DI-8xx Server

Web API

DATAQ Instruments, Inc.
241 Springside Dr.
Akron, OH 44333

Version: 1.0
Last Modified: 2017-10-09

Version	Date	Description	Change Request ID	Modified By
0.1	2013-10-23	Initial release.		Ioan Popescu
0.2	2013-10-23	Removed Limits API. These can be hard-coded per product.		Ioan Popescu
0.3	2013-10-23	Removed Product/Series. Moved to limits file.		Ioan Popescu
0.4	2013-10-23	Added API to get configuration file. Comments.		Ioan Popescu
0.5	2013-10-29	Added API to get data.		Ioan Popescu
0.6	2013-10-30	Added API to get log. Notes about changing static content.		Ioan Popescu
0.7	2013-11-07	Replaced some APIs with static URLs.		Ioan Popescu
0.8	2013-11-08	Added some file handling support.		Ioan Popescu
0.9	2013-11-12	Added compression factor.		Ioan Popescu
0.10	2013-11-14	Added authentication keep alive.		Ioan Popescu
0.11	2013-11-20	Removed listing of USB files. Separated getting/setting of configuration file into getting/setting each of the new configuration files.		Ioan Popescu
0.12	2013-12-11	Fixed use of "nil" with correct "null".		Ioan Popescu
0.13	2014-01-24	Better permissions indication.		Ioan Popescu
0.14	2014-01-31	Changed API to use WebSockets.		Ioan Popescu
0.15	2014-01-31	Changed Socket.IO channels.		Ioan Popescu
0.16	2014-02-04	Changed use of possibly reserved keywords.		Ioan Popescu
0.17	2014-02-07	Moved some APIs to special URLs.		Ioan Popescu
0.18	2014-02-28	Added APIs and channel for maintaining state.		Ioan Popescu
0.19	2014-04-16	Removed individual alarm, trigger, and session states. Replaced with alarm out states and recording state.		Ioan Popescu
0.20	2014-09-16	Updated WebSocket API to match current implementation.		Ioan Popescu
0.21	2015-06-16	Changed API based on new understandings. May not be complete.		Ioan Popescu
0.22	2015-06-17	Removed getBufferedDataStream as unnecessary.		Ioan Popescu
0.23	2015-06-17	Name change: Update → ApiUpdate		Ioan Popescu
0.24	2015-06-17	Name change: Data → ApiValue		Ioan Popescu
0.25	2015-06-26	WebSocket API change, reduced channels.		Ioan Popescu
0.26	2015-06-29	Replaced API arguments with common value.		Ioan Popescu
0.27	2015-07-22	Corrected old argument references.		Ioan Popescu
0.28	2015-08-12	Disabled security sensitive APIs.		Ioan Popescu
0.29	2015-08-12	Disabled network configuration API.		Ioan Popescu
0.30	2015-09-08	Added missing start/stop API.		Ioan Popescu
0.31	2015-09-23	Added APIs to handle high-level session errors and internal storage usage.		Ioan Popescu
0.32	2015-09-25	Added APIs to handle high-level session successes.		Ioan Popescu
0.33	2015-10-01	Added API to format local storage.		Ioan Popescu

Version	Date	Description	Change Request ID	Modified By
0.34	2015-10-22	Removed USB activity state and added API to reset alarms.		Ioan Popescu
0.35	2015-11-20	Added new API to signal start of session data stream.		Ioan Popescu
0.36	2015-12-09	Updated support/access of some special URLs.		Ioan Popescu
0.37	2015-12-17	Updated alarm outs to be tri-state.		Ioan Popescu
0.38	2016-01-08	Added URL for web GUI JSON file.		Ioan Popescu
0.39	2016-02-02	Added new API to signal connection of administrators.		Ioan Popescu
0.40	2016-02-09	Added URL for version JSON file.		Ioan Popescu
0.41	2016-02-09	Added URLs for default configuration files.		Ioan Popescu
0.42	2016-02-15	Changed URL of version JSON file.		Ioan Popescu
1.0	2016-09-30	Updated to version 1.0 for production release.		Ioan Popescu
1.1	2017-10-09	Enabled access to accounts configuration special URL.		Ioan Popescu

Table of Contents

1 Introduction

This document explains and describes the API for the DI-8xx server series of products. Unless otherwise noted, the API protocol will use JSON formatting. JSON was selected as the protocol format for its generic nature, compatibility with JavaScript, and textual representation. WebSockets will be used as the transport for the API.

Each special URL or API shall indicate the permissions necessary for access as follows:

- anonymous: non-authenticated/any users
- authenticated: any currently logged in users
- administrative: any currently logged in and administrative users

Authentication is handled using browser-based HTTP Digest authentication.

2 Special URLs

2.1 /version.json

"GETting" from this URL will return the current version information in JSON format. It can be accessed regardless of permissions or authentication.

`GET` Permissions: n/a

2.2 /configuration/

2.2.1 current.network.json

"`GET`ting" from this URL will return the current network configuration in JSON format. "`POST`ing" to this URL will save the provided and JSON formatted network configuration to the device. If the supplied configuration is valid and successfully applied, the old configuration shall be overwritten; otherwise, the supplied configuration shall be ignored and the old one restored. Changing the network configuration shall result in disconnection of all users.

`GET` Permissions: authenticated
`POST` Permissions: administrative

2.2.2 current.general.json

"`GET`ting" from this URL will return the current general configuration in JSON format. "`POST`ing" to this URL will save the provided and JSON formatted general configuration to the device. If the supplied configuration is valid and successfully applied, the old configuration shall be overwritten; otherwise, the supplied configuration shall be ignored and the old one restored. If the supplied configuration is written, all Socket.IO clients are disconnected.

`GET` Permissions: authenticated
`POST` Permissions: administrative

2.2.3 current.users.json

"`GET`ting" from this URL will return the current users configuration in JSON format. "`POST`ing" to this URL will save the provided and JSON formatted users configuration to the device. If the supplied configuration is valid and successfully applied, the old configuration shall be overwritten; otherwise, the supplied configuration shall be ignored and the old one restored.

`GET` Permissions: administrative
`POST` Permissions: administrative

2.2.4 current.accounts.json

"GETting" from this URL will return the current accounts configuration in JSON format. "POSTing" to this URL will save the provided and JSON formatted accounts configuration to the device. If the supplied configuration is valid and successfully applied, the old configuration shall be overwritten; otherwise, the supplied configuration shall be ignored and the old one restored.

GET Permissions: administrative
POST Permissions: administrative

2.2.5 current.network.json

"GETting" from this URL will return the current network configuration in JSON format. "POSTing" to this URL will save the provided and JSON formatted network configuration to the device. If the supplied configuration is valid and successfully applied, the old configuration shall be overwritten; otherwise, the supplied configuration shall be ignored and the old one restored. Changing the network configuration shall result in disconnection of all users.

GET Permissions: authenticated
POST Permissions: administrative

2.2.6 default.general.json

"GETting" from this URL will return the default general configuration in JSON format.

GET Permissions: authenticated

2.2.7 default.users.json

"GETting" from this URL will return the default users configuration in JSON format.

GET Permissions: authenticated

2.2.8 default.accounts.json

"GETting" from this URL will return the default accounts configuration in JSON format.

GET Permissions: authenticated

2.2.9 default.network.json

"GETting" from this URL will return the default network configuration in JSON format.

GET Permissions: authenticated

2.2.10 limits.json

"GETting" from this URL will return a set of limits for the device and configuration in JSON format.

GET Permissions: authenticated

2.2.11 web.gui.json

"GETting" from this URL will return the current web GUI configuration in JSON format. "POSTing" to this URL will save the provided and JSON formatted web GUI configuration to the device. There is no special treatment of this file, other than a reasonable limit to its size (100kb) and that it's a valid JSON file.

`GET` Permissions: authenticated
`POST` Permissions: administrative

2.3 /di8xx.log.zip

This URL will always serve all log data as a ZIP-compressed archive file. The log is continually appended and older entries are removed based on internal settings of age and size.

GET Permissions: administrative

NOTE: not to be used directly by the web interface, but only as directed in special cases.

2.4 /data/files/

2.4.1 local/

Web server generated list of data files stored internally. It is recommended to be displayed in a container so that navigation remains on the page. Individual files may be downloaded using the generated links. All files may be deleted using appropriate API. (`GET` only)

Permissions: authenticated

2.4.2 usb/

Web server generated list of all files stored on external USB storage (if one is plugged in). It is recommended to be displayed in a container so that navigation remains on the page. Individual files may be downloaded using the generated links. (`GET` only)

Permissions: authenticated

3 API Channel

The API is handled via WebSockets using JSON objects in the Socket.IO channel `apiChannel`.

API calls are of the general form:
`{"ApiCall":"...","ApiValue":...}`

Members:
- **`ApiCall`**: The name of the API call.
- **`ApiValue`**: API-specific.

API returns are communicated via acknowledgments to the original API call and are of the general form:
`{"ApiReturn":"...","ApiError":"..."}`
`{"ApiReturn":"...","ApiValue":...}`

Members:
- **`ApiReturn`**: The API call returning a value, or "Unknown" if the API doesn't match.
- **`ApiError`**: Exists only when the return indicates an error; this member being the error description string. May be shown directly to the user.
- **`ApiValue`**: Exists only when the return indicates success; this member being API-specific, or `null` for generic/simple success.

Updates may be pushed by the server asynchronously to clients and are of the general form:
`{"ApiUpdate":"...","ApiValue":...}`

Members:

- **`ApiUpdate`**: Name of the update; see following subsections.
- **`ApiValue`**: Update-specific.

3.1 API Calls

The following sections describe possible API calls.

3.1.1 `start`

Allows recording triggers to be evaluated (if defined).

Permissions: administrative

3.1.2 `stop`

Prevents recording triggers from being evaluated (even if defined).

Permissions: administrative

3.1.3 `getStatusState`

Retrieves the current status of the device.

The return `ApiValue` is a string.

Return values:
- `Idle`: device isn't processing trigger conditions.
- `Armed`: device is processing trigger conditions, but not recording data.
- `Recording`: device is processing trigger conditions and recording data.
- `Error`: device encountered an error and stopped the session.
- `UsbActivity`: the external USB storage is in use; it is not safe to remove it during this time.

Permissions: authenticated

3.1.4 `reboot`

Reboots device, no warning, hard reboot. Will require re-authentication when reboot complete.

Call example:
```
{"ApiCall":"reboot"}
```

Permissions: administrative

3.1.5 `getDateTime`

Gets the current device date and time in ISO-8601 formatting.

The return `ApiValue` is a string containing the ISO-8601 formatted date and time.

Return example using UTC timezone:
```
{"ApiReturn":"getDateTime","ApiValue":"2013-10-16T16:00:00.000Z"}
```

Permissions: authenticated

3.1.6 `setDateTime`

Sets (explicitly or initiates a synchronization to the configured NTP server) the current device date and time in ISO-8601 formatting (UTC timezone expected).

`ApiValue`:
- `null`: Initiates synchronization of device clock with configured NTP server.
- `"..."`: ISO-8601 formatted string.

Call example using specific date/time (UTC timezone expected):
```
{"ApiCall":"setDateTime","ApiValue":"2013-10-16T16:00:00Z"}
```

Permissions: administrative

3.1.7 `pingDevice`

Tests network connectivity to the provided hostname or IP address by sending a ping and checking for a response. Note: this is primarily used to test connectivity to a NTP server; it doesn't indicate successful use of it, only that it can be reached by pinging.

`ApiValue`:
- `"..."`: Hostname or IP address to ping.

Call example using a public NTP server:
```
{"ApiCall":"pingDevice","ApiValue":"pool.ntp.org"}
```

Permissions: administrative

3.1.8 `getAlarmOutStates`

Retrieves all enabled alarm out states.

The return `ApiValue` is an array of booleans/null. The array index is the alarm out channel and the value is the boolean state of that alarm, or `null` if the alarm is not set. This API shall return the state of all alarm out channels.

Return example:
```
{"ApiReturn":"getAlarmOutStates","ApiValue":[true,false,null,false]}
```

Permissions: authenticated

3.1.9 `getSessionErrors`

Gets a list of the most recent high-level errors for this session.

The return `ApiValue` is an array of objects with members indicating `timestamp` (number) and `description` (string) for each error.

Return example using UTC timezone:
```
{"ApiReturn":"getSessionErrors","ApiValue":[{"timestamp":0,"description":""}]}
```

Permissions: authenticated

3.1.10 `getSessionSuccesses`

Gets a list of the most recent high-level successes for this session.

The return `ApiValue` is an array of objects with members indicating `timestamp` (number) and `description` (string) for each success.

Return example using UTC timezone:
{**"ApiReturn"**:"getSessionSuccesses",**"ApiValue"**:[{"timestamp":0,"description":""}]}

Permissions: authenticated

3.1.11 `getInternalStorageUsage`

Gets the usage statistics of the internal storage (amount of used/available space).

The return `ApiValue` is an object with members indicating `used` (number) and `available` (number) space in bytes.

Return example using UTC timezone:
{**"ApiReturn"**:"getInternalStorageUsage","**ApiValue**":{"used":0,"available":0}}

Permissions: authenticated

3.1.12 `formatLocalStorage`

Formats the internal local storage, thereby reclaiming space for further data acquisition.

Permissions: administrative

3.1.13 `resetAlarm`

Resets the indicated alarm out (useful for resetting alarms that are held); fails if alarm out is not set.

`ApiValue`:
- `...`: Zero-based alarm out number, limited by total alarms.

Call example:
{**"ApiCall"**:"resetAlarm","ApiValue":0}

Permissions: administrative

3.1.14 `streamData`

Signals the device to start streaming the session data stream. The stream is stopped whenever the websocket connection is disconnected (such as when reloading the web page or posting a new configuration). The session data stream is will only be sent when this call is made to allow the website to function over limited connections without interference from the bandwidth heavy streaming.

Permissions: authenticated

3.2 Updates

The following sections describe possible updates from the device.

3.2.1 `statusState`

Issued whenever the current status of the device changes.

The `ApiValue` value is a string.

Data values:

- `Idle`: device isn't processing trigger conditions.
- `Armed`: device is processing trigger conditions, but not recording data.
- `Recording`: device is processing trigger conditions and recording data.
- `Error`: device encountered an error and stopped the session.

Permissions: authenticated

3.2.2 alarmOutStates

Issued whenever an alarm out state changes.

The `ApiValue` value is an array of booleans/null. The array index is the alarm out channel and the value is the boolean state of that alarm, or `null` if the alarm is not set. This API shall include the state of all alarm out channels.

Update example:
```
{"ApiUpdate":"alarmOutStates","ApiValue":[true,false,null,false]}
```

Permissions: authenticated

3.2.3 sessionDataStream

Issued whenever there is new data. This is not the recorded data, it's a stream of data solely to be displayed to the user.

The `ApiValue` value is an `ArrayBuffer` of scans as doubles (to use, convert to `Float64Array`). A scan contains one sample from each channel.

Permissions: authenticated

3.2.4 sessionError

Issued whenever there is a high-level session error.

The `ApiValue` value is an object with members indicating `timestamp` (number) and `description` (string) for the error.

Update example:
```
{"ApiUpdate":"sessionError","ApiValue":{"timestamp":0,"description":""}}
```

Permissions: authenticated

3.2.5 sessionSuccess

Issued whenever there is a high-level session success.

The `ApiValue` value is an object with members indicating `timestamp` (number) and `description` (string) for the success.

Update example:
```
{"ApiUpdate":"sessionSuccess","ApiValue":{"timestamp":0,"description":""}}
```

Permissions: authenticated

3.2.6 adminConnected

Issued whenever an administrative user connects via web socket.

The `ApiValue` value is a string indicating the username of the connecting administrative user. There's no indication when an administrative user disconnects. Web socket connections may be frequent from the same user.

Update example:
{"**ApiUpdate**":"adminConnected","**ApiValue**":"Administrator"}

Permissions: authenticated

Alphabetical Index