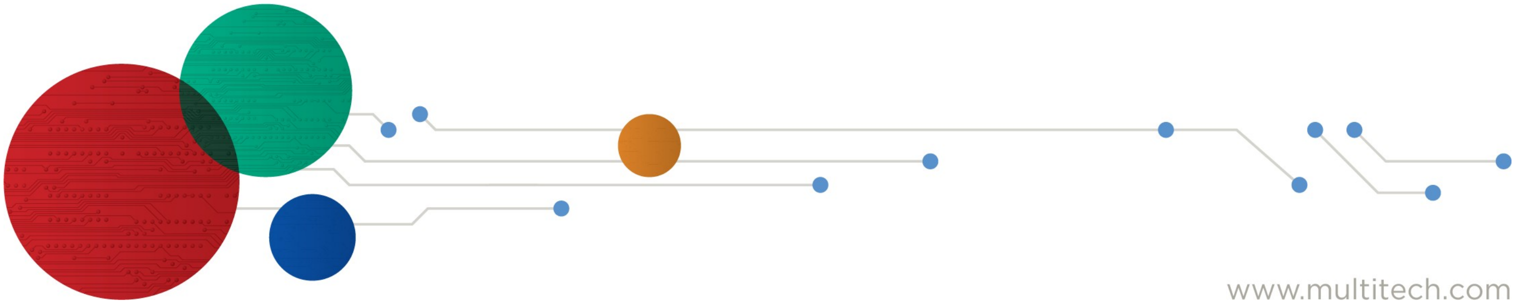
MULTITECH®

DeviceHQ®

DeviceHQ®

API Guide



www.multitech.com

DeviceHQ Reference Guide

Part Number: S000769, Version 1.1

Copyright

Multi-Tech Systems, Inc. makes no representations or warranties, whether express, implied or by estoppels, with respect to the content, information, material and recommendations herein and specifically disclaims any implied warranties of merchantability, fitness for any particular purpose and non-infringement.
Multi-Tech Systems, Inc. reserves the right to revise this publication and to make changes from time to time in the content hereof without obligation of Multi-Tech Systems, Inc. to notify any person or organization of such revisions or changes.

Legal Notices

The MultiTech products are not designed, manufactured or intended for use, and should not be used, or sold or re-sold for use, in connection with applications requiring fail-safe performance or in applications where the failure of the products would reasonably be expected to result in personal injury or death, significant property damage, or serious physical or environmental damage. Examples of such use include life support machines or other life preserving medical devices or systems, air traffic control or aircraft navigation or communications systems, control equipment for nuclear facilities, or missile, nuclear, biological or chemical weapons or other military applications ("Restricted Applications"). Use of the products in such Restricted Applications is at the user's sole risk and liability.

Contacting MultiTech

Knowledge Base

The Knowledge Base provides immediate access to support information and resolutions for all MultiTech products. Visit http://www.multitech.com/kb.go.

Support Portal

To create an account and submit a support case directly to our technical support team, visit: https://support.multitech.com.

Support

Business Hours: M-F, 8am to 5pm CT

Country	By Email	By Phone
Europe, Middle East, Africa:	support@multitech.co.uk	+(44) 118 959 7774
U.S., Canada, all others:	support@multitech.com	(800) 972-2439 or (763) 717-5863

Warranty

To read the warranty statement for your product, visit www.multitech.com/warranty.go. For other warranty options, visit www.multitech.com/es.go.

World Headquarters

Multi-Tech Systems, Inc.

2205 Woodale Drive, Mounds View, MN 55112 Phone: (800) 328-9717 or (763) 785-3500

Fax (763) 785-9874

Contents

DeviceHQ Account API

Starting with release 3.0, DeviceHQ has supported a json-format, token-authorized account API designed for automated access to account and device information for any user with a valid DeviceHQ account.

Prerequisites

To use the API, you need an enabled, valid DeviceHQ account. You do not need to own any MultiTech devices or have any devices checked in to the DeviceHQ server.

To enable API access to their account information, users must generate an **Account API Key** and enable API access.

Enabling API Access

To enable API access:

1. Log in to devicehq.com with manager-level access.
2. Click on user email displayed in upper right corner.
3. Click **Account Info** and then click **Edit.**
4. Check Account API enabled.
5. Click **Update Account** and return to the account edit screen

Generating API Keys

An account API key is like a password for your account on deviceHQ.com The account API key (not the same as the account key) allows the API client to access the account with the same privileges as a manager-level user belonging to that account.

To generate API keys:

1. Log in to devicehq.com with manager-level access.
2. Click on user email displayed in upper right corner.
3. Click **Account Info** and then click **Edit.**
4. Click Generate New Account API Key (as needed).

New keys will be generated for access to the account API.

NOTES:

- MultiTech recommends you save the generated keys in a safe place, as they will not be displayed again as they are stored using a 1-way cryptographic hash algorithm.
- New keys may be generated at any time.
- Generating new keys will invalidate any key that had previously been generated.
- At any given time, only 1 client per account can have a session open.

Format

The [json-api](http://jsonapi.org/format/) specification will be used for all communication to and from the account API, with the exception of uploading and downloading files, which use typical **Send-File** http messages.

DeviceHQ supports a slightly relaxed standard of json-api for client requests, with the following deviations from strict json-api:

- It is not be necessary for requests targeting the API to have the header Content-Type fields set to "application/vnd.api+json". Clients may use this heade, or set their Content-Type to "application/json".
- Clients may use *either* PUT or PATCH requests for updating resources. Json-api requires the use of PATCH only for updates, but since this may not be possible for all clients, either PUT or PATCH will be accepted.

All responses *from* the server will be jsonapi 1.0 compliant.

Example

PUT /api/v2/devices/12345678 HTTP/1.1

Content-Type: application/vnd.api+json

Accept: application/vnd.api+json

```
{
  "data": {
    "type": "devices",
    "id": "12345678",
    "attributes": {
      "description": "Test device"
    }
  }
}
```

Authentication

Authentication consists of two steps:

1. Creating a session.
2. Passing the session token with each subsequent request.

To access the API, create a session by submitting a valid account API key, which will return a session token that can be used for subsequent requests. For help generating account API keys, refer to *Generating API Keys*

Create a session by making a POST or PUT request with a header field set to "X-API-KEY" or a URL parameter "api_key" set to the account API key. For example, any of the following three requests would create a session.

NOTE: You must set the Content-Length header for any POST request. Many http clients will automatically include this header field on any POST request. If using the first form listed below (no data field sent), you must set a header to "Content-Length: 0".

Example Account API Key: 5sHszZNuTfmgs58VGA

```
POST /api/v2/session?api_key=5sHszZNuTfmgs58VGA
Headers: "Content-Length: 0"
curl -H "Content-Type:application/json" -X POST
"https://www.devicehq.com/api/v2/session?api_key=5sHszZNuTfmgs58VGA" -H "Content-
Length:0"
```

OR

```
POST /api/v2/session
{ "api_key": "5sHszZNuTfmgs58VGA" }
curl -H "Content-Type:application/json" -X POST https://www.devicehq.com/api/v2/session
-d'{"api_key": "5sHszZNuTfmgs58VGA"}'
```

OR

```
POST /api/v2/session
Headers: "X-API-KEY: 5sHszZNuTfmgs58VGA"
curl -H "Content-Type:application/json" -X POST https://www.devicehq.com/api/v2/session
-H "X-API-KEY: 5sHszZNuTfmgs58VGA"
```

Creating a valid new session closes any existing session for the account that was open. The new session replaces it. Accounts can only have one open session at a time.

- The response to any of the above requests will be json formatted as follows:

```
"meta":{
  "token": "<session_token>",
  "expires":"<ISO formatted date string >",
  "jsonapi": {"version":"1.0"}
}
```

The token is a short string of approximately 14 characters. Subsequent requests to the API will be honored if the token is included either as a URL request parameter "token" or as a header field "X-AUTH-TOKEN"

- Sessions expire 5 minutes after the last request to the server.
- The session may also be closed with the following route (the token must be included as with all other requests):

```
DELETE /api/v2/session
```

Routes

All account API access will be done with routes starting with

```
/api/v2
```

For example, to display information for a single device, the HTTPS request below yields a json-formatted string containing device information for a device with a serial # of 12341234:

```
GET https://devicehq.com/api/v2/devices/12341234
```

Devices may be specified by their UUID or serial number.

NOTE:

- The UUID must contain exactly 32 hexadecimal characters.
- Any whitespace or non-alphanumeric characters will be stripped.
- It is not case sensitive.

The following routes are available in the API:

- `GET /api/v2/info`

  Returns basic info about the api. Authentication not required. Useful for testing if api/v2 is successfully reached, and is up and running.

- `GET /api/v2/test`

  Session and authentication required. Useful for testing if client is establishing session and reaching api. Returns "OK" and timestamp if successfully reached.

- `GET /api/v2/account`

  Returns account information for present account

- `GET /api/v2/account/users`

  Returns list of users for the present account

- `GET /api/v2/devices`

  Returns list of devices. Supports search and pagination (see below for more details)

- `GET /api/v2/devices/(serial or uuid)`

  Returns single device identified by serial# or UUID. Will include all fields that are presently visible when user clicks on a device in DeviceHQ.com

- `GET /api/v2/devices/(serial or uuid)/check_ins`

  Returns all checkins for a device going back 30 days

- `GET /api/v2/devices/(serial or uuid)/signal_strength`

  Returns reported signal strength information for a device going back 30 days

- `GET /api/v2/devices/(serial or uuid)/network`

  Returns network usage for a device going back 30 days

- `GET /api/v2/devices/(serial or uuid)/lora`

  Returns LoRa card information for a device going back 30 days

- `GET /api/v2/devices/(serial or uuid)/logs`

  Returns last uploaded (tar/zip formatted) device logs

- `GET /api/v2/devices/(serial or uuid)/configs`

  Returns last uploaded (tar/zip formatted) configuration for the device

- `PUT /api/v2/devices/(serial or uuid)`

  Accepts updated information for device and returns the device. Only fields changeable in DeviceHQ can be changed in the API

- `POST /api/v2/devices/register {serial: 12345678, uuid:"776334cb-8e42-4839-835c-899d075d63ae"}`

  Registers a device so as to be associated with the current account, allowing the "call-home" feature to work

- `DELETE /api/v2/devices/(serial or uuid)`

  Removes a device and all associated records from the database of DeviceHQ. If the device subsequently checks in, it will be recreated in the same account

Search and Pagination

Devices can be searched and paginated within the /api/v2/devices route. This search is similar to searches on the DeviceHQ website.

- A query parameter ("q") will match against various fields of the devices collection
- Limit and skip parameters may be included. If no limit parameter is included, it will default to 500

Examples:

- Retrieves the first 30 devices for an account

  ```
  GET https://devicehq.com/api/v2/devices?limit=30
  ```

- Retrieves devices 30 through 49 for an account:

  ```
  GET https://devicehq.com/api/v2/devices?limit=20&skip=30
  ```

- Retrieves all the devices for an account matching on the phrase "MTCDT":

  ```
  GET https://devicehq.com/api/v2/devices?q=MTCDT
  ```

HTTP Codes

The API will always respond with 1 of 3 different http codes:

- `200 - OK`
- `404 - Not found - when a specific resource is requested (a device by its uuid for example) and the resource is not found`
- `423 - unprocessable entity : any other error such as not being logged in, using incorrect tokens`

Examples

Test

- `GET /api/v2/test`

  No session or authentication required. Returns "OK" and timestamp if successfully reached. Useful for testing if api/v2 is successfully reached, and is up and running.

```
{
  "meta": {
    "message": "OK",
    "timestamp": "2017-06-03T13:06:45.592Z",
  }
}
```

Info

- `GET /api/v2/info`

  Returns basic info about the API. Useful for testing if client is establishing session and reaching api.

```
{
  "meta": {
    "version": "v3.0.2.6",
    "url": "http://example.org/api/v2/info"
  },
  "jsonapi": {
    "version"=>1.0
  }
}
```

Account

- `GET /api/v2/account`

  Returns account information for present account.

```
{
  "data": {
    "id": "jeff@ez-tech.com", # a comment
    "type": "accounts",
    "attributes": {
      "name": "EZ-TECH",
      "address1": "1704 Standish Ave",
      "address2": "",
      "city": "Woodsville",
      "state": "AK",
      "postal_code": "32118",
      "country": "United States",
      "phone": "706-242-2222",
      "email": "jeff@ez-tech.com",
      "website": "https://ez-tech.biz",
      "alerts_config": {
        "enabled": "1",
        "low_signal": {
          "enabled": "1",
          "threshold": "-100",
          "recipient_groups": [
            ""
          ],
          "max_notifications_per_day": "daily",
          "email": "1",
          "sms": "0",
          "details": "Signal < -100 dBm"
        },
        "missed_checkin": {
          "enabled": "1",
          "recipient_groups": [
            ""
          ],
          "max_notifications_per_day": "every_checkin",
          "email": "1",
          "sms": "0",
          "details": "device fails to check in"
        },
        "high_data_usage": {
          "enabled": "1",
          "recipient_groups": [
            ""
          ],
          "max_notifications_per_day": "every_checkin",
          "email": "1",
          "sms": "0",
          "details": "data usage > threshold"
        },
        "failed_actions": {
          "enabled": "1",
          "recipient_groups": [
            ""
          ],
          "max_notifications_per_day": "every_checkin",
          "email": "1",
          "sms": "0",
          "details": "scheduled actions failed to execute"
        },
        "device_reboot": {
          "enabled": "1",
          "recipient_groups": [
            ""
          ],
          "max_notifications_per_day": "every_checkin",
          "email": "1",
          "sms": "0",
          "details": "device rebooted"
        },
        "recipient_groups": {
          "1": {
            "name": "t"
          },
          "2": {
            "name": "rr"
          }
        }
      },
      "enabled": true,
      "confirmed": true
    }
  }
}
```

Users

- `GET /api/v2/account/users`

  Returns list of users for the present account.

```
{
  "data": {
    "id": "jeff@ez-tech.com",
    "type": "accounts",
    "attributes": {
      "users": [
        {
          "email": "jeff@ez-tech.com",
          "first_name": "jeff",
          "last_name": "Wilson",
          "last_login_at": "2016-09-20T15:48:02.790+00:00",
          "created_at": "2014-11-20T15:53:26.000Z",
          "enabled": true,
          "role": "manager"
        },
        {
          "email": "guest_user@ez-tech.com",
          "first_name": "Guest",
          "last_name": "User",
          "last_login_at": "2015-06-23T19:44:26.647+00:00",
          "created_at": "2014-12-02T15:42:24.000Z",
          "enabled": true,
          "role": "basic"
        }
      ],
      "name": "EZ-TECH",
      "email": "jeff@ez-tech.com",
      "created_at": "2014-11-20T15:53:25.000Z"
    }
  }
}
```

Devices

- `GET /api/v2/devices?limit=1`

  Returns list of devices. Supports search and pagination.

```
{
  "data": [
    {
      "id": "MTS-JP-103",
      "type": "devices",
      "attributes": {
        "product": "MTR-H5",
        "group": "",
        "serial": "MTS-JP-103",
        "provider": "AT&T",
        "description": "",
        "firmware": "3.0.9",
        "position": null,
        "last_checkin": "2015-12-15T20:30:17.129+00:00",
        "active_apps": null,
        "dhcp": {
          "enabled": true,
          "range_start": "192.168.2.100",
          "range_end": "192.168.2.254",
          "lease_time": 86400,
          "netmask": "255.255.255.0",
          "gateway": "192.168.2.1",
          "dns": "192.168.2.1"
        },
        "enabled": true,
        "hardware_version": "REV C",
        "lora_network": {},
        "ethernet": {
          "name": "eth0",
          "link_type": "LINK_TYPE_ETHER",
          "mtu": 1500,
          "mac": "5B:3C:1D:BF:0F:E4",
          "gateway": "0.0.0.0",
          "name_servers": [],
          "flags": {
            "up": true,
            "lower_up": true,
            "loopback": false,
            "broadcast": true,
            "pointtopoint": false,
            "multicast": false,
            "dynamic": false,
            "noarp": false,
            "allmulti": false,
            "promisc": false
          },
          "link_statistics": {
            "rx_bytes": 69985,
            "rx_packets": 551,
            "rx_errors": 48,
            "rx_dropped": 6,
            "rx_overrun": 9,
            "rx_multicast": 0,
            "tx_bytes": 72739,
            "tx_packets": 572,
            "tx_errors": 43,
            "tx_dropped": 2,
            "tx_carrier": 9,
            "tx_collisions": 0
          }
        },
        "cell_radio": {
          "firmware": "12.00.024",
          "phone_number": "1107069877",
          "provider": "AT&T",
          "rssi": -107,
          "imsi": null,
          "up_time": null
        },
        "wifi": {
          "name": "wlan0",
          "link_type": "LINK_TYPE_ETHER",
          "mtu": 1500,
          "ip_addr": "192.168.235.19",
          "netmask": "255.255.255.0",
          "gateway": "192.168.235.1",
          "name_servers": [],
          "flags": {
            "up": true,
            "lower_up": true,
            "loopback": false,
            "broadcast": true,
            "pointtopoint": false,
            "multicast": false,
            "dynamic": false,
            "noarp": false,
            "allmulti": false,
            "promisc": false
          },
          "link_statistics": {
            "rx_bytes": 119803,
            "rx_packets": 943,
            "rx_errors": 41,
            "rx_dropped": 1,
            "rx_overrun": 8,
            "rx_multicast": 0,
            "tx_bytes": 338173,
            "tx_packets": 2662,
            "tx_errors": 77,
            "tx_dropped": 5,
            "tx_carrier": 5,
            "tx_collisions": 0
          },
          "wifi": {
            "mode": "client",
            "client": {
              "mode": "WAN",
              "security_mode": "WPA2-PSK",
              "ssid": "ez-test",
```

~~~~
              "rssi": -85
            }
          }
        },
        "up_time": " 127 days",
        "up_time_seconds": null,
        "supports" : {
          "logs" : true,
          "configs" : false,
          "apps" : false,
        },
        "has_radio": true
      }
    }
  ],
  "meta": {
    "skip": 0,
    "limit": 1,
    "total": 67
  }
}
```
~~~~

Device

- `GET /api/v2/devices/(serial or uuid)`

  Returns single device identified by serial# or UUID, includes all fields presently visible when user clicks on a device in DeviceHQ.com

```
{
  "result": "ok",
  "token": "i9eTvNmbinALex",
  "data": {
    "id": "MTS-JP-003",
    "type": "devices",
    "attributes": {
      "product": "MTR-H5",
      "group": "Group 00X",
      "serial": "MTS-JP-003",
      "provider": "AT&T",
      "description": "East wing of library building",
      "location_description": [-27.6234,93.2],
      "software_version": "3.0.9",
      "hardware_version": "1.2b",
      "last_checkin": "2015-12-15T20:30:13.107+00:00",
```

```
"enabled": true,
"hardware_version": "REV C",
"lora_network": {},
"active_apps": [
  { "status": "READY",
    "info": "aaaa",
    "name": " Hello World",
    "version": "1.0.1",
    "type": "app",
    "id": "57ee3aa4ed3e892698000057"
  },
  { "status": "READY",
    "info": null,
    "name":"APPTEST",
    "version": "22.22",
    "type": "app",
    "id": "581cb7dbed3e89c60100004c"
  }
],
"dhcp": {
  "enabled": true,
  "range_start": "192.168.2.100",
  "range_end": "192.168.2.254",
  "lease_time": 86400,
  "netmask": "255.255.255.0",
  "gateway": "192.168.2.1",
  "dns": "192.168.2.1"
},
"ethernet": {
  "name": "eth0",
  "link_type": "LINK_TYPE_ETHER",
  "mtu": 1500,
  "mac": "F7:F8:86:0D:68:91",
  "gateway": "0.0.0.0",
  "name_servers": [],
  "flags": {
    "up": true,
    "lower_up": true,
    "loopback": false,
    "broadcast": true,
    "pointtopoint": false,
    "multicast": false,
    "dynamic": false,
    "noarp": false,
    "allmulti": false,
    "promisc": false
  },
  "link_statistics": {
    "rx_bytes": 74498,
    "rx_packets": 586,
    "rx_errors": 58,
    "rx_dropped": 1,
    "rx_overrun": 8,
    "rx_multicast": 0,
    "tx_bytes": 73870,
    "tx_packets": 581,
    "tx_errors": 77,
    "tx_dropped": 1,
    "tx_carrier": 5,
    "tx_collisions": 0
  }
},
"cell_radio": {
  "firmware": "12.00.024",
  "phone_number": "1914591206",
  "provider": "AT&T",
  "rssi": -103,
  "imsi": null,
  "up_time": null
},
"wifi": {
  "name": "wlan0",
  "link_type": "LINK_TYPE_ETHER",
  "mtu": 1500,
  "ip_addr": "192.168.211.211",
  "netmask": "255.255.255.0",
  "gateway": "192.168.211.1",
  "name_servers": [],
  "flags": {
    "up": true,
    "lower_up": true,
    "loopback": false,
    "broadcast": true,
    "pointtopoint": false,
    "multicast": false,
    "dynamic": false,
    "noarp": false,
    "allmulti": false,
      "promisc": false
    },
    "link_statistics": {
      "rx_bytes": 123939,
      "rx_packets": 975,
      "rx_errors": 50,
      "rx_dropped": 6,
      "rx_overrun": 8,
      "rx_multicast": 0,
      "tx_bytes": 287959,
      "tx_packets": 2267,
      "tx_errors": 75,
      "tx_dropped": 6,
      "tx_carrier": 4,
      "tx_collisions": 0
    },
    "wifi": {
      "mode": "client",
      "client": {
        "mode": "WAN",
        "security_mode": "WPA2-PSK",
        "ssid": "mts-test",
        "rssi": -75
      }
    }
  },
  "up_time": " 127 days",
  "up_time_seconds": 23422,
  "supports" : {
    "logs" : true,
    "configs" : false,
    "apps" : false,
  },
  "has_radio": true
}
}
}
```

Update Device

Although a large amount of information is available when reading a device through the API, only a limited set of items can be updated through it. This is because the device directly executes updates when it checks in and updates DeviceHQ regarding its state.

Fields updatable through the api/v2 interface include description, group, and location_description. These can be viewed at devicehq.com, under the device/edit page for a given device. Description and group are simply arbitrary strings.

- `"location_description" should be an array of [latitude, longitude]`

- `PUT /api/v2/devices/(serial or uuid)`

  Accepts updated information for device and returns the device. Only fields changeable in DeviceHQ can be changed in the API

Example:

```
PUT /api/v2/devices/12345678
Content-Type: application/json
Accept: application/json
```

```
{
  "data": {
    "id": "12345678",
    "type": "devices",
    "attributes": {
      "description": "Device installed in building 33"
      "location_description": [44.211, -92.12],
      "group": "product test"
    }
  }
}
```

Network

- `GET /api/v2/devices/(serial or uuid)/network`

  Returns network usage statistics.

  Parameters:

  - type: (cell|ethernet|wifi) Default: cell
  - start_date - defaults to 10 days ago
  - end_date - defaults to tomorrow (so as to get all data)
  - dates must be YYYY-MM-DD formatted

```
{
  "id": "12345678",
  "type": "devices",
  "description": "99889908",
  "product_id": "MTCAP-LEU1-868-001L",
  "vendor_id": "Multi-Tech Systems",
  "device_id": "99889908",
  "attributes": {
    "network_usages": [
      {
        "2017-01-15": {
          "cell": {
            "rx_bytes": 0,
            "tx_bytes": 0
          }
        }
      },
      {
        "2017-01-16": {
          "cell": {
            "rx_bytes": 20997455,
            "tx_bytes": 35255516
          },
          "ethernet": {
            "rx_bytes": 4236739667,
            "tx_bytes": 4281006485
          }
        }
      }
    ]
  }
}
```

Signal Strength History

- `GET /api/v2/devices/(serial or uuid)/signal_strength`

  Returns history of cell radio signal strength.

  Parameters:

  - start_date - defaults to 10 days ago
  - end_date - defaults to tomorrow (so as to get all data)
  - dates must be YYYY-MM-DD formatted

```
{
  "id": "MTS-JP-501",
  "type": "devices",
  "attributes": {
    "signal_strength": [
      {
        "timestamp": "2017-01-09T16:02:18.626+00:00",
        "rssi": -77,
        "fer": null
      },
      {
        "timestamp": "2017-01-26T16:02:15.918+00:00",
        "rssi": -71,
        "fer": null
      }
    ],
    "description": "",
    "product_id": "MTR-G3",
    "vendor_id": "MultiTech",
    "device_id": "MTS-JP-501"
  }
}
```

Check-in History

- `GET /api/v2/devices/(serial or uuid)/check_ins`

  Returns information about check-ins the device has made.

  Parameters:

  - start_date - defaults to 10 days ago
  - end_date - defaults to tomorrow (so as to get all data)
  - dates must be YYYY-MM-DD formatted

```
{
  "id": "77779514",
  "type": "devices",
  "description": "Agricultural data collector",
  "product_id": "MTCDT-H5-210A",
  "vendor_id": "Multi-Tech Systems",
  "device_id": "77779514",
  "attributes": {
    "check_ins": [
      {
        "timestamp": "2017-01-17T19:02:44.213+00:00",
        "action": "upload_log",
        "result": "200",
        "ssl": true,
        "duration": 0
      },
      {
        "timestamp": "2017-01-17T19:02:44.281+00:00",
        "action": "update",
        "result": "200",
        "ssl": true,
        "duration": 0
      }
    ]
  }
}
```

LoRa Network

- `GET /api/v2/devices/(serial or uuid)/lora`

  If the device has a LoRa network card installed, returns information about it, including usage and node connections.

  Parameters:

  - start_date - defaults to 10 days ago
  - end_date - defaults to tomorrow (so as to get all data)
  - dates must be YYYY-MM-DD formatted

```
{
  "id": "77778211",
  "type": "devices",
  "description": "",
  "product": "MTCDT-H5-210L",
  "vendor": "Multi-Tech Systems",
  "device": "77778211",
  "attributes": {
    "lora": {
      "nodes": [
        {
          "node_addr": "00:00:00:01",
          "dev_eui": "00-80-00-00-00-00-9c-3d",
          "class": "A",
          "joined": "2016-09-14T17:17:04.000Z",
          "seq_num": 770,
          "pkts_up": 377,
          "pkts_down": 0,
          "pkts_1st": 0,
          "pkts_2nd": 0,
          "dropped": 0,
          "rssi_min": -57,
          "rssi_max": -35,
          "rssi_avg": -42,
          "snr_min": 9.800000190734863,
          "snr_max": 14,
          "snr_avg": 12.199999809265137,
          "per": 0
        }
      ],
      "link_statistics": {
        "rx_mic_fails": 36420,
        "rx_crc_errors": 75275,
        "rx_dups": 202,
        "rx_join_dups": 964,
        "rx_joins": 36,
        "rx_pkts": 123616,
        "tx_ack_dups": 0,
        "tx_ack_pkts": 0,
        "tx_joins_1st_wnd": 36,
        "tx_joins_2nd_wnd": 0,
        "tx_joins_dropped": 0,
        "tx_pkts": 36,
        "tx_pkts_1st_wnd": 36,
        "tx_pkts_2nd_wnd": 0,
        "tx_pkts_dropped": 0
      },
      "enabled": true,
      "running": true
    },
  }
}
```

Logs

- `GET /api/v2/devices/(serial or uuid)/logs`

  Returns devices log file information (if it exists), and a link to retrieve the .tar.gz file containing the device logs.

  Parameters:

  - start_date - defaults to 10 days ago
  - end_date - defaults to tomorrow (so as to get all data)
  - dates must be YYYY-MM-DD formatted

```
{
  "id": "77778211",
  "type": "devices",
  "attributes": {
    "log": {
      "uploaded_at": "09-23-16 11:12:29UTC",
      "md5sum": "69ed906fbec08f0ee7d66550e0f590ec",
      "description": "09-23-16_11:12:29UTC.zip"
    }
  },
  "links": {
    "self": "https://devicehq.com/api/v2/files/57e50e1d11c3a8fd12711fff"
  }
}
```

Configs

- `GET /api/v2/devices/(serial or uuid)/configs`

  Returns device configuration file information, and a link to retrieve the .tar.gz file containing it.

  Parameters:

  - start_date - defaults to 10 days ago
  - end_date - defaults to tomorrow (so as to get all data)
  - dates must be YYYY-MM-DD formatted

```
{
  "id": "77778332",
  "type": "devices",
  "attributes": {
    "config": {
      "md5sum": "ADFA1341234131341234AFDDDD",
      "name": "config22",
      "description": "Config #22"
    }
  },
  "links": {
    "self": "https://devicehq.com/api/v2/files/5894cd7f95b54a75e732f952"
  }
}
```

Register

- `POST /api/v2/devices/register {serial: 12345678, uuid:"776334cb-8e42-4839-835c-899d075d63ae"}`

  You can register a device if you know its serial # and UUID. This associates the device with your account, so that when it checks in to DeviceHQ.com, you and other account users will be able to manage it. It is not necessary to register the device once it appears in your device list on DeviceHQ.com.

  Parameters:

  - serial - the serial # of the device
  - uuid - the uuid (32 characters) of the device. The uuid must contain exactly 32 hexadecimal characters. Any whitespace or non-alphanumeric characters will be stripped. It is not case sensitive.

```
{
  "id": "77778332",
  "type": "devices",
  "attributes": {
    "config": {
      "md5sum": "ADFA1341234131341234AFDDDD",
      "name": "config22",
      "model": "MTCDT-210A",
      "description": "Config #22"
    }
  },
  "links": {
    "self": "https://devicehq.com/api/v2/files/5894cd7f95b54a75e732f952"
  }
}
```

Account API Extensions for Task Management

The following allows DeviceHQ users to create and manage tasks for devices in their DeviceHQ account using an API. Apps, firmware, or other files the tasks need must be uploaded through the normal DeviceHQ web portal at https://www.devicehq.com. If a file ID for a firmware, app, or other dependency is needed, the following also defines plain routes that fetch the items of interest.

Applies To

API task management extensions apply to any device in the account. Some task types are not supported by some device types. Current firmware level and other factors can affect whether creating a task succeeds or not for a given device. A descriptive error message will be returned if the task creation request fails.

Task Types

- Install app on device
- Uninstall app from device
- Install app config on device
- Install full device configuration onto device
- Install full device firmware upgrade onto device
- Install full radio firmware upgrade

New Endpoints

- `GET /api/v2/devices/:id/tasks`

  Returns a list of all tasks for a device. They will be listed in descending order by creation time.

- `POST /api/v2/devices/:id/tasks`

  Creates a task for a device. It will be handled by the device on the next subsequent check-in when it is able to do so.

- `GET /api/v2/devices/:id/tasks/:task_id`

  Returns detailed information about the type, status, return codes, etc for the task.

- `GET /api/v2/devices/:id/apps`

  Returns detailed information about the apps installed on the given device.

- `PUT/PATCH /api/v2/devices/:id/tasks/:task_id/cancel`

  Cancels the task. If the device has not yet received the task, then the device never receives the task. If the task is being processed, the device sees the cancelled state and gracefully attempts to complete or if possible, cancel the task.

- `DELETE /api/v2/devices/:id/tasks/:task_id`

  Completely destroys the task. If the task is in pending state, this may cause it to fail if the device is currently processing the task.

- `GET /api/v2/apps/:id`

  Shows more details such as available versions and available app configs.

- `GET /api/v2/apps`

  A list of all apps on the account. Provides ID and name for each.

- `GET /api/v2/device-firmwares`

  A list of all device firmwares on the account. Provides id, name, version for each.

- `GET /api/v2/radio-firmwares`

  A list of all radio firmwares on the account. Provides id, name, device model, version for each.

- `GET /api/v2/device-configurations`

  A list of all device configurations on the account. Provides id, name, version for each.

Limitations

The total number of new, not completed tasks that can be created at one time per account is limited to 25.

Examples

Example 1 Set up a task for a device to install a custom app on its next check-in.

1. User logs in to www.devicehq.com and uploads a custom app.
2. API client contacts the API and does GET /api/v2/apps.
3. Within the retrieved list, the account client fetches the ID field from the desired app.
4. The API client makes a request: POST /api/v2/devices/{device- id}/task, with the payload completed with the necessary data (app id, app version, etc.). The request succeeds.
5. The device checks in to DeviceHQ
6. The device retrieves the task, and completes it by installing the app.

Example 2 Set up a task for a device to install an app config

1. User logs in to www.devicehq.com and uploads a custom app.
2. API client contacts API and does GET /api/v2/apps
3. Within the retrieved list the account client fetches the ID field from the desired app.
4. The account client makes a request: GET /api/v2/apps/{app id}, and examines its app_configs field to find the ID of the desired app config.
5. The API client makes a request: POST /api/v2/devices/{device- id}/task, with the payload completed with the necessary data containing the app config ID. The request succeeds.
6. The device checks in to DeviceHQ
7. The device carries out the command to install the requested app config.

Detailed Request Examples

Retrieve Tasks for a Device

GET /api/v2/devices/996874991/tasks

RESPONSE:

```
{
  "data": [
    {
      "id": "60d0a5eb31eddd285865a781",
      "type": "tasks",
      "attributes": {
        "function": "UNINSTALL_APP",
        "created_at": "2021-06-21T14:44:59.247Z",
        "app_id": "60d0aabc31eddd2de579995c"
        "state": "queued"
      }
    }
  ]
}
```

Create an Install App Task

POST /api/v2/devices/997512397/tasks

```
{
  "data": {
    "type": "tasks",
    "attributes": {
      "type": "INSTALL_APP",
      "app_id": "60d0a5eb31eddd285865a784"
    }
  }
}
```

RESPONSE:

```
{
  "data": {
    "id": "60d0a5eb31eddd285865a78d",
    "type": "tasks",
    "attributes": {
      "function": "INSTALL_APP",
      "created_at": "2021-06-21T14:44:59.663Z",
      "app": {
        "name": "app_37278085",
        "version": "1.4.7",
        "config_ids": "",
        "id": "60d0a5eb31eddd285865a784"
      }
    }
  }
}
```

Create an Install App Task on a Device, Specifying App Version

```
POST /api/v2/devices/996082621/tasks
```

```
{
  "data": {
    "type": "tasks",
    "attributes": {
      "type": "INSTALL_APP",
      "app_id": "60d0dece31eddd4d97e2e549",
      "version": "1.2.0"
    }
  }
}
```

RESPONSE:

```
{
  "data": {
    "id": "60d0dece31eddd4d97e2e554",
    "type": "tasks",
    "attributes": {
      "function": "INSTALL_APP",
      "created_at": "2021-06-21T18:47:42.625Z",
      "state": "queued",
      "app_id": "60d0dece31eddd4d97e2e549",
      "version": "1.2.0",
      "app": {
        "name": "app_45854484",
        "version": "1.2.0",
        "config_ids": "",
        "id": "60d0dece31eddd4d97e2e549"
      }
    }
  }
}
```

Create an Install App Task on a Device, Specifying App Version and Configuration

This example creates an install app task on a device, specifys the app version, and the app configurations to apply after the install. App configurations are specified by ID in the config_ids array.

```
POST /api/v2/devices/997270969/tasks
```

```
{
  "data": {
    "type": "tasks",
    "attributes": {
      "type": "INSTALL_APP",
      "app_id": "60d0e0d631eddd4f5a2f5d85",
      "version": "4.3.3",
      "config_ids": [
        "60d0e0d631eddd4f5a2f5d8c"
      ]
    }
  }
}
```

RESPONSE

```
{
  "data": {
    "id": "60d0e0d631eddd4f5a2f5d90",
    "type": "tasks",
    "attributes": {
      "function": "INSTALL_APP",
      "created_at": "2021-06-21T18:56:22.638Z",
      "state": "queued",
      "app_id": "60d0e0d631eddd4f5a2f5d85",
      "config_ids": "60d0e0d631eddd4f5a2f5d8c",
      "app": {
        "name": "app_56448630",
        "version": "4.3.3",
        "config_ids": "60d0e0d631eddd4f5a2f5d8c",
        "id": "60d0e0d631eddd4f5a2f5d85"
      },
      "version": "4.3.3"
    }
  }
}
```

Create an Uninstall App Task on a Device

```
POST /api/v2/devices/997512397/tasks
```

```
{
  "data": {
    "type": "tasks",
    "attributes": {
      "type": "UNINSTALL_APP",
      "app_id": "60d0a5eb31eddd285865a784"
    }
  }
}
```

RESPONSE:

```
{
  "data": {
    "id": "60d0dc1631eddd4bc3b5ac66",
    "type": "tasks",
    "attributes": {
      "function": "UNINSTALL_APP",
      "created_at": "2021-06-21T18:36:06.295Z",
      "state": "queued",
      "app_id": "60d0dc1631eddd4bc3b5ac5d",
      "app": {
        "name": "app_60196004",
        "version": "0.3.7",
        "config_ids": null,
        "id": "60d0dc1631eddd4bc3b5ac5d"
      }
    }
  }
}
```

Create an Upgrade Firmware Task on a Device

```
POST /api/v2/devices/997414120/tasks
```

```
{
  "data": {
    "type": "tasks",
    "attributes": {
      "type": "SOFTWARE_UPGRADE",
      "firmware_id": "60db20dc31eddd3a911f90b5"
    }
  }
}
```

RESPONSE:

```
{
  "data": {
    "id": "60db20dc31eddd3a911f90b6",
    "type": "tasks",
    "attributes": {
      "function": "SOFTWARE_UPGRADE",
      "created_at": "2021-06-29T13:32:12.490Z",
      "state": "queued",
      "firmware_id": "60db20dc31eddd3a911f90b5",
      "version": "4.8"
    }
  }
}
```

Create an Upgrade Radio Firmware Task on a Device

```
POST /api/v2/devices/992798923/tasks
```

```
{
  "data": {
    "type": "tasks",
    "attributes": {
      "type": "RADIO_FIRMWARE_UPGRADE",
      "firmware_id": "60db22fa31eddd3c1e89867e"
    }
  }
}
```

RESPONSE:

```
{
  "data": {
    "id": "60db22fa31eddd3c1e89867f",
    "type": "tasks",
    "attributes": {
      "function": "RADIO_FIRMWARE_UPGRADE",
      "created_at": "2021-06-29T13:41:14.941Z",
      "state": "queued",
      "firmware_id": "60db22fa31eddd3c1e89867e",
      "version": "4.8"
    }
  }
}
```

Create an Apply Configuration Task on a Device

```
POST /api/v2/devices/995356139/tasks
```

```
{
  "data": {
    "type": "tasks",
    "attributes": {
      "type": "CONFIG_UPGRADE",
      "config_file_id": "60db20df31eddd3a911f90f8"
    }
  }
}
```

RESPONSE:

```
{
  "data": {
    "id": "60db20df31eddd3a911f910e",
    "type": "tasks",
    "attributes": {
      "function": "CONFIG_UPGRADE",
      "created_at": "2021-06-29T13:32:15.831Z",
      "state": "queued",
      "config_file_id": "60db20df31eddd3a911f90f8"
    }
  }
}
```

Create an Apply App Configuration Task on a Device

```
POST /api/v2/devices/995560841/tasks
```

```
{
  "data": {
    "type": "tasks",
    "attributes": {
      "type": "INSTALL_CONFIGS",
      "config_ids": [
        "60db225031eddd3bb1dcc43e"
      ]
    }
  }
}
```

RESPONSE:

```
{
  "data": {
    "id": "60db225131eddd3bb1dcc445",
    "type": "tasks",
    "attributes": {
      "function": "INSTALL_CONFIGS",
      "created_at": "2021-06-29T13:38:25.481Z",
      "state": "queued",
      "app_id": "60db225031eddd3bb1dcc43a",
      "config_ids": "60db225031eddd3bb1dcc43e",
      "app": {
        "name": "app_28832400",
        "version": "*",
        "config_ids": "60db225031eddd3bb1dcc43e",
        "id": "60db225031eddd3bb1dcc43a"
      },
      "app_configs": [
        {
          "id": "60db225031eddd3bb1dcc43e",
          "description": "ac desc 0.211886161970272",
          "install_requested_at": "2021-06-29T13:28:25.000+00:00",
          "version": "0.4.5"
        }
      ],
      "version": "*"
    }
  }
}
```

Retrieve Detailed Information about a Single Task on a Device

```
GET /api/v2/devices/992741055/tasks/60d0aabb31eddd2de579992c
```

```
{
  "data": {
    "id": "60d0aabb31eddd2de579992c",
    "type": "tasks",
    "attributes": {
      "function": "INSTALL_APP",
      "state": "queued",
      "created_at": "2021-06-21T15:05:31.162Z",
      "app": {
        "name": "app_11148179",
        "version": "3.9.6",
        "config_ids": "",
        "id": "60d0aabb31eddd2de5799923"
      }
    }
  }
}
```

Cancel a Task Created for a Device

```
PUT/PATCH /api/v2/devices/993608179/tasks/60d3463531eddd136e3f154c/cancel
```

```
{
  "data": {
    "id": "60d3463531eddd136e3f154c",
    "type": "tasks",
    "attributes": {
      "function": "INSTALL_APP",
      "created_at": "2021-06-23T14:33:25.145Z",
      "state": "cancelled",
      "app_id": "60d3463431eddd136e3f1542",
      "app": {
        "name": "app_90535018",
        "version": "7.1.3",
        "config_ids": "",
        "id": "60d3463431eddd136e3f1542"
      },
      "version": "7.1.3"
    }
  }
}
```

Delete a Task on a Device

```
DELETE /api/v2/devices/992821453/tasks/60d0a3ce31eddd271866d2ba
```

RESPONSE:

```
NO CONTENT (204)
```

Retrieve Detailed Information about Apps Installed on a Device

```
GET /api/v2/devices/992557995/apps
```

RESPONSE:

```
{
  "data": {
    "type": "devices",
    "id": "992557995",
    "attributes": {
      "id": "992557995",
      "description": "",
      "product": "MTCDT-H5-210L",
      "vendor": "Multi-Tech Systems Inc.",
      "apps": [
        {
          "name": "app_37278085",
          "id": "60d0a5eb31eddd285865a784",
          "version": "0.2.6",
          "status": "Active",
          "type": "custom",
          "info": "What the app reported."
        }
      ]
    }
  }
}
```

Retrieve Detailed App Information

```
GET /api/v2/apps/60d098a631eddd1f6a725d23
```

RESPONSE:

```
{
  "data": {
    "id": "60d098a631eddd1f6a725d23",
    "type": "apps",
    "attributes": {
      "name": "app_92595685",
      "versions": [
        {
          "id": "60d098a631eddd1f6a725d24",
          "version": "9.6.3",
          "description": "description for version 9.6.3",
          "status": "Active"
        }
      ],

      "app_configs": [
        {
          "id": "60d098a631eddd1f6a725d27",
          "version": "0.3.6",
          "created_at": null,
          "description": "description for app config version 0.3.6"
        }
      ],
      "platform": "python",
      "status": "Active",
      "description": null
    }
  }
}
```

Retrieve a List of Apps Belonging to Your Account

`GET /api/v2/apps`

RESPONSE:

```
{
  "data": [
    {
      "id": "60d098a631eddd1f6a725d19",
      "type": "apps",
      "attributes": {
        "name": "app_30225500"
      }
    }
  ]
}
```

Retrieve a List of Device Firmwares that are Public or Belong to Your Account

`GET /api/v2/device-firmwares`

RESPONSE:

```
{
  "data": [
    {
      "id": "60d09a5731eddd2125ca1c26",
      "type": "firmwares",
      "attributes": {
        "sub_type": "device",
        "upload_date": "2021-05-21T13:55:35.975+00:00",
        "version": "4.8",
        "model": "MTCDT",
        "radio_module": null
      }
    }
  ]
}
```

Retrieve a List of Radio Firmwares that are Public or Belong to Your Account

`GET /api/v2/radio-firmwares`

RESPONSE:

```
{
  "data": [
    {
      "id": "60d09a5731eddd2125ca1c29",
      "type": "firmwares",
      "attributes": {
        "sub_type": "cell_radio",
        "upload_date": "2021-05-21T13:55:35.987+00:00",
        "version": "4.8",
        "model": "MTCDT",
        "radio_module": "HE910-D"
      }
    }
  ]
}
```

Retrieve a List of Device Configurations Belonging to Your Account

```
GET /api/v2/device-configurations
```

RESPONSE:

```
{
  "data": [
    {
      "id": "60d0a12f31eddd245ccc9322",
      "type": "device_configurations",
      "attributes": {
        "name": "config 67120",
        "description": "description for config 67120",
        "model": "MTCDT-H5-210L"
      }
    }
  ]
}
```

Error Responses

All responses that result in errors have a statue > 400. The payload in the response contains an errors array. Each item in the errors array consists of an object with at least the detail field present with a human-readable error explanation:

```
{
  "errors": [
    {
      "detail": "Cannot create new command because account is already at command limit
(25)."
    }
  ]

}
```