CONNECTING DEVELOPERS. DEVICES. APIS

connect2.me API Documentation 2015

70°
65% chance of rain
Pack an umbrella!
574 Pedestrians crossed this hour
Energy consumed 15% less than last month
Bulb life low change soon!
75 Park Check-ins Today
Train 5 minutes behind scheudle
5,624 steps taken today

Document Contents

An Overview

Authentication

Authentication Overview

Check user credentials

Companies

Companies Overview

Get all companies

Channels

Channels Overview

Get published channels

Get published channel schema

Onboard an API type channel

Register a device type channel

Channel Instantiation

Channel Instantiation Overview

Get all subscribed channels

Get subscribed channel status

Set channel status

Feed

Feed Overview

Get channel feeds

Get channel feed status

Set channel feed status

Delete channel feed

Push feed to channel

Get data from channel

Get a subscribed channel

Delete subscribed channel

connect2.meAPI Overview

What is connect2.me™ ?

connect2.me™ is an easy to use, dynamic Internet of Things connector that integrates all of your devices and APIs on your application without hundreds of development hours.

A Unique Connection to the Devices & Data - Connect to all enterprise infrastructure including smart devices, sensors, homegrown systems and third party SaaS/Social platforms without complex integrations.

Simple, Real-Time Access - The Data Feeds dynamic connection to connect2.me™ provides and produces consumable, mashed data while connect2.me™ manages the device, data and security.

Simplicity within the Cloud - A simple, intuitive, flexible, cloud-based platform that incorporates a simple Plug & Play interface that allows for easily accessible integration with smart IoT sources.

How connect2.me™ works?

connect2.me™ acts as an IOT connector middleware, which can enable connection to any kind of device or API dynamically. It replaces the traditional model of creating and coding adaptors to talk to various devices and/or API's. connect2.me™ provides the layers that can talk to hundreds and thousands of devices and API's.

connect2.me™ has the inbuilt intelligence that can sense the native protocol, data format and the native support of those devices. In terms of extraction of data, creating a reliable connection, securing the connection as well as the ability to control the device. Based on the native capability of the device without putting a client on it, connect2.me™ can enable it to exist in its ecosystem and instantiate hundreds or thousand times for a particular implementation. It not only provides the connection, but can also do analytics and visualization.

Who is the audience?

This developers guide is intended for developers and device manufacturers.

Schema

All APIs access is over HTTPS, and all resources are accessed using the following URL:
https://ice.connect2.me

All data is received in XML format.

All timestamps are received and returned in ISO 8601 format, with 3 decimal fractions of a second: YYYY-MM-DDTHH:MM:SS:nnnZ

Parameters

Most parameters used to identify a specific resource are passed as http query strings or request headers.

"https://ice.connect2.me/ice.svc/GetPublishChannels?companyid={companyid}&apikey={apikey}"

In the above example, the parameters companyid (company id) and apikey (api key) are passed into the query string variables to retrieve the list of published channels for a company.

API Errors

connect2.me™ uses standard HTTP status codes to indicate success or failure of an API request. connect2.me™ also returns a status message (Success or Fail).

HTTP Status Code Summary

HTTP Status Code	Description
200	OK - Request has succeeded.
201	Created - Request has been fulfilled and a new resource created.
202	Accepted - The request has been accepted for processing, but the processing will be completed asynchronously
204	No Content - The server has fulfilled the request but does not need to return an entity-body.
400	Bad Request - (e.g. sending an array when the API is expecting a hash.)
401	Unauthorized - No valid API key provided.
403	Forbidden - You tried to access a disabled feed, or your API key is not allowed to access that resource, etc.
404	Not Found - The requested item could not be found.
405	Method Not Allowed - The method specified is not allowed.
422	Unprocessable Entity - Can result from sending invalid fields.

500, 502, 503, 504	Server errors - Something went wrong in the C2M server.

HTTP Verbs used

Theconnect2.me™APIs enables the developer to perform various actions to retrieve data, create new data, update data and delete data.

Use the appropriate HTTP verbs for each action:

GET

The HTTP GET method is used to retrieve (or read) a representation of a resource. The GET method returns a representation in XML and an HTTP response code of 200 (OK). The GET requests are used only to read data and not change it.

POST

The POST verb is used for creation of new resources within a Channel

On successful creation, the systemreturns HTTP status 201, with the message that the resource has successfully being created.

DELETE

DELETE is used to delete a resource within C2M.

On successful deletion, system returns HTTP status 200 (OK) along with a response body.

DELETE operations are idempotent. If you DELETE a resource, it's removed. Repeatedly calling DELETE on that resource ends up the same: the resource is gone. If calling DELETE say, decrements a counter (within the resource), the DELETE call is no longer idempotent. As mentioned previously, usage statistics and measurements may be updated while still considering the service idempotent as long as no resource data is changed. Using POST for non-idempotent resource requests is recommended.

There is a caveat about DELETEidempotence, however. Calling DELETE on a resource a second time will often return a 404 (NOT FOUND) since it was already removed and therefore is no longer findable. This makes DELETE operations no longer idempotent, but is an appropriate compromise if resources are removed from the database instead of being simply marked as deleted.

Authentication

When a request is sent to the API, it needs to be authenticated by including a valid API key as a parameter in the query string request.

If the API key is missing or empty or does not exit , the server will respond with a 401 - Unauthorized - No valid key provided message. If the API Key is used to access a resource from a different account, the server will respond with the 403 - Forbidden - You tried to access a disabled feed, or your API key is not allowed to access that resource, etc., message

API Keys

Public API Key

Each time a new user is created in connect2.me™, a new API Public Key is created and associated with the new User that has just been created. This key cannot be edited or deleted: it can only be regenerated if needed (e.g. if there's any risk that someone else has gained access to that key). The newly generated Public API key replaces the old for new or existing API/device calls.

These Public API Keys can only be used to access the feeds, values and location of the User which they are associated with. They can't access data from any other User.

Restrictions of Public API Key

The Public API Key in an API key that has been restricted to a User, who will only have access to the resources of that user, all other cases will either return a error response or no data.

Private API Key

Each time a new user is created in connect2.me™, a new API Private Key is created and associated with the new User that has just been created. This key cannot be edited or deleted: it can only be regenerated if needed (e.g. if there's any risk that someone else has gained access to that key). The newly generated Private API key replaces the old for new or existing API/device calls.

These Private API Keys can only be used to encrypt the data using AES Encryption technique. At the time of sending the data user need to use this private key to encrypt the data before making call to ICE server.

Restrictions of Private API Key

The Private API Key in an API key that has been restricted to a User, who will only use to encrypt the data before sending to ICE server so that in between no one can access this data.

Resources

Data Source

A Data Source is any individual generator of data that can be connected to connect2.me™. It could be a devices from various manufacturers, APIs, Files or Databases.

Company

Company represents a device manufacturer or an API publisher

Channel

Channel is a blueprint of a actual device (Eg. Xirgo, Arduino,etc) or third party web service like facebook, LinkedIn etc.

Channel definition :Superuser defines a channel by inputting attributes of a device/API.

Channel Instantiation

After the channel is created by a user, the channel is instantiated when the user adds devices, actual values and identifiers to the devices.

Values

Values are individual measurable data points that are derived from device/API (i.e. a thermostat will give the date and time value as well as humidity and atmospheric pressure values).

Below example shows values received from a thermostat

C2MDateTime	Temperature	Humidity
12/31/2014 23:59	32 °F	39%

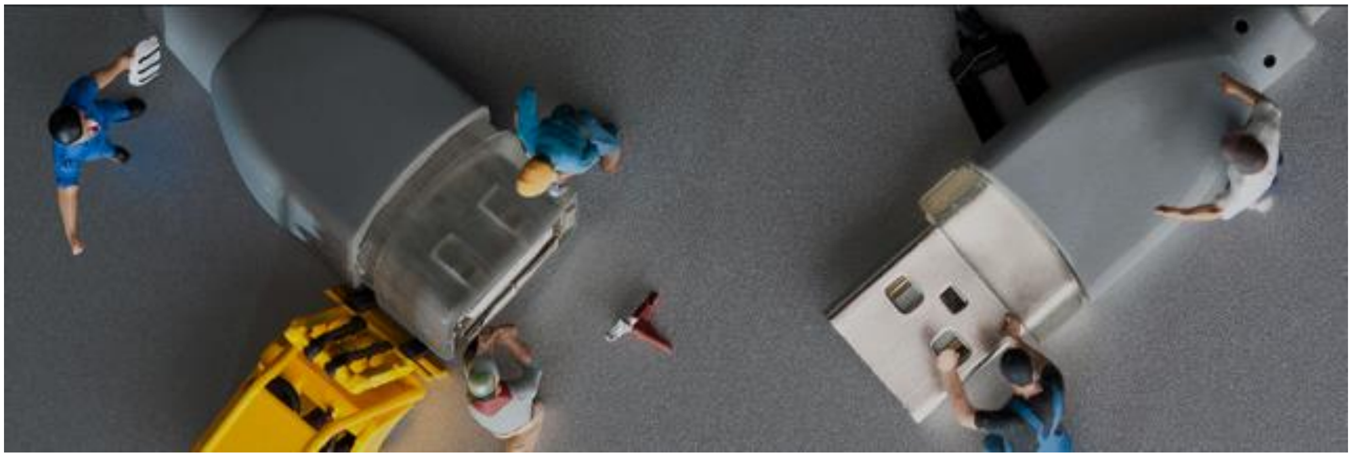
Feed

Feeds comprise of Values. A Feed is a sequence of timestamped values for a single source of measurement data. For example, a thermostat that measures temperature, humidity and atmospheric pressure, provides the a collective output of these values as Feed.

The table below shows a set of values that are received from a thermostat. These values collectively comprise as a Feed

C2MDateTime	Temperature	Humidity
12/31/2014 23:59	32 °F	39%

1/1/2015 22:34	58 °F	20%
4/2/2015 2:23	67 °F	60%



What is publicThings?

connect2.me™ has signed onboard several companies and device manufacturers , who use the portal to share APIs, device information and other resources with the developers at large. The information that each company makes public and that is available to the users in connect2.me™,falls under the publicThings segment.

What is myThings?

The user can instantiate a channel in publicThings and during that process, the user provides the actual parameter values of a device/API. Once the instance of the channel is created , it is saved under myThings for that user.

Authentication Overview

The authentication process in connect2.me™ involves the user entering their username and password. The API authenticates the validity of the user and gives them access as per their level of subscription to connect2.me™.

An Pubic Key is created and assigned to the new user upon successful login. This Pubic Key cannot be edited or delete but can only be regenerated if needed. (in case of login information being compromised). The newly generated Pubic Key replaces the old Pubic Key.

The Pubic Key gives the user access to all the APIs , devices and other resources available within their own portfolio. The user is denied access to all other portfolios within the system.

Check user credentials

GET /ice.svc/CheckUserCredentials

This method is used to authenticate the user login into connect2.me.

User passes the username and password as a header parameter.

In success calls the user receives the Pubic Key.

Method Details	
HTTP Method :	GET
Response Format :	xml
Authentication Required?	Yes

Parameters used in the API

Parameter	Required?	Type	Description
emailid	Yes	String	Username of the user
password	Yes	String	Password of the user

Example URL:http//ice.connect2.me/ice.svc/CheckUserCredentials

Response Schema

```
<response>
<code>200</code>
<status>SUCCESS</status>
<message>dvH5tcKnrQu5CcXz</message>
</response>
```

Companies API overview

Use ofCompanies API

The connect2.me™ portal has partnered with several leading companies to provide the latest and the most advanced connected / smart devices and APIs.

Thesuperuser alone has the permission to create, delete and edit companies and their properties.

The Companies API allows the developer to retrieve list of all companies that are partnered with connect2.me™.

Get all companies

GET /ice.svc/GetCompanies

Get a list of all companies within the connect2.me™ portal.

The API call returns Company ID and Company Name.

Method Details	
HTTP Method :	GET
Response Format :	xml
Authentication Required?	Yes

Parameters used in the API

Parameter	Required?	Type	Description
apikey	Yes	String	User's API Key

Example URL: http//ice.connect2.me/ice.svc/GetCompanies?apikey={apikey}

Response Schema

```
<response>
<code>200</code>
<status>SUCCESS</status>
<icedata><companies><company>
<ID>89</ID><COMPANYNAME>37
signals</COMPANYNAME><URL>https://campfirenow.com/</URL>
<NOOFCHANNELS>2</NOOFCHANNELS>
<CATEGORYNAME>Social APIs</CATEGORYNAME>
</company></companies></icedata>
</response>
```

Channels API overview

Use ofChannels API

A Channel is a blueprint of a actual device (Xirgo, Arduino etc) or third party web service like facebook, LinkedIn etc.

The authorized user logs into their profile in the connect2.me™ portal, using their credentials and creates a Channel. The user inputs different information about the channel for different types of audience.The channel acts as a hub for each company , where they are able to share all information about their company, device, API etc; in a collaborative fashion. The end user can avail all relevant resource (marketing info, APIs, device info etc.) for that company through one public channel.

The Channel API can be used to:

- Retrieve list of all published Channels
- Retrieve the schema and the identification info of specific Channels
- Onboard or instantiate a new channel
- Register channel to myThings from publicThings

Get published channels

GET /ice.svc/GetPublishChannels

A Channel is a blueprint of a actual device like Xirgo, Arduino etc or third party web service like facebook, LinkedIn etc;

This method is used to show all the channels of a specific company. The call returns Channel Id ,Channel name and Channel type, which in turn can be used in other API calls.

Method Details	
HTTP Method :	GET
Response Format :	xml
Authentication Required?	Yes

Parameters used in the API

Parameter	Required?	Type	Description
apikey	Yes	String	Pubic Key
companyid	Yes	Integer	Id of the company for which user wants to get the channels.

Example URL:

https://ice.connect2.me/ice.svc/GetPublishChannels?companyid={companyid}&apikey={apikey}

Response Schema

```
<response>
<code>200</code>
<status>SUCCESS</status>
<icedata><company><channel><ID>511</ID><NAME>Arduino Leonardo
Battery</NAME><CHANNELTYPE>Device</CHANNELTYPE><CHANNELSUBTYPE>connect2me</C
HANNELSUBTYPE>
</channel></company></icedata>
</response>
```

Get published channel schema

GET /ice.svc/GetPublishChannelSchema

This method is used to get schema and identification information of the channel.

Schema is the XSD of the parameters that the device sends to C2M where as Identification info is the XML that tells us how the device is uniquely identified.

Method Details	
HTTP Method :	GET
Response Format :	xml
Authentication Required?	Yes

Parameters used in the API

Parameter	Required?	Type	Description
apikey	Yes	String	Pubic Key
publishchannelid	Yes	Integer	ID of the channel for which the user want to get the channel schema and identification information

Example URL:
https://ice.connect2.me/ice.svc/GetPublishChannelSchema?publishchannelid={publishchannelid}&apikey={apikey}

Response Schema

```
<?xml version="1.0" encoding="utf-8"?>
<response xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"><code>200</code><status>SUCCESS</status><ic
edata><channel><name>Arduino Uno</name><schemaxsd><xs:element name="dm"
xmlns:xs="http://www.w3.org/2001/XMLSchema"><xs:complexType><xs:sequence><xs:element
name="temp1" type="xs:int"/><xs:element name="temp2" type="xs:int"/><xs:element
name="temp3" type="xs:int"/><xs:element name="temp4" type="xs:int"/><xs:element
name="light1" type="xs:int"/><xs:element name="light2" type="xs:int"/><xs:element name="light3"
type="xs:int"/><xs:element name="light4"
type="xs:int"/></xs:sequence></xs:complexType></xs:element></schemaxsd><identification><root>
<Column selected="true" type="string"
name="mac"/></root></identification></channel></icedata></response>
```

Onboard channel

GET /ice.svc/OnboardChannel

connect2.me™ comprise of either an API or device type of channels. A user instantiates an API type of channel to myThings from publicThings.

In such an event, this method is used to onboard or initiate an API type of channel from publicThings into the user's myThings

Method Details	
HTTP Method :	GET
Response Format :	xml
Authentication Required?	Yes

Parameters used in the API

Parameter	Required?	Type	Description
apikey	Yes	String	Pubic Key
name	Yes	String	Friendly name of the channel instance that user wants to give
channelidefid	Yes	Integer	ID of the channel in Public Things that user wants to on-board (instantiate)

Example URL:https://ice.connect2.me/ice.svc/OnboardChannel?apikey={apikey}&name={name}&channeldefid={channelID}

Notes: If this call is successful, user gets an URL in message node of the response . The user will need to take that URL from the response and run that in the browser or in the app.The user will be taken to the authorization page of the API provider.After inputting credentials and giving access rights, a successful message shows up and on-boarding is a success.

Response Schema

```
<response>
<code>200</code>
<status>SUCCESS</status>
<message>https://www.connect2.me/Apiprovider.aspx?accessToken=https://graph.faceboo
k.com/oauth/access_token?grant_type=authorization_code@accessTokenparam=access_to
ken@requestToken=@authorizeToken=https://graph.facebook.com/oauth/authorize?auth_
type=reauthenticate@consumerKey=649822138478766@consumerKeyparam=client_id@co
nsumerSecret=84c5707598407ec61880e6ead5600ee4@consumerSecretparam=client_secre
t@ApiCallUrl=@BaseUrl=https://graph.facebook.com/v2.2@Version=OAuth
2.0@methodType=GET@fromPg=Ice@Scope=user_about_me,user_actions.books,user_acti
ons.music,user_actions.news,user_actions.video,user_activities,user_birthday,
user_education_history,user_events,user_friends,user_games_activity,user_groups,user_ho
metown,user_interests,user_likes,
user_location,user_photos,user_relationship_details,user_relationships,user_religion_politic
s,user_status,
user_tagged_places,user_videos,user_website,user_work_history,email,manage_notificatio
ns,manage_pages,publish_actions,
read_friendlists,read_insights,read_mailbox,read_page_mailboxes,read_stream,rsvp_event,
publish_likes@channeldefId=315@name=ApiTest2@type=API@apikey=dvH5tcKnrQu5CcXz<
/message>
</response>
```

Register channel

POST /ice.svc/RegisterChannel

connect2.me™ comprise of either an API or device type of channels. A user can instantiate a device type of channel to myThings from publicThings.

In such an event, this method is used to onboard or initiate a device type of channel from publicThings into the user's myThings.

Method Details	
HTTP Method :	POST
Response Format :	xml
Authentication Required?	Yes

Parameters used in the API

Parameter	Required?	Type	Description
apikey	Yes	String	Pubic Key
name	Yes	String	Friendly name of the channel instance that the user wants to give
channelid	Yes	String	ID of the channel in public Things that the user wants to on-board (instantiate)
Identifier	Yes	String	Pass identifier with paramname1, paramvalue1|paramvalue2,paramvalue2|paramvalue3,param value3
conninfo	No	String	Connection information of the channel
schema	No	String	Schema information of the channel
legallyagree	Yes	String	User has to agree legally by sending 1 in this parameter

Example URL:
https://ice.connect2.me/ice.svc/RegisterChannel?apikey={apikey}&name={name}&channelid={channelid}&identifier={identifier} &conninfo={conninfo}&schema={schema}&legallyagree={legallyagree}

Response Schema

```
<response>
<code>200</code>
<status>SUCCESS</status>
<message>Channel successfully on-boarded.</message>
</response>
```

Channel instance API overview

Use ofChannel Instance APIs

The user logs successfully into the connect2.me™ portal and is able to access various channels and avail the blueprint of the actual devices and APIs and all their resources. The user can further instantiate a channel by adding devices, actual values and identifiers to the devices into the myThings within connect2.me™.

The Channel Instance APIs allows the developer to :

- Get a list of all subscribed channels by a specific user
- Set a channel status
- Get a particular subscribed channel
- Get a subscribed channel status
- Delete a subscribed channel instance
- Onboard or Instantiate a channel

Get all subscribed channels

GET /ice.svc/GetAllSubscribeChannel

This method is used to show all on-boarded (myThings) channels to the user.

Method Details	
HTTP Method :	GET
Response Format :	xml
Authentication Required?	Yes

Parameters used in the API

Parameter	Required?	Type	Description
apikey	Yes	String	Public Key
pagenumber	Yes	String	1 for first 100 records 2 for next 100 records and so on.

Example URL:
https://ice.connect2.me/ice.svc/GetAllSubscribeChannel?apiKey={apiKey}&pagenumber={pagenumber}

Response Schema

```
<response>
<code>200</code>
<status>SUCCESS</status>
<icedata><subscribechannel><channels><channel>
<ID>16204</ID><NAME>ParulLinkedin</NAME><ICON>https://54.148.178.54/UserResource/Compan
y-108/CompLogo/sthumb/b76152d2-822a-411b-b1fb-2cfc81ba108e.jpg</ICON>
<COMPANYID>108</COMPANYID><COMPANYNAME>LinkedIn</COMPANYNAME><CHANNELNAME>L
inkedIn
API</CHANNELNAME><ACTIVESTATUS>0</ACTIVESTATUS></channel></channels></subscribechannel
></icedata>
</response>
```

Get subscribed channel commands

GET /ice.svc/GetSubscribeChannelCommands

This method is used to get the commands of the channel instance.

Method Details	
HTTP Method :	GET
Response Format :	xml
Authentication Required?	Yes

Parameters used in the API

Parameter	Required?	Type	Description
apikey	Yes	String	Public Key
subscribechannelid	Yes	Integer	Channel instance ID

Example URL:
https://ice.connect2.me/ice.svc/GetSubscribeChannelcommands?apiKey={apikey}&subscribechannelid={subscribechannelid}

Response Schema

```
<?xml version="1.0" encoding="utf-8"?>
<response xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema">
<code>200</code>
<status>SUCCESS</status>
<message>commands</message>
</response>
```

Get subscribed channel status

GET /ice.svc/GetSubscribeChannelStatus

Method Details	
HTTP Method :	GET
Response Format :	xml
Authentication Required?	Yes

This method is used to get the active status of the channel instance.

Parameters used in the API

Parameter	Required?	Type	Description
apikey	Yes	String	Public Key
subscribechannelid	Yes	Integer	Channel instance ID

Example URL:
https://ice.connect2.me/ice.svc/GetSubscribeChannelStatus?apiKey={apikey}&subscribechannelid={subscribechannelid}

Response Schema

```
<?xml version="1.0" encoding="utf-8"?>
<response xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema">
<code>200</code>
<status>SUCCESS</status>
<message>Inactive</message>
</response>
```

Set Subscribe channel configuration

POST /ice.svc/SetSubscribeChannelConfiguration

This method is used to set the subscribe channel refresh policy.

Method Details	
HTTP Method :	POST
Response Format :	xml
Authentication Required?	Yes

Parameters used in the API

Parameter	Required?	Type	Description
apikey	Yes	String	Public Key
subscribechannelid	Yes	Integer	Channel instance Id
refreshrate	Yes	Integer	1 means None 2 means Second 3 means Minute 4 means Hour 5 means Weekly 6 means Hourly 7 means Monthly 8 means Yearly
refreshperiod	Yes	Integer	Value of refresh period means if refresh rate 2 and refresh period 5 it means data will receive on server after every 5 second.

Example URL:
https://ice.connect2.me/ice.svc/SetSubscribeChannelConfiguration?apiKey={apiKey}&subscribechannelid={subscribechannelid}&refreshrate={refreshrate}&refreshperiod={refreshperiod}

Response Schema

```
<response>
<code>200</code>
<status>SUCCESS</status>
<message>Configuration updated successfully.</message>
</response>
```

Set channel status

POST /ice.svc/SetChannelStatus

This method is used to set the active status ON or OFF of the channel instance.

When the channel instance is OFF, connect2.me will not accept data for the feeds under that channel instance.

Method Details	
HTTP Method :	POST
Response Format :	xml
Authentication Required?	Yes

Parameters used in the API

Parameter	Required?	Type	Description
apikey	Yes	String	Public Key
subscribechannelid	Yes	Integer	Channel instance Id
status	Yes	Integer	1 means ON and 0 means OFF

Example URL:
https://ice.connect2.me/ice.svc/SetChannelStatus?apikey={apikey}&subscribechannelid={subscribechannelid}&status={status}

Response Schema

```
<response>
<code>200</code>
<status>SUCCESS</status>
<message>Channel status changed successfully.</message>
</response>
```

Get subscribed channel

GET /ice.svc/GetSubscribeChannel

This method is used to show all on-boarded channels under a specific channel definition

Method Details	
HTTP Method :	GET
Response Format :	xml
Authentication Required?	Yes

Parameters used in the API

Parameter	Required?	Type	Description
apikey	Yes	String	Public Key
publishedchannelid	Yes	Integer	This is the value which user gets from the get company call
pagenumber	Yes	Integer	1 for first 100 records and 2 for next 100 records and so on

Example URL:
https://ice.connect2.me/ice.svc/GetSubscribeChannel?apiKey={apiKey}&publishchannelid={publishchannelid}&
pagenumber={pagenumber}

Response Schema

```
<response xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
<code>200</code>
<status>SUCCESS</status>
<icedata><company><channel>
<ID>16909</ID><NAME>SaleemMega</NAME>
</channel></company></icedata>
</response>
```

Delete subscribed channel

DELETE /ice.svc/DeleteSubscribeChannel

Method Details	
HTTP Method :	DELETE
Response Format :	xml
Authentication Required?	Yes

This method is used to delete a subscribed channel or the channel instance under myThings.

Parameters used in the API

Parameter	Required?	Type	Description
apikey	Yes	String	Public Key
subscribechannelid	Yes	Integer	Channel instance ID

Example URL:
https://ice.connect2.me/ice.svc/DeleteSubscribeChannel?apiKey={apikey}&subscribechannelid={subscribechannelid}

Response Schema

```
<response>
<code>200</code>
<status>SUCCESS</status>
<message>Channel successfully deleted.</message>
</response>
```

Feed API overview

Use ofFeed API

Upon instantiation of a device or an API channel. The user can input values into the system. These values collectively form a Feed.

A Feed is a sequence of timestamped values for a single source of measurement data.

The Feed API allows the developer to :

- Get channel feed
- Get channel feed status
- Delete feed in a channel
- Push Data to a channel
- Get data from a channel

Get subscribe channel feeds

GET /ice.svc/GetSubscribeChannelFeeds

Once the user instantiates a channel under myThings, they are able to add values into the parameters within each segment of the channel. The values collective form a feed of that channel.

This method is used to return all the feeds under an on-boarded or instantiated channel in myThings.

Method Details	
HTTP Method :	GET
Response Format :	xml
Authentication Required?	Yes

Parameters used in the API

Parameter	Required?	Type	Description
apikey	Yes	String	Public Key
subscribechannelid	Yes	Integer	Channel instance Id

Example URL:
https://ice.connect2.me/ice.svc/GetSubscribeChannelFeeds?subscribechannelid={subscribechannelid}&apikey={apikey}

Response Schema

```
<response>
<code>200</code>
<status>SUCCESS</status>
<icedata><feeds><feed>
<CHANNELDATAID>22813</CHANNELDATAID><FRIENDLYNAME>PushFeed</FRIENDLYNAME><ACTI
VESTATUS>1</ACTIVESTATUS></feed><feed>
<CHANNELDATAID>22814</CHANNELDATAID><FRIENDLYNAME>AvgTempAQB</FRIENDLYNAME><
QUERYTYPE>AQB</QUERYTYPE><ACTIVESTATUS>0</ACTIVESTATUS></feed><ENCRYPTEDFEEDID>X
XXXXXX</ENCRYPTEDFEEDID>
</feeds></icedata></response>
```

Get channel feed status

GET /ice.svc/GetChannelFeedStatus

The user is able to add values into the parameters within each segment of the channel under their myThings. These values collectively form a feed of that channel.

This method is used to get the active status of the channel feed.It returns "Active" or "Inactive" as the feed status. If the feed status is Inactive ,connect2.me™ does not accept data for that feed.

Method Details	
HTTP Method :	GET
Response Format :	xml
Authentication Required?	Yes

Parameters used in the API

Parameter	Required?	Type	Description
apikey	Yes	String	Public Key
feedid	Yes	String	Encrypted feed id

Example URL:https://ice.connect2.me/ice.svc/GetChannelFeedStatus?apiKey={apikey}&feedid={feedid}

Response Schema

```
<?xml version="1.0" encoding="utf-8"?>
<response xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema">
<code>200</code>
<status>SUCCESS</status>
<message>Active</message>
</response>
```

Set feed configuration

POST /ice.svc/SetFeedConfiguration

This method is used to set the feed refresh policy and feed storage type (Append or overwrite).

Method Details	
HTTP Method :	POST
Response Format :	xml
Authentication Required?	Yes

Parameters used in the API

Parameter	Required?	Type	Description
apikey	Yes	String	Public Key
feedid	Yes	Integer	Channel instance Id
refreshrate	Yes	Integer	1 means None 2 means Second 3 means Minute 4 means Hour 5 means Weekly 6 means Hourly 7 means Monthly 8 means Yearly
refreshperiod	Yes	Integer	Value of refresh period means if refresh rate 2 and refresh period 5 it means data will receive on server after every 5 second.
isappend	Yes	Integer	O means overwrite and 1 means append data.

Example URL:
https://ice.connect2.me/ice.svc/SetfeedConfiguration?apiKey={apiKey}&feedid={feedid}&refreshrate={refreshrate}&refreshperiod={refreshperiod}&isappend={isappend}

Response Schema

```
<response>
<code>200</code>
<status>SUCCESS</status>
<message>Feed Configuration updated successfully.</message>
</response>
```

Set channel feed status

POST /ice.svc/SetChannelFeedStatus

This method is used to set the active status ON or OFF for the channel feed.

When the Channel Feed is OFF, connect2.me™ will not accept data for the feeds under that Feed.

Method Details	
HTTP Method :	POST
Response Format :	xml
Authentication Required?	Yes

Parameters used in the API

Parameter	Required?	Type	Description
apikey	Yes	String	Public Key
feedid	Yes	Integer	ID of the feed for which the user wants to set the status
status	Yes	Integer	1 means ON and 0 means OFF

Example URL:
https://ice.connect2.me/ice.svc/SetChannelFeedStatus?apikey={apikey}&feedid={feedid}&status={status}

Response Schema

```
<response>
<code>200</code>
<status>SUCCESS</status>
<message>Feed status successfully changed.</message>
</response>
```

Delete channel feed

DELETE /ice.svc/DeleteChannelFeed

The user instantiates a channel and onboards devices and/or APIs from that channel. The user is then able to add values into the parameters for a device. The time series data collected from a device is called a Feed.

This method is used to delete the Channel Feed.

Method Details	
HTTP Method :	DELETE
Response Format :	xml
Authentication Required?	Yes

Parameters used in the API

Parameter	Required?	Type	Description
apikey	Yes	String	Public Key
feedid	Yes	String	Encrypted feed id

Example URL:https://ice.connect2.me/ice.svc/DeleteChannelFeed?apiKey={apikey}&feedid={feedid}

Response Schema

```
<?xml version="1.0" encoding="utf-8"?>
<response xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema">
<code>200</code>
<status>SUCCESS</status>
<message>Feed successfully deleted</message>
</response>
```

Push feed

GET /ice.svc/PushFeed

This method is used to send the data from different devices that support HTTPS calls to connect2.me™.

User passes the feed as a header parameter and before sending the data it must be encrypted by Private Key using AES encryption.

Method Details	
HTTP Method :	GET
Response Format :	xml
Authentication Required?	Yes

Parameters used in the API

Parameter	Required?	Type	Description
apikey	Yes	String	Pubic Key
feedid	Yes	String	Encrypted feed id
feed	Yes	String	In this user needs to pass data as parameter value pair. E.g. param1,value1|param2,value2|param3,value3

Example URL:https://ice.connect2.me/ice.svc/pushfeed?apiKey={apiKey}&feedID={feedID}

Response Schema

```
<response>
<code>200</code>
<status>SUCCESS</status>
<message>Request has been accepted successfully. Processing will be completed
asynchronously.</message>
</response>
```

Postfeeddata feed

Interface URLs and Methods

Below is PostFeed URL with the method.

Method Details	
HTTP Method :	POST
Request Format :	XML
Response Format :	XML
Authentication Required?	Yes

POST /ice.svc/Postfeeddata

This method is used to push the data to C2M.

Parameters used in the API

Parameter	Required?	Type	Description
apikey	Yes	String	Encrypted User API Key
feedid	Yes	String	Encrypted feed id
feed	Yes	String	In this base64 encoded JSON/XML string of the data. Example {“first_name”:”John”, “last_name”: “Miller”, “address”:”123 Main St”……..} Pass data as a header parameter

URL: https://ice.connect2.me/ce.svc/PostFeedData?apikey={apikey}&feedid{feedid}

Note: Pass Feed parameter as a header string.

Response Schema

```
<response> <code>200</code> <status>SUCCESS</status> <message>Request has been accepted successfully. Processing will be completed asynchronously. </message></response>
```

Get data

GET /ice.svc/GetData

This method is used to retrieve data in a feed.

Method Details	
HTTP Method :	GET
Response Format :	xml
Authentication Required?	Yes

Parameters used in the API

Parameter	Required?	Type	Description
apikey	Yes	String	Pubic Key
feedid	Yes	String	Encrypted feed id
parameter type	Yes	String	LIST : If we want to return raw data (rows), data can be filtered by other parameters that will be passed into the function. AGGR_OPR: If we want to perform any type of aggregation operation on the values that will be in the {parameters} parameter. parameters=AVG: param1,param2 - - which means we want to obtain AVG of both the columns. We can pass all the Solr supported analytical functionsin place of AVG here. parameters-PERCENTAGE: param1,param2 -- If we want to calculate the percentage of param1 based on the values of param1 and param2. E.g:PERCENTAGE: RAM_Used,RAM_Free, here AVG(RAM_Used)/(AVG (RAM_Used)+AVG(RAM_Free)) parameters-PERCENTAGE:IsWifi:Yes - suppose devices are sending data and if device has wifi feature in itself it will send

			Yes in IsWifiparam otherwise No, now if we want to calculate the percentage of devices having Wifi feature then we send this format in parameters , now calculation is COUNT(IsWifi=Yes)/(COUNT(IsWifi=Yes)+COUNT(IsWifi=No)
parameters	No	String	Comma separated parameter list that we want to fetch from the feed. If user does not pass this parameter, all the parameters of the feed are retrieved.
filtertype	No	String	Date:If filtering is based on date, then filtervalue is a date in YYYY-MM-DDORYYYY-MM-DD HH:MI:SS format DateRange: If filtering is based on a date range, then filtervalue is a date range in YYYY-MM-DDTOYYYY-MM-DD format TimePeriodName: If filtering is based on specific keywords in filtervalue filtervalue=Daily then daily data, Weekly then weekly data, Monthly then monthly data, TillDate then whole data TotalNoOfRecords:To fetch the specified number of rows filterValue: value for rows to be fetched.
startrecord	No	Integer	An integer value to skip the records from the top
whereCondition	No	String	Additional conditions to put further filtration e.g.param1=India && param2=UP param1>100 && param1<1000

Example
URL:https://ice.connect2.me/ice.svc/GetData?apikey={apikey}&feedID={feedID}&parameterType={parameterType}&filterType={TotalNoOfRecords}
&filterValue={filterValue}

Response Schema

<root><response status="SUCCESS" startrecord="0" fetchedrecords="10" totalrecords="36011">
<NewDataSet><Table1><c2mdatetime>2015-01-20T22:03:00.436+00:00</c2mdatetime><v5>8</v5><v4>0</v4><v3>1240</v3><v2>1240</v2><c2mseqno>1</c2mseqno><c2mfeedid>897</c2mfeedid><c2muserid>526</c2muserid><battery>2529</battery><potentiometer>25</potentiometer></Table1><Table1><c2mdatetime>2015-01-20T22:01:25.681+00:00</c2mdatetime><v5>0</v5><v4>0</v4><v3>1054</v3><v2>1015</v2><c2mseqno>1</c2mseqno><c2mfeedid>897</c2mfeedid><c2muserid>526</c2muserid><battery>2109</battery><potentiometer>21</potentiometer></Table1></ NewDataSet>
</ response></root>

Execute feed

GET /ice.svc/ExecuteFeed

This method is used to execute the feeds that are already created on connect2.me™ with different parameters values and get the response.

Method Details	
HTTP Method :	GET
Response Format :	xml
Authentication Required?	Yes

Parameters used in the API

Parameter	Required?	Type	Description
apikey	Yes	String	Pubic Key
feedid	Yes	String	Encrypted feed id
parameters	Yes	String	In this user needs to pass data as parameter value pair. E.g. param1,value1|param2,value2|param3,value3

Example URL:

https://ice.connect2.me/ice.svc/ExecuteFeed?apiKey={apiKey}&feedID={feedID}&parameters={parameters}

Response Schema

```
<response>
<code>200</code>
<status>SUCCESS</status>
<message>Response return from dynamic api</message>
</response>
```