
RequestBody body = RequestBody.create(mediaType, " 0");

```

```
Request request = new Request.Builder()
    .url("http://localhost:5000/api/CashDevice/GetCoinAcceptance?
deviceID=SMART_COIN_SYSTEM-COM10")
    .method("GET", body)
    .addHeader("Content-Type", "application/json")
    .addHeader("Authorization", "Bearer eyJhb.....")
    .build();
Response response = client.newCall(request).execute();
```

REST API - SetCashboxLevels

Description

Sets the coin levels of the cashbox

| | |
|----------------------|--|
| Endpoint | SetCashboxLevels |
| Method | POST |
| URL | {server_url}/api/CashDevice/SetCashboxLevels |
| Parameters | <ul style="list-style-type: none">• deviceID (string): The value of "deviceID" obtained from the response to the OpenConnection request |
| Authorisation | Bearer Token |
| Body | <p>SetCashboxLevelsRequest: The request body containing NumCoinsToAdd, Denomination and CountryCode.</p> <ul style="list-style-type: none">• ushort NumCoinsToAdd: Number of coins to add to level (0 will clear the level)• uint Denomination: Value of denomination to set• string CountryCode: ASCII country code of denomination <div style="border: 1px solid #ccc; padding: 10px; margin-top: 10px;"><pre>{ "NumCoinsToAdd": 10, "Denomination": 10, "CountryCode": "{{Currency}}" }</pre></div> |

Responses

| Status | Body | Notes |
|--------|---|---|
| 200 | SMART_COIN_SYSTEM-COM10: Cashbox levels set successfully. | OK: If the cashbox levels were set successfully. |
| 400 | | Bad Request: If there is an error setting the cashbox levels. |

Code Examples

C#

```
var options = new RestClientOptions("http://localhost:5000")
{
    MaxTimeout = -1,
};
var client = new RestClient(options);
```

```

var request = new RestRequest("/api/CashDevice/SetCashboxLevels?
deviceID=SMART_COIN_SYSTEM-COM10", Method.Post);
request.AddHeader("Content-Type", "application/json");
request.AddHeader("Authorization", "Bearer eyJhb.....");
var body = @ "{" + "\n" +
@"    ""numCoinsToAdd"": 10," + "\n" +
@"    ""denomination"": 10," + "\n" +
@"    ""countryCode"": ""ITL"""+ "\n" +
@"}" + "\n" +
@"";
request.AddStringBody(body, DataFormat.Json);
RestResponse response = await client.ExecuteAsync(request);
Console.WriteLine(response.Content);

```

NodeJS

```

var request = require('request');
var options = {
  'method': 'POST',
  'url': 'http://localhost:5000/api/CashDevice/SetCashboxLevels?
deviceID=SMART_COIN_SYSTEM-COM10',
  'headers': {
    'Content-Type': 'application/json',
    'Authorization': 'Bearer eyJhb.....'
  },
  body: JSON.stringify({
    "numCoinsToAdd": 10,
    "denomination": 10,
    "countryCode": "ITL"
  })
};

request(options, function (error, response) {
  if (error) throw new Error(error);
  console.log(response.body);
});

```

Python

```

import requests
import json

url = "http://localhost:5000/api/CashDevice/SetCashboxLevels?
deviceID=SMART_COIN_SYSTEM-COM10"

payload = json.dumps({
    "numCoinsToAdd": 10,
    "denomination": 10,
    "countryCode": "ITL"
})
headers = {
    'Content-Type': 'application/json',
    'Authorization': 'Bearer eyJhb.....'
}

response = requests.request("POST", url, headers=headers, data=payload)

```

```
print(response.text)
```

Java

```
OkHttpClient client = new OkHttpClient().newBuilder()
    .build();
MediaType mediaType = MediaType.parse("application/json");
RequestBody body = RequestBody.create(mediaType, "{\"numCoinsToAdd\": 10,\r\n\"denomination\": 10,\r\n\"countryCode\": \"ITL\"\r\n}\r\n");
Request request = new Request.Builder()
    .url("http://localhost:5000/api/CashDevice/SetCashboxLevels?
deviceID=SMART_COIN_SYSTEM-COM10")
    .method("POST", body)
    .addHeader("Content-Type", "application/json")
    .addHeader("Authorization", "Bearer eyJhb.....")
    .build();
Response response = client.newCall(request).execute();
```

REST API - ClearCashboxLevels

Description

Clears the cashbox levels on the SSP device.

| | |
|----------------------|---|
| Endpoint | ClearCashboxLevels |
| Method | POST |
| URL | {server_url}/api/CashDevice/ClearCashboxLevels |
| Parameters | <ul style="list-style-type: none">deviceID (string): The value of "deviceID" obtained from the response to the OpenConnection request |
| Authorisation | Bearer Token |
| Body | None |

Responses

| Status | Body | Notes |
|--------|---|--|
| 200 | SMART_COIN_SYSTEM-COM10: Cashbox levels cleared successfully. | OK: If the cashbox levels were cleared successfully. |
| 400 | | Bad Request: If there is an error clearing the cashbox levels. |

Code Examples

C#

```
var options = new RestClientOptions("http://localhost:5000")
{
    MaxTimeout = -1,
};
var client = new RestClient(options);
var request = new RestRequest("/api/CashDevice/ClearCashboxLevels?
deviceID=SMART_COIN_SYSTEM-COM10", Method.Post);
request.AddHeader("Authorization", "Bearer eyJhb.....");
RestResponse response = await client.ExecuteAsync(request);
Console.WriteLine(response.Content);
```

NodeJS

```
var request = require('request');
var options = {
```

```

'method': 'POST',
'url': 'http://localhost:5000/api/CashDevice/ClearCashboxLevels?
deviceID=SMART_COIN_SYSTEM-COM10',
'headers': {
    'Authorization': 'Bearer eyJhb.....'
}
};

request(options, function (error, response) {
    if (error) throw new Error(error);
    console.log(response.body);
});

```

Python

```

import requests

url = "http://localhost:5000/api/CashDevice/ClearCashboxLevels?
deviceID=SMART_COIN_SYSTEM-COM10"

payload = {}
headers = {
    'Authorization': 'Bearer eyJhb.....'
}

response = requests.request("POST", url, headers=headers, data=payload)

print(response.text)

```

Java

```

OkHttpClient client = new OkHttpClient().newBuilder()
    .build();
MediaType mediaType = MediaType.parse("text/plain");
RequestBody body = RequestBody.create(mediaType, "");
Request request = new Request.Builder()
    .url("http://localhost:5000/api/CashDevice/ClearCashboxLevels?
deviceID=SMART_COIN_SYSTEM-COM10")
    .method("POST", body)
    .addHeader("Authorization", "Bearer eyJhb.....")
    .build();
Response response = client.newCall(request).execute();

```

REST API - GetCashboxLevels

Description

- Retrieves the coins that have been flushed down to the cashbox.
- Reports in the same format as [GetAllLevels\(\)](#)

| | |
|----------------------|---|
| Endpoint | GetCashboxLevels |
| Method | GET |
| URL | {server_url}/api/CashDevice/GetCashboxLevels |
| Parameters | <ul style="list-style-type: none">• deviceID (string): The value of "deviceID" obtained from the response to the OpenConnection request |
| Authorisation | Bearer Token |
| Body | None |

Responses

| Status | Body | Notes |
|--------|--|--|
| 200 | <pre>SMART_COIN_SYSTEM-COM10: { CashboxLevels = Number of denominations in device: 2 10 x 10 ITL 0 x 50 ITL Quantity of unknown coins: 0 }</pre> | OK: If the cashbox levels were retrieved successfully. |
| 400 | | Bad Request: If there is an error retrieving the cashbox levels. |

Code Examples

C#

```
var options = new RestClientOptions("http://localhost:5000")
{
    MaxTimeout = -1,
};
var client = new RestClient(options);
var request = new RestRequest("/api/CashDevice/GetCashboxLevels?
deviceID=SMART_COIN_SYSTEM-COM10", Method.Get);
request.AddHeader("Authorization", "Bearer eyJhb.....");
RestResponse response = await client.ExecuteAsync(request);
Console.WriteLine(response.Content);
```

NodeJS

```
var request = require('request');
var options = {
  'method': 'GET',
  'url': 'http://localhost:5000/api/CashDevice/GetCashboxLevels?
deviceID=SMART_COIN_SYSTEM-COM10',
  'headers': {
    'Authorization': 'Bearer eyJhb.....'
  }
};
request(options, function (error, response) {
  if (error) throw new Error(error);
  console.log(response.body);
});
```

Python

```
import requests

url = "http://localhost:5000/api/CashDevice/GetCashboxLevels?
deviceID=SMART_COIN_SYSTEM-COM10"

payload = {}
headers = {
  'Authorization': 'Bearer eyJhb.....'
}

response = requests.request("GET", url, headers=headers, data=payload)

print(response.text)
```

Java

```
OkHttpClient client = new OkHttpClient().newBuilder()
  .build();
MediaType mediaType = MediaType.parse("text/plain");
RequestBody body = RequestBody.create(mediaType, "");
Request request = new Request.Builder()
  .url("http://localhost:5000/api/CashDevice/GetCashboxLevels?
deviceID=SMART_COIN_SYSTEM-COM10")
  .method("GET", body)
  .addHeader("Authorization", "Bearer eyJhb.....")
  .build();
Response response = client.newCall(request).execute();
```

REST API - GetNoteCashboxLevels

Description

Retrieves the notes that have been stored the cashbox.

Levels can be cleared with [ClearNoteCashboxLevels](#) when the cashbox is emptied, or can auto clear when the cashbox is removed and replaced by setting the option in the config file: serverconfig.json.

| | |
|----------------------|---|
| Endpoint | GetNoteCashboxLevels |
| Method | GET |
| URL | {server_url}/api/CashDevice/GetNoteCashboxLevels |
| Parameters | <ul style="list-style-type: none">deviceID (string): The value of "deviceID" obtained from the response to the OpenConnection request |
| Authorisation | Bearer Token |
| Body | None |

Responses

| Status | Body | Notes |
|--------|--|--|
| 200 | <pre>[{ "value": 500, "countryCode": "GBP", "storedInCashbox": 0 }, { "value": 1000, "countryCode": "GBP", "storedInCashbox": 2 }, { "value": 2000, "countryCode": "GBP", "storedInCashbox": 5 }, { "value": 5000, "countryCode": "GBP", "storedInCashbox": 5 }]</pre> | OK: If the cashbox levels were retrieved successfully. |
| 400 | | Bad Request: If there is an error retrieving the cashbox levels. |

Code Examples

C#

```
var client = new HttpClient();
var request = new HttpRequestMessage(HttpMethod.Get, "http://localhost:5000/api/CashDevice/GetNoteCashboxLevels?deviceID=SPECTRAL_PAYOUT-COM5");
request.Headers.Add("Authorization", "Bearer eyJhb.....");
var response = await client.SendAsync(request);
response.EnsureSuccessStatusCode();
Console.WriteLine(await response.Content.ReadAsStringAsync());
```

NodeJS

```
var request = require('request');
var options = {
  'method': 'GET',
  'url': 'http://localhost:5000/api/CashDevice/GetNoteCashboxLevels?deviceID=SPECTRAL_PAYOUT-COM5',
  'headers': {
    'Authorization': 'Bearer eyJhb.....'
  }
};
request(options, function (error, response) {
  if (error) throw new Error(error);
  console.log(response.body);
});
```

Python

```
import requests

url = "http://localhost:5000/api/CashDevice/GetNoteCashboxLevels?deviceID=SPECTRAL_PAYOUT-COM5"

payload = {}
headers = {
  'Authorization': 'Bearer eyJhb.....'
}

response = requests.request("GET", url, headers=headers, data=payload)

print(response.text)
```

Java

```
OkHttpClient client = new OkHttpClient().newBuilder()
  .build();
MediaType mediaType = MediaType.parse("text/plain");
RequestBody body = RequestBody.create(mediaType, "");
Request request = new Request.Builder()
  .url("http://localhost:5000/api/CashDevice/GetNoteCashboxLevels?deviceID=SPECTRAL_PAYOUT-COM5")
  .method("GET", body)
  .addHeader("Authorization", "Bearer eyJhb.....")
```

```
.build();
Response response = client.newCall(request).execute();
```

REST API - ClearNoteCashboxLevels

Description

Clears the cashbox levels on the SSP device.

| | |
|----------------------|---|
| Endpoint | ClearNoteCashboxLevels |
| Method | POST |
| URL | {server_url}/api/CashDevice/ClearNoteCashboxLevels |
| Parameters | <ul style="list-style-type: none">• deviceID (string): The value of "deviceID" obtained from the response to the OpenConnection request |
| Authorisation | Bearer Token |
| Body | None |

Responses

| Status | Body | Notes |
|--------|---|--|
| 200 | SPECTRAL_PAYOUT-COM5: Cashbox levels cleared. | OK: If the cashbox levels were cleared successfully. |
| 400 | | Bad Request: If there is an error clearing the cashbox levels. |

Code Examples

C#

```
var client = new HttpClient();
var request = new HttpRequestMessage(HttpMethod.Post, "http://localhost:5000/api/
CashDevice/ClearNoteCashboxLevels?deviceID=SPECTRAL_PAYOUT-COM5");
request.Headers.Add("Authorization", "Bearer eyJhb.....");
var response = await client.SendAsync(request);
response.EnsureSuccessStatusCode();
Console.WriteLine(await response.Content.ReadAsStringAsync());
```

NodeJS

```
var request = require('request');
var options = {
  'method': 'POST',
  'url': 'http://localhost:5000/api/CashDevice/ClearNoteCashboxLevels?
deviceID=SPECTRAL_PAYOUT-COM5',
```

```

'headers': {
    'Authorization': 'Bearer eyJhb.....'
}
};

request(options, function (error, response) {
    if (error) throw new Error(error);
    console.log(response.body);
});

```

Python

```

import requests

url = "http://localhost:5000/api/CashDevice/ClearNoteCashboxLevels?
deviceID=SPECTRAL_PAYOUT-COM5"

payload = {}
headers = {
    'Authorization': 'Bearer eyJhb.....'
}

response = requests.request("POST", url, headers=headers, data=payload)

print(response.text)

```

Java

```

OkHttpClient client = new OkHttpClient().newBuilder()
    .build();
MediaType mediaType = MediaType.parse("text/plain");
RequestBody body = RequestBody.create(mediaType, "");
Request request = new Request.Builder()
    .url("http://localhost:5000/api/CashDevice/ClearNoteCashboxLevels?
deviceID=SPECTRAL_PAYOUT-COM5")
    .method("POST", body)
    .addHeader("Authorization", "Bearer eyJhb.....")
    .build();
Response response = client.newCall(request).execute();

```

REST API - SetSorterRoute

Description

Sets the route for the specified denomination to go to either PRIMARY or SECONDARY hopper of the Twin SMART Coin System

| | |
|----------------------|--|
| Endpoint | SetSorterRoute |
| Method | POST |
| URL | {server_url}/api/CashDevice/SetSorterRoute |
| Parameters | <ul style="list-style-type: none">• deviceId (string): The value of "deviceId" obtained from the response to the OpenConnection request |
| Authorisation | Bearer Token |
| Body | <p>SetSorterRouteRequest: The request body containing ValueCountryCode and SorterRoute.</p> <ul style="list-style-type: none">• ValueCountryCode valueCountryCode: is an object representing the denomination and its respective country code:<ul style="list-style-type: none">• UInt32 Value: Denomination value e.g 500, 1000 etc• string countryCode: The country code of the currency e.g. "GBP", "EUR", "USD" etc• byte sorterRoute: The sorter route to set for the specified denomination. Use 0x00 for PRIMARY hopper and 0x01 for SECONDARY hopper. <div style="border: 1px solid #ccc; padding: 10px; margin-top: 10px;"><pre>{ "Value": 100, "CountryCode": "{{Currency}}", "SorterRoute": 0 }</pre></div> |

Responses

| Status | Body | Notes |
|--------|--|---|
| 200 | TWIN_SMART_COIN_SYSTEM-COM10: Sorter route set successfully. | OK: If the sorter route was set successfully. |
| 400 | | Bad Request: If there is an error setting the sorter route. |

Code Examples

C#

```
var options = new RestClientOptions("http://localhost:5000")  
{
```

```

    MaxTimeout = -1,
};

var client = new RestClient(options);
var request = new RestRequest("/api/CashDevice/SetSorterRoute?
deviceID=TWIN_SMART_COIN_SYSTEM-COM10", Method.Post);
request.AddHeader("Content-Type", "application/json");
request.AddHeader("Authorization", "Bearer eyJhb....");

var body = @ "{" + "\n" +
@"    ""Value"": 100, ""CountryCode"": ""ITL"" , " + "\n" +
@"    ""SorterRoute"": 0" + "\n" +
@"}" + "\n" +
@"";

request.AddStringBody(body, DataFormat.Json);
RestResponse response = await client.ExecuteAsync(request);
Console.WriteLine(response.Content);

```

NodeJS

```

var request = require('request');
var options = {
  'method': 'POST',
  'url': 'http://localhost:5000/api/CashDevice/SetSorterRoute?
deviceID=TWIN_SMART_COIN_SYSTEM-COM10',
  'headers': {
    'Content-Type': 'application/json',
    'Authorization': 'Bearer eyJhb....'
  },
  body: JSON.stringify({
    "Value": 100,
    "CountryCode": "ITL",
    "SorterRoute": 0
  })
};

request(options, function (error, response) {
  if (error) throw new Error(error);
  console.log(response.body);
});

```

Python

```

import requests
import json

url = "http://localhost:5000/api/CashDevice/SetSorterRoute?
deviceID=TWIN_SMART_COIN_SYSTEM-COM10"

payload = json.dumps({
    "Value": 100,
    "CountryCode": "ITL",
    "SorterRoute": 0
})
headers = {
    'Content-Type': 'application/json',
    'Authorization': 'Bearer eyJhb....'
}

response = requests.request("POST", url, headers=headers, data=payload)

```

```
print(response.text)
```

Java

```
OkHttpClient client = new OkHttpClient().newBuilder()
    .build();
MediaType mediaType = MediaType.parse("application/json");
RequestBody body = RequestBody.create(mediaType, "{\"Value\": 100,
\"CountryCode\": \"ITL\", \"SorterRoute\": 0}");
Request request = new Request.Builder()
    .url("http://localhost:5000/api/CashDevice/SetSorterRoute?
deviceID=TWIN_SMART_COIN_SYSTEM-COM10")
    .method("POST", body)
    .addHeader("Content-Type", "application/json")
    .addHeader("Authorization", "Bearer eyJhb.....")
    .build();
Response response = client.newCall(request).execute();
```

REST API - GetSorterRouteAssignment

Description

Retrieves sorter route assignments.

| | |
|----------------------|---|
| Endpoint | GetSorterRouteAssignment |
| Method | GET |
| URL | {server_url}/api/CashDevice/GetSorterRouteAssignment |
| Parameters | <ul style="list-style-type: none">• deviceID (string): The value of "deviceID" obtained from the response to the OpenConnection request |
| Authorisation | Bearer Token |
| Body | None |

Responses

| Status | Body | Notes |
|--------|---|--|
| 200 | <pre>[{ "currentSorterRoute":0 }, { "currentSorterRoute":1 }, { "currentSorterRoute":0 }]</pre> | Ok: If the sorter route assignments were retrieved successfully. |
| 404 | | Not Found: If no sorter route assignments were found. |
| 400 | | Bad Request: If there is an error retrieving the sorter route assignments. |

Code Examples

C#

```
var options = new RestClientOptions("http://localhost:5000")  
{  
    MaxTimeout = -1,  
};
```

```

var client = new RestClient(options);
var request = new RestRequest("/api/CashDevice/GetSorterRouteAssignment?
deviceID=TWIN_SMART_COIN_SYSTEM-COM10", Method.Get);
request.AddHeader("Authorization", "Bearer eyJhb.....");
RestResponse response = await client.ExecuteAsync(request);
Console.WriteLine(response.Content);

```

NodeJS

```

var request = require('request');
var options = {
  'method': 'GET',
  'url': 'http://localhost:5000/api/CashDevice/GetSorterRouteAssignment?
deviceID=TWIN_SMART_COIN_SYSTEM-COM10',
  'headers': {
    'Authorization': 'Bearer eyJhb.....'
  }
};
request(options, function (error, response) {
  if (error) throw new Error(error);
  console.log(response.body);
});

```

Python

```

import requests

url = "http://localhost:5000/api/CashDevice/GetSorterRouteAssignment?
deviceID=TWIN_SMART_COIN_SYSTEM-COM10"

payload = {}
headers = {
  'Authorization': 'Bearer eyJhb.....'
}

response = requests.request("GET", url, headers=headers, data=payload)

print(response.text)

```

Java

```

OkHttpClient client = new OkHttpClient().newBuilder()
  .build();
MediaType mediaType = MediaType.parse("text/plain");
RequestBody body = RequestBody.create(mediaType, "");
Request request = new Request.Builder()
  .url("http://localhost:5000/api/CashDevice/GetSorterRouteAssignment?
deviceID=TWIN_SMART_COIN_SYSTEM-COM10")
  .method("GET", body)
  .addHeader("Authorization", "Bearer eyJhb.....")
  .build();
Response response = client.newCall(request).execute();

```

REST API - SetPayoutLimit

Description

- Limits the number of coins that can be dispensed in one transaction
- Should be sent during the setup of the device

| | |
|----------------------|---|
| Endpoint | SetPayoutLimit |
| Method | POST |
| URL | {server_url}/api/CashDevice/SetPayoutLimit |
| Parameters | <ul style="list-style-type: none">• deviceID (string): The value of "deviceID" obtained from the response to the OpenConnection request |
| Authorisation | Bearer Token |
| Body | ushort value: The maximum number of coins that can be dispensed in one transaction |

Responses

| Status | Body | Notes |
|--------|--|---|
| 200 | TWIN_SMART_COIN_SYSTEM-COM10: Payout limit set successfully. | OK: If the payout limit was set successfully. |
| 400 | | Bad Request: If there is an error setting the payout limit. |

Code Examples

C#

```
var options = new RestClientOptions("http://localhost:5000")
{
    MaxTimeout = -1,
};
var client = new RestClient(options);
var request = new RestRequest("/api/CashDevice/SetPayoutLimit?
deviceID=TWIN_SMART_COIN_SYSTEM-COM10", Method.Post);
request.AddHeader("Content-Type", "application/json");
request.AddHeader("Authorization", "Bearer eyJhb.....");
var body = @"5";
request.AddStringBody(body, DataFormat.Json);
RestResponse response = await client.ExecuteAsync(request);
Console.WriteLine(response.Content);
```

NodeJS

```
var request = require('request');
var options = {
  'method': 'POST',
  'url': 'http://localhost:5000/api/CashDevice/SetPayoutLimit?',
  deviceID=TWIN_SMART_COIN_SYSTEM-COM10',
  'headers': {
    'Content-Type': 'application/json',
    'Authorization': 'Bearer eyJhb.....'
  },
  body: JSON.stringify(5)

};

request(options, function (error, response) {
  if (error) throw new Error(error);
  console.log(response.body);
});
```

Python

```
import requests
import json

url = "http://localhost:5000/api/CashDevice/SetPayoutLimit?"
deviceID=TWIN_SMART_COIN_SYSTEM-COM10"

payload = json.dumps(5)
headers = {
  'Content-Type': 'application/json',
  'Authorization': 'Bearer eyJhb.....'
}

response = requests.request("POST", url, headers=headers, data=payload)

print(response.text)
```

Java

```
OkHttpClient client = new OkHttpClient().newBuilder()
  .build();
MediaType mediaType = MediaType.parse("application/json");
RequestBody body = RequestBody.create(mediaType, "5");
Request request = new Request.Builder()
  .url("http://localhost:5000/api/CashDevice/SetPayoutLimit?")
  deviceID=TWIN_SMART_COIN_SYSTEM-COM10")
  .method("POST", body)
  .addHeader("Content-Type", "application/json")
  .addHeader("Authorization", "Bearer eyJhb.....")
  .build();
Response response = client.newCall(request).execute();
```

REST API - GetPayoutCount

Description

Gets the number of coins the hopper wants to pay out after it calculates the coin split on a test payout.

| | |
|----------------------|---|
| Endpoint | GetPayoutCount |
| Method | GET |
| URL | {server_url}/api/CashDevice/GetPayoutCount |
| Parameters | <ul style="list-style-type: none">• deviceID (string): The value of "deviceID" obtained from the response to the OpenConnection request |
| Authorisation | Bearer Token |
| Body | None |

Responses

| Status | Body | Notes |
|--------|--|--|
| 200 | TWIN_SMART_COIN_SYSTEM-COM10: { PayoutCount = 0 } | OK: If the payout count was retrieved successfully. |
| 400 | | Bad Request: If there is an error retrieving the payout count. |

Code Examples

C#

```
var options = new RestClientOptions("http://localhost:5000")
{
    MaxTimeout = -1,
};
var client = new RestClient(options);
var request = new RestRequest("/api/CashDevice/GetPayoutCount?
deviceID=TWIN_SMART_COIN_SYSTEM-COM10", Method.Get);
request.AddHeader("Authorization", "Bearer eyJhb.....");
RestResponse response = await client.ExecuteAsync(request);
Console.WriteLine(response.Content);
```

NodeJS

```
var request = require('request');
var options = {
```

```

'method': 'GET',
'url': 'http://localhost:5000/api/CashDevice/GetPayoutCount?
deviceID=TWIN_SMART_COIN_SYSTEM-COM10',
'headers': {
    'Authorization': 'Bearer eyJhb.....'
}
};

request(options, function (error, response) {
    if (error) throw new Error(error);
    console.log(response.body);
});

```

Python

```

import requests

url = "http://localhost:5000/api/CashDevice/GetPayoutCount?
deviceID=TWIN_SMART_COIN_SYSTEM-COM10"

payload = {}
headers = {
    'Authorization': 'Bearer eyJhb.....'
}

response = requests.request("GET", url, headers=headers, data=payload)

print(response.text)

```

Java

```

OkHttpClient client = new OkHttpClient().newBuilder()
    .build();
MediaType mediaType = MediaType.parse("text/plain");
RequestBody body = RequestBody.create(mediaType, "");
Request request = new Request.Builder()
    .url("http://localhost:5000/api/CashDevice/GetPayoutCount?
deviceID=TWIN_SMART_COIN_SYSTEM-COM10")
    .method("GET", body)
    .addHeader("Authorization", "Bearer eyJhb.....")
    .build();
Response response = client.newCall(request).execute();

```

REST API - SetTwinMode

Description

A command to select the operation mode of the Twin SCS

| | |
|----------------------|---|
| Endpoint | SetTwinMode |
| Method | POST |
| URL | {server_url}/api/CashDevice/SetTwinMode |
| Parameters | <ul style="list-style-type: none">• deviceID (string): The value of "deviceID" obtained from the response to the OpenConnection request |
| Authorisation | Bearer Token |
| Body | <ul style="list-style-type: none">• byte twinMode: The operation mode. Currently there are 4 modes available:<ul style="list-style-type: none">• 0: Normal Smart Coin System: The master SCS works alone as a normal SCS. There is no need to connect the secondary hopper as it is unused in this mode.• 1: Twin Smart Coin System: The unit works as a complete Twin Smart Coin System. The secondary hopper needs to be connected to the main hopper.• 2: Twin Single Smart Coin System: The Twin Feeder is able to pay in coins to both routes (main hopper or lateral) based on host selection, but there is no control over the secondary hopper. That means in this mode only the main hopper is in control for the payouts. The slave hopper, if connected, should be controlled by the host individually. For this mode the following commands/events can be used by the host<ul style="list-style-type: none">• Get coin amount to lateral route: To account for the coins sent to the lateral path.• Set coin amount to lateral route: To set/reset the coins amount to the lateral path.• When working with individual coin events during pay-in (event coin credit 0x0D in CC2), every coin paid-in reports the value and the country code as usual, but also an extra byte with the route (main hopper or lateral). In this way the coins sent to lateral can be accounted in this Single Twin mode during a pay-in.• 3: Twin 1ec Balanced mode: This mode is the same as the Twin mode 0x01, but it handles the EUR 1ec in a way that the quantities are balanced in both hoppers. This mode improves the quantity of 1ec coins that the system can handle in the hoppers. Please note:<ul style="list-style-type: none">• During payins, the 1ec coins will be routed to the hopper with less quantity of coins.• During payouts, the 1ec coins will be taken from the hopper with more 1ec coins (or from both if necessary).• If the set coin amount command is sent, the levels will be updated in the slave hopper only (as for operation it might be easier to manipulate the levels). If a level 0 is sent both hoppers levels will be cleared of 1ec. |

Responses

| Status | Body | Notes |
|--------|---|--|
| 200 | TWIN_SMART_COIN_SYSTEM-COM10: Twin mode set successfully. | OK: If the twin mode was set successfully. |
| 400 | | Bad Request: If there is an error setting the twin mode. |

Code Examples

C#

```
var options = new RestClientOptions("http://localhost:5000")
{
    MaxTimeout = -1,
};

var client = new RestClient(options);
var request = new RestRequest("/api/CashDevice/SetTwInMode?
deviceID=TWIN_SMART_COIN_SYSTEM-COM10", Method.Post);
request.AddHeader("Content-Type", "application/json");
request.AddHeader("Authorization", "Bearer eyJhbG.....");
var body = @"1";
request.AddStringBody(body, DataFormat.Json);
RestResponse response = await client.ExecuteAsync(request);
Console.WriteLine(response.Content);
```

NodeJS

```
var request = require('request');
var options = {
    'method': 'POST',
    'url': 'http://localhost:5000/api/CashDevice/SetTwInMode?
deviceID=TWIN_SMART_COIN_SYSTEM-COM10',
    'headers': {
        'Content-Type': 'application/json',
        'Authorization': 'Bearer eyJhbG.....'
    },
    body: JSON.stringify(1)
};

request(options, function (error, response) {
    if (error) throw new Error(error);
    console.log(response.body);
});
```

Python

```
import requests
import json
```

```
url = "http://localhost:5000/api/CashDevice/SetTwInMode?  
deviceID=TWIN_SMART_COIN_SYSTEM-COM10"  
  
payload = json.dumps(1)  
headers = {  
    'Content-Type': 'application/json',  
    'Authorization': 'Bearer eyJhb.....'  
}  
  
response = requests.request("POST", url, headers=headers, data=payload)  
  
print(response.text)
```

Java

```
OkHttpClient client = new OkHttpClient().newBuilder()  
    .build();  
MediaType mediaType = MediaType.parse("application/json");  
RequestBody body = RequestBody.create(mediaType, "1");  
Request request = new Request.Builder()  
    .url("http://localhost:5000/api/CashDevice/SetTwInMode?  
deviceID=TWIN_SMART_COIN_SYSTEM-COM10")  
    .method("POST", body)  
    .addHeader("Content-Type", "application/json")  
    .addHeader("Authorization", "Bearer eyJhb.....")  
    .build();  
Response response = client.newCall(request).execute();
```

REST API - comPortReadError

Description

Checks if there is any read error on the COM port

| | |
|----------------------|---|
| Endpoint | comPortReadError |
| Method | GET |
| URL | {server_url}/api/CashDevice/comPortReadError |
| Parameters | <ul style="list-style-type: none">• deviceID (string): The value of "deviceID" obtained from the response to the OpenConnection request |
| Authorisation | Bearer Token |
| Body | None |

Responses

| Status | Body | Notes |
|--------|--|---|
| 200 | <pre>NV4000-COM5: { ComPortReadError = False }</pre> | OK: If the com port read error status was retrieved successfully. |
| 400 | | Bad Request |

Code Examples

C#

```
var options = new RestClientOptions("http://localhost:5000")
{
    MaxTimeout = -1,
};
var client = new RestClient(options);
var request = new RestRequest("/api/CashDevice/comPortReadError?deviceID=NV4000-
COM5", Method.Get);
request.AddHeader("Authorization", "Bearer eyJhb.....");
RestResponse response = await client.ExecuteAsync(request);
Console.WriteLine(response.Content);
```

NodeJS

```
var request = require('request');
var options = {
```

```

'method': 'GET',
'url': 'http://localhost:5000/api/CashDevice/comPortReadError?deviceID=NV4000-
COM5',
'headers': {
    'Authorization': 'Bearer eyJhb.....'
}
};

request(options, function (error, response) {
    if (error) throw new Error(error);
    console.log(response.body);
});

```

Python

```

import requests

url = "http://localhost:5000/api/CashDevice/comPortReadError?deviceID=NV4000-COM5"

payload = {}
headers = {
    'Authorization': 'Bearer eyJhb.....'
}

response = requests.request("GET", url, headers=headers, data=payload)

print(response.text)

```

Java

```

OkHttpClient client = new OkHttpClient().newBuilder()
    .build();
MediaType mediaType = MediaType.parse("text/plain");
RequestBody body = RequestBody.create(mediaType, "");
Request request = new Request.Builder()
    .url("http://localhost:5000/api/CashDevice/comPortReadError?deviceID=NV4000-COM5")
    .method("GET", body)
    .addHeader("Authorization", "Bearer eyJhb.....")
    .build();
Response response = client.newCall(request).execute();

```

REST API - GetLifterStatus

Description

Allows to get the current status of the lifter

| | |
|----------------------|---|
| Endpoint | GetLifterStatus |
| Method | GET |
| URL | {server_url}/api/CashDevice/GetLifterStatus |
| Parameters | <ul style="list-style-type: none">• deviceID (string): The value of "deviceID" obtained from the response to the OpenConnection request |
| Authorisation | Bearer Token |
| Body | None |

Responses

| Status | Body | Notes |
|--------|--|---|
| 200 | <pre>SMART_COIN_SYSTEM-COM10: { LifterConnected = True, LifterOptoClear = True, LifterJammed = False }</pre> | OK: If the lifter status was retrieved successfully. |
| 400 | | Bad Request: If there is an error retrieving the lifter status. |

Code Examples

C#

```
var options = new RestClientOptions("http://localhost:5000")
{
    MaxTimeout = -1,
};
var client = new RestClient(options);
var request = new RestRequest("/api/CashDevice/GetLifterStatus?
deviceID=SMART_COIN_SYSTEM-COM10", Method.Get);
request.AddHeader("Authorization", "Bearer eyJhb.....");
RestResponse response = await client.ExecuteAsync(request);
Console.WriteLine(response.Content);
```

NodeJS

```
var request = require('request');
var options = {
  'method': 'GET',
  'url': 'http://localhost:5000/api/CashDevice/GetLifterStatus?
deviceID=SMART_COIN_SYSTEM-COM10',
  'headers': {
    'Authorization': 'Bearer eyJhb.....'
  }
};
request(options, function (error, response) {
  if (error) throw new Error(error);
  console.log(response.body);
});
```

Python

```
import requests

url = "http://localhost:5000/api/CashDevice/GetLifterStatus?
deviceID=SMART_COIN_SYSTEM-COM10"

payload = {}
headers = {
  'Authorization': 'Bearer eyJhb.....'
}

response = requests.request("GET", url, headers=headers, data=payload)

print(response.text)
```

Java

```
OkHttpClient client = new OkHttpClient().newBuilder()
  .build();
MediaType mediaType = MediaType.parse("text/plain");
RequestBody body = RequestBody.create(mediaType, "");
Request request = new Request.Builder()
  .url("http://localhost:5000/api/CashDevice/GetLifterStatus?
deviceID=SMART_COIN_SYSTEM-COM10")
  .method("GET", body)
  .addHeader("Authorization", "Bearer eyJhb.....")
  .build();
Response response = client.newCall(request).execute();
```

REST API - GetLastRejectCode

Description

Gets the reason the device rejected the last note

| | |
|----------------------|---|
| Endpoint | GetLastRejectCode |
| Method | GET |
| URL | {server_url}/api/CashDevice/GetLastRejectCode |
| Parameters | <ul style="list-style-type: none">deviceID (string): The value of "deviceID" obtained from the response to the OpenConnection request |
| Authorisation | Bearer Token |
| Body | None |

Responses

| Status | Body | Notes |
|--------|--|--|
| 200 | <pre>SPECTRAL_PAYOUT-COM5: { RejectCategory = Reject Reason: Channel Inhibit }</pre> | OK: If the last reject code was retrieved successfully. |
| 400 | | Bad Request: If there is an error retrieving the last reject code. |

Code Examples

C#

```
var options = new RestClientOptions("http://localhost:5000")
{
    MaxTimeout = -1,
};
var client = new RestClient(options);
var request = new RestRequest("/api/CashDevice/GetLastRejectCode?
deviceID=SPECTRAL_PAYOUT-COM5", Method.Get);
request.AddHeader("Authorization", "Bearer eyJhb.....");
RestResponse response = await client.ExecuteAsync(request);
Console.WriteLine(response.Content);
```

NodeJS

```
var request = require('request');
var options = {
  'method': 'GET',
  'url': 'http://localhost:5000/api/CashDevice/GetLastRejectCode?
deviceID=SPECTRAL_PAYOUT-COM5',
  'headers': {
    'Authorization': 'Bearer eyJhb.....'
  }
};
request(options, function (error, response) {
  if (error) throw new Error(error);
  console.log(response.body);
});
```

Python

```
import requests

url = "http://localhost:5000/api/CashDevice/GetLastRejectCode?
deviceID=SPECTRAL_PAYOUT-COM5"

payload = {}
headers = {
  'Authorization': 'Bearer eyJhb.....'
}

response = requests.request("GET", url, headers=headers, data=payload)

print(response.text)
```

Java

```
OkHttpClient client = new OkHttpClient().newBuilder()
  .build();
MediaType mediaType = MediaType.parse("text/plain");
RequestBody body = RequestBody.create(mediaType, "");
Request request = new Request.Builder()
  .url("http://localhost:5000/api/CashDevice/GetLastRejectCode?
deviceID=SPECTRAL_PAYOUT-COM5")
  .method("GET", body)
  .addHeader("Authorization", "Bearer eyJhb.....")
  .build();
Response response = client.newCall(request).execute();
```

REST API - GetSmartCurrencyData

Description

Retrieves the available currency files stored on the Smart Currency SD Card.

 Can be used on the SCU enabled NV200 Spectral only. Devices must be purchased as SCU devices as they cannot be configured retrospectively to SCU devices.

| | |
|----------------------|---|
| Endpoint | GetSmartCurrencyData |
| Method | GET |
| URL | {server_url}/api/CashDevice/GetSmartCurrencyData |
| Parameters | <ul style="list-style-type: none">• deviceID (string): The value of "deviceID" obtained from the response to the OpenConnection request |
| Authorisation | Bearer Token |
| Body | None |

Responses

| Field | Description | Example |
|-------------|---|----------------------------|
| enabled | True for the currently enabled currency, otherwise false. | "enabled": false , |
| country | The ISO 4217 Currency Code. | "country": "GBP", |
| datasetCode | The dataset name of the available file. | "datasetCode": "GBP07057", |
| denomCount | The number of denomination channels in the dataset. | "denomCount": 4, |
| index | The index of the dataset. | "index": 0 |

Example Response

| Status | Body | Notes |
|--------|---|---|
| 200 | <pre>[{ "enabled": false, "country": "GBP", "datasetCode": "GBP07057", "denomCount": 4, "index": 0 }, { "enabled": false, "country": "USD", "datasetCode": "USD01010", "denomCount": 7, "index": 1 }, { "enabled": false, "country": "EUR", "datasetCode": "EUR01045", "denomCount": 7, "index": 2 }]</pre> | OK: Successfully retrieved the Smart Currency data. |
| 400 | | Bad Request: If there is an error retrieving the Smart Currency Data. |

Code Examples

C#

```
var client = new HttpClient();
var request = new HttpRequestMessage(HttpMethod.Get, "http://localhost:5000/api/CashDevice/?deviceID=SMART_CURRENCY_SYSTEM-COM5");
request.Headers.Add("Authorization", "Bearer eyJhb.....");
var response = await client.SendAsync(request);
response.EnsureSuccessStatusCode();
Console.WriteLine(await response.Content.ReadAsStringAsync());
```

NodeJS

```
var request = require('request');
var options = {
  'method': 'GET',
  'url': 'http://localhost:5000/api/CashDevice/?deviceID=SMART_CURRENCY_SYSTEM-COM5',
  'headers': {
    'Authorization': 'Bearer eyJhb.....'
  }
};
request(options, function (error, response) {
```

```
if (error) throw new Error(error);
console.log(response.body);
});
```

Python

```
import requests

url = "http://localhost:5000/api/CashDevice/?deviceID=SMART_CURRENCY_SYSTEM-COM5"

payload = {}
headers = {
    'Authorization': 'Bearer eyJhb.....'
}

response = requests.request("GET", url, headers=headers, data=payload)

print(response.text)
```

Java

```
OkHttpClient client = new OkHttpClient().newBuilder()
    .build();
MediaType mediaType = MediaType.parse("text/plain");
RequestBody body = RequestBody.create(mediaType, "");
Request request = new Request.Builder()
    .url("http://localhost:5000/api/CashDevice/?deviceID=SMART_CURRENCY_SYSTEM-COM5")
    .method("GET", body)
    .addHeader("Authorization", "Bearer eyJhb.....")
    .build();
Response response = client.newCall(request).execute();
```

REST API - UpdateSmartCurrencyDataset

Description

Sets the the dataset for the currency you wish to accept from the inserted SD card on the Smart Currency NV200 Spectral. Available datasets can be retrieved using the [GetSmartCurrencyData](#) endpoint.

 Can be used on the SCU enabled NV200 Spectral only. Devices must be purchased as SCU devices as they cannot be configured retrospectively to SCU devices.

| | |
|----------------------|--|
| Endpoint | UpdateSmartCurrencyDataset |
| Method | POST |
| URL | {server_url}/api/Download/UpdateSmartCurrencyDataset |
| Parameters | None |
| Authorisation | Bearer Token |
| Body | The dataset code for the currency file you wish to accept. "EUR01045" |

Responses

| Status | Body | Notes |
|--------|--|---------------------------------------|
| 200 | Successfully updated smart currency dataset to EUR01045. | OK: Successfully updated the dataset. |
| 400 | | Bad Request |
| 500 | | Internal Server Error |

Code Examples

C#

```
var client = new HttpClient();
var request = new HttpRequestMessage(HttpMethod.Post, "http://localhost:5000/api/CashDevice/UpdateSmartCurrencyDataset?deviceID=SMART_CURRENCY_SYSTEM-COM5");
request.Headers.Add("Authorization", "Bearer eyJhb.....");
var content = new StringContent("\"EUR01045\"", null, "application/json");
request.Content = content;
var response = await client.SendAsync(request);
response.EnsureSuccessStatusCode();
```

```
Console.WriteLine(await response.Content.ReadAsStringAsync());
```

NodeJS

```
var request = require('request');
var options = {
  'method': 'POST',
  'url': 'http://localhost:5000/api/CashDevice/UpdateSmartCurrencyDataset?
deviceID=SMART_CURRENCY_SYSTEM-COM5',
  'headers': {
    'Content-Type': 'application/json',
    'Authorization': 'Bearer eyJhb.....'
  },
  body: JSON.stringify("EUR01045")
};

request(options, function (error, response) {
  if (error) throw new Error(error);
  console.log(response.body);
});
```

Python

```
import requests
import json

url = "http://localhost:5000/api/CashDevice/UpdateSmartCurrencyDataset?
deviceID=SMART_CURRENCY_SYSTEM-COM5"

payload = json.dumps("EUR01045")
headers = {
  'Content-Type': 'application/json',
  'Authorization': 'Bearer eyJhb.....'
}

response = requests.request("POST", url, headers=headers, data=payload)

print(response.text)
```

Java

```
OkHttpClient client = new OkHttpClient().newBuilder()
  .build();
MediaType mediaType = MediaType.parse("application/json");
RequestBody body = RequestBody.create(mediaType, "\"EUR01045\"");
Request request = new Request.Builder()
  .url("http://localhost:5000/api/CashDevice/UpdateSmartCurrencyDataset?
deviceID=SMART_CURRENCY_SYSTEM-COM5")
  .method("POST", body)
  .addHeader("Content-Type", "application/json")
  .addHeader("Authorization", "Bearer eyJhb.....")
  .build();
Response response = client.newCall(request).execute();
```

REST API - StartDownload

Description

Begins the download of the specified firmware/dataset file to the ITL cash device.

 Not Available in Android APK at this time.

| | |
|----------------------|---|
| Endpoint | StartDownload |
| Method | POST |
| URL | {server_url}/api/Download/StartDownload |
| Parameters | None |
| Authorisation | Bearer Token |
| Body | <pre>{ "DownloadFileName": "{{DownloadFilePath}}\\{{UpdateFileName}}.bv1", "ComPort": "{{ComPort}}", "SspAddress": {{SspAddress}} }</pre> |

Body Parameters

| Parameter | Type | Description | Required | Default |
|------------------|---------|---|----------|---------|
| DownloadFileName | string | Full path to the firmware/dataset file (.bv1) | Yes | |
| ComPort | string | The COM port to use | Yes | |
| SSPAddress | byte | SSP address | Yes | |
| PacketDownload | boolean | Enable Packet Download | No | true |
| BaudRate | int | The data transfer rate | No | 115200 |

Responses

| Status | Body | Notes |
|--------|--|--------------------------------------|
| 200 | Download initiated. Check progress with /GetDownloadStatus. | OK: Successfully initiated download. |
| 400 | Invalid parameters. Please provide a valid file name and COM port. | Bad Request |
| 500 | Failed to initiate download. | Internal Server Error |

Code Examples

C#

```
var options = new RestClientOptions("http://localhost:5000")
{
    MaxTimeout = -1,
};
var client = new RestClient(options);
var request = new RestRequest("/api/Download/StartDownload", Method.Post);
request.AddHeader("Content-Type", "application/json");
request.AddHeader("Authorization", "Bearer eyJhb.....");
var body = @"" + "\n" +
@"
    ""DownloadFileName"": ""C:\\InnovativeTechnology\\UpdateFile\\
    \\ITL01003_NVS2004311048000_IF_01.bv1""," + "\n" +
@"
    ""ComPort"": ""COM5""," + "\n" +
@"
    ""SspAddress"": 0" + "\n" +
@"}";
request.AddStringBody(body, DataFormat.Json);
RestResponse response = await client.ExecuteAsync(request);
Console.WriteLine(response.Content);
```

NodeJS

```
var request = require('request');
var options = {
    'method': 'POST',
    'url': 'http://localhost:5000/api/Download/StartDownload',
    'headers': {
        'Content-Type': 'application/json',
        'Authorization': 'Bearer eyJhb.....'
    },
    body: JSON.stringify({
        "DownloadFileName": "C:\\InnovativeTechnology\\UpdateFile\\
        \\ITL01003_NVS2004311048000_IF_01.bv1",
        "ComPort": "COM5",
        "SspAddress": 0
    })
};
```

```

        });

};

request(options, function (error, response) {
  if (error) throw new Error(error);
  console.log(response.body);
});

```

Python

```

import requests
import json

url = "http://localhost:5000/api/Download/StartDownload"

payload = json.dumps({
  "DownloadFileName": "C:\\InnovativeTechnology\\UpdateFile\\
  \\ITL01003_NVS2004311048000_IF_01.bv1",
  "ComPort": "COM5",
  "SspAddress": 0
})
headers = {
  'Content-Type': 'application/json',
  'Authorization': 'Bearer eyJhb....'
}

response = requests.request("POST", url, headers=headers, data=payload)

print(response.text)

```

Java

```

OkHttpClient client = new OkHttpClient().newBuilder()
  .build();
MediaType mediaType = MediaType.parse("application/json");
RequestBody body = RequestBody.create(mediaType, "{\r\n  \"DownloadFileName\": \"C:\\\\
  \\\\InnovativeTechnology\\\\\\UpdateFile\\\\\\ITL01003_NVS2004311048000_IF_01.bv1\",\\r\\n
  \"ComPort\": \"COM5\",\\r\\n  \"SspAddress\": 0\\r\\n}");
Request request = new Request.Builder()
  .url("http://localhost:5000/api/Download/StartDownload")
  .method("POST", body)
  .addHeader("Content-Type", "application/json")
  .addHeader("Authorization", "Bearer eyJhb....")
  .build();
Response response = client.newCall(request).execute();

```

REST API - GetDownloadStatus

Description

Used to monitor the progress of a firmware/dataset update to the ITL cash device in real-time.



Not Available in Android APK at this time.

| | |
|----------------------|---|
| Endpoint | GetDownloadStatus |
| Method | Get |
| URL | {server_url}/api/Download/GetDownloadStatus |
| Parameters | None |
| Authorisation | Bearer Token |
| Body | None |

Responses

| Field | Type | Description |
|----------------------|---------|---|
| State | string | Current state of the download: <ul style="list-style-type: none">· IDLE· UPDATING· COMPLETE |
| CurrentDownloadBlock | integer | The current download block number |
| TotalDownloadBlock | integer | Total number of download blocks |
| Success | boolean | True if the download is complete, otherwise false |

Example Responses

| Status | Body | Notes |
|--------|--|--------------------------------------|
| 200 | <pre>{ "state": "UPDATING", "currentDownloadBlock": 4453, "totalDownloadBlock": 4655, "success": false }</pre> | OK: Successfully initiated download. |

| Status | Body | Notes |
|--------|------|-----------------------|
| 400 | | Bad Request |
| 500 | | Internal Server Error |

Code Examples

C#

```
var options = new RestClientOptions("http://localhost:5000")
{
    MaxTimeout = -1,
};
var client = new RestClient(options);
var request = new RestRequest("/api/Download/GetDownloadStatus", Method.Get);
request.AddHeader("Authorization", "Bearer eyJhb.....");
RestResponse response = await client.ExecuteAsync(request);
Console.WriteLine(response.Content);
```

NodeJS

```
var request = require('request');
var options = {
    'method': 'GET',
    'url': 'http://localhost:5000/api/Download/GetDownloadStatus',
    'headers': {
        'Authorization': 'Bearer eyJhb.....'
    }
};
request(options, function (error, response) {
    if (error) throw new Error(error);
    console.log(response.body);
});
```

Python

```
import requests

url = "http://localhost:5000/api/Download/GetDownloadStatus"

payload = []
headers = [
    'Authorization': 'Bearer eyJhb.....'
]

response = requests.request("GET", url, headers=headers, data=payload)

print(response.text)
```

Java

```
OkHttpClient client = new OkHttpClient().newBuilder()
    .build();
MediaType mediaType = MediaType.parse("text/plain");
RequestBody body = RequestBody.create(mediaType, "");
Request request = new Request.Builder()
    .url("http://localhost:5000/api/Download/GetDownloadStatus")
    .method("GET", body)
    .addHeader("Authorization", "Bearer eyJhb.....")
    .build();
Response response = client.newCall(request).execute();
```