CSL HTTP API Specifications

Version 1.6

2024 10 22

1. Release Notes

Dates	Release	Description
2024 10 22	1.6	Add new Tag Group APIs, Revised Layouts

2. Content

3. Introduction

3.1 CSL Intelligent Reader

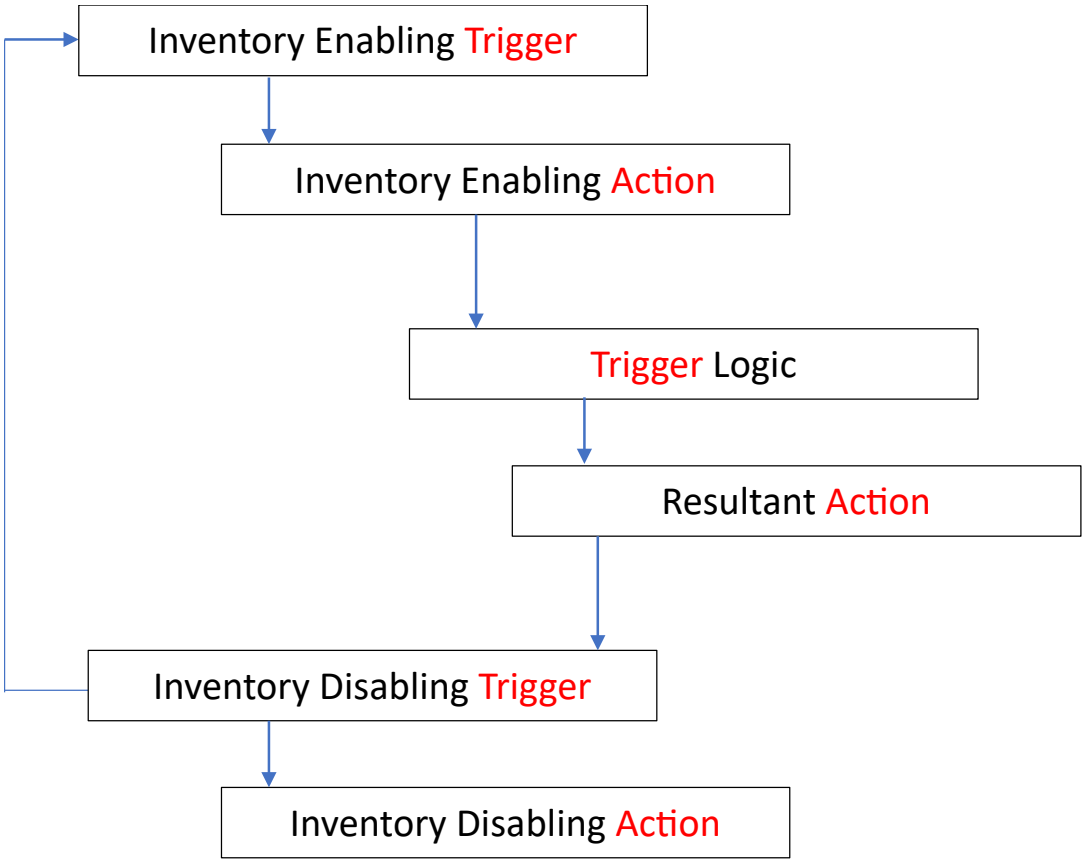
The CSL Intelligent Reader family are intelligent reader designed to work standalone in an autonomous manner. An intelligent Event Engine is embedded with configurable complex logic sequence, triggers and resultant actions that are automatically activated on power up. All reader settings, including the events, can be saved and further deployed to as many readers, as many sites as you want, thus providing easy scalability for the system integrators.

The readers can act alone and be configured to send tag data to Cloud servers anywhere, or, on top of that, work together with other readers in the site in Master/Slave manner. Operators can remote control the readers, adding and modifying tag groups for membership management in the case of RFID access control.

This document describes how to remote manage the readers using HTTP protocols.

3.2 CSL Event Engine

The CSL Event Engine is a multilevel event loop that allows configurable triggers and resultant actions. Such event, looped around endlessly, provides autonomous control and management of all kinds of business scenarios.



Definitions

The CSL Event Engine consists of layers of triggers and actions. Here are the definitions of various types of triggers and actions:

Inventory Enabling Trigger: This is the trigger that cause the reader to start doing inventory

Inventory Enabling Action: This is the actions that the reader would do when starting inventory

Trigger Logic: This is the trigger that would cause the reader to act on some particular RFID tag IDs collected during inventory

Resultant Action: This is the action that the reader would do on those RFID tag IDs singled out because of the above Trigger Logic.

Inventory Disabling Trigger: This is the trigger that would cause the reader to stop inventory and loop back to the start of the Event Loop

Inventory Disabling Action: This is the action that accompanies the reader when it stops inventory.

Triggers:

Trigger Mode	can be used in		
	Inventory Enabling Trigger	Trigger Logic	Inventory Disabling Trigger
Always On	x		
Never Stop			x
Read Any Tags (any ID, 1 trigger per tag)		X	
Input Sensor State	X		X
No Tag Read in Specified Time Span from Start of Inventory			X
No Tag Read in Specified Time Span from last Tag Read and Triggered			X
Trigger in Tag Group		X	
Trigger in Tag Database		X	
Trigger if RSSI larger than or equal to		X	
Trigger if Moisture is larger than or equal to		X	
Trigger if Moisture is less than or equal to		X	
Trigger if Temperature is larger than or equal to		X	
Trigger if Temperature is less than or equal to		X	
Specified Time Span elapsed from start of event		X	X
Periodic Time	X		
UDP Trigger	X		

Resultant Actions:

Action Mode	can be used in		
	Inventory Enabling Action	Resultant Action	Inventory Disabling Action
Do Nothing (Only Show on Screen)		X	
Batch Alert to Server		X	
Instant Alert to Server (No Duplicate Elimination)		X	
Low Latency Alert to Server	X	X	X
Alert on TCP Listening Port		X	
Output Port	X	X	X
Save to External USB Memory		X	
Display Tag Database Record		X	
Display Tag Group Record		X	
Debug Serial Port (No Duplicate Elimination)		X	
Debug Serial Port (with Duplicate Elimination)		X	
UDP Message	X		X

4. CSL HTTP API Categories

This document contains 8 categories of HTTP query strings and the XML based response document layout:

1. Users Management
2. System Management
3. Network Management
4. Time and Timer Management
5. Tag and Tag Filter Management
6. GPIO Management
7. Events Management
8. Version Management

This table of API is for Web Application 1.4.82

5. CSL HTTP API Syntax

(1) Server → Reader

The format of High-level HTTP API query from server to reader is as follows:

http://*<IP_address_of_Reader>*/API?session_id=*<session_id>*&command=*<command>*[&*<param1>*=*<param1_value>*]

where:

Variable	Description
<IP_address_of_Reader>	IP address of the CSL intelligent fixed reader
<session_id>	The session ID obtained in the XML response message from reader after user login (not necessary for some commands, e.g. login)
<command>	High-level API command
<param1>	Setting parameter for the corresponding command. It can be optional or more than one parameter
<param1_value>	Value for the corresponding parameter setting

(2) Reader → Server

The format of XML/HTTP response from reader to server is as follows:

```
<?xml version="1.0" ?>
  <CSL>
    <Command>command</Command>
    <Ack>ack_value</Ack>
  </CSL>
```

or

```
<?xml version="1.0" ?>
  <CSL>
    <Command>command</Command>
    <name attribute1="attribute1_value" attribute2="attribute2_value"
      />
  </CSL>
```

Note:

- This document is applicable to CS463 web application 1.1.8 or above.
- All High-Level HTTP API query strings are ***Case-Sensitive***.

6. User's Management

<table>
<tr><th></th><th>query_string</th><th>Description</th></tr>
<tr><td colspan="3">Users Management</td></tr>
<tr><td>6.1</td><td>session_id=<login_session_id>&
command=addUser&
username=username&
password=password&
[desc=desc]
[&item=permission]
⋮</td><td>Adds a new user with name username, password password, and permission of accessing item.
e.g.
session_id=<login_session_id>&command=addUser&username=BruceLi&password=pw123&Status=1&LogFileConfiguration=1&DownloadLogFile=4&TagInventory=1

Valid attributes:
<table>
<tr><th>item</th><th>permission</th></tr>
<tr><td>Status</td><td>1</td></tr>
<tr><td>UserManagement</td><td>1, 2</td></tr>
<tr><td>ForceLogout</td><td>4</td></tr>
<tr><td>ReaderId</td><td>1, 2</td></tr>
<tr><td>CompanyLabel</td><td>1, 2</td></tr>
<tr><td>CapturePointName</td><td>1, 2</td></tr>
<tr><td>AccessMode</td><td>1, 2</td></tr>
<tr><td>CustomEmbeddedApplication</td><td>1, 2</td></tr>
<tr><td>FrequencyConfiguration</td><td>1, 2</td></tr>
<tr><td>OperationProfile</td><td>1, 2</td></tr>
<tr><td>ConfigureLNAGain</td><td>1, 2</td></tr>
<tr><td>MemoryInformation</td><td>1, 2</td></tr>
<tr><td>PowerUpNotification</td><td>1, 2</td></tr>
<tr><td>HeartBeatNotification</td><td>1, 2</td></tr>
<tr><td>ReaderErrorNotification</td><td>1, 2</td></tr>
<tr><td>GPIInterruptNotification</td><td>1, 2</td></tr>
<tr><td>ConfigurationBackupRestore</td><td>1, 2</td></tr>
<tr><td>LogFileConfiguration</td><td>1, 2</td></tr>
<tr><td>DownloadLogFile</td><td>4</td></tr>
</table>
</td></tr>
</table>

<table>
<tr>
<td></td>
<td></td>
<td>
<table>
<tr><td>ScheduledReboot</td><td>1, 2</td></tr>
<tr><td>RebootSystem</td><td>4</td></tr>
<tr><td>EthernetWiFiConfiguration</td><td>1, 2</td></tr>
<tr><td>CloudServer</td><td>1, 2</td></tr>
<tr><td>DataFormat</td><td>1, 2</td></tr>
<tr><td>TimeSetting</td><td>1, 2</td></tr>
<tr><td>TagGroup</td><td>1, 2</td></tr>
<tr><td>TagDatabase</td><td>1, 2</td></tr>
<tr><td>IOPortControl</td><td>1, 2, 4</td></tr>
<tr><td>Trigger</td><td>1, 2</td></tr>
<tr><td>ResultactionAction</td><td>1, 2</td></tr>
<tr><td>Event</td><td>1, 2</td></tr>
<tr><td>DisplayFormat</td><td>1, 2</td></tr>
<tr><td>TagInventory</td><td>1</td></tr>
<tr><td>FirmwareUpgrade</td><td>4</td></tr>
<tr><td>SSLCertificate</td><td>1, 2</td></tr>
<tr><td>TagFilter</td><td>1, 2</td></tr>
</table>

Note:
permission : 1=read, 2=write, 4=execute or bitwise OR of these values

result:
<pre>
<?xml version="1.0" ?>
<CSL>
  <Command>addUser</Command>
  <Ack>OK:</Ack>
</CSL>
</pre>
</td>
</tr>
<tr>
<td>6.2</td>
<td>session_id=<login_session_id>&
command=modUser&
username=username&
[&item=permission]
⋮</td>
<td>Modify the permission of accessing item of a user with name username.
e.g.
session_id=<login_session_id>&command=modUser&username=BruceLi&Status=1&LogFileConfiguration=3&DownloadLogFile=4&TagInventory=1&Event=3&DisplayFormat=3

Valid attributes:</td>
</tr>
</table>

		item	*permission*
		Status	1
		UserManagement	1, 2
		ForceLogout	4
		ReaderId	1, 2
		CompanyLabel	1, 2
		CapturePointName	1, 2
		AccessMode	1, 2
		CustomEmbeddedApplication	1, 2
		FrequencyConfiguration	1, 2
		OperationProfile	1, 2
		ConfigureLNAGain	1, 2
		MemoryInformation	1, 2
		PowerUpNotification	1, 2
		HeartBeatNotification	1, 2
		ReaderErrorNotification	1, 2
		GPIInterruptNotification	1, 2
		ConfigurationBackupRestore	1, 2
		LogFileConfiguration	1, 2
		DownloadLogFile	4
		ScheduledReboot	1, 2
		RebootSystem	4
		EthernetWiFiConfiguration	1, 2
		CloudServer	1, 2
		DataFormat	1, 2
		TimeSetting	1, 2
		TagGroup	1, 2
		TagDatabase	1, 2
		IOPortControl	1, 2, 4
		Trigger	1, 2
		ResultactionAction	1, 2
		Event	1, 2
		DisplayFormat	1, 2
		TagInventory	1
		FirmwareUpgrade	4

<table>
<tr><td></td><td></td><td>
<table>
<tr><td>SSLCertificate</td><td>1, 2</td></tr>
<tr><td>TagFilter</td><td>1, 2</td></tr>
</table>

Note:
permission : 1=read, 2=write, 4=execute or bitwise OR of these values

result:
<?xml version="1.0" ?>
<CSL>
<Command>modUser</Command>
<Ack>OK:</Ack>
</CSL>
</td></tr>
<tr><td>6.3</td><td>session_id=<login_session_id>&
command=delUser&
username=username</td><td>Removes the user with name username.
e.g.
session_id=<login_session_id>&command=delUser&username=Bruce Li

result:
<?xml version="1.0" ?>
<CSL>
<Command>delUser</Command>
<Ack>OK:</Ack>
</CSL>
</td></tr>
<tr><td>6.4</td><td>session_id=<login_session_id>&
command=setUserPassword&
username=username&
password=password</td><td>Sets the user password for the user with name username.
Only “root” user can invoke this command.

e.g.
session_id=<login_session_id>&command=setUserPassword&username=Bruce Li&password=mod123

result:
<?xml version="1.0" ?>
<CSL>
<Command>setUserPassword</Command>
<Ack>OK:</Ack>
</CSL>
</td></tr>
</table>

6.5	session_id=<login_session_id>&
	command=***listUsers***
		List all users information.
		e.g.
		session_id=<login_session_id>&command=listUsers
		
		result:
		`<?xml version="1.0" ?>`
		`<CSL>`
		`<Command>`**listUsers**`</Command>`
		`<Account desc="`**top level administrator**`"`
		`username="`**root**`"`
		`Status="`**1**`"`
		`UserManagement="`**3**`"`
		`ForceLogout="`**4**`"`
		`ReaderId="`**3**`"`
		`CompanyLabel="`**3**`"`
		`CapturePointName="`**3**`"`
		`AccessMode="`**3**`"`
		`CustomEmbeddedApplication="`**3**`"`
		`FrequencyConfiguration="`**3**`"`
		`OperationProfile="`**3**`"`
		`ConfigureLNAGain="`**3**`"`
		`MemoryInformation="`**3**`"`
		`PowerUpNotification="`**3**`"`
		`HeartBeatNotification="`**3**`"`
		`ReaderErrorNotification="`**3**`"`
		`GPIInterruptNotification="`**3**`"`
		`ConfigurationBackupRestore="`**3**`"`
		`LogFileConfiguration="`**3**`"`
		`DownloadLogFile="`**4**`"`
		`ScheduledReboot="`**3**`"`
		`RebootSystem="`**4**`"`
		`EthernetWiFiConfiguration="`**3**`"`
		`CloudServer="`**3**`"`
		`DataFormat="`**3**`"`
		`TimeSetting="`**3**`"`
		`TagGroup="`**3**`"`
		`TagDatabase="`**3**`"`
		`IOPortControl="`**7**`"`
		`Trigger="`**3**`"`
		`ResultantAction="`**3**`"`
		`Event="`**3**`"`
		`DisplayFormat="`**3**`"`
		`TagInventory="`**1**`"`
		`FirmwareUpgrade="`**4**`"`
		`SSLCertificate="`**3**`"`
		`ReaderErrorNotification="`**3**`"`
		`TagFilter="`**3**`" />`
		`<Account desc=""`
		`username="`**BruceLi**`"`
		`Status="`**1**`"`
		`LogFileConfiguration="`**3**`"`
		`DownloadLogFile="`**4**`"`
		`Event="`**3**`"`
		`DisplayFormat="`**3**`"`

<table>
<tr><td></td><td></td><td>TagInventory="1" />
</CSL></td></tr>
<tr><td>6.6</td><td>command=<i>login</i>&
username=<i>username</i>&
password=<i>password</i></td><td>Login is required for access to the reader.
Login is successful if <i>password</i> for the <i>user</i> is correct.

e.g.
command=login&username=root&password=csl

result 1:
(Login successfully)
<?xml version="1.0" ?>
<CSL>
<Command>login</Command>
<Ack>OK: session_id=768f32f8</Ack>
</CSL>

result 2:
(If other user has already logged-in)
<?xml version="1.0" ?>
<CSL>
<Command>login</Command>
<Error alreadyLoginIP="192.168.25.124" alreadyLoginUser="root" code="-10" msg="Error: Only one user can login the system at the same time!Another User root has already logged-in the system (by browser or API command) at location 192.168.25.124.Please logout the other user and retry login." />
</CSL></td></tr>
<tr><td>6.7</td><td>session_id=<login_session_id>&
command=logout</td><td>Log out is recommended to ensure the security and integrity of the system.
e.g.
session_id=<login_session_id>&command=logout

result:
<?xml version="1.0" ?>
<CSL>
<Command>logout</Command>
<Ack>OK:</Ack>
</CSL></td></tr>
</table>

6.8	command=***forceLogout***& username=*username*& password=*password*	Force logout the system and intend to login another session for operation. e.g. command=forceLogout&username=root&password=<password> Remark : username must be “root” result: <?xml version="1.0" ?> <CSL> <Command>**forceLogout**</Command> <Ack>**OK:**</Ack> </CSL>
6.9	session_id=<login_session_id>& command=***setAutoLogoutTime***& time=*time*	Set the auto logout time to the Edge Server such that the it will automatically be logout after the idle time, *time*. e.g. session_id=<login_session_id>&command=setAutoLogoutTime&time=30 Valid attributes: time : unit = minute, 0 = login session never expire result: <?xml version="1.0" ?> <CSL> <Command>**setAutoLogoutTime**</Command> <Ack>**OK:**</Ack> </CSL>
6.10	session_id=<login_session_id>& command=***getAutoLogoutTime***	Get the auto logout time. e.g. session_id=<login_session_id>&command=getAutoLogoutTime result: <?xml version="1.0" ?>

		<CSL> <Command>**getAutoLogoutTime**</Command> <Logout time="**30**" unit="**minute**" /> </CSL>

7. System Management

	query_string	Description
System Management		
7.1	session_id=<login_session_id>& command=***setReaderID***& reader_id=*reader_id*&desc=*desc*	Set Reader ID. e.g. session_id=<login_session_id>&command=setReaderID&reader_id=CS463 Demo Reader&desc=Demo Reader result: <?xml version="1.0" ?> <CSL> <Command>**setReaderID**</Command> <Ack>**OK:**</Ack> </CSL>
7.2	session_id=<login_session_id>& command=***getReaderID***	Get Reader ID. e.g. session_id=<login_session_id>&command=getReaderID result: <CSL> <Command>**getReaderID**</Command> <Reader desc="**CS463 Demo Reader**" reader_id="**Demo Reader**" /> </CSL>
7.3	session_id=<login_session_id>& command=***setCapturePointName*** &capturepoint_id=*capturepoint_id*&name=*name*	Set Antenna Read Point Names. e.g.1 http://192.168.25.160/API?session_id=75cf3f18&command=setCapturePointName&capturepoint_id=Antenna1&name=Room1 e.g.2 http://192.168.25.160/API?session_id=75cf3f18&command=setCapturePointName&capturepoint_id=Antenna2&name=Room2

<table>
<tr><td></td><td></td><td>e.g.3
http://192.168.25.160/API?session_id=75cf3f18&command=setCapturePointName&capturepoint_id=Antenna3&name=Room3

e.g.4
http://192.168.25.160/API?session_id=75cf3f18&command=setCapturePointName&capturepoint_id=Antenna4&name=Room4

Valid attributes :
capturepoint_id : Antenna1, Antenna2 …… Antenna16</td></tr>
<tr><td>7.4</td><td>session_id=<login_session_id>&command=<i>getCapturePointName</i></td><td>Get Capture Point Name (Antenna Name).

e.g.
http://192.168.25.160/API?session_id=75cf3f18&command=getCapturePointName

<?xml version="1.0" ?>
<CSL>
<Command>getCapturePointName</Command>
<capturepoint id="Antenna1" name="Capture Point 1" selected="true" />
<capturepoint id="Antenna2" name="Capture Point 2" selected="true" />
<capturepoint id="Antenna3" name="Capture Point 3" selected="true" />
<capturepoint id="Antenna4" name="Capture Point 4" selected="true" />
<capturepoint id="Antenna5" name="Capture Point 5" selected="true" />
<capturepoint id="Antenna6" name="Capture Point 6" selected="true" />
<capturepoint id="Antenna7" name="Capture Point 7" selected="true" />
<capturepoint id="Antenna8" name="Capture Point 8" selected="true" />
<capturepoint id="Antenna9"</td></tr>
</table>

		name="**Capture Point 9**" selected="**true**" /> <capturepoint id="**Antenna10**" name="**Capture Point 10**" selected="**true**" /> <capturepoint id="**Antenna11**" name="**Capture Point 11**" selected="**true**" /> <capturepoint id="**Antenna12**" name="**Capture Point 12**" selected="**true**" /> <capturepoint id="**Antenna13**" name="**Capture Point 13**" selected="**true**" /> <capturepoint id="**Antenna14**" name="**Capture Point 14**" selected="**true**" /> <capturepoint id="**Antenna15**" name="**Capture Point 15**" selected="**true**" /> <capturepoint id="**Antenna16**" name="**Capture Point 16**" selected="**true**" /> </CSL>
7.5	session_id=<login_session_id>& command=***setAccessMode***& mode=*mode*	To set Access Mode of the reader. e.g. session_id=<login_session_id>&command=setAccessMode& mode=http Valid attributes: mode : high *or* http = High Level HTTP API Mode low = Low Level Mach1 API Mode cslapi = CSL Unified API High Level Mode cslapilow = CSL Unified API Low Level Mode llrp = LLRP API Mode bluetooth = CS108 Bluetooth API Mode customembedded = Custom Embedded RFID HTTP API Mode cslapirs232 = CSL Unified API Mode via RS232 Control Serial Port cslapilowrs232 = CSL Unified API Low Level Mode via RS232 Control Serial Port

		result:
		`<?xml version="1.0" ?>`
		`<CSL>`
		`  <Command>`**`setAccessMode`**`</Command>`
		`  <Ack>`**`OK:`**`</Ack>`
		`</CSL>`

7.6	session_id=<login_session_id>&
	command=***getAccessMode***
		To get Access Mode of the reader.
		
		e.g.
		session_id=<login_session_id>&command=getAccessMode
		
		result 1:
		`<?xml version="1.0" ?>`
		`<CSL>`
		`  <Command>`**`getAccessMode`**`</Command>`
		`  <Access mode="`**`1`**`" name=" HTTP/XML" />`
		`</CSL>`
		
		result 2:
		`<?xml version="1.0" ?>`
		`<CSL>`
		`  <Command>`**`getAccessMode`**`</Command>`
		`  <Access mode="`**`2`**`" name=" CS461 Low Level API (MACH1)" />`
		`</CSL>`
		
		result 3:
		`<?xml version="1.0" ?>`
		`<CSL>`
		`  <Command>`**`getAccessMode`**`</Command>`
		`  <Access mode="`**`3`**`" name="CSL Unified API/High Level" />`
		`</CSL>`
		
		result 4:
		`<?xml version="1.0" ?>`
		`<CSL>`
		`  <Command>`**`getAccessMode`**`</Command>`
		`  <Access mode="`**`4`**`" name="LLRP" />`
		`</CSL>`
		
		result 5:
		`<?xml version="1.0" ?>`
		`<CSL>`
		`  <Command>`**`getAccessMode`**`</Command>`
		`  <Access mode="`**`5`**`" name="CS108 Bluetooth API" />`

		</CSL> result 6: <?xml version="1.0" ?> <CSL> <Command>**getAccessMode**</Command> <Access mode="**6**" name="**Custom Embedded RFID HTTP**" /> </CSL> result 7: <?xml version="1.0" ?> <CSL> <Command>**getAccessMode**</Command> <Access mode="**7**" name="**CSL Unified API/High Level via RS232 Control Serial Port**" /> </CSL> result 8: <?xml version="1.0" ?> <CSL> <Command>**getAccessMode**</Command> <Access mode="**8**" name="**CSL Unified API/Low Level** " /> </CSL> result 9: <?xml version="1.0" ?> <CSL> <Command>**getAccessMode**</Command> <Access mode="**9**" name="**CSL Unified API/Low Level via RS232 Control Serial Port**" /> </CSL>
7.7	session_id=<login_session_id>& command=***setEmbeddedRFIDApp***& path=*path*& cmd=*cmd*	Set Custom Embedded RFID Application. e.g. session_id=<login_session_id>&command=setEmbeddedRFIDApp&path=%2Fopt%2Fcsl_embedded_rfid_example_2.6_20190828&cmd=.%2Fexample+-conf+config_HK.txt result: <CSL> <Command>**setEmbeddedRFIDApp**</Command> <Ack>**OK:**</Ack> </CSL>

7.8	session_id=<login_session_id>& command=***getEmbeddedRFIDApp***	Get Reader ID. e.g. session_id=<login_session_id>&command=getReaderID result: <CSL> <Command>**getEmbeddedRFIDApp**</Command> <embeddedRFIDApp path="**/opt/csl_embedded_rfid_example_2.6_20190828**" cmd="**./example -conf config_HK.txt**" /> </CSL>
7.9	session_id=<login_session_id>& command=***setOperProfile***& profile_id= *profile_id*& linkProfile= *linkProfile*& populationEst= *populationEst*& sessionNo=*sessionNo*& target=*target*& queryAlgorithm=*queryAlgorithm*& reflectedPowerThreshold=*reflectedPowerThreshold*& tagModel=*tagModel*& antenna_port=*antenna_port* [&transmitPower=*transmitPower*] [&transmitPower1=*transmitPower1* &transmitPower2=*transmitPower2* &transmitPower3=*transmitPower3* &transmitPower4=*transmitPower4* &transmitPower5=*transmitPower5*	Configure Operation Profile. e.g. 1 (same transmit power on all antenna) http://192.168.25.160/API?session_id=75cf3f18&command=setOperProfile&profile_id=Default Profile&linkProfile=1&populationEst=50&sessionNo=0&target=2&queryAlgorithm=DynamicQ&reflectedPowerThreshold=24&tagModel=ANY&antenna_port=1,2,3,4&transmitPower=30.00&dwellTime1=2000&dwellTime2=2000&dwellTime3=2000&dwellTime4=2000 e.g. 2 (different transmit power on each antenna) http://192.168.25.160/API?session_id=75cf3f18&command=setOperProfile&profile_id=Default Profile&linkProfile=1&populationEst=50&sessionNo=0&target=0&queryAlgorithm=DynamicQ&reflectedPowerThreshold=24&tagModel=ANY&antenna_port=1,2,3,4&transmitPower1=21.00&transmitPower2=22.00&transmitPower3=23.00&transmitPower4=24.00&dwellTime1=2000&dwellTime2=2000&dwellTime3=2000&dwellTime4=2000 Valid attributes : linkProfile : 0 = Multipath Interface Resistance 1 = Range/Dense Reader 2 = Range/Throughput/Dense Reader

	&transmitPower6=*transmitPower6* &transmitPower7=*transmitPower7* &transmitPower8=*transmitPower8* &transmitPower9=*transmitPower9* &transmitPower10=*transmitPower10* &transmitPower11=*transmitPower11* &transmitPower12=*transmitPower12* &transmitPower13=*transmitPower13* &transmitPower14=*transmitPower14* &transmitPower15=*transmitPower15* &transmitPower16=*transmitPower16*] [&dwellTime1= *dwellTime1* &dwellTime2= *dwellTime2* &dwellTime3= *dwellTime3* &dwellTime4= *dwellTime4* &dwellTime5= *dwellTime5* &dwellTime6= *dwellTime6* &dwellTime7= *dwellTime7* &dwellTime8= *dwellTime8* &dwellTime9= *dwellTime9* &dwellTime10= *dwellTime10* &dwellTime11= *dwellTime11* &dwellTime12= *dwellTime12* &dwellTime13= *dwellTime13* &dwellTime14= *dwellTime14*	3 = Max Throughput populationEst : 1 – 8192 sessionNo : 0 = S0 , 1 = S1 , 2 = S2, 3 = S3 target : 0 = A, 1 = B, 2 = A/B Togggle queryAlgorithm : FixedQ, DynamicQ reflectedPowerThreshold : 1.0 – 32.0 in step of 0.1 dBm tagModel : ANY, Magnus_S2, Magnus_S3, Ctesius antenna_port : 1 – 16, any combinations with comma separated, e.g. 1,2,3,4 transmitPower : 0.0 – 32.0 in step of 0.1 dBm transmitPower1 – 16 : 0.0 – 32.0 in step of 0.1 dBm dwellTime1 – 16 : unit=ms, >= 0ms retry : >= 0 tagFocus : true, false (if it is true, sessionNo is set to 1 and target is set to 0 automatically) Optional attributes: memoryBank1 : Bank0, Bank1, Bank2, Bank3 memoryBank1Offset : >= 0 memoryBank1Length : unit=no. of words, >= 0 memoryBank2 : Bank0, Bank1, Bank2, Bank3 memoryBank2Offset : >= 0 memoryBank2Length : unit=no. of words, >= 0 fastId : true, false minOnChipRSSI : 0 – 31, unit=dBm maxOnChipRSSI : 0 – 31, unit=dBm moistAvgWindow : 1 – 50 tempAvgWindow : 1 – 50 reconfigAntennaPortError : true, false retryErrorAntennaPortTime : unit=second, >=0, 0=never retry preFilter1 – 7 : ID of Tag Filter, the type of the filter must be PRE_FILTER postFilter : ID of Tag Filter, the type of the filter must be POST_FILTER Note: If tagModel is ANY, there can be 7 pre-filters. If tagModel is Magnus_S2, there can be 5 pre-filters.

	&dwellTime15= *dwellTime15* &dwellTime16= *dwellTime16*] [&memoryBank1=*memoryBank1* &memoryBank1Offset=*memoryBank1Offset* &memoryBank1Length=*memoryBank1Length*] [&memoryBank2=*memoryBank2* &memoryBank2Offset=*memoryBank2Offset* &memoryBank2Length=*memoryBank2Length*] [&retry=*retry*] [&tagFocus=*tagFocus*] [&fastId=*fastId*] [&minOnChipRSSI=*minOnChipRSSI*] [&maxOnChipRSSI=*maxOnChipRSSI*] [&moistAvgWindow=*moistAvgWindow*] [&tempAvgWindow=*tempAvgWindow*] [&reconfigAntennaPortError=*reconfigAntennaPortError*] [&retryErrorAntennaPortTime=*retryErrorAntennaPortTime*] [&preFilter1=*preFilter1*] [&preFilter2=*preFilter2*] [&preFilter3=*preFilter3*] [&preFilter4=*preFilter4*] [&preFilter5=*preFilter5*] [&preFilter6=*preFilter6*] [&preFilter7=*preFilter7*] [&postFilter=*postFilter*]	If tagModel is Magnus_S3, there can be 4 pre-filters. If tagModel is Ctesius, there can be 6 pre-filters. result : <?xml version="1.0" ?> <CSL> <Command>**setOperProfile**</Command> <Ack>**OK:**</Ack> </CSL>

7.10	session_id=<login_session_id>& command=***getOperProfile***	Get Operation Profile information. e.g. http://192.168.25.160/API?session_id=7C1286DE&command=getOperProfile <?xml version="1.0" ?> <CSL> <Command>**getOperProfile**</Command> <ProfileList> <profile profile_id="**Default Profile**" active="**true**" linkProfile="**1**" populationEst="**50**" sessionNo="**0**" target="**2**" queryAlgorithm="**DynamicQ**" reflectedPowerThreshold="**24.0**" tagModel="**ANY**" retry="**0**" tagFocus="**false**" fastId="**false**" minOnChipRSSI="**16**" maxOnChipRSSI="**21**" moistAvgWindow="**5**" tempAvgWindow="**5**" reconfigAntennaPortError="**false**" retryErrorAntennaPortError="**0**" antenna_port="**1,2,3,4**" transmitPower="**30.0**" transmitPower1="**30.0**" transmitPower2="**30.0**" transmitPower3="**30.0**" transmitPower4="**30.0**" transmitPower5="**30.0**" transmitPower6="**30.0**" transmitPower7="**30.0**" transmitPower8="**30.0**" transmitPower9="**30.0**" transmitPower10="**30.0**" transmitPower11="**30.0**" transmitPower12="**30.0**" transmitPower13="**30.0**" transmitPower14="**30.0**" transmitPower15="**30.0**" transmitPower16="**30.0**" dwellTime1="**2000**" dwellTime2="**2000**" dwellTime3="**2000**" dwellTime4="**2000**" dwellTime5="**2000**" dwellTime6="**2000**" dwellTime7="**2000**" dwellTime8="**2000**"

		dwellTime9="**2000**" dwellTime10="**2000**" dwellTime11="**2000**" dwellTime12="**2000**" dwellTime13="**2000**" dwellTime14="**2000**" dwellTime15="**2000**" dwellTime16="**2000**" *memoryBank1="**Bank2**"* *memoryBank1Offset="0"* *memoryBank1Length="**2**"* *memoryBank2="**Bank3**"* *memoryBank2Offset="0"* *memoryBank2Length="**2**" />* </ProfileList> </CSL>
7.11	session_id=<login_session_id>& command=***delOperProfile***& profile_id= *profile_id*	Remove operation profile. e.g. session_id=<login_session_id>&command=delOperProfile&server_id=ExampleProfile result: <?xml version="1.0" ?> <CSL> <Command>**delOperProfile**</Command> <Ack>**OK:**</Ack> </CSL>
7.12	session_id=<login_session_id>& command=***setRFLNAIFLNAGain***& rf_lna_compression_mode=*rf_lna_compression_mode*& rf_lna_gain=*rf_lna_gain*& if_lna_gain=*if_lna_gain*& agc_gain=*agc_gain*	Set RF LNA Gain and IF LNA Gain settings. e.g. session_id=<login_session_id>&command=setRFLNAIFLNAGain&rf_lna_compression_mode=1&rf_lna_gain=1dB&if_lna_gain=24dB&agc_gain=-6dB Valid attributes: rf_lna_compression_mode : 0, 1 (this must be 0 if rf_lna_gain is 13dB) rf_lna_gain : 1dB, 7dB, 13dB if_lna_gain : 24dB, 18dB, 12dB, 6dB agc_gain : -12dB, -6dB, 0dB, 6dB result:

		<?xml version="1.0" ?> <CSL> <Command>**setRFLNAIFLNAGain**</Command> <Ack>**OK:**</Ack> </CSL>
7.13	session_id=<login_session_id>& command=***getRFLNAIFLNAGain***	Get RF LNA Gain and IF LNA Gain settings. e.g. session_id=<login_session_id>&command=getRFLNAIFLNAGain result: <?xml version="1.0" ?> <CSL> <Command>**getRFLNAIFLNAGain**</Command> <Settings rf_lna_compression_mode="**1**" rf_lna_gain="**1dB**" if_lna_gain="**24dB**" agc_gain="**-6dB**" /> </CSL>
7.14	session_id=<login_session_id>& command=***getRAMMemory***	Get system memory information. e.g. session_id=<login_session_id>&command=getRAMMemory result: <?xml version="1.0" ?> <CSL> <Command>**getRAMMemory**</Command> <SystemMemory> <Total>**64638976**</Total> <Used>**50401280**</Used> <Free>**14237696**</Free> </SystemMemory> </CSL>
7.15	session_id=<login_session_id>& command=***getFlashMemory***	Get flash memory information. e.g. session_id=<login_session_id>&command=getFlashMemory result: <?xml version="1.0" ?> <CSL> <Command>**getFlashMemory**</Command> <FlashMemory>

		<Total>**4194304**</Total> <Used>**3212000**</Used> <Free>**982304**</Free> </FlashMemory> </CSL>
7.16	command=***getReaderStatus***& username=*username*& password=*password*	Get the reader run-time status for inspection without login the reader. e.g. http://192.168.25.160/API?command=getReaderStatus&username=root&password=csl result: <?xml version="1.0" ?> <CSL> <Command>**getReaderStatus**</Command> <Model name="**CS463-2**" protocol="**EPC Class1 Gen 2**" /> <Reader desc="**CS463 Demo Reader**" reader_id="**Demo Reader**" reader_serial_number="**ABC0123456789**" pcb_serial_number="**DEF9876543210024**" /> <ReaderVersion cs108_bluetooth_api_library="**1.0.2**" cs461_low_level_api_mach1_library=" **1.0.4**" csl_unified_api_library="**1.0.3**" java=" **1.8.0_221**" jni_library="**1.0.4**" llrp_library="**1.0.7**" os=" **Linux v4.14.78-imx_4.14.78_1.0.0_ga+g94da7bd**" pcb_version="**2.4**" rfid_firmware="**2.6.29**" web_application="**1.1.9**" /> <Timezone daylight_saving="**0**" tz="**GMT+08:00**" /> <Logout time="**30**" unit="**minute**" /> <UserStatus client_ip="**192.168.25.126**" login_status="**yes**" session_id="**00000000**" username="**root**" /> <AccessMode mode="**1**" name="**HTTP/XML**" /> <ActiveOperationProfile antenna_power="**1:30.0,2:30.0,3:30.0,4:30.0,**" profile_id="**Default Profile**" /> <ActiveEventList> <Event desc="**Event Demo**" event_id="**DemoEvent**" />

		<Event desc="" event_id="**e45**" /> </ActiveEventList> <CurrentLocalTime day="**9**" hour="**15**" minute="**41**" month="**3**" second="**9**" year="**2020**" /> <CurrentUTCTime day="**9**" hour="**7**" minute="**41**" month="**3**" second="**9**" year="**2020**" /> </CSL>
7.17	command=***healthCheck***& username=*username*& password=*password*	Make a health check of the reader without login the reader first. e.g. http://192.168.25.160/API?command=healthCheck&username=root&password=csl result: <?xml version="1.0" ?> <CSL> <Command>**healthCheck**</Command> <result checkTime="**Mon Mar 9 15:43:20 2020**" freeRAM="**15416368**" upTime="**16.25**" /> </CSL>
7.18	session_id=<login_session_id>& command=***getReaderError***	Get the unresolved errors from the reader. e.g. http://192.168.25.160/API?session_id=f13b3074&command=getReaderError result: <?xml version="1.0" ?> <CSL> <Command>**getReaderError**</Command> <readerError error_code="**0309**" desc=" **Reverse Power Too High – may be antenna mismatch**" antennaPort="**2**" reflected_power="**26**" reflected_power_threshold="**24**" upTime="**16.25**" /> <readerError error_code="**0309**" desc=" **Reverse Power Too High – may be antenna mismatch**" antennaPort="**3**" reflected_power="**27**" reflected_power_threshold="**24**" upTime="**16.25**" /> </CSL>

7.19	session_id=<login_session_id>& command=***restartSystem***	Reboot the system. e.g. http://192.168.25.160/API?session_id=f13b3074&command=restartSystem result: <?xml version="1.0" ?> <CSL> <Command>**restartSystem**</Command> <Ack>**OK:**</Ack> </CSL>
7.20	session_id=<login_session_id>& command=***isOnline***	To check if the reader is on-line. e.g. http://192.168.25.160/API?session_id=f13b3074&command=isOnline result: <?xml version="1.0" ?> <CSL> <Command>**isOnline**</Command> <Ack>**OK: Online,CS463.**</Ack> </CSL>
7.21.	session_id=<login_session_id>& command=***getScheduledRestart***	Get the scheduled restart settings e.g. http://192.168.25.160/API?session_id=f13b3074&command=getScheduledRestart result: <?xml version="1.0" ?> <CSL> <Command>**getScheduledRestart**</Command> <ScheduledRebootList> <scheduleReboot enable="**false**" mode="**Monday**" month="" day="" time1="**17:47**" time2="" />

		</ScheduledRebootList> </CSL>
7.22	session_id=<login_session_id>& command=***setScheduledRestart*** &mode=*mode* &enable=*enable* [&month=*month*] [&day=*day*] &time1=*time1* [&time2=*time2*]	The system may be scheduled to restart. Valid attributes : mode : ANNUAL, SEMI_ANNUAL, QUARTERLY, BI_MONTHLY, MONTHLY, SUNDAY, MONDAY, TUESDAY, WEDNESDAY, THURSDAY, FRIDAY, SATURDAY, DAILY, TWICE_PER_DAY enable : true, false month : 1 – 12 for mode ANNUAL 1 – 6 for mode SEMI_ANNUAL 1 – 3 for mode QUARTERLY 1 or 2 for mode BI_MONTHLY Not required for other mode day : 1 – 31, only required for mode ANNUAL, SEMI_ANNUAL, QUARTERLY, BI_MONTHLY, MONTHLY time1 : hh:mm (hh is hour in 24 hour format, mm is minute) time2 : hh:mm (hh is hour in 24 hour format, mm is minute), only required for mode TWICE_PER_DAY e.g. http://192.168.25.160/API?session_id=f13b3074&command=setScheduledRestart&mode=MONDAY&enable=true&time1=23:00 result: if successful, <?xml version="1.0" ?> <CSL> <Command>**setScheduledRestart**</Command> <Ack>**OK:**</Ack> </CSL> if fail (the example is wrong mode), <?xml version="1.0" ?>

		<CSL> <Command>**setScheduledRestart**</Command> <Error code="**-10**" msg="**Error: mode is not valid**" /> </CSL>
7.23	session_id=<login_session_id>& command=***addPowerUpNotification*** &notification_id=*notification_id* &type=*type* &server_id=*server_id* &data_format_id=*data_format_id* &enable=*enable*	Add Power Up Notification to be sent to server. e.g. session_id=<login_session_id>&command=addPowerUpNotification&notification_id=Example Power Up Notification&type=HTTP POST&server_id=Example CSL Demo Cloud Server&data_format_id=Example Power Up Notification Data Format&enable=true Valid attributes : type : HTTP POST, MQTT enable : true, false result: <?xml version="1.0" ?> <CSL> <Command>**addPowerUpNotification**</Command> <Ack>**OK:**</Ack> </CSL>
7.24	session_id=<login_session_id>& command=***modPowerUpNotification*** &notification_id=*notification_id* &type=*type* &server_id=*server_id* &data_format_id=*data_format_id* &enable=*enable*	Modify Power Up Notification to be sent to server. e.g. session_id=<login_session_id>&command=modPowerUpNotification&notification_id=Example Power Up Notification&type=HTTP POST&server_id=Example CSL Demo Cloud Server&data_format_id=Example Power Up Notification Data Format&enable=false Valid attributes : type : HTTP POST, MQTT enable : true, false

		result: <?xml version="1.0" ?> <CSL> <Command>**modPowerUpNotification** </Command> <Ack>**OK:**</Ack> </CSL>
7.25	session_id=<login_session_id>& command=***delPowerUpNotification*** &notification_id=*notification_id*	Remove Power Up Notification. e.g. session_id=<login_session_id>&command=delPowerUpNotification&notification_id=Example Power Up Notification result: <?xml version="1.0" ?> <CSL> <Command>**delPowerUpNotification** </Command> <Ack>**OK:**</Ack> </CSL>
7.26	session_id=<login_session_id>& command=***listPowerUpNotification***	List Power Up Notification. e.g. session_id=<login_session_id>&command=listPowerUpNotification result: <?xml version="1.0" ?> <CSL> <Command>**listPowerUpNotification** </Command> <PowerUpNotificationList> <notification data_format_id="**Example Power Up Notification Data Format**" enable="**false**" notification_id="**Example Power Up Notification**" server_id="**Example CSL Demo Cloud Server**" type="**HTTP POST**" /> <notification data_format_id="**Example Power Up Notification Data Format**" enable="**false**" notification_id="**Example Power Up Notification to MQTT Broker**"

		server_id="**Example MQTT Broker**" type="**MQTT**" /> </PowerUpNotificationList> </CSL>
7.27	session_id=<login_session_id>& command=***addReaderErrorNotification*** &notification_id=*notification_id* &type=*type* &server_id=*server_id* &data_format_id=*data_format_id* &enable=*enable*	Add Reader Error Notification to be sent to server. e.g. session_id=<login_session_id>&command=addReaderErrorNotification&notification_id=Example Reader Error Notification&type=HTTP POST&server_id=Example CSL Demo Cloud Server&data_format_id=Example Reader Error Notification Data Format&enable=true Valid attributes : type : HTTP POST, MQTT enable : true, false result: <?xml version="1.0" ?> <CSL> <Command>**addReaderErrorNotification** </Command> <Ack>**OK:**</Ack> </CSL>
7.28	session_id=<login_session_id>& command=***modReaderErrorNotification*** &notification_id=*notification_id* &type=*type* &server_id=*server_id* &data_format_id=*data_format_id* &enable=*enable*	Modify Reader Error Notification to be sent to server. e.g. session_id=<login_session_id>&command=modReaderErrorNotification&notification_id=Example Reader Error Notification&type=HTTP POST&server_id=Example CSL Demo Cloud Server&data_format_id=Example Reader Error Notification Data Format&enable=false Valid attributes : type : HTTP POST, MQTT enable : true, false result: <?xml version="1.0" ?>

		<CSL> <Command>**modReaderErrorNotification** </Command> <Ack>**OK:**</Ack> </CSL>
7.29	session_id=<login_session_id>& command=***delReaderErrorNotification*** &notification_id=*notification_id*	Remove Reader Error Notification. e.g. session_id=<login_session_id>&command=delReaderErrorNotification&notification_id=Example Reader Error Notification result: <?xml version="1.0" ?> <CSL> <Command>**delReaderErrorNotification** </Command> <Ack>**OK:**</Ack> </CSL>
7.30	session_id=<login_session_id>& command=***listReaderErrorNotification***	List Reader Error Notification. e.g. session_id=<login_session_id>&command=listReaderErrorNotification result: <?xml version="1.0" ?> <CSL> <Command>**listReaderErrorNotification** </Command> <ReaderErrorNotificationList> <notification data_format_id="**Example Reader Error Notification Data Format**" enable="**false**" notification_id="**Example Reader Error Notification**" server_id="**Example CSL Demo Cloud Server**" type="**HTTP POST**" /> <notification data_format_id="**Example Reader Error Notification Data Format**" enable="**false**" notification_id="**Example Reader Error Notification to MQTT Broker**" server_id="**Example MQTT Broker**" type="**MQTT**" /> </ReaderErrorNotificationList>

		</CSL>
7.31	session_id=<login_session_id>& command=***addGPIInterruptNotification*** &notification_id=*notification_id* &interrupt_type=*interrupt_type* &gpi_port=*gpi_port* &type=*type* &server_id=*server_id* &data_format_id=*data_format_id* &enable=*enable*	Add GPI Interrupt Notification to be sent to server. e.g. session_id=<login_session_id>&command=addGPIInterruptNotification&notification_id=Example GPI Interrupt Notification&interrupt_type=Rising Edge&gpi_port=1&type=HTTP POST&server_id=Example CSL Demo Cloud Server&data_format_id=Example GPI Interrupt Notification Data Format&enable=true Valid attributes : interrupt_type : Rising Edge, Falling Edge, Both gpi_port : 1, 2, 3 or 4 type : HTTP POST, MQTT enable : true, false result: <?xml version="1.0" ?> <CSL> <Command>**addGPIInterruptNotification**</Command> <Ack>**OK:**</Ack> </CSL>
7.32	session_id=<login_session_id>& command=***modGPIInterruptNotification*** &notification_id=*notification_id* &interrupt_type=*interrupt_type* &gpi_port=*gpi_port* &type=*type* &server_id=*server_id* &data_format_id=*data_format_id* &enable=*enable*	Modify GPI Interrupt Notification to be sent to server. e.g. session_id=<login_session_id>&command=modGPIInterruptNotification&notification_id=Example GPI Interrupt Notification&interrupt_type=Falling Edge&gpi_port=1&type=HTTP POST&server_id=Example CSL Demo Cloud Server&data_format_id=Example GPI Interrupt Notification Data Format&enable=false Valid attributes : interrupt_type : Rising Edge, Falling Edge, Both gpi_port : 1, 2, 3 or 4 type : HTTP POST, MQTT

		enable : true, false result: <?xml version="1.0" ?> <CSL> <Command>**modGPIInterruptNotification** </Command> <Ack>**OK:**</Ack> </CSL>
7.33	session_id=<login_session_id>& command=***delGPIInterruptNotification*** &notification_id=*notification_id*	Remove GPI Interrupt Notification. e.g. session_id=<login_session_id>&command=delGPIInterruptNotification&notification_id=Example GPI Interrupt Notification result: <?xml version="1.0" ?> <CSL> <Command>**delGPIInterruptNotification** </Command> <Ack>**OK:**</Ack> </CSL>
7.34	session_id=<login_session_id>& command=***listGPIInterruptNotification***	List GPI Interrupt Notification. e.g. session_id=<login_session_id>&command=listGPIInterruptNotification result: <?xml version="1.0" ?> <CSL> <Command>**listGPIInterruptNotification** </Command> <GPIInterruptNotificationList> <notification data_format_id="**Example GPI Interrupt Notification Data Format**" enable="**false**" gpi_port="**1**" interrupt_type="**Falling Edge**" notification_id="**Example GPI Interrupt Notification**" server_id="**Example CSL Demo Cloud Server**" type="**HTTP POST**" /> </GPIInterruptNotificationList> </CSL>

7.35	session_id=<login_session_id>& command=***configurationBackup***	Backup reader configuration. e.g. session_id=<login_session_id>&command=configurationBackup result: <?xml version="1.0" ?> <CSL> <Command>**configurationBackup**</Command> <Configuration>***{configuration in Json format}***</Configuration> </CSL>
7.36	session_id=<login_session_id>& command=***configurationRestore*** & configuration=*configuration*	Restore reader configuration. e.g. session_id=<login_session_id>&command=configurationRestore&configuration=*{configuration in Json format}* Valid attributes : configuration : configuration of reader in Json format result: <?xml version="1.0" ?> <CSL> <Command>**configurationRestore**</Command> <Ack>**OK: Please wait a moment... System restarting...**</Ack> </CSL>
7.37	session_id=<login_session_id>& command=***addHeartBeat*** &heart_beat_id=*heart_beat_id* &type=*type* &interval=*interval* &enable=*enable* [&address=*address*] [&server_id=*server_id*] [&data_format_id=*data_format_id*] [&enableReset=*enableReset*]	Add Heart Beat to be sent to server. e.g. session_id=<login_session_id>&command=addHeartBeat&heart_beat_id=Heart Beat to Demo Cloud Server&type=HTTP POST&interval=60&server_id=Example CSL Demo Cloud Server&data_format_id=Example Heart Beat Data Format&enableReset=true&resetPort=ethernet&tryBeforeReset=5&enable=true Valid attributes :

	[&resetPort=*resetPort*] [&tryBeforeReset=*tryBeforeReset*]	type : ICMP Ping, HTTP POST, MQTT, arp, arp Gateway interval : unit=second, 30 – 86400 enable : true, false address : required if type is ICMP Ping server_id : required if type is HTTP POST or MQTT data_format_id : required if type is HTTP POST or MQTT enableReset: required if type is not arp resetPort : ethernet, wifi or both, required if enableReset is true tryBeforeReset : 1 – 10, required if enableReset is true result: <?xml version="1.0" ?> <CSL> <Command>**addHeartBeat**</Command> <Ack>**OK:**</Ack> </CSL>
7.38	session_id=<login_session_id>& command=***modHeartBeat*** &heart_beat_id=*heart_beat_id* &type=*type* &interval=*interval* &enable=*enable* [&address=*address*] [&server_id=*server_id*] [&data_format_id=*data_format_id*] [&enableReset=*enableReset*] [&resetPort=*resetPort*] [&tryBeforeReset=*tryBeforeReset*]	Modify Heart Beat to be sent to server. e.g. session_id=<login_session_id>&command=modHeartBeat&heart_beat_id=Heart Beat to Demo Cloud Server&type=HTTP POST&interval=60&server_id=Example CSL Demo Cloud Server&data_format_id=Example Heart Beat Data Format&enableReset=true&resetPort=ethernet&tryBeforeReset=5&enable=false Valid attributes : type : ICMP Ping, HTTP POST, MQTT, arp, arp Gateway interval : unit=second, 30 – 86400 enable : true, false address : required if type is ICMP Ping server_id : required if type is HTTP POST or MQTT data_format_id : required if type is HTTP POST or MQTT enableReset: required if type is not arp resetPort : ethernet, wifi or both, required if enableReset is true tryBeforeReset : 1 – 10, required if enableReset is true

		result: <?xml version="1.0" ?> <CSL> <Command>**modHeartBeat**</Command> <Ack>**OK:**</Ack> </CSL>
7.39	session_id=<login_session_id>& command=***delHeartBeat*** &heart_beat_id=*heart_beat_id*	Remove Heart Beat. e.g. session_id=<login_session_id>&command=delHeartBeat&hea rt_beat_id=Heart Beat to Demo Cloud Server result: <?xml version="1.0" ?> <CSL> <Command>**delHeartBeat**</Command> <Ack>**OK:**</Ack> </CSL>
7.40	session_id=<login_session_id>& command=***listHeartBeat***	List Heart Beat. e.g. session_id=<login_session_id>&command=listHeartBeat result: <?xml version="1.0" ?> <CSL> <Command>**listHeartBeat**</Command> <HeartBeatList> <heartbeat address="" data_format_id="**Example Heart Beat Data Format**" enable="**true**" enableReset="**false**" heart_beat_id="**Heart Beat to Demo Cloud Server**" interval="**60**" resetPort="**ethernet**" server_id="**Example CSL Demo Cloud Server**" tryBeforeReset="**5**" type="**HTTP POST**" /> <heartbeat address="" data_format_id="" enable="**true**" enableReset="**true**" heart_beat_id="**ARPING of Local Gateway**"

		interval="**30**" resetPort="**ethernet**" server_id="" tryBeforeReset="**5**" type="**arping Gateway**" /> </ReaderErrorNotificationList> </CSL>
7.41	session_id=<login_session_id>& command=***uploadFile***& fileName=*fileName*	Upload file to the reader. Here below is an example showing how to upload the SSL certificate that will be used for Secure Web Access via HTTP POST protocol written in C# (printed in blue color). HttpClient client = new HttpClient(); var stream = new FileStream("C:\\temp\\certificate.pem", FileMode.Open); var content = new StreamContent(stream); var requestUri = "http://192.168.25.160/API?session_id=a33219dc&command=uploadFile&fileName=certificate.pem"; var response = await client.PostAsync(requestUri, content); result: <?xml version="1.0" ?> <CSL> <Command>**uploadFile**</Command> <Ack>**OK:**</Ack> </CSL>
7.42	session_id=<login_session_id>& command=***setSecureWebAccess*** &useSelfSignedCert=*useSelfSignedCert* [&certFile=*certFile*] [&keyFile=*keyFile*] [&keyPassword=*keyPassword*]	Configure to use HTTP or HTTPS for accessing the web interface of the reader and the SSL certificate to be used for HTTPS. e.g. 1 (use the ex-factory self-signed certificate and key for HTTPS) session_id=<login_session_id>&command=setSecureWebAccess&useSelfSignedCert=true e.g. 2 (use the client provided certificate and private key for

		HTTPS, the certificate and key files must be uploaded to the reader first by using the ***uploadFile*** command, the certificate and key files must be in PEM format, keyPassword must be provided if the key file is encrypted) session_id=<login_session_id>&command=setSecureWebAccess&useSelfSignedCert=false&certFile=cert.pem&keyFile=key.pem e.g. 3 (use HTTP for web access) session_id=<login_session_id>&command=setSecureWebAccess&useSelfSignedCert=false&certFile=&keyFile= result: <?xml version="1.0" ?> <CSL> <Command>**setSecureWebAccess**</Command> <Ack>**OK:**</Ack> </CSL>
7.43	session_id=<login_session_id>&command=***getSecureWebAccess***	Get the configuration of Secure Web Access. e.g. session_id=<login_session_id>&command=getSecureWebAccess result: <?xml version="1.0" ?> <CSL> <Command>**getSecureWebAccess**</Command> <SecureWebAccess certFile="" keyFile="" keyPassword="" useSelfSignedCert="false" /> </CSL>
7.44	session_id=<login_session_id>&command=***setTwoWayAuth*** &certFile=*certFile* &keyFile=*keyFile* &keyPassword=*keyPassword*	Configure the certificate, private key files and key password of the reader to be used for HTTPS two way authentication. e.g. 1 (the certificate and key files must be uploaded to the reader first by using the ***uploadFile*** command, the

		certificate and key files must be in PEM format, keyPassword must be provided if the key file is encrypted) session_id=<login_session_id>&command=setTwoWayAuth&certFile=cert.pem&keyFile=key.pem e.g. 2 (remove the certificate and key files from the reader) session_id=<login_session_id>&command=setTwoWayAuth&certFile= &keyFile= result: <?xml version="1.0" ?> <CSL> <Command>**setTwoWayAuth**</Command> <Ack>**OK:**</Ack> </CSL>
7.45	session_id=<login_session_id>&command=***getTwoWayAuth***	Get the configuration of Two Way Authentication. e.g. session_id=<login_session_id>&command=getTwoWayAuth result: <?xml version="1.0" ?> <CSL> <Command>**getTwoWayAuth**</Command> <SecureWebAccess certFile="" keyFile="" keyPassword="" /> </CSL>

8. Network Management

	query_string	Description
Network Management		
8.1	session_id=<login_session_id>& command=***setNetworkConfig***& type=*type*& [enable=*enable*&] dhcpmode=*dhcpmode* [&ip=*ip* &mask=*mask* &gateway=*gateway*] [&dns_server1=*dns_server1*] [&dns_server2=*dns_server2*] [&security=*security*] [&ssid=*ssid* &psk=*psk*]	Set Network Properties of the following setting: Type, DHCP Mode, IP Address, Subnet Mask, Default Gateway, DNS Server 1 and DNS Server 2 e.g.1 http://192.168.25.160/API?session_id=a33219dc&command=setNetworkConfig&type=ethernet&dhcpmode=1 e.g.2 http://192.168.25.160/API?session_id=a33219dc&command=setNetworkConfig&type=wifi&dhcpmode=0&ip=192.168.25.102&mask=255.255.255.0&gateway=192.168.25.1&dns_server1=192.168.25.2&dns_server2=8.8.8.8&security=wpa-psk&ssid=TestAP&psk=password Valid attributes : type : ethernet, wifi enable : true or false, only valid for type wifi dhcpmode : 0 = Static IP 1 = DHCP Mode security : none or wpa-psk, required for type wifi result: <?xml version="1.0" ?> <CSL> <Command>**setNetworkConfig**</Command> <Ack>**OK:**</Ack> </CSL>
8.2	session_id=<login_session_id>& command=***getNetworkConfig***	Get Network Properties such as IP Address, Subnet Mask, Default Gateway, MAC Address.

<table>
<tr><td></td><td></td><td>e.g.
session_id=<login_session_id>&command=getNetworkConfig

result:
<?xml version="1.0" ?>
<CSL>
<Command>getNetworkConfig</Command>
<NetworkConfigList>
<NetworkConfig type="ethernet"
dhcpmode="0"
gateway="192.168.25.1"
ip="192.168.25.248"
mask="255.255.255.0"
MAC="00:0D:60:A5:8F:E3" />
<NetworkConfig type="wifi"
enable="true"
dhcpmode="1"
gateway="192.168.25.1"
ip="192.168.25.238"
mask="255.255.255.0"
MAC="00:0D:60:34:56:78"
security="wpa-psk"
ssid="TestAP" />
</NetworkConfigList>
</CSL></td></tr>
<tr><td>8.3</td><td>session_id=<login_session_id>&
command=setServerID&
server_id=server_id&
desc=desc&
type=type&
server_ip=server_ip
[&server_port=server_port]
[&client_id=client_id]
[&username =username]
[&password=password]
[&enable_ssl=enable_ssl]
[&ssl_version=ssl_version]
[&two_way_authentication=two_way_authentication]
[&topic=topic]
[&clean_session=clean_session]
[&qos =qos]</td><td>Set host notification url and port to be communicated with.

e.g.1 Example CSL Demo Cloud Server

http://192.168.25.160/API?session_id=a33219dc&command=setServerID&server_id=Example CSL Demo Cloud Server&desc=Demo Http Cloud Server&type=HTTP&server_ip=https://democloud.convergence.com.hk:29090/WebServiceRESTs/1.0/reg/create-update-delete/update-entity/tagdata

e.g.2 Example TCP Server

http://192.168.25.160/API?session_id=a33219dc&command=setServerID&server_id=Demo TCP Server&desc=Demo TCP Server&type=TCP&server_ip=192.168.25.100&server_port=9090

e.g.2 Example MQTT Server</td></tr>
</table>

		http://192.168.25.160/API?session_id=a33219dc&command=setServerID&server_id=Demo_MQTT_Server&desc=Demo_MQTT_Server&type=MQTT&server_ip=test.mosquitto.org&server_port=8883&enable_ssl=true&ssl_version=TLSv1.2&two_way_authentication=false&topic=csl/tagdata&clean_session=true&qos=0 Valid attributes : type : HTTP, TCP, MQTT server_port : required if type is TCP or MQTT client_id : optional depends on MQTT server username : optional depends on MQTT server password : optional depends on MQTT server enable_ssl : true, false, required if type is MQTT ssl_version : TLSv1.2, TLSv1.1, TLSv1, SSLv3, SSLv2, required if enable_ssl is true two_way_authentication : true, false, required if type is enable_ssl is true clean_session : true, false, required if type is MQTT qos : 0 (at most once), 1 (at least once), 2 (exactly once), required if type is MQTT result: <?xml version="1.0" ?> <CSL> <Command>**setServerID**</Command> <Ack>**OK:**</Ack> </CSL>
8.4	session_id=<login_session_id>& command=***modServerID***& server_id=*server_id*& desc=*desc*& type=*type*& server_ip=*server_ip* [&server_port=*server_port*] [&client_id=*client_id*] [&username =*username*]	Modify host notification url and port to be communicated with. Valid attributes : type : HTTP, TCP, MQTT server_port : required if type is TCP client_id : optional depends on MQTT server username : optional depends on MQTT server password : optional depends on MQTT server

	[&password=*password*] [&enable_ssl=*enable_ssl*] [&ssl_version=*ssl_version*] [&two_way_authentication=*two_way_authentication*] [&topic=*topic*] [&clean_session=*clean_session*] [&qos =*qos*]	enable_ssl : true, false, required if type is MQTT ssl_version : TLSv1.2, TLSv1.1, TLSv1, SSLv3, SSLv2, required if enable_ssl is true two_way_authentication : true, false, required if type is enable_ssl is true clean_session : true, false, required if type is MQTT qos : 0 (at most once), 1 (at least once), 2 (exactly once), required if type is MQTT result: <?xml version="1.0" ?> <CSL> <Command>**modServerID**</Command> <Ack>**OK:**</Ack> </CSL>
8.5	session_id=<login_session_id>& command=***delServerID***& server_id= *server_id*	Remove server from the server list. e.g. session_id=<login_session_id>&command=delServerID&server_id=DemoServer result: <?xml version="1.0" ?> <CSL> <Command>**delServerID**</Command> <Ack>**OK:**</Ack> </CSL>
8.6	session_id=<login_session_id>& command=***listServer***	List server table. e.g. session_id=<login_session_id>&command=listServer result: <?xml version="1.0" ?> <CSL> <Command>**listServer**</Command> <ServerList> <Server desc="**Demo Http Cloud Server**" server_id="**Examp CSL Demo Cloud Server**" server_ip="**https://democloud.convergence.com.hk:29090/WebServiceRESTs/1.0**

		/req/create-update-delete/update-entity/tagdata" server_port="" type="**HTTP**" /> <Server desc="**Demo TCP Server**" server_id="**Demo TCP Server**" server_ip="**192.168.25.100**" server_port="**9090**" type="**TCP**" /> </ServerList> </CSL>

8.7	session_id=<login_session_id>&
	command=***setServerCertificate***&
	server_id=*server_id*&
	serverCertFile=*serverCertFile*
		Upload the SSL certificate of the specified server. The certificate must be in PEM format.
		
		Here below is an example showing how to upload the SSL certificate via HTTP POST protocol written in C# (printed in blue color).
		
		HttpClient client = new HttpClient();
		var stream = new FileStream("C:\\temp\\certificate.pem", FileMode.Open);
		var content = new StreamContent(stream);
		var requestUri = "http://192.168.25.160/API?session_id=a33219dc&command=setServerCertificate&server_id=Demo MQTT Server&serverCertFile=certificate.pem";
		var response = await client.PostAsync(requestUri, content);
		
		result:
		<?xml version="1.0" ?>
		<CSL>
		<Command>**setServerCertificate**</Command>
		<Ack>**OK:**</Ack>
		</CSL>

8.8	session_id=<login_session_id>& command=***delServerCertificate***& server_id=*server_id*	Remove the SSL certificate of the specified server. e.g. session_id=<login_session_id>&command=delServerCertificat

		e&server_id=Demo MQTT Server result: <?xml version="1.0" ?> <CSL> <Command>**delServerCertificate**</Command> <Ack>**OK:**</Ack> </CSL>
8.9	session_id=<login_session_id>& command=***addDataFormat***& data_format_id=*data_format_id* & desc=*desc*& format=*format* &field{m}=*field* &label{m}=*label* [&tagDataField{n}=*tagDataFiel* *d* &tagDataLabel{n}=*tagDataLabe* *l*]	Add data format of packet to be sent to server. e.g. session_id=<login_session_id>&command=addDataFormat&d ata_format_id=ExampleDataFormat&desc=Example Data Format&format=JSON&field1=RFIDReaderName&label1=rfi dReaderName&field2=EthernetMACAddress&label2=ethernet MACAddress&field3=NumberOfTags&label3=numberOfTags &field4=TagDataList&label4=tags&tagDataField1=EPC&tag DataLabel1=epc&tagDataField2=AntennaPort&tagDataLabel2 =antennaPort&tagDataField3=RSSI&tagDataLabel3=rssi&tag DataField4=TimeOfRead&tagDataLabel4=time Valid attributes : format : JSON, XML, CSV field : SequenceNumber, NumberOfTags, TagDataList, RFIDReaderName, RFIDReaderSerialNumber, RFIDReaderInternalSerialNumber, EthernetMACAddress, WiFiMACAddress, EthernetMACAddressWithColon, WiFiMACAddressWithColon, HeartBeatFlag, PowerUpFlag, ReaderErrorFlag, ReaderErrorCode,

		ReaderErrorDescription, ReaderErrorAntennaPort, ReaderErrorReflectedPower, ReaderErrorReflectedPowerThreshold, TimeOfHeartBeat, TimeOfPowerUp, TimeOfReaderError, TimeStampOfHeartBeat, TimeStampOfPowerUp, TimeStampOfReaderError, TimeZone tagDataField (effective only if field TagDataList exists) : PC, EPC, TidBank, UserBank, TimeOfRead, TimeStampOfRead, TimeZone, AntennaPort, AntennaPort_Number, RSSI, RSSI_Number, Frequency, Phase, EventId, HeartBeatFlag, PowerUpFlag, ReaderErrorFlag, ReaderErrorCode, ReaderErrorDescription, ReaderErrorAntennaPort, ReaderErrorReflectedPower, ReaderErrorReflectedPowerThrehold, TimeOfHeartBeat, TimeOfPowerUp,

		TimeOfReaderError, TimeStampOfHeartBeat, TimeStampOfPowerUp, TimeStampOfReaderError result: <?xml version="1.0" ?> <CSL> <Command>**addDataFormat**</Command> <Ack>**OK:**</Ack> </CSL>
8.10	session_id=<login_session_id>& command=***modDataFormat***& data_format_id=*data_format_id* & [desc=*desc*&] format=*format* &field{m}=*field* &label{m}=*label* [&tagDataField{n}=*tagDataFiel* *d* &tagDataLabel{n}=*tagDataLabe* *l*]	Modify data format of packet to be sent to server. e.g. session_id=<login_session_id>&command=modDataFormat& data_format_id=ExampleDataFormat&desc=Example Data Format&format=XML&field1=RFIDReaderName&label1=rfi dReaderName&field2=EthernetMACAddress&label2=ethernet MACAddress&field3=NumberOfTags&label3=numberOfTags &field4=TagDataList&label4=tags&tagDataField1=EPC&tag DataLabel1=epc&tagDataField2=AntennaPort&tagDataLabel2 =antennaPort&tagDataField3=RSSI&tagDataLabel3=rssi&tag DataField4=TimeStampOfRead&tagDataLabel4=timeStamp Valid attributes : format : JSON, XML, CSV field : SequenceNumber, NumberOfTags, TagDataList, RFIDReaderName, RFIDReaderSerialNumber, RFIDReaderInternalSerialNumber, EthernetMACAddress, WiFiMACAddress, EthernetMACAddressWithColon, WiFiMACAddressWithColon, HeartBeatFlag,

		PowerUpFlag, ReaderErrorFlag, ReaderErrorCode, ReaderErrorDescription, ReaderErrorAntennaPort, ReaderErrorReflectedPower, ReaderErrorReflectedPowerThreshold, TimeOfHeartBeat, TimeOfPowerUp, TimeOfReaderError, TimeStampOfHeartBeat, TimeStampOfPowerUp, TimeStampOfReaderError, TimeZone tagDataField (effective only if field TagDataList exists) : PC, EPC, TidBank, UserBank, TimeOfRead, TimeStampOfRead, TimeZone, AntennaPort, AntennaPort_Number, RSSI, RSSI_Number, Frequency, Phase, EventId, HeartBeatFlag, PowerUpFlag, ReaderErrorFlag, ReaderErrorCode, ReaderErrorDescription, ReaderErrorAntennaPort, ReaderErrorReflectedPower,

		ReaderErrorReflectedPowerThrehold, TimeOfHeartBeat, TimeOfPowerUp, TimeOfReaderError, TimeStampOfHeartBeat, TimeStampOfPowerUp, TimeStampOfReaderError result: <?xml version="1.0" ?> <CSL> <Command>**modDataFormat**</Command> <Ack>**OK:**</Ack> </CSL>
8.11	session_id=<login_session_id>& command=***delDataFormat***& data_format_id=*data_format_id*	Remove data format. e.g. session_id=<login_session_id>&command=delDataFormat&data_format_id=ExampleDataFormat result: <?xml version="1.0" ?> <CSL> <Command>**delDataFormat**</Command> <Ack>**OK:**</Ack> </CSL>
8.12	session_id=<login_session_id>& command=***listDataFormat***	List data format. e.g. session_id=<login_session_id>&command=listDataFormat result: <?xml version="1.0" ?> <CSL> <Command>**listDataFormat**</Command> <DataFormatList> <dataFormat data_format_id="**ExampleDataFormat**" desc="**Example Data Format**" field1="**RFIDReaderName**" field2="**EthernetMACAddress**"

		field3="**NumberOfTags**" field4="**TagDataList**" format="**XML**" label1="**rfidReaderName**" label2="**ethernetMACAddress**" label3="**numberOfTags**" label4="**tags**" tagDataField1="**EPC**" tagDataField2="**AntennaPort**" tagDataField3="**RSSI**" tagDataField4="**TimeStampOfRead**" tagDataLabel1="**epc**" tagDataLabel2="**antennaPort**" tagDataLabel3="**rssi**" tagDataLabel1="**timeStamp**" /> </DataFormatList> </CSL>

9. Time & Timer Management

	query_string	Description
Time and Timer Management		
9.1	session_id=<login_session_id>& command=***setDateTime***& Year=*year*&Month=*month*& Day=*day*&Hour=*hour*& Minute=*minute*&Second=*second*	Set system **UTC** date time as the parameters pass. Time is in the 24-hours format. e.g. session_id=<login_session_id>&command=setDateTime&Year=2020&Month=5&Day=1&Hour=15&Minute=32&Second=58
9.2	session_id=<login_session_id>& command=***setTimeZone***& time_zone=*time_zone*&dst=*dst*	Set Time Zone and Daylight Saving Time (DST). The setting can be read by calling getDateTime command. e.g. session_id=<login_session_id>&command=setTimeZone& time_zone=08:00&dst=0 Valid attributes : time_zone=<in hh:mm or -hh:mm format where hh=hour, mm=minute> dst=-1,0,1 result: <?xml version="1.0" ?> <CSL> <Command>**setTimeZone**</Command> <Ack>**OK:**</Ack> </CSL>
9.3	session_id=<login_session_id>& command=***getDateTime***	Get date/time in the format of asctime() (ANSI C). e.g. session_id=<login_session_id>&command=getDateTime

		result: <?xml version="1.0" ?> <CSL> <Command>**getDateTime**</Command> <DateTime>**Wed Nov 29 09:43:48 2020**</DateTime> <UTCDateTime>**Wed Nov 29 01:43:48 2020**</UTCDateTime> <TimeZone>**GMT+08:00**</TimeZone> <DaylightSavingTime>**0**</DaylightSavingTime> <UpTime>**325.23**</UpTime> </CSL>
9.4	session_id=<login_session_id>& command=***getNTP***	Get NTP server information. e.g. http://192.168.25.160/API?session_id=12AC12DE&command=getNTP result: <?xml version="1.0" ?> <CSL> <Command>**getNTP**</Command> <ntp enable="**true**" ip1="**207.46.130.100**" ip2="**pool.ntp.org**" mode="**Saturday**" time="**00:00**" /> </CSL>
9.5	session_id=<login_session_id>& command=***setNTP*** &ip1=***ip1*** &ip2=***ip2*** &mode=***mode*** &time=***time*** &enable=***enable*** [&immedidateUpdate=***immediateUpdate***]	Configure NTP server. e.g. http://192.168.25.160/API?session_id=12AC12DE&command=setNTP&ip1=207.46.130.100&ip2=pool.ntp.org&mode=Saturday&time=00:00&enable=true Valid attributes : ip1, ip2 : NTP server address in form of dot-notation xxx.xxx.xxx.xxx or valid URL mode : Every, Monday, Tuesday, Wednesday, Thursday, Friday, Saturday, Sunday time : in 24-hour form of hh:mm, e.g. 00:00, 23:59 enable : true,false immedidateUpdate=true (false by default), synchronize the date/time with time server immediately

		result: <?xml version="1.0" ?> <CSL> <Command>**setNTP**</Command> <Ack>**OK:**</Ack> </CSL>

10. Tag & Tag Filter Management

#	query_string	Description
10.1	session_id=<login_session_id>&command=newTagGroup&group_id=group_id	ADD TAGGROUP EMPTY GROUPID: Create new tag group without tag data
10.2	session_id=<login_session_id>&command=addTagGroupMember&group_id=group_id&tag_id=EPCID	ADD TAG GROUPID EPCID : Add one tag data to tag group
10.3	session_id=<login_session_id>&command=delTagGroupMember&group_id=group_id&tag_id=EPCID	DEL TAG GROUPID EPCID (delete one tag in tag group)
10.4	session_id=<login_session_id>&command=delTagGroupMemberAll&group_id=group_id	DELALL TAG GROUPID (delete all tag data in tag group)
10.5	session_id=<login_session_id>&command=importTagGroup&group_id=group_id&tagGroupContent=tagGroupContent	IMPORT TAG GROUPID EPCIDSTRING_CSV_FORMAT (add multiple new tag data to old tag group, new tag data in CSV format)
10.6.	http://<ip>/importTagGroupCSV session_id=<login_session_id>& tagGroupFilename=*tagGroupFilename*& tagGroupContent=*tagGroupContent*	create new Tag Group in reader and import data from CSV file in remote PC server. Here below is an example showing how to import CSV file via HTTP POST protocol written in C# (printed in blue color). HttpClient client = new HttpClient(); var str = File.ReadAllText("C:\\temp\\DemoGroup.csv"); var map = new Dictionary<string, string>

		{
		{ “session_id”, “a33219dc” },
		{ “tagGroupFilename”, “DemoGroup.csv” },
		{ “tagGroupContent”, str }
		}
		var content = new FormUrlEncodedContent(map);
		var requestUri =
		“http://192.168.25.160/importTagGroupCSV”;
		var response = await client.PostAsync(requestUri, content);
		
		result:
		<?xml version="1.0" ?>
		<CSL>
		<Command>**importTagGroupCSV**</Command>
		<Ack>**OK:**</Ack>
		</CSL>
		
		or
		
		<?xml version="1.0" ?>
		<CSL>
		<Command>**importTagGroupCSV**</Command>
		<Ack>**Error: Tag Group already existed. (Remark: Tag Group = <filename>)**</Ack>
		</CSL>

10.7	session_id=<login_session_id>& command=***delTagGroup***& group_id=*group_id*	Remove a tag group. e.g. session_id=<login_session_id>&command=delTagGroup &group_id=DemoGroup result: <?xml version="1.0" ?> <CSL> <Command>**delTagGroup**</Command> <Ack>**OK:**</Ack> </CSL>
10.8	session_id=<login_session_id>& command=***listTagGroup***	List tag group. e.g.

		session_id=<login_session_id>&command=listEvent result: <?xml version="1.0" ?> <CSL> <Command>**listTagGroup**</Command> > <TagGroupList> <tagGroup group_id="**DemoGroup**"> <tag id="**012345678901234567 89ABCD**" /> <tag id="**000000020090505095227234**" /> </tagGroup> </ TagGroupList> </CSL>
10.9	session_id=<login_session_id>& command=***setDatabaseConfiguration***& databasePath=*databasePath*	Set database configuration. e.g. http://192.168.25.160/API?session_id=a33219dc&command=setDatabaseConfiguration&databasePath=%2Frun%2Fmedia%2Fmmcblk2p5%2Fmysql result: <?xml version="1.0" ?> <CSL> <Command>**setDatabaseConfiguration** </Command> <Ack>**OK:** </Ack> </CSL>
10.10	session_id=<login_session_id>& command=***getDatabaseConfiguration***	Set database configuration. e.g. session_id=<login_session_id>&command=getDatabaseConfiguration result: <?xml version="1.0" ?> <CSL> <Command>**getDatabaseConfiguration**</Command> <Database path="**/run/media/mmcblk2p5/mysql**" /> </CSL>

10.11	session_id=<login_session_id>& command=***addDatabase***& databaseName=*databaseName* [&field{n}=*field* &dataType{n}=*datatype*]	Add database. e.g. http://192.168.25.160/API?session_id=a33219dc&command=addDatabase&databaseName=ProductDB&field1=EPC&dataType1=STRING&field2=ProductId&dataType2=STRING&field3=ProductName&dataType3=STRING&field4=ProductPrice&dataType4=NUMBER Valid attributes : dataType : STRING, NUMBER, IMAGE Note : A **STRING** dateType **EPC** field is always used as the primary key field in the database. If the HTTP message does not contain a **EPC** field, the system adds it. result: <?xml version="1.0" ?> <CSL> <Command>**addDatabase**</Command> <Ack>**OK:** </Ack> </CSL>
10.12	session_id=<login_session_id>& command=***modDatabase***& databaseName=*databaseName* [&field{n}=*field* &dataType{n}=*datatype*]	Modify database. e.g. http://192.168.25.160/API?session_id=a33219dc&command=modDatabase&databaseName=ProductDB&field1=EPC&dataType1=STRING&field2=ProductId&dataType2=STRING&field3=ProductName&dataType3=STRING&field4=ProductImage&dataType4=IMAGE&field5=ProductPrice&dataType5=NUMBER Valid attributes : dataType : STRING, NUMBER, IMAGE Note : A **STRING** dateType **EPC** field is always used as the

<table>
<tr><td></td><td></td><td>primary key field in the database. The field should not be modified.

result:
<?xml version="1.0" ?>
<CSL>
<Command>modDatabase</Command>
<Ack>OK: </Ack>
</CSL></td></tr>
<tr><td>10.13</td><td>session_id=<login_session_id>&
command=delDatabase&
databaseName=databaseName</td><td>Delete database.
e.g.
session_id=<login_session_id>&command=delDatabase
&databaseName=ProductDB

result:
<?xml version="1.0" ?>
<CSL>
<Command>delDatabase</Command>
<Ack>OK: </Ack>
</CSL></td></tr>
<tr><td>10.14</td><td>session_id=<login_session_id>&
command=listDatabase</td><td>List the field names and the data types of all databases.
e.g.
session_id=<login_session_id>&command=listDatabase

result:
<?xml version="1.0" ?>
<CSL>
<Command>listDatabase</Command>
<DatabaseList>
<database dataType1="STRING"
dataType2="STRING"
dataType3="IMAGE"
dataType4="NUMBER"
databaseName="ProductDB"
field1="ProductId"
field2="ProductName"
field3="ProductImage"
field4="ProductPrice"
keyDataType="STRING"
keyField="EPC"/>
</DatabaseList>
</CSL></td></tr>
</table>

10.15	session_id=<login_session_id>& command=***addTagDatabaseRecord*** &databaseName=*databaseName* &EPC=*EPC* [&*fieldname*=*value*]	Add tag record to the database. e.g. http://192.168.25.160/API?session_id=a33219dc&command=addTagDatabaseRecord&databaseName=ProductDB&EPC=012345678901234567 89ABCD&ProductId=1234&ProductName=Orange%20Juice&ProductPrice=1.5 Note : **EPC** field is always used as the primary key field in the database. It must be included in the addTagDatabaseRecord message and must not be null or empty. Use **setTagDatabaseRecordImage** command to set the image field. result: <?xml version="1.0" ?> <CSL> <Command>**addTagDatabaseRecord**</Command> <Ack>**OK:** </Ack> </CSL>
10.16	session_id=<login_session_id>& command=***modTagDatabaseRecord*** &databaseName=*databaseName* &EPC=*EPC* [&*fieldname*=*value*]	Modify tag record in the database. e.g. http://192.168.25.160/API?session_id=a33219dc&command=modTagDatabaseRecord&databaseName=ProductDB&EPC=012345678901234567 89ABCD&ProductId=1234&ProductName=Orange%20Juice&ProductPrice=1.75 Note : **EPC** field is always used as the primary key field in the database. It must be included in the modTagDatabaseRecord message and must not be null or empty. Use **setTagDatabaseRecordImage** command to set the image field.

		result: <?xml version="1.0" ?> <CSL> <Command>**modTagDatabaseRecord**</Command> <Ack>**OK:** </Ack> </CSL>
10.17	session_id=<login_session_id>& command=***delTagDatabaseRecord*** &databaseName=*databaseName* &EPC=*EPC*	Delete tag record from the database. e.g. http://192.168.25.160/API?session_id=a33219dc&command=delTagDatabaseRecord&databaseName=ProductDB&EPC=012345678901234567890ABCD Note : **EPC** field is always used as the primary key field in the database. It must be included in the delTagDatabaseRecord message. result: <?xml version="1.0" ?> <CSL> <Command>**delTagDatabaseRecord**</Command> <Ack>**OK:** </Ack> </CSL>
10.18	session_id=<login_session_id>& command=***listTagDatabaseRecord*** &databaseName=*databaseName*	List tag records in the database. e.g. http://192.168.25.160/API?session_id=a33219dc&command=listTagDatabaseRecord&databaseName=ProductDB Note : Use **getTagDatabaseRecordImage** command to get the image field. result: <?xml version="1.0" ?> <CSL> <Command>**listTagDatabaseRecord**</Comm

<table>
<tr><td></td><td></td><td>and>
<TagDatabaseRecordList>
<tagDatabaseRecord
EPC="0123456789012345678ABCD"
ProductId="1234"
ProductName="Orange Juice"
ProductImage=""
ProductPrice="1.75" />
<tagDatabaseRecord
EPC="000000020090505095227234"
ProductId="2345"
ProductName="Apple Juice"
ProductImage=""
ProductPrice="1.65" />
</TagDatabaseRecordList>
</CSL></td></tr>
<tr><td>10.19</td><td>session_id=<login_session_id>&
command=setTagDatabaseRecordImage
&databaseName=databaseName
&EPC=EPC
&imageFieldName=imageFieldName
&imageFileExtension=imageFileExtension</td><td>Set an image to the image field of a record in the database.

Here below is an example showing how to set an image to the image field of a record in the database via HTTP POST protocol written in C# (printed in blue color).

HttpClient client = new HttpClient();
var stream = new FileStream(“C:\\temp\\image.jpg”, FileMode.Open);
var content = new StreamContent(stream);
var requestUri = “http://192.168.25.160/API?session_id=a33219dc&command=setTagDatabaseRecordImage&databaseName=ProductDB&EPC=01234567890123456789ABCD&imageFieldName=ProductImage&imageFileExtension=jpg”;
var response = await client.PostAsync(requestUri, content);

result:
<?xml version="1.0" ?>
<CSL>
<Command>setTagDatabaseRecordImage</Command>
<Ack>OK:</Ack>
</CSL></td></tr>
</table>

10.20	session_id=<login_session_id>&
	command=***getTagDatabaseRecordImage***
	&databaseName=*databaseName*
	&EPC=*EPC*
	&imageFieldName=*imageFieldName*
		Get the image from a record in the database.
		
		Here below is an example showing how to get the image from a record in the database via HTTP POST protocol written in C# (printed in blue color).
		
		HttpClient client = new HttpClient();
		var requestUri =
		"http://192.168.25.160/API?session_id=a33219dc&command=getTagDatabaseRecordImage&databaseName=ProductDB&EPC=0123456789012345678 9ABCD&imageFieldName=ProductImage";
		var response = await client.GetAsync(requestUri);
		var content = response.Content;
		if(content.Headers.ContentType.MediaType.Contains("image"))
		{
		var fileName =
		content.Headers.ContentDisposition.FileName;
		var stream =
		content.ReadAsStreamAsync().Result;
		using (var fileStream = new
		FileStream("C:\\temp\\" + fileName,
		FileMode.Create, FileAccess.Write,
		FileShare.None))
		{
		Await stream.CopyToAsync(fileStream);
		}
		}

10.21	session_id=<login_session_id>& command=***databaseBackup*** &databaseName=*databaseName*	Backup the database and get the backup zip file. e.g. http://192.168.25.160/API?session_id=a33219dc&command=databaseBackup&databaseName=ProductDB

		Here below is an example, written in C# (printed in blue color), showing how to backup database ProductDB and save it as **Backup.zip** in **C:\temp**.
		
		HttpClient client = new HttpClient();
		var requestUri = "http://192.168.25.160/API?session=a33219dc&command=databaseBackup&databaseName=ProductDB";
		var response = await client.GetAsync(requestUri);
		var stream = response.Content.ReadAsStreamAsync().Result;
		using (var fileStream = new FileStream("C:\\temp\\Backup.zip", FileMode.Create, FileAccess.Write, FileShare.None))
		{
		  await stream.CopyToAsync(fileStream);
		}

10.22	session_id=<login_session_id>&
	command=***databaseRestore***
		Restore the database by loading the backup file to the reader.
		
		Here below is an example showing how to restore the database via HTTP POST protocol written in C# (printed in blue color).
		
		HttpClient client = new HttpClient();
		var stream = new FileStream("C:\\temp\\Backup.zip", FileMode.Open);
		var content = new StreamContent(stream);
		var requestUri = "http://192.168.25.160/API?session=a33219dc&command=databaseRestore";
		var response = await client.PostAsync(requestUri,

<table>
<tr><td></td><td></td><td>content);</td></tr>
<tr><td>10.23</td><td>session_id=<login_session_id>&
command=fm13dt160ReadTemperature
&linkProfile=linkProfile
&antennaPort=antennaPort
&transmitPower=transmitPower
&dwellTime=dwellTime
&reflectedPowerThreshold=reflectedPowerThreshold
&maskBank=maskBank
&mask=mask
&accessPassword=accessPassword</td><td>Read Temperature in degree Celsius from FM13DT160 tag.

e.g.
session_id=<login_session_id>&command=fm13dt160ReadTemperature&linkProfile=1&antennaPort=1&transmitPower=30&dwellTime=2000&reflectedPowerThreshold=24&maskBank=Bank1&mask=E2827001000000000000001&accessPassword=00000000

Valid attributes :
linkProfile : 0 = Multipath Interface Resistance
1 = Range/Dense Reader
2 = Range/Throughput/Dense Reader
3 = Max Throughput
antennaPort : 1 – 16
transmitPower : 0.0 – 32.0 in step of 0.1 dBm
dwellTime : unit=ms, >= 0ms
reflectedPowerThreshold : 1.0 – 32.0 in step of 0.1 dBm
maskBank : Bank0, Bank1, Bank2, Bank3

result:
<?xml version="1.0" ?>
<CSL>
<Command>fm13dt160ReadTemperature</Command>
<EPC>E2827001000000000000001</EPC>
<RSSI>-44.00</RSSI>
<Temperature>26.25</Temperature>
</CSL></td></tr>
<tr><td>10.24</td><td>session_id=<login_session_id>&
command=fm13dt160ReadBatteryVoltage
&linkProfile=linkProfile
&antennaPort=antennaPort
&transmitPower=transmitPower</td><td>Read Battery Voltage in volt from FM13DT160 tag.

e.g.
session_id=<login_session_id>&command=fm13dt160ReadBatteryVoltage&linkProfile=1&antennaPort=1&transmitPower=30&dwellTime=2000&reflectedPowerThreshold</td></tr>
</table>

	&dwellTime=*dwellTime* &reflectedPowerThreshold=*reflected PowerThreshold* &maskBank=*maskBank* &mask=*mask* &accessPassword=*accessPassword*	=24&maskBank=Bank1&mask=E2827001000000000000 0001&accessPassword=00000000 Valid attributes : linkProfile : 0 = Multipath Interface Resistance 1 = Range/Dense Reader 2 = Range/Throughput/Dense Reader 3 = Max Throughput antennaPort : 1 – 16 transmitPower : 0.0 – 32.0 in step of 0.1 dBm dwellTime : unit=ms, >= 0ms reflectedPowerThreshold : 1.0 – 32.0 in step of 0.1 dBm maskBank : Bank0, Bank1, Bank2, Bank3 result: <?xml version="1.0" ?> <CSL> <Command>**fm13dt160ReadBatteryVoltage**</Command> <BatteryVoltage>**1.47**</BatteryVoltage> <EPC>**E2827001000000000000001**</EPC> <RSSI>**-44.00**</RSSI> </CSL>
10.25	session_id=<login_session_id>& command=***fm13dt160ReadExtVoltage*** &linkProfile=*linkProfile* &antennaPort=*antennaPort* &transmitPower=*transmitPower* &dwellTime=*dwellTime* &reflectedPowerThreshold=*reflected PowerThreshold* &maskBank=*maskBank* &mask=*mask* &accessPassword=*accessPassword*	Read External Voltage in volt from FM13DT160 tag. e.g. session_id=<login_session_id>&command=fm13dt160ReadExtVoltage&linkProfile=1&antennaPort=1&transmitPower=30&dwellTime=2000&reflectedPowerThreshold=24&maskBank=Bank1&mask=E2827001000000000000001&accessPassword=00000000 Valid attributes : linkProfile : 0 = Multipath Interface Resistance 1 = Range/Dense Reader 2 = Range/Throughput/Dense Reader 3 = Max Throughput

		antennaPort : 1 – 16 transmitPower : 0.0 – 32.0 in step of 0.1 dBm dwellTime : unit=ms, >= 0ms reflectedPowerThreshold : 1.0 – 32.0 in step of 0.1 dBm maskBank : Bank0, Bank1, Bank2, Bank3 result: <?xml version="1.0" ?> <CSL> <Command>**fm13dt160ReadExtVoltage**</Command> <EPC>**E2827001000000000000001**</EPC> <ExtVoltage>**1.47**</ExtVoltage> <RSSI>**-44.00**</RSSI> </CSL>
10.26	session_id=<login_session_id>& command=***fm13dt160ReadExtSensorVoltage*** &linkProfile=*linkProfile* &antennaPort=*antennaPort* &transmitPower=*transmitPower* &dwellTime=*dwellTime* &reflectedPowerThreshold=*reflectedPowerThreshold* &maskBank=*maskBank* &mask=*mask* &accessPassword=*accessPassword*	Read External Sensor Voltage in volt from FM13DT160 tag. e.g. session_id=<login_session_id>&command=fm13dt160ReadExtSensorVoltage&linkProfile=1&antennaPort=1&transmitPower=30&dwellTime=2000&reflectedPowerThreshold=24&maskBank=Bank1&mask=E2827001000000000000001&accessPassword=00000000 Valid attributes : linkProfile : 0 = Multipath Interface Resistance 1 = Range/Dense Reader 2 = Range/Throughput/Dense Reader 3 = Max Throughput antennaPort : 1 – 16 transmitPower : 0.0 – 32.0 in step of 0.1 dBm dwellTime : unit=ms, >= 0ms reflectedPowerThreshold : 1.0 – 32.0 in step of 0.1 dBm maskBank : Bank0, Bank1, Bank2, Bank3 result: <?xml version="1.0" ?>

		<CSL> <Command>**fm13dt160ReadExtSensorVoltage**</Command> <EPC>**E2827001000000000000001**</EPC> <ExtSensorVoltage>**1.47**</ExtSensorVoltage> <RSSI>**-44.00**</RSSI> </CSL>
10.27	session_id=<login_session_id>& command=***fm13dt160ReadMemory*** &linkProfile=*linkProfile* &antennaPort=*antennaPort* &transmitPower=*transmitPower* &dwellTime=*dwellTime* &reflectedPowerThreshold=*reflectedPowerThreshold* &maskBank=*maskBank* &mask=*mask* &accessPassword=*accessPassword* &address=*address* &length=*length*	Read data from memory in FM13DT160 tag. e.g. session_id=<login_session_id>&command=fm13dt160ReadMemory&linkProfile=1&antennaPort=1&transmitPower=30&dwellTime=2000&reflectedPowerThreshold=24&maskBank=Bank1&mask=E2827001000000000000001&accessPassword=00000000&address=b040&length=4 Valid attributes : linkProfile : 0 = Multipath Interface Resistance 1 = Range/Dense Reader 2 = Range/Throughput/Dense Reader 3 = Max Throughput antennaPort : 1 – 16 transmitPower : 0.0 – 32.0 in step of 0.1 dBm dwellTime : unit=ms, >= 0ms reflectedPowerThreshold : 1.0 – 32.0 in step of 0.1 dBm maskBank : Bank0, Bank1, Bank2, Bank3 address : 0000 – C1FC, hex value and must be divisible by 4 length : 0 – 500, must be a multiple of 4 result: <?xml version="1.0" ?> <CSL> <Command>**fm13dt160ReadMemory**</Command> <Data>**4FB030CF**</Data> <EPC>**E2827001000000000000001**</EPC> <RSSI>**-44.00**</RSSI> </CSL>

10.28	session_id=<login_session_id>& command=***fm13dt160WriteMemory*** &linkProfile=*linkProfile* &antennaPort=*antennaPort* &transmitPower=*transmitPower* &dwellTime=*dwellTime* &reflectedPowerThreshold=*reflectedPowerThreshold* &maskBank=*maskBank* &mask=*mask* &accessPassword=*accessPassword* &address=*address* &length=*length*	Write data to memory in FM13DT160 tag. e.g. session_id=<login_session_id>&command=fm13dt160WriteMemory&linkProfile=1&antennaPort=1&transmitPower=30&dwellTime=2000&reflectedPowerThreshold=24&maskBank=Bank1&mask=E2827001000000000000001&accessPassword=00000000&address=0&data=8C9F7E60 Valid attributes : linkProfile : 0 = Multipath Interface Resistance 1 = Range/Dense Reader 2 = Range/Throughput/Dense Reader 3 = Max Throughput antennaPort : 1 – 16 transmitPower : 0.0 – 32.0 in step of 0.1 dBm dwellTime : unit=ms, >= 0ms reflectedPowerThreshold : 1.0 – 32.0 in step of 0.1 dBm maskBank : Bank0, Bank1, Bank2, Bank3 address : 0000 – C1FF, hex value data : hex string, at most 4 bytes result: <?xml version="1.0" ?> <CSL> <Command>**fm13dt160WriteMemory**</Command> <Ack>**OK:**</Ack> <EPC>**E2827001000000000000001**</EPC> <RSSI>**-44.00**</RSSI> </CSL>
10.29	session_id=<login_session_id>& command=***fm13dt160ReadUserCfg*** &linkProfile=*linkProfile* &antennaPort=*antennaPort*	Read user_cfg data from FM13DT160 tag. e.g. session_id=<login_session_id>&command=fm13dt160Re

	&transmitPower=*transmitPower* &dwellTime=*dwellTime* &reflectedPowerThreshold=*reflected PowerThreshold* &maskBank=*maskBank* &mask=*mask* &accessPassword=*accessPassword* &user_cfg=*user_cfg*	adUserCfg&linkProfile=1&antennaPort=1&transmitPower=30&dwellTime=2000&reflectedPowerThreshold=24&maskBank=Bank1&mask=E2827001000000000000001&accessPassword=00000000&user_cfg=0 Valid attributes : linkProfile : 0 = Multipath Interface Resistance 1 = Range/Dense Reader 2 = Range/Throughput/Dense Reader 3 = Max Throughput antennaPort : 1 – 16 transmitPower : 0.0 – 32.0 in step of 0.1 dBm dwellTime : unit=ms, >= 0ms reflectedPowerThreshold : 1.0 – 32.0 in step of 0.1 dBm maskBank : Bank0, Bank1, Bank2, Bank3 user_cfg : 0 – 3 result: <?xml version="1.0" ?> <CSL> <Command>**fm13dt160ReadUserCfg**</Command> <Data>**4F**</Data> <EPC>**E2827001000000000000001**</EPC> <RSSI>**-44.00**</RSSI> </CSL>
10.30	session_id=<login_session_id>& command=***fm13dt160WriteUserCfg*** &linkProfile=*linkProfile* &antennaPort=*antennaPort* &transmitPower=*transmitPower* &dwellTime=*dwellTime* &reflectedPowerThreshold=*reflected PowerThreshold* &maskBank=*maskBank* &mask=*mask* &accessPassword=*accessPassword*	Write user_cfg data to FM13DT160 tag. e.g. session_id=<login_session_id>&command=fm13dt160WriteUserCfg&linkProfile=1&antennaPort=1&transmitPower=30&dwellTime=2000&reflectedPowerThreshold=24&maskBank=Bank1&mask=E2827001000000000000001&accessPassword=00000000&user_cfg=1&data=30 Valid attributes : linkProfile : 0 = Multipath Interface Resistance

	&user_cfg=*user_cfg* &data =*data*	1 = Range/Dense Reader 2 = Range/Throughput/Dense Reader 3 = Max Throughput antennaPort : 1 – 16 transmitPower : 0.0 – 32.0 in step of 0.1 dBm dwellTime : unit=ms, >= 0ms reflectedPowerThreshold : 1.0 – 32.0 in step of 0.1 dBm maskBank : Bank0, Bank1, Bank2, Bank3 user_cfg : 0 – 3 data : 00 – FF, hex value, 1 byte only result: <?xml version="1.0" ?> <CSL> <Command>**fm13dt160WriteUserCfg**</Command> <Ack>**OK:**</Ack> <EPC>**E2827001000000000000001**</EPC> <RSSI>**-44.00**</RSSI> </CSL>
10.31	session_id=<login_session_id>& command=***fm13dt160ReadReg*** &linkProfile=*linkProfile* &antennaPort=*antennaPort* &transmitPower=*transmitPower* &dwellTime=*dwellTime* &reflectedPowerThreshold=*reflectedPowerThreshold* &maskBank=*maskBank* &mask=*mask* &accessPassword=*accessPassword* &address=*address*	Read data from register in FM13DT160 tag. e.g. session_id=<login_session_id>&command=fm13dt160ReadReg&linkProfile=1&antennaPort=1&transmitPower=30&dwellTime=2000&reflectedPowerThreshold=24&maskBank=Bank1&mask=E2827001000000000000001&accessPassword=00000000&address=c000 Valid attributes : linkProfile : 0 = Multipath Interface Resistance 1 = Range/Dense Reader 2 = Range/Throughput/Dense Reader 3 = Max Throughput antennaPort : 1 – 16 transmitPower : 0.0 – 32.0 in step of 0.1 dBm dwellTime : unit=ms, >= 0ms

		reflectedPowerThreshold : 1.0 – 32.0 in step of 0.1 dBm maskBank : Bank0, Bank1, Bank2, Bank3 address : c000 – c0ff, hex value result: <?xml version="1.0" ?> <CSL> <Command>**fm13dt160ReadReg**</Command> <Data>**0600**</Data> <EPC>**E2827001000000000000001**</EPC> <RSSI>**-44.00**</RSSI> </CSL>
10.32	session_id=<login_session_id>& command=***fm13dt160WriteReg*** &linkProfile=*linkProfile* &antennaPort=*antennaPort* &transmitPower=*transmitPower* &dwellTime=*dwellTime* &reflectedPowerThreshold=*reflectedPowerThreshold* &maskBank=*maskBank* &mask=*mask* &accessPassword=*accessPassword* &address=*address* &data =*data*	Write data to register in FM13DT160 tag. e.g. session_id=<login_session_id>&command=fm13dt160WriteReg&linkProfile=1&antennaPort=1&transmitPower=30&dwellTime=2000&reflectedPowerThreshold=24&maskBank=Bank1&mask=E2827001000000000000001&accessPassword=00000000&address=c000&data=0600 Valid attributes : linkProfile : 0 = Multipath Interface Resistance 1 = Range/Dense Reader 2 = Range/Throughput/Dense Reader 3 = Max Throughput antennaPort : 1 – 16 transmitPower : 0.0 – 32.0 in step of 0.1 dBm dwellTime : unit=ms, >= 0ms reflectedPowerThreshold : 1.0 – 32.0 in step of 0.1 dBm maskBank : Bank0, Bank1, Bank2, Bank3 address : c000 – c0ff, hex value data : 0000 – ffff, hex value result: <?xml version="1.0" ?> <CSL> <Command>**fm13dt160WriteReg**</Comma

		nd> <Ack>**OK:**</Ack> <EPC>**E2827001000000000000001**</EPC> <RSSI>**-44.00**</RSSI> </CSL>
10.33	session_id=<login_session_id>& command=***fm13dt160DeepSleep*** &linkProfile=*linkProfile* &antennaPort=*antennaPort* &transmitPower=*transmitPower* &dwellTime=*dwellTime* &reflectedPowerThreshold=*reflectedPowerThreshold* &maskBank=*maskBank* &mask=*mask* &accessPassword=*accessPassword* &enable=*enable*	Send Deep Sleep command to FM13DT160 tag. e.g. session_id=<login_session_id>&command=fm13dt160DeepSleep&linkProfile=1&antennaPort=1&transmitPower=30&dwellTime=2000&reflectedPowerThreshold=24&maskBank=Bank1&mask=E2827001000000000000001&accessPassword=00000000&enable=true Valid attributes : linkProfile : 0 = Multipath Interface Resistance 1 = Range/Dense Reader 2 = Range/Throughput/Dense Reader 3 = Max Throughput antennaPort : 1 – 16 transmitPower : 0.0 – 32.0 in step of 0.1 dBm dwellTime : unit=ms, >= 0ms reflectedPowerThreshold : 1.0 – 32.0 in step of 0.1 dBm maskBank : Bank0, Bank1, Bank2, Bank3 enable : true, false result: <?xml version="1.0" ?> <CSL> <Command>**fm13dt160DeepSleep**</Command> <Ack>**OK:**</Ack> <EPC>**E2827001000000000000001**</EPC> <RSSI>**-44.00**</RSSI> </CSL>
10.34	session_id=<login_session_id>& command=***fm13dt160OpModeChk*** &linkProfile=*linkProfile*	Send Op_Mode_Chk command to FM13DT160 tag. e.g.

	&antennaPort=*antennaPort* &transmitPower=*transmitPower* &dwellTime=*dwellTime* &reflectedPowerThreshold=*reflectedPowerThreshold* &maskBank=*maskBank* &mask=*mask* &accessPassword=*accessPassword* &refreshTempMeasurement=*refreshTempMeasurement*	session_id=<login_session_id>&command=fm13dt160OpModeChk&linkProfile=1&antennaPort=1&transmitPower=30&dwellTime=2000&reflectedPowerThreshold=24&maskBank=Bank1&mask=E2827001000000000000001&accessPassword=00000000&refreshTempMeasurement=false Valid attributes : linkProfile : 0 = Multipath Interface Resistance 1 = Range/Dense Reader 2 = Range/Throughput/Dense Reader 3 = Max Throughput antennaPort : 1 – 16 transmitPower : 0.0 – 32.0 in step of 0.1 dBm dwellTime : unit=ms, >= 0ms reflectedPowerThreshold : 1.0 – 32.0 in step of 0.1 dBm maskBank : Bank0, Bank1, Bank2, Bank3 refreshTempMeasurement : true, false result: <?xml version="1.0" ?> <CSL> <Command>**fm13dt160OpModeChk**</Command> <EPC>**E2827001000000000000001**</EPC> <OpMode>**user_access_en,vbat_pwr_flag**</OpMode> <RSSI>**-44.00**</RSSI> </CSL> OpMode : user_access_en : user has valid access right rtc_logging : RTC logging in progress vdet_process_flag : instant temperature measurement is interrupted light_chk_flag : light strength over preset value vbat_pwr_flag : battery voltage is higher than 0.9V

10.35	session_id=<login_session_id>& command=***fm13dt160InitialRegfile*** &linkProfile=*linkProfile* &antennaPort=*antennaPort* &transmitPower=*transmitPower* &dwellTime=*dwellTime* &reflectedPowerThreshold=*reflected PowerThreshold* &maskBank=*maskBank* &mask=*mask* &accessPassword=*accessPassword*	Send Initial Regfile command to FM13DT160 tag. e.g. session_id=<login_session_id>&command=fm13dt160InitialRegfile&linkProfile=1&antennaPort=1&transmitPower=30&dwellTime=2000&reflectedPowerThreshold=24&maskBank=Bank1&mask=E2827001000000000000000 1&accessPassword=00000000 Valid attributes : linkProfile : 0 = Multipath Interface Resistance 1 = Range/Dense Reader 2 = Range/Throughput/Dense Reader 3 = Max Throughput antennaPort : 1 – 16 transmitPower : 0.0 – 32.0 in step of 0.1 dBm dwellTime : unit=ms, >= 0ms reflectedPowerThreshold : 1.0 – 32.0 in step of 0.1 dBm maskBank : Bank0, Bank1, Bank2, Bank3 result: <?xml version="1.0" ?> <CSL> <Command>**fm13dt160InitialRegfile**</Command> <Ack>**OK:**</Ack> <EPC>**E2827001000000000000000 1**</EPC> <RSSI>**-44.00**</RSSI> </CSL>
10.36	session_id=<login_session_id>& command=***fm13dt160LedCtrl*** &linkProfile=*linkProfile* &antennaPort=*antennaPort* &transmitPower=*transmitPower* &dwellTime=*dwellTime* &reflectedPowerThreshold=*reflected PowerThreshold*	Send Led Ctrl command to FM13DT160 tag. e.g. session_id=<login_session_id>&command=fm13dt160LedCtrl&linkProfile=1&antennaPort=1&transmitPower=30&dwellTime=2000&reflectedPowerThreshold=24&maskBank=Bank1&mask=E2827001000000000000000 1&accessPassword=00000000&enable=true

	&maskBank=*maskBank* &mask=*mask* &accessPassword=*accessPassword* &enable=*enable*	Valid attributes : linkProfile : 0 = Multipath Interface Resistance 1 = Range/Dense Reader 2 = Range/Throughput/Dense Reader 3 = Max Throughput antennaPort : 1 – 16 transmitPower : 0.0 – 32.0 in step of 0.1 dBm dwellTime : unit=ms, >= 0ms reflectedPowerThreshold : 1.0 – 32.0 in step of 0.1 dBm maskBank : Bank0, Bank1, Bank2, Bank3 enable : true, false result: <?xml version="1.0" ?> <CSL> <Command>**fm13dt160LedCtrl**</Command> <Ack>**OK:**</Ack> <EPC>**E2827001000000000000001**</EPC> > <RSSI>**-44.00**</RSSI> </CSL>
10.37	session_id=<login_session_id>& command=***fm13dt160StartLogging*** &linkProfile=*linkProfile* &antennaPort=*antennaPort* &transmitPower=*transmitPower* &dwellTime=*dwellTime* &reflectedPowerThreshold=*reflectedPowerThreshold* &maskBank=*maskBank* &mask=*mask* &accessPassword=*accessPassword* &startDelay=*startDelay* &timeStep=*timeStep* &sampleNumber=*sampleNumber* &sampleFlash=*sampleFlash* [&sampleFlashLength=*sampleFlashL*	Send Start Logging command to FM13DT160 tag. e.g. session_id=<login_session_id>&command=fm13dt160StartLogging&linkProfile=1&antennaPort=1&transmitPower=30&dwellTime=2000&reflectedPowerThreshold=24&maskBank=Bank1&mask=E2827001000000000000001&accessPassword=00000000&startDelay=5&timeStep=30&sampleNumber=5&sampleFlash=true&sampleFlashLength=1&outOfLimitFlash=true&outOfLimitFlashLength=0.5&outOfLimitFlashNumber=3&loggingMode=Out_Of_Range_Only&minLimit=0&maxLimit=20 Valid attributes : linkProfile : 0 = Multipath Interface Resistance 1 = Range/Dense Reader

	ength] [&outOfLimitFlash=*outOfLimitFlash* &outOfLimitFlashLength=*outOfLimitFlashLength* &outOfLimitFlashNumber=*outOfLimitFlashNumber*] &loggingMode=*loggingMode* [&minLimit=*minLimit* &maxLimit=*maxLimit*]	2 = Range/Throughput/Dense Reader 3 = Max Throughput antennaPort : 1 – 16 transmitPower : 0.0 – 32.0 in step of 0.1 dBm dwellTime : unit=ms, >= 0ms reflectedPowerThreshold : 1.0 – 32.0 in step of 0.1 dBm maskBank : Bank0, Bank1, Bank2, Bank3 startDelay : unit minute, 0 – 65535, time delay to start logging after command is received timeStep : unit second, 1 – 65535, time interval between each temperature measurement sample sampleNumber : 1 – 4864, number of samples to be taken sampleFlash : true, false, LED flash after each sampling sampleFlashLength : unit second, 0.1 – 1.6, LED flash time duration after each sampling outOfLimitFlash : true, false, LED flash if temperature sample is lower than the minLimit or higher than the maxLimit outOfLimitFlash : unit second, 0.1 – 1.6, LED flash time duration if temperature sample is out of the preset limit outOfLimitFlashNumber : 1 – 15, number of flashes if temperature sample is out of the preset limit loggingMode : Normal, Out_Of_Range_Only minLimit : unit degree Celsius, -127.75 – 127.75 maxLimit : unit degree Celsius, -127.75 – 127.75 result: <?xml version="1.0" ?> <CSL> <Command>**fm13dt160StartLogging**</Command> <Ack>**OK:**</Ack> <EPC>**E2827001000000000000001**</EPC>

		<RSSI>**-44.00**</RSSI> </CSL>
10.38	session_id=<login_session_id>& command=***fm13dt160StopLogging*** &linkProfile=*linkProfile* &antennaPort=*antennaPort* &transmitPower=*transmitPower* &dwellTime=*dwellTime* &reflectedPowerThreshold=*reflected PowerThreshold* &maskBank=*maskBank* &mask=*mask* &accessPassword=*accessPassword*	Send Stop Logging command to FM13DT160 tag. e.g. session_id=<login_session_id>&command=fm13dt160StopLogging&linkProfile=1&antennaPort=1&transmitPower=30&dwellTime=2000&reflectedPowerThreshold=24&maskBank=Bank1&mask=E2827001000000000000001&accessPassword=00000000 Valid attributes : linkProfile : 0 = Multipath Interface Resistance 1 = Range/Dense Reader 2 = Range/Throughput/Dense Reader 3 = Max Throughput antennaPort : 1 – 16 transmitPower : 0.0 – 32.0 in step of 0.1 dBm dwellTime : unit=ms, >= 0ms reflectedPowerThreshold : 1.0 – 32.0 in step of 0.1 dBm maskBank : Bank0, Bank1, Bank2, Bank3 result: <?xml version="1.0" ?> <CSL> <Command>**fm13dt160StopLogging**</Command> <Ack>**OK:**</Ack> <EPC>**E2827001000000000000001**</EPC> <RSSI>**-44.00**</RSSI> </CSL>
10.39	session_id=<login_session_id>& command=***fm13dt160GetLogging*** &linkProfile=*linkProfile* &antennaPort=*antennaPort* &transmitPower=*transmitPower* &dwellTime=*dwellTime*	Send Get Logging command to FM13DT160 tag. e.g. session_id=<login_session_id>&command=fm13dt160GetLogging&linkProfile=1&antennaPort=1&transmitPower=30&dwellTime=2000&reflectedPowerThreshold=24&m

	&reflectedPowerThreshold=*reflectedPowerThreshold* &maskBank=*maskBank* &mask=*mask* &accessPassword=*accessPassword*	askBank=Bank1&mask=E2827001000000000000001&accessPassword=00000000 Valid attributes : linkProfile : 0 = Multipath Interface Resistance 1 = Range/Dense Reader 2 = Range/Throughput/Dense Reader 3 = Max Throughput antennaPort : 1 – 16 transmitPower : 0.0 – 32.0 in step of 0.1 dBm dwellTime : unit=ms, >= 0ms reflectedPowerThreshold : 1.0 – 32.0 in step of 0.1 dBm maskBank : Bank0, Bank1, Bank2, Bank3 result: <?xml version="1.0" ?> <CSL> <Command>**fm13dt160GetLogging**</Command> <EPC>**E2827001000000000000001**</EPC> <RSSI>**-44.00**</RSSI> <LocalStartTime>**Tue Apr 20 17:26:56 2021**</LocalStartTime> <UTCStartTime>**Tue Apr 20 09:26:56 2021**</UTCStartTime> <StartDelay>**5**</StartDelay> <StartDelayUnit>**minute**</StartDelayUnit> <TimeStep>**30**</TimeStep> <TimeStepUnit>**second**</TimeStepUnit> <LoggingMode>**Out_Of_Range_Only**</LoggingMode> <MinLimit>**0**</MinLimit> <MaxLimit>**20**</MaxLimit> <LogList> <log temperature="**22.25**" localTime="**Tue Apr 20 17:31:56 2021**" utcTime="**Tue Apr 20 09:31:56 2021**" /> <log temperature="**22.50**" localTime="**Tue Apr 20 17:32:26 2021**" utcTime="**Tue Apr 20 09:32:26 2021**" /> <log temperature="**22.50**" localTime="**Tue Apr 20 17:32:56 2021**" utcTime="**Tue Apr 20 09:32:56 2021**" />

		<log temperature="22.25" localTime="Tue Apr 20 17:33:26 2021" utcTime="Tue Apr 20 09:33:26 2021" /> <log temperature="22.50" localTime="Tue Apr 20 17:33:56 2021" utcTime="Tue Apr 20 09:33:56 2021" /> </LogList> </CSL>
10.40	session_id=<login_session_id>& command=***readTag*** &linkProfile=*linkProfile* &antennaPort=*antennaPort* &transmitPower=*transmitPower* &dwellTime=*dwellTime* &reflectedPowerThreshold=*reflectedPowerThreshold* &maskBank=*maskBank* &mask=*mask* &accessPassword=*accessPassword* &readAccessPassword=*readAccessPassword* &readKillPassword=*readKillPassword* &readTidBank=*readTidBank* [&tidBankOffset=*tidBankOffset* &tidBankLength=*tidBankLength*] &readUserBank=*readUserBank* [&userBankOffset=*userBankOffset* &userBankLength=*userBankLength*]	Read tag data. e.g. session_id=<login_session_id>&command=readTag&linkProfile=1&antennaPort=1&transmitPower=30&dwellTime=2000&reflectedPowerThreshold=24&maskBank=Bank1&mask=E2827001000000000000001&accessPassword=00000000&readAccessPassword=true&readKillPassword=true&readTidBank=true&tidBankOffset=0&tidBankLength=2&readUserBank=true&userBankOffset=0&userBankLength=2 Valid attributes : linkProfile : 0 = Multipath Interface Resistance 1 = Range/Dense Reader 2 = Range/Throughput/Dense Reader 3 = Max Throughput antennaPort : 1 – 16 transmitPower : 0.0 – 32.0 in step of 0.1 dBm dwellTime : unit=ms, >= 0ms reflectedPowerThreshold : 1.0 – 32.0 in step of 0.1 dBm maskBank : Bank0, Bank1, Bank2, Bank3 readAccessPassword : true, false readKillPassword : true, false readTidBank : true, false readUserBank : true, false result: <?xml version="1.0" ?> <CSL>

		<Command>**readTag**</Command> <PC>**3000**</PC> <EPC>**E2827001000000000000001**</EPC> <AccessPassword>**00000000**</AccessPassword> <KillPassword>**00000000**</KillPassword> <TidBank>**E2827001**</TidBank> <UserBank>**33192F61**</UserBank> </CSL>
10.41	session_id=<login_session_id>& command=***writeTag*** &linkProfile=*linkProfile* &antennaPort=*antennaPort* &transmitPower=*transmitPower* &dwellTime=*dwellTime* &reflectedPowerThreshold=*reflectedPowerThreshold* &maskBank=*maskBank* &mask=*mask* &accessPassword=*accessPassword* [&writePC=*writePC* &newPC=*newEPC*] [&writeEPC=*writeEPC* &newEPC=*newEPC*] [&writeAccessPassword=*writeAccessPassword* &newAccessPassword=*newAccessPassword*] [&writeKillPassword=*writeKillPassword* &newKillPassword=*newKillPassword*] [&writeTidBank=*writeTidBank* &newTidBankOffset=*newTidBankOffset* &newTidBank=*newTidBank*] [&writeUserBank=*writeUserBank* &newUserBankOffset=*newBankOffse*	Write tag data. e.g. session_id=<login_session_id>&command=writeTag&linkProfile=1&antennaPort=1&transmitPower=30&dwellTime=2000&reflectedPowerThreshold=24&maskBank=Bank1&mask=6161616160000000000000000&accessPassword=00000000&writePC=true&newPC=3000&writeEPC=true&newEPC=626263636000000000000000&writeTidBank=true&newTidBankOffset=0&newTidBank=E2801160 Valid attributes : linkProfile : 0 = Multipath Interface Resistance 1 = Range/Dense Reader 2 = Range/Throughput/Dense Reader 3 = Max Throughput antennaPort : 1 – 16 transmitPower : 0.0 – 32.0 in step of 0.1 dBm dwellTime : unit=ms, >= 0ms reflectedPowerThreshold : 1.0 – 32.0 in step of 0.1 dBm maskBank : Bank0, Bank1, Bank2, Bank3 writeAccessPassword : true, false writeKillPassword : true, false writeTidBank : true, false writeUserBank : true, false result: <?xml version="1.0" ?> <CSL> <Command>**writeTag**</Command> <PC>**3000**</PC>

<table>
<tr><td></td><td>t
&newUserBank=newUserBank]</td><td><EPC>616161616000000000000000</EPC>
<WritePC>OK</WritePC>
<WriteEPC>OK</WriteEPC>
<WriteTidBank>Error</WriteTidBank>
</CSL></td></tr>
<tr><td>10.42</td><td>session_id=<login_session_id>&
command=addTagFilter
&tag_filter_id=tag_filter_id
&type=type
&bank=bank
&offset=offset
&mask=mask
&action=action</td><td>Add Tag Filter.

e.g.
http://192.168.25.160/API?session_id=a33219dc&command=addTagFilter&tag_filter_id=Pre%20Filter%201&type=PRE_FILTER&bank=Bank1&offset=0&mask=6161&action=0

Valid attributes :
type : PRE_FILTER, POST_FILTER
bank : Bank0, Bank1, Bank2, Bank3
offset : unit = bits
action :
If type is PRE_FILTER :
<table>
<tr><th>Action</th><th>Tag Matching</th><th>Tag Not Matching</th></tr>
<tr><td>0</td><td>assert SL or inventoried -> A</td><td>deassert SL or inventoried -> B</td></tr>
<tr><td>1</td><td>assert SL or inventoried -> A</td><td>do nothing</td></tr>
<tr><td>2</td><td>do nothing</td><td>deassert SL or inventoried -> B</td></tr>
<tr><td>3</td><td>negate SL or (A -> B, B -> A)</td><td>do nothing</td></tr>
<tr><td>4</td><td>deassert SL or inventoried -> B</td><td>assert SL or inventoried -> A</td></tr>
<tr><td>5</td><td>deassert SL or inventoried -> B</td><td>do nothing</td></tr>
<tr><td>6</td><td>do nothing</td><td>assert SL or inventoried -> A</td></tr>
<tr><td>7</td><td>do nothing</td><td>negate SL or (A -> B, B -> A)</td></tr>
</table>
If type is POST_FILTER :
action 0 = Match mask</td></tr>
</table>

<table>
<tr><td></td><td></td><td>action 1 = Not match mask

result:
<pre><?xml version="1.0" ?>
<CSL>
  <Command>addTagFilter</Command>
  <Ack>OK:</Ack>
</CSL></pre></td></tr>
<tr><td>10.43</td><td>session_id=<login_session_id>&
command=modTagFilter
&tag_filter_id=tag_filter_id
&type=type
&bank=bank
&offset=offset
&mask=mask
&action=action</td><td>Modiy Tag Filter.

e.g.
http://192.168.25.160/API?session_id=a33219dc&command=modTagFilter&tag_filter_id=Pre%20Filter%201&type=PRE_FILTER&bank=Bank1&offset=0&mask=6262&action=0

Valid attributes :
type : PRE_FILTER, POST_FILTER
bank : Bank0, Bank1, Bank2, Bank3
offset : unit = bits
action :
If type is PRE_FILTER :
<table>
<tr><th>Action</th><th>Tag Matching</th><th>Tag Not Matching</th></tr>
<tr><td>0</td><td>assert SL or inventoried -> A</td><td>deassert SL or inventoried -> B</td></tr>
<tr><td>1</td><td>assert SL or inventoried -> A</td><td>do nothing</td></tr>
<tr><td>2</td><td>do nothing</td><td>deassert SL or inventoried -> B</td></tr>
<tr><td>3</td><td>negate SL or (A -> B, B -> A)</td><td>do nothing</td></tr>
<tr><td>4</td><td>deassert SL or inventoried -> B</td><td>assert SL or inventoried -> A</td></tr>
<tr><td>5</td><td>deassert SL or inventoried -> B</td><td>do nothing</td></tr>
<tr><td>6</td><td>do nothing</td><td>assert SL or inventoried -> A</td></tr>
<tr><td>7</td><td>do nothing</td><td>negate SL or (A -> B, B -> A)</td></tr>
</table></td></tr>
</table>

		If type is POST_FILTER : action 0 = Match mask action 1 = Not match mask result: <?xml version="1.0" ?> <CSL> <Command>**modTagFilter**</Command> <Ack>**OK:**</Ack> </CSL>
10.44	session_id=<login_session_id>& command=***delTagFilter*** &tag_filter_id=*tag_filter_id*	Delete Tag Filter e.g. http://192.168.25.160/API?session_id=a33219dc&command=delTagFilter&tag_filter_id=Pre%20Filter%201 result: <?xml version="1.0" ?> <CSL> <Command>**delTagFilter**</Command> <Ack>**OK:**</Ack> </CSL>
10.45	session_id=<login_session_id>& command=***listTagFilter***	List Tag Filter e.g. http://192.168.25.160/API?session_id=a33219dc&command=delTagFilter&tag_filter_id=Pre%20Filter%201 result: <?xml version="1.0" ?> <CSL> <Command>**listTagFilter**</Command> <TagFilterList> <tagfilter action="**0**" bank="**Bank1**" mask="**6161**" offset="**0**" tag_filter_id="**Pre Filter 1**" type="**PRE_FILTER**" />

		<tagfilter action="**1**" bank="**Bank3**" mask="**3005**" offset="**0**" tag_filter_id="**Post Filter 1**" type="**POST_FILTER**" /> </TagFilterList> </CSL>

11. GPIO Management

11.1	session_id=<login_session_id>& command=***runIO_output***& port=*port*& oper_logic=*oper_logic*	Set the output port *port* to the logic value *oper_logic*. e.g. http://192.168.25.160/API?session_id=a33219dc&command=runIO_output&port=1&oper_logic=1 result: <?xml version="1.0" ?> <CSL> <Command>**runIO_output**</Command> <Ack>**OK:**</Ack> </CSL> Valid attributes : port : 1,2,3,4 oper_logic: 0,1
11.2	session_id=<login_session_id>& command=***runIO_output8bits***& logic=*logic*	Set the output port to the logic value *logic*. e.g. http://192.168.25.245/API?session_id=f9125ad4&command=runIO_output8bits&logic=0F result: <?xml version="1.0" ?> <CSL> <Command>**runIO_output8bits**</Command> <Ack>**OK:**</Ack> </CSL> Valid attributes : logic : 2 hex digits, i.e. 00 - 0F

11.3	session_id=<login_session_id>& command=***runIO_input***	Get input status from I/O ports. e.g.1 Synchronized mode http://192.168.25.160/API?session_id=a33219dc&command=runIO_input result: <?xml version="1.0" ?> <CSL> <Command>**runIO_input**</Command> <Input input_logic_list="**0,1,0,0**" port_list="**1,2,3,4**" /> </CSL> i.e. port1 : logic '0' port2 : logic '1' port3 : logic '0' port4 : logic '0' The 0 or 1 in the input_logic_list represents the corresponding logic of port number (port 1-4) in the port_list.
11.4	session_id=<login_session_id>& command=***directIOOutput***& port=*port*& oper_logic=*oper_logic*& username=*username*& passwor=*password*	Set or reset the output port *port* according to the logic *oper_logic* (without login). e.g. http://192.168.25.160/API?command=directIOOutput&port=1&oper_logic=1&username=root&password=csl result: <?xml version="1.0" ?> <CSL> <Command>**directIOOutput**</Command> <Ack>**OK:**</Ack> </CSL> Valid attributes : port : 1,2,3,4 oper_logic: 0,1

11.5	session_id=<login_session_id>& command=***directIOOutput8bits***& logic=*logic*& username=*username*& passwor=*password*	Set or reset the output port *port* according to the logic *oper_logic* (without login). e.g. http://192.168.25.245/API?command=directIOOutput8bits&logic=0F&username=root&password=csl result: <?xml version="1.0" ?> <CSL> <Command>**directIOOutput8bits**</Command> <Ack>**OK:**</Ack> </CSL> Valid attributes : logic : 2 hex digits, i.e. 00 - 0F
11.6	session_id=<login_session_id>& command=***directIOInput***& username=*username*& passwor=*password*	Get input status from I/O ports (without login). e.g.1 Synchronized mode http://192.168.25.160/API?command=directIOInput&username=root&password=csl result: <?xml version="1.0" ?> <CSL> <Command>**directIOInput**</Command> <Input input_logic_list="**0,1,0,0**" port_list="**1,2,3,4**" /> </CSL> Valid attributes : The 0 or 1 in the input_logic_list represents the corresponding logic of port number (port 1-4) in the port_list.

12. Events Management

Events Management		
12.1	session_id=<login_session_id> & command=***addTriggeringLogic*** & logic_id=*logic_id*& desc=*desc*& mode=*mode* [&logic=*logic*] [&state_mode=*state_mode*] [&capturePoint=*capturePoint*] [&referenceTagId=*ref_tag*]	Create a triggering logic in <TriggeringLogic> table. e.g.1 This mode is used for “InventoryEnablingTrigger” or “Trigger Logic” in Event definition http://192.168.25.160/API?session_id=a33219dc&command=addTriggeringLogic&logic_id=DemoTrigger&desc=Demo Trigger&mode=Read Any Tags (any ID, 1 trigger per tag) e.g.2 http://192.168.25.160/API?session_id=a33219dc&command=addTriggeringLogic&logic_id=Sensor1&desc=Sesnor 1&mode=Input Sensor State&logic=Sensor1:0&state_mode=CHANGE e.g.3 This mode is used for “InventoryDisablingTrigger” with reference tag in Event definition http://192.168.25.160/API?session_id=a33219dc&command=addTriggeringLogic&logic_id=NoTagAndStop&desc=Stop Inventory if no tag read more than 2 seconds&mode=No Tag Read in Specified Time Span&logic=2000&referenceTagId=012345678901234 56789ABCD e.g.4 http://192.168.25.160/API?session_id=a33219dc&command=addTriggeringLogic&logic_id=TagTest&desc=Tag Read test&mode=Tag Group Filtering&logic=TagGroup1

		Valid attributes : mode : Read Any Tags (any ID, 1 trigger per tag), Input Sensor State, No Tag Read in Specified Time Span, Trigger in Tag Group, Trigger in Tag Database, Trigger if RSSI larger than or equal to, Trigger if Moisture is larger than or equal to, Trigger if Moisture is less than or equal to, Trigger if Temperature is larger than or equal to, Trigger if Temperature is less than or equal to, Specified Time Span elapsed, If mode==Input Sensor State logic=Sensor and input level in form of 'Sensor[n]:[0,1]' where n=1,2,3,4. eg. Sensor1:0 ==> Sensor1 with input in high level, Sensor2:1 ==> Sensor2 with input in low level If mode==No Tag Read in Specified Time Span logic=<time span in which no tag read, unit:ms>, refer to e.g.3 referenceTagId=<EPC>, EPC of the reference tag, which is ignore in counting the time If mode==Trigger in Tag Group logic=<Tag Group>, refer to e.g.4 If mode==Trigger if RSSI larger than or equal to, Trigger if Moisture is larger than or equal to, Trigger if Moisture is less than or equal to, Trigger if Temperature is larger than or equal to, Trigger if Temperature is less than or equal to logic=<threshold value> state_mode : CHANGE = sensor input is changed to the specified logic LEVEL = sensor input meets the specified

		logic capturePoint : 1 – 16, any combinations with comma separated result: <?xml version="1.0" ?> <CSL> <Command>**addTriggeringLogic**</Command> <Ack>**OK:**</Ack> </CSL>
12.2	session_id=<login_session_id> & command=***modTriggeringLogic*** & logic_id=*logic_id* [&desc=*desc*] [&mode=*mode*] [&logic=*logic*] [&state_mode=*state_mode*] [&capture_point=*capture_point*] [&referenceTagId=*ref_tag*]	Modify an existing triggering logic in <TriggeringLogic> table by logic_id. e.g.1 modify capture point http://192.168.25.160/API?session_id=a33219dc&command=modTriggeringLogic&logic_id=DemoTrigger&capturePoint=1,3 e.g.2 modify desc http://192.168.25.160/API?session_id=a33219dc&command=modTriggeringLogic&logic_id=DemoTrigger&desc=ModifiedDemoTrigger Valid attributes : mode : Read Any Tags (any ID, 1 trigger per tag), Input Sensor State, No Tag Read in Specified Time Span, Trigger in Tag Group, Trigger in Tag Database, Trigger if RSSI larger than or equal to, Trigger if Moisture is larger than or equal to, Trigger if Moisture is less than or equal to, Trigger if Temperature is larger than or equal to, Trigger if Temperature is less than or equal to, Specified Time Span elapsed,

		If mode==Input Sensor State logic=Sensor and input level in form of 'Sensor[n]:[0,1]' where n=1,2,3,4. eg. Sensor1:0 ==> Sensor1 with input in high level, Sensor2:1 ==> Sensor2 with input in low level If mode==No Tag Read in Specified Time Span logic=<time span in which no tag read, unit:ms>, refer to e.g.3 referenceTagId=<EPC>, EPC of the reference tag, which is ignore in counting the time If mode==Trigger in Tag Group logic=<Tag Group>, refer to e.g.5 If mode==Trigger if RSSI larger than or equal to, Trigger if Moisture is larger than or equal to, Trigger if Moisture is less than or equal to, Trigger if Temperature is larger than or equal to, Trigger if Temperature is less than or equal to logic=<threshold value> state_mode : CHANGE = sensor input is changed to the specified logic LEVEL = sensor input meets the specified logic capturePoint : 1 – 16, any combinations with comma separated result: <?xml version="1.0" ?> <CSL> <Command>**modTriggeringLogic**</Command> <Ack>**OK:**</Ack> </CSL>
12.3	session_id=<login_session_id> & command=***delTriggeringLogic***	Remove a triggering logic from the <TriggeringLogic> table. e.g.

<table>
<tr><td></td><td>&
logic_id=logic_id</td><td>session_id=<login_session_id>&command=delTriggeringLogic&logic _id=logic1

result:
<?xml version="1.0" ?>
<CSL>
<Command>delTriggeringLogic</Command>
<Ack>OK:</Ack>
</CSL></td></tr>
<tr><td>12.4</td><td>session_id=<login_session_id>
&
command=listTriggeringLogic</td><td>List Triggering Logic table.

e.g.
session_id=<login_session_id>&command=listTriggeringLogic

result:
<?xml version="1.0" ?>
<CSL>
<Command>listTriggeringLogic</Command>
<TriggeringLogic>
<logic capture_point="1234"
desc="Read Burn-in Trigger Logic"
logic=""
logic_id="ReadBurninTrigger"
mode="Read Any Tags (any ID, 1 trigger per tag)"
referenceTagId=""
state_mode="" />
<logic capture_point="1234"
desc="Read Burn-in Disabling Trigger Logic"
logic="15000"
logic_id="ReadBurninDisableTrigger"
mode="No Tag Read in Specified Time Span"
referenceTagId="000000020090505095227234"
state_mode="" />
</TriggeringLogic>
</CSL></td></tr>
<tr><td>12.5</td><td>session_id=<login_session_id>
&
command=addResultantAction
&
action_id=action_id&
desc=desc</td><td>Create a resultant action in <ResultantAction> table.

e.g.
http://192.168.25.160/API?session_id=a33219dc&command=addResultantAction&action_id=DemoAction&desc=Demo%20Action&action_mode=Batch%20Alert%20t</td></tr>
</table>

	[&condition=*condition* &condition_logic=*condition_logic*] &action_mode=*action_mode* [&server_id=*server_id* &pre_action_wait=*pre_action_wait* &post_action_delay=*post_action_delay* &action=*action* &pulse_logic=*pulse_logic* &pulse_mode=*pulse_mode* &pulse_width=*pulse_width* &dutycycle=*dutycycle* &duration=*duration* &transport=*transport* &data_format_id=*data_format_id* &display_format_id=*display_format_id* &display_time_factor_type=*display_time_factor_type* &display_time_factor=*display_time_factor* &batch_alert_time_cycle=*batch_alert_time_cycle*]	o%20Server&server_id=DemoServer Valid attributes : condition : None, Input Sensor State condition_logic : If condition is Input Sensor State, it represents Sensor and input level in form of 'Sensor[n]:[0,1]' where n=1,2,3,4. eg. Sensor1:0 ==> Sensor1 with input in high level, Sensor2:1 ==> Sensor2 with input in low level action_mode : Do Nothing (Only Show on Screen), Batch Alert to Server, Instant Alert to Server, Low Latency Alert to Server, Output Port, Display Tag Database Record, Display Tag Group Record, server_id : <Cloud Server> pre_acton_wait : unit = ms post_acton_delay : unit = ms action : Port[n]:[0,1,Pulse] where n=1,2,3,4; 0 ==> Open switch, 1 ==> Close switch pulse_logic : Positive ==> Open, Close, Open Negative ==> Close, Open, Close (for action=Pulse only) pulse_mode : One Shot Pulse, Pulse Train pulse_width : unit = ms dutycycle : unit = % (for Pulse Train only) duration : unit = ms (for Pulse Train only) transport : TCP, HTTP POST, MQTT display_time_factor_type : Additive, Multiplicative batch_alert_time_cycle : unit = ms Note : If action_mode is Display Tag Database Record or Display Tag Group Record, the display time is determined by

		display_time_factor_type and display_time_factor. e.g.1 display_time_factor_type = Additive display time = Tag Duplicate Elimination Window + display_time_factor (ms) e.g.2 display_time_factor_type = Multiplicative display time = Tag Duplicate Elimination Window × display_time_factor (ms) If display_time_factor = 0, record shown on the page forever until next record comes in. result: <?xml version="1.0" ?> <CSL> <Command>**addResultantAction**</Command> <Ack>**OK:**</Ack> </CSL>
12.6	session_id=<login_session_id> & command=***modResultantAction*** & action_id=*actiont_id* [&desc=*desc*] [&condition=*condition*] [&condition_logic=*condition_logic*] [&action_mode=*action_mode*] [&server_id=*server_id*] [&pre_action_wait=*pre_action_wait*] [&post_action_delay=*post_action_delay*] [&action=*action*] [&pulse_logic=*pulse_logic*] [&pulse_mode=*pulse_mode*] [&pulse_width=*pulse_width*] [&dutycycle=*dutycycle*] [&duration=*duration*]	Modify an existing resultant action in <ResultantAction> table by action_id. e.g.1 http://192.168.25.160/API?session_id=a33219dc&command=modResultantAction&action_id=DemoAction&desc=Demo%20Action e.g.2 http://192.168.25.160/API?session_id=a33219dc&command=modResultantAction&action_id=DemoAction&server_id=DemoServer Valid attributes : condition : None, Input Sensor State condition_logic : If condition is Input Sensor State, it represents Sensor and input level in form of 'Sensor[n]:[0,1]' where n=1,2,3,4. eg. Sensor1:0 ==> Sensor1 with input in high level, Sensor2:1 ==> Sensor2 with input in low level

	[&transport=*transport*] [&data_format_id=*data_format_id*] [&display_format_id=*display_format_id*] [&display_time_factor_type=*display_time_factor_type*] [&display_time_factor=*display_time_factor*] [&batch_alert_time_cycle=*batch_alert_time_cycle*]	action_mode : server_id : pre_acton_wait : post_acton_delay : action : pulse_logic : pulse_mode : pulse_width : dutycycle : duration : transport : display_time_factor_type : batch_alert_time_cycle : Note : If action_mode is Display Tag Database Record or Display Tag Group Record, the display time is determined by display_time_factor_type and display_time_factor. e.g.1 display_time_factor_type = Additive display time = Tag Duplicate Elimination Window + display_time_factor (ms) e.g.2 display_time_factor_type = Multiplicative display time = Tag Duplicate Elimination Window × display_time_factor (ms) If display_time_factor = 0, record shown on the page forever until next record comes in.	Do Nothing (Only Show on Screen), Batch Alert to Server, Instant Alert to Server, Low Latency Alert to Server, Output Port, Display Tag Database Record, Display Tag Group Record, <Cloud Server> unit = ms unit = ms Port[n]:[0,1,Pulse] where n=1,2,3,4; 0 ==> Open switch, 1 ==> Close switch Positive ==> Open, Close, Open Negative ==> Close, Open, Close (for action=Pulse only) One Shot Pulse, Pulse Train unit = ms unit = % (for Pulse Train only) unit = ms (for Pulse Train only) TCP, HTTP POST, MQTT Additive, Multiplicative unit = ms

<table>
<tr><td></td><td></td><td>result:
<?xml version="1.0" ?>
<CSL>
<Command>modResultantAction</Command>
<Ack>OK:</Ack>
</CSL></td></tr>
<tr><td>12.7</td><td>session_id=<login_session_id>
&
command=delResultantAction&
action_id=action_id</td><td>Remove an events action from <ResultantActionList> table.

e.g.
session_id=<login_session_id>&command=delResultantActio
n&action _id=DemoAction

result:
<?xml version="1.0" ?>
<CSL>
<Command>delResultantAction</Command>
<Ack>OK:</Ack>
</CSL></td></tr>
<tr><td>12.8</td><td>session_id=<login_session_id>
&
command=listResultantAction</td><td>List all events action from <ResultantActionList> table.

e.g.
session_id=<login_session_id>&command=listResultantActio
n

result:
<?xml version="1.0" ?>
<CSL>
<Command>listResultantAction</Command>
<ResultantActionList>
<resultantaction action=""
action_id="DemoAction"
action_mode="Instant Alert to Server"
batch_alert_time_cycle="0"
condition="None"
condition_logic=" "
data_format_id="Example Tag Format"
desc="Demo Action"
display_time_factor="0"
display_time_factor_type=""
duration="0"
dutycycle="0"
post_action_delay="0"
pre_action_wait="0"
pulse_logic=""
pulse_mode=""</td></tr>
</table>

		pulse_width="**0**" server_id="**DemoServer**" transport="**HTTP POST**" /> </ResultantActionList> </CSL>
12.9	session_id=<login_session_id> & command=***addEvent***& event_id=*event_id*& desc=*desc*& operProfile_id=*operProfile_id*& exclusivity=*exclusivity*& duplicateEliminationWindow= *duplicateEliminationWindow*& antennaDifferentiation= *antennaDifferentiation*& triggering_logic=*triggering_logi* *c*& resultant_action=*resultant_actio* *n*& enable=*enable* [&inventoryEnablingTrigger= *inventoryEnablingTrigger* &inventoryDisablingTrigger= *inventoryDisablingTrigger*] [&inventoryEnablingAction= *inventoryEnablingAction* &inventoryDisablingAction= *inventoryDisablingAction*]	Create event definition. e.g. 1 http://192.168.25.160/API?session_id=a33219dc&command=addEvent&event_id=DemoEvent&desc=Demo%20Event&operProfile_id=Default%20Profile&exclusivity=Non-exclusive&duplicateEliminationWindow=10000&antennaDifferentiation=false&triggering_logic=DemoTrigger&resultant_action=DemoAction&enable=true Valid attributes : exclusivity : Exclusive, Non-exclusive duplicateEliminationWindow : unit = ms antennaDifferentiation : true, false enable : true, false triggering_logic : <Read Any Tags (any ID, 1 trigger per tag)>, <Trigger in Tag Group>, <Trigger in Tag Database>, <Trigger if RSSI larger than or equal to>, <Specified Time Span elapsed> resultant_action : NONE, <Do Nothing (Only Show on Screen)>, <Batch Alert to Server>, <Instant Alert to Server>, <Low Latency Alert to Server>, <Output Port>, <Display Tag Database Record>,

		<Display Tag Group Record> inventoryEnablingTrigger : Always On, <Input Sensor State> inventoryDisablingTrigger : Never Stop, <Input Sensor State>, <No Tag Read in Specified Time Span>, <Specified Time Span elapsed> inventoryEnablingAction : NONE, <Output Port> inventoryDisablingAction : NONE, <Output Port> The valid resultant action operation can be used in the *inventoryEnablingAction*, *inventoryDisablingAction* and *resultant_action* attributes are as follows: **AND, THEN** result: <?xml version="1.0" ?> <CSL> <Command>**addEvent**</Command> <Ack>**OK:**</Ack> </CSL>
12.10	session_id=<login_session_id> & command=***modEvent***& event_id=*event_id*& desc=*desc*& operProfile_id=*operProfile_id*& exclusivity=*exclusivity*& duplicateEliminationWindow= *duplicateEliminationWindow*& antennaDifferentiation= *antennaDifferentiation*& triggering_logic=*triggering_logi*	Modify event definition. e.g. 1 http://192.168.25.160/API?session_id=a33219dc&command=modEvent&event_id=DemoEvent&desc=Demo%20Event&operProfile_id=Default%20Profile&triggering_logic=DemoTrigger&resultant_action=DemoAction&event_log=false&enable=true Valid attributes : exclusivity : Exclusive, Non-exclusive duplicateEliminationWindow : unit = ms antennaDifferentiation : true, false

	c& resultant_action=*resultant_action*& enable=*enable* [&inventoryEnablingTrigger= *inventoryEnablingTrigger*] [&inventoryDisablingTrigger= *inventoryDisablingTrigger*] [&inventoryEnablingAction= *inventoryEnablingAction* &inventoryDisablingAction= *inventoryDisablingAction*]	enable :	true, false
		triggering_logic :	<Read Any Tags (any ID, 1 trigger per tag)>, <Trigger in Tag Group>, <Trigger in Tag Database>, <Trigger if RSSI larger than or equal to>, <Specified Time Span elapsed>
		resultant_action :	NONE, <Do Nothing (Only Show on Screen)>, <Batch Alert to Server>, <Instant Alert to Server>, <Low Latency Alert to Server>, <Output Port>, <Display Tag Database Record>, <Display Tag Group Record>,
		inventoryEnablingTrigger :	Always On, <Input Sensor State>
		inventoryDisablingTrigger :	Never Stop, <Input Sensor State>, <No Tag Read in Specified Time Span>, <Specified Time Span elapsed>
		inventoryEnablingAction :	NONE, <Output Port>
		inventoryDisablingAction :	NONE, <Output Port>
		The valid resultant action operation can be used in the *inventoryEnablingAction*, *inventoryDisablingAction* and *resultant_action* attributes are as follows:	

		AND, THEN
		
		result:
		`<?xml version="1.0" ?>`
		`<CSL>`
		`<Command>`**modEvent**`</Command>`
		`<Ack>`**OK:**`</Ack>`
		`</CSL>`

12.11	session_id=<login_session_id>
	&
	command=***enableEvent***&
	event_id=*event_id*&
	enable=*enable*
		Enable/disable an event to be active/inactive.
		
		e.g.
		session_id=<login_session_id>&command=enableEvent&even t _id=DemoEvent&enable=true
		
		Valid attributes :
		enable : true => enable an event to be active
		false => disable an event to be inactive
		
		result:
		`<?xml version="1.0" ?>`
		`<CSL>`
		`<Command>`**enableEvent**`</Command>`
		`<Ack>`**OK:**`</Ack>`
		`</CSL>`

12.12	session_id=<login_session_id>
	&
	command=***delEvent***&
	event_id=*event_id*
		Remove an event definition from the table.
		
		e.g.
		session_id=<login_session_id>&command=delEvent&event _id=DemoEvent
		
		result:
		`<?xml version="1.0" ?>`
		`<CSL>`
		`<Command>`**delEvent**`</Command>`
		`<Ack>`**OK:**`</Ack>`
		`</CSL>`

12.13	session_id=<login_session_id> & command=***listEvent***	List Event definition table. e.g. session_id=<login_session_id>&command=listEvent

		result: <?xml version="1.0" ?> <CSL> <Command>**listEvent**</Command> <EventMode mode="**0**" /> <EventList> <event antennaDifferentiation="**false**" desc="**Event Demo**" duplicateEliminationWindow="**10000**" enable="**true**" event_id="**DemoEvent**" exclusivity="**Non-exclusive**" inventoryDisablingAction="**NONE**" inventoryDisablingTrigger="**Never Stop**" inventoryEnablingAction="**NONE**" inventoryEnablingTrigger="**Always On**" operProfile_id="**Default Profile**" resultant_action="**DemoAction**" triggering_logic="**DemoTrigger**" /> <event antennaDifferentiation="**false**" desc="" duplicateEliminationWindow="**10000**" enable="**true**" event_id="**EventTest**" exclusivity="**Non-exclusive**" inventoryDisablingAction="**NONE**" inventoryDisablingTrigger="**Never Stop**" inventoryEnablingAction="**NONE**" inventoryEnablingTrigger="**Always On**" operProfile_id="**Default Profile**" resultant_action="**NONE**" triggering_logic="**DemoTrigger**" /> </EventList> </CSL>
12.14	session_id=<login_session_id> & command=***addDisplayFormat***& display_format_id=*display_format_id* [&databaseName=*databaseName*] &fieldName{n}=*fieldName* [&displayLabel{n}=*displayLabel*] &topPosition{n}=*topPosition* &leftPosition{n}=*leftPosition* [&fontSize{n}=*fontSize*	Add display format of tag data for displaying on web browser. e.g. session_id=<login_session_id>&command=addDisplayFormat&display_format_id=DBDisplayFormat&databaseName=ProductDB&fieldName1=ProductId&displayLabel1=ID&topPosition1=10&leftPosition1=10&fontSize1=16&fontColor1=%23000000&fieldName2=ProductName&displayLabel2=Name&topPosition2=30&leftPosition2=10&fontSize2=16&fontColor2=%23000000&fieldName3=ProductPrice&displayLabel3=Price&topPosition3=50&leftPosition3=10&fontSize3=16&fontColor3=%23000000&fieldName4=ProductImage&topPosition4=10&leftPosition4=200&imageHeight4=0&imageWidth4=0

	&fontColor{n}=*fontColor*] [&imageHeight{n}=*imageHeigh* *t* &imageWidth{n}=*imageWidth*]	Valid attributes : For displaying database record data : databaseName : must be included fieldName : DatabaseName, Time or the name of field in the database displayLabel : label displayed before data topPosition : unit = px leftPosition : unit = px fontSize : unit = px fontColor : color name (like “red”) or hex code (like “#ff0000”) imageHeight : unit = px, or 0 means auto imageWidth : unit = px, or 0 means auto For displaying tag group data : databaseName : must be ignored fieldName : TagGroupId, Time, TagId displayLabel : label displayed before data topPosition : unit = px leftPosition : unit = px fontSize : unit = px fontColor : color name (like “red”) or hex code (like “#ff0000”) result: <?xml version="1.0" ?> <CSL> <Command>**addDisplayFormat**</Command> <Ack>**OK:**</Ack> </CSL>
12.15	session_id=<login_session_id> & command=***modDisplayFormat*** & display_format_id=*display_form*	Modify display format of tag data for displaying on web browser. e.g. session_id=<login_session_id>&command=modDisplayForma

	at_id [&databaseName=*databaseNam* *e*] &fieldName{n}=*fieldName* [&displayLabel{n}=*displayLabe* *l*] &topPosition{n}=*topPosition* &leftPosition{n}=*leftPosition* [&fontSize{n}=*fontSize* &fontColor{n}=*fontColor*] [&imageHeight{n}=*imageHeigh* *t* &imageWidth{n}=*imageWidth*]	t&display_format_id=DBDisplayFormat&databaseName=Prod uctDB&fieldName1=Time&displayLabel1=Time&topPosition 1=10&leftPosition1=10&fontSize1=16&fontColor1=%230000 00&fieldName2=ProductId&displayLabel2=ID&topPosition2= 30&leftPosition2=10&fontSize2=16&fontColor2=%23000000 &fieldName3=ProductName&displayLabel3=Name&topPositi on3=50&leftPosition3=10&fontSize3=16&fontColor3=%2300 0000&fieldName4=ProductPrice&displayLabel4=Price&topPo sition4=70&leftPosition4=10&fontSize4=16&fontColor4=%23 000000&fieldName5=ProductImage&topPosition5=10&leftPo sition5=200&imageHeight5=0&imageWidth5=200 Valid attributes : For displaying database record data : databaseName : must be included fieldName : DatabaseName, Time or the name of field in the database displayLabel : label displayed before data topPosition : unit = px leftPosition : unit = px fontSize : unit = px fontColor : color name (like “red”) or hex code (like “#ff0000”) imageHeight : unit = px, or 0 means auto imageWidth : unit = px, or 0 means auto For displaying tag group data : databaseName : must be ignored fieldName : TagGroupId, Time, TagId displayLabel : label displayed before data topPosition : unit = px leftPosition : unit = px fontSize : unit = px fontColor : color name (like “red”) or hex code (like “#ff0000”)

<table>
<tr><td></td><td></td><td>result:
<?xml version="1.0" ?>
<CSL>
<Command>addDisplayFormat</Command>
<Ack>OK:</Ack>
</CSL></td></tr>
<tr><td>12.16</td><td>session_id=<login_session_id>
&
command=delDisplayFormat&
display_format_id=display_format_id</td><td>Remove display format.

e.g.
session_id=<login_session_id>&command=delDisplayFormat
&display_format_id=DBDisplayFormatFormat

result:
<?xml version="1.0" ?>
<CSL>
<Command>delDisplayFormat</Command>
<Ack>OK:</Ack>
</CSL></td></tr>
<tr><td>12.17</td><td>session_id=<login_session_id>
&
command=listDisplayFormat</td><td>List display format.

e.g.
session_id=<login_session_id>&command=listDisplayFormat

result:
<?xml version="1.0" ?>
<CSL>
<Command>listDisplayFormat</Command>
<DisplayFormatList>
<displayFormat
databaseName="ProductDB"
display_format_id="DBDisplayFormat"
displayLabel1="Time"
displayLabel2="ID"
displayLabel3="Name"
displayLabel4="Price"
displayLabel5=""
fieldName1="Time"
fieldName2="ProductId"
fieldName3="ProductName"
fieldName4="ProductPrice"
fieldName5="ProductImage"
fontColor1="#000000"
fontColor2="#000000"
fontColor3="#000000"
fontColor4="#000000"
fontColor5="#000000"</td></tr>
</table>

		fontSize1="**16.0**" fontSize1="**16.0**" fontSize1="**16.0**" fontSize1="**16.0**" fontSize1="**16.0**" imageHeight1="**0.0**" imageHeight2="**0.0**" imageHeight3="**0.0**" imageHeight4="**0.0**" imageHeight5="**0.0**" imageWidth1="**0.0**" imageWidth2="**0.0**" imageWidth3="**0.0**" imageWidth4="**0.0**" imageWidth5="**200.0**" leftPosition1="**10.0**" leftPosition2="**10.0**" leftPosition3="**10.0**" leftPosition4="**10.0**" leftPosition5="**200.0**" topPosition1="**10.0**" topPosition2="**30.0**" topPosition3="**50.0**" topPosition4="**70.0**" topPosition5="**10.0**" /> </DataFormatList> </CSL>

13. Version Management

13.1	session_id=<login_session_id>& command=***getReaderVersion***	Get version information of the Reader. e.g. session_id=<login_session_id>&command=getReaderVersion result: <CSL> <Command>**getReaderVersion**</Command> <ReaderVersion cs108_bluetooth_api_library="**1.0.2**" cs461_low_level_api_mach1_library=" **1.0.4**" csl_unified_api_library="**1.0.3**" java=" **1.8.0_221**" jni_library="**1.0.4**" llrp_library="**1.0.7**" os=" **Linux v4.14.78-imx_4.14.78_1.0.0_ga+g 94da7bd**" pcb_version="**2.4**" rfid_firmware=" **2.6.29**" web_application="**1.1.9**" /> </CSL>

14. Legacy CS461 TCP Format to TCP Server

If a resultant action has a TCP transport type defined, tag data will be sent to the server by TCP protocol. The protocol is configurable, except for one legacy format: CS461 TCP Format. The following describes the legacy CS461 TCP Format of tag data.

(1) Tag Data

cmd=evtNtf&evt_id=%s&src_ip=%s&ant=Antenna%d&cp_id=%s&idx=A%d&tag_id=%s&rssi=%d&time=%s&cnt=%d&freq=%d&PC=%04X&usec=%d\n

cmd=evtNtf&evt_id=DemoEvent&src_ip=192.168.25.224&ant=Antenna1&cp_id=CapturePoint1&tag_id=100000000000000000000004&rssi=-35&time=1587524604&freq=924.25&phase=22.5&PC=3000&usec=158000

cmd is the command type, and in this case is event notification;
evt_id is the event ID;
src_ip is the reader IP address;
ant is the antenna port where the tag is received and is of the form ant=Antenna1 or ant=Antenna2 etc;
cp_id is the capture point (alias read point) name;
rssi is the tag rssi in unit of dBm;
time is the time of tag capture based on Linux epoch time.
freq is the frequency in Hz.
phase is the phase in degree.
PC is the protocol control bits.
usec is the micro-seconds part of the time of tag capture based on Linux epoch time.

(2) End of batch message

This message is sent after the last tag data in each packet.

cmd=evtNtf&batchEnd=yes\n

cmd=evtNtf is the command type.

(3) Tag Data with additional bank (bank0, bank2, bank3)

If bank0 is selected in the Active Operation Profile,
cmd=evtNtf&evt_id=%s&src_ip=%s&ant=Antenna%d&cp_id=%s&tag_id=%s&rssi=%d&time=%s&bank0=%s&freq=%s&phase=%s&PC=%04X&usec=%d\n

If bank2 is selected in the Active Operation Profile,
cmd=evtNtf&evt_id=%s&src_ip=%s&ant=Antenna%d&cp_id=%s&tag_id=%s&rssi=%d&time=%s&bank2=%s&freq=%s&phase=%s&PC=%04X&usec=%d\n

If bank3 is selected in the Active Operation Profile,

cmd=evtNtf&evt_id=%s&src_ip=%s&ant=Antenna%d&cp_id=%s&tag_id=%s&rssi=%d&time=%s&bank3=%s&freq=%s&phase=%s&PC=%04X&usec=%d\n