FIWARE IoT Agents API

DATE: December 2016

This version:

https://telefonicaid.github.io/iotagent-node-lib/api/stable

Latest version:

https://telefonicaid.github.io/iotagent-node-lib/api/latest

Copyright

License

Table of Contents

API Summary

Authentication and Authorization

If the IoT Agent is in authenticated environment, this API requires a token, which you obtain from authentication system. This system and its API is out of scope of present documentation. In this environment, a mandatory header is needed: `X-Auth-Token`.

Mandatory HTTP Headers

The API needs two headers in order to manage requests:

Http Header	Level	If not present	Observations
Fiware-Service	Represents a tenant. Higher level in resources hierachy in IoT Platform	An error is returned	Must only contain less than 50 Underscores and Alphanumeric lowercase characters
Fiware-ServicePath	Represents the second level	We assume '/'. Allowed operation '/*'	Must only contain less than 50 Underscores and Alphanumeric characters. Must start with character '/'. Max. length is 51 characters (with /)

API Access

All API base URLs are relative and depend on where the IoT Agent is listening (depends on each service). For example, `http://127.0.0.1:8080/iot/`.

API Specification

Configuration API

Services [/services{?apikey,device,limit,offset,resource}]

Services are the higher level in IoT Platform. When you manipulate a service, you use a Fiware-Service header with its name. Parameters apply to different operations.

Fields in JSON object representing a service are:

Fields in JSON Object	
`apikey`	It is a key used for devices belonging to this service. If "", service does not use apikey, but it must be specified.
`token`	If authentication/authorization system is configured, IoT Agent works as user when it publishes information. That token allows that other components to verify the identity of IoT Agent. Depends on authentication and authorization system.
`cbroker`	Context Broker endpoint assigned to this service, it must be a real uri.
`outgoing_route`	It is an identifier for VPN/GRE tunnel. It is used when device is into a VPN and a command is sent.
`resource`	Path in IoTAgent. When protocol is HTTP a device could send information to this uri. In general, it is a uri in a HTTP server needed to load and execute a module.
`entity_type`	Entity type used in entity publication (overload default).
`attributes`	Mapping for protocol parameters to entity attributes. `object_id` (string, mandatory): protocol parameter to be mapped. `name` (string, mandatory): attribute name to publish. `type`: (string, mandatory): attribute type to publish.
`static_attributes`	Attributes published as defined. `name` (string, mandatory): attribute name to publish. `type` (string, mandatory): attribute type to publish. `value` (string, mandatory): attribute value to publish.

Mandatory fields are identified in every operation.

`static_attributes` and `attributes` are used if device has not this information.

Retrieve a service

GET /services{?apikey,device,limit,offset,resource}

With Fiware-ServicePath you can retrieve a subservice or all subservices.

Parameters [limit, offset, resource]

`limit` (optional, number). In order to specify the maximum number of services (default is 20, maximun allowed is 1000).

`offset` (optional, number). In order to skip a given number of elements at the beginning (default is 0) .

`resource` (optional, string). URI for the iotagent, return only services for this iotagent.

Create a service

POST /services{?apikey,device,limit,offset,resource}

With one subservice defined in Fiware-ServicePath header. From service model, mandatory fields are: apikey, resource (cbroker field is temporary mandatory).

Update a service/subservice

PUT /services{?apikey,device,limit,offset,resource}

If you want modify only a field, you can do it. You cannot modify an element into an array field, but whole array. ("/*" is not allowed).

Parameters [apikey, resource]

`apikey` (optional, string). If you don't specify, apikey=" " is applied.

`resource` (mandatory, string). URI for service into iotagent.

Remove a subservice/service

DELETE /services{?apikey,device,limit,offset,resource}

You remove a subservice into a service. If Fiware-ServicePath is '/*' or '/#' remove service and all subservices.

Parameters [apikey, resource, device]

`apikey` (optional, string). If you don't specify, apikey="" is applied.

`resource` (mandatory, string). URI for service into iotagent.

`device` (optional, boolean). Default value is false. Remove devices in service/subservice. This parameter is not valid when Fiware-ServicePath is '/*' or '/#'.

Devices **[/devices{?detailed,entity,limit,offset,protocol}]**

A device is a resource that publish information to IoT Platform and it uses the IoT

Agent.

Device Model

`device_id`. Unique identifier into a service.

`protocol`. Protocol assigned to device. This field is easily provided by IoTA Manager if it is used. Every module implmenting a protocol has an identifier.

`entity_name`. Entity name used for entity publication (overload default)

`entity_type`. Entity type used for entity publication (overload entity_type defined in service).

`timezone`. Not used in this version.

`attributes`. Mapping for protocol parameters to entity attributes. `object_id` (string, mandatory): protocol parameter to be mapped. `name` (string, mandatory): attribute name to publish. `type`: (string, mandatory): attribute type to publish.

`static_attributes` (optional, array). Attributes published as defined. `name` (string, mandatory): attribute name to publish. `type` (string, mandatory): attribute type to publish. `value` (string, mandatory): attribute value to publish.

`endpoint` (optional, string): when a device uses push commands.

`commands` (optional, array). Attributes working as commands. `name` (string, mandatory): command identifier. `type` (string, mandatory). It must be 'command'. `value` (string, mandatory): command representation depends on protocol.

Mandatory fields are identified in every operation.

Retrieve all devices

GET /devices{?detailed,entity,limit,offset,protocol}

Parameters [limit, offset, detailed, entity, protocol]

`limit` (optional, number). In order to specify the maximum number of devices (default is 20, maximun allowed is 1000).

`offset` (optional, number). In order to skip a given number of elements at the beginning (default is 0) .

`detailed` (optional, string). `on` return all device information, `off` (default) return only name.

`entity` (optional, string). It allows get a device from entity name.

`protocol` (optional, string). It allows get devices with this protocol.

Create a device

POST /devices{?detailed,entity,limit,offset,protocol}

From device model, mandatory fields are: device_id and protocol.

Device **[/devices/{device_id}]**

Retrieve a device

GET /devices/{device_id}

Update a device

PUT /devices/{device_id}

If you want modify only a field, you can do it, except field `protocol` (this field, if provided it is removed from request).

Remove a device

DELETE /devices/{device_id}

If specific device is not found, we work as deleted.

Examples

Configuration API

Services

[/services{?apikey,device,limit,offset,resource}]

Retrieve a service

GET /services{?apikey,device,limit,offset,resource}

Request /services (application/json)

Headers

```
Content-Type: application/json
Fiware-Service: testservice
Fiware-ServicePath: /*
```

Response 200

Body

```
{
 "count": 1,
 "services": [
   {
      "apikey": "apikey3",
      "service": "service2",
      "service_path": "/srvpath2",
      "token": "token2",
      "cbroker": "http://127.0.0.1:1026",
      "entity_type": "thing",
      "resource": "/iot/d"
   }
 ]
}
```

Request /services (application/json)

Headers

```
Content-Type: application/json
Fiware-Service: testservice
Fiware-ServicePath: /TestSubservice
```

Response 200

Body

```
{
 "count": 1,
 "services": [
   {
      "apikey": "apikey3",
      "service": "service2",
      "service_path": "/srvpath2",
      "token": "token2",
      "cbroker": "http://127.0.0.1:1026",
      "entity_type": "thing",
      "resource": "/iot/d"
   }
 ]
}
```

Create a service

POST /services{?apikey,device,limit,offset,resource}

Request /services (application/json)

Headers

Headers

```
Content-Type: application/json
Fiware-Service: testservice
Fiware-ServicePath: /TestSubservice
```

Body

```
{
  "services": [
    {
      "apikey": "apikey3",
      "token": "token2",
      "cbroker": "http://127.0.0.1:1026",
      "entity_type": "thing",
      "resource": "/iot/d"
    }
  ]
}
```

Response 201

Update a service/subservice

PUT /services{?apikey,device,limit,offset,resource}

Request /services (application/json)

Headers

```
Content-Type: application/json
Fiware-Service: testservice
Fiware-ServicePath: /TestSubservice
```

Body

```
{
   "entity_type": "entity_type"
}
```

Response 204

Remove a subservice/service

DELETE /services{?apikey,device,limit,offset,resource}

Request /services (application/json)

Headers

```
Content-Type: application/json
Fiware-Service: testservice
Fiware-ServicePath: /TestSubservice
```

Response 204

Devices

[/devices{?detailed,entity,limit,offset,protocol}]

Retrieve all devices

GET /devices{?detailed,entity,limit,offset,protocol}

Request /devices (application/json)

Headers

```
Content-Type: application/json
Fiware-Service: testService
Fiware-ServicePath: /TestSubservice
```

Response 200

Body

```
{
  "count": 1,
  "devices": [
    {
      "device_id": "device_id",
      "protocol": "12345",
      "entity_name": "entity_name",
      "entity_type": "entity_type",
      "timezone": "America/Santiago",
      "attributes": [
        {
          "object_id": "source_data",
          "name": "attr_name",
          "type": "int"
        }
      ],
      "static_attributes": [
        {
          "name": "att_name",
          "type": "string",
          "value": "value"
        }
      ]
    }
  ]
}
```

Create a device

POST /devices{?detailed,entity,limit,offset,protocol}

Request /devices (application/json)

Headers

```
Content-Type: application/json
Fiware-Service: testservice
Fiware-ServicePath: /TestSubservice
```

Body

```
{
    "devices": [
    {
      "device_id": "device_id",
      "protocol": "12345",
      "entity_name": "entity_name",
      "entity_type": "entity_type",
      "timezone": "America/Santiago",
      "attributes": [
        {
          "object_id": "source_data",
          "name": "attr_name",
          "type": "int"
        }
      ],
      "static_attributes": [
        {
          "name": "att_name",
          "type": "string",
          "value": "value"
        }
      ]
    }
    ]
 }
```

Response 201

Body

```
+ Headers (only if ONE device is in request)

    Location: /iot/devices/device_id
```

Device

[/devices/{device_id}]

Retrieve a device

GET /devices/{device_id}

Request /devices/{device_id} (application/json)

Headers

```
Content-Type: application/json
Fiware-Service: testservice
Fiware-ServicePath: /TestSubservice
```

Response 200

Body

```
{
 "device_id": "device_id",
 "protocol": "121345",
 "entity_name": "entity_name",
 "entity_type": "entity_type",
 "timezone": "America/Santiago",
 "attributes": [
  {
   "object_id": "source_data",
   "name": "attr_name",
   "type": "int"
  }
 ],
 "static_attributes": [
  {
   "name": "att_name",
   "type": "string",
   "value": "value"
  }
 ]
}
```

Update a device

PUT /devices/{device_id}

Request /devices/{device_id} (application/json)

Headers

```
Content-Type: application/json
Fiware-Service: testservice
Fiware-ServicePath: /TestSubservice
```

Body

```
{
  "entity_name": "entity_name"
}
```

Response 204

Remove a device

DELETE /devices/{device_id}

Request /devices/{device_id} (application/json)

Headers

```
Content-Type: application/json
Fiware-Service: testservice
Fiware-ServicePath: /TestSubservice
```

Response 204

References

Apiary project (http://docs.telefonicaiotiotagents.apiary.io/#reference)

Github source (https://github.com/telefonicaid/iotagent-node-lib)

FIWARE Open Specification License (implicit patent license) (https://forge.fiware.org/plugins/mediawiki/wiki/fiware/index.php/Implicit_Patents_License)

IoT Agent (https://github.com/telefonicaid/fiware-IoTAgent-Cplusplus)