

Network Server

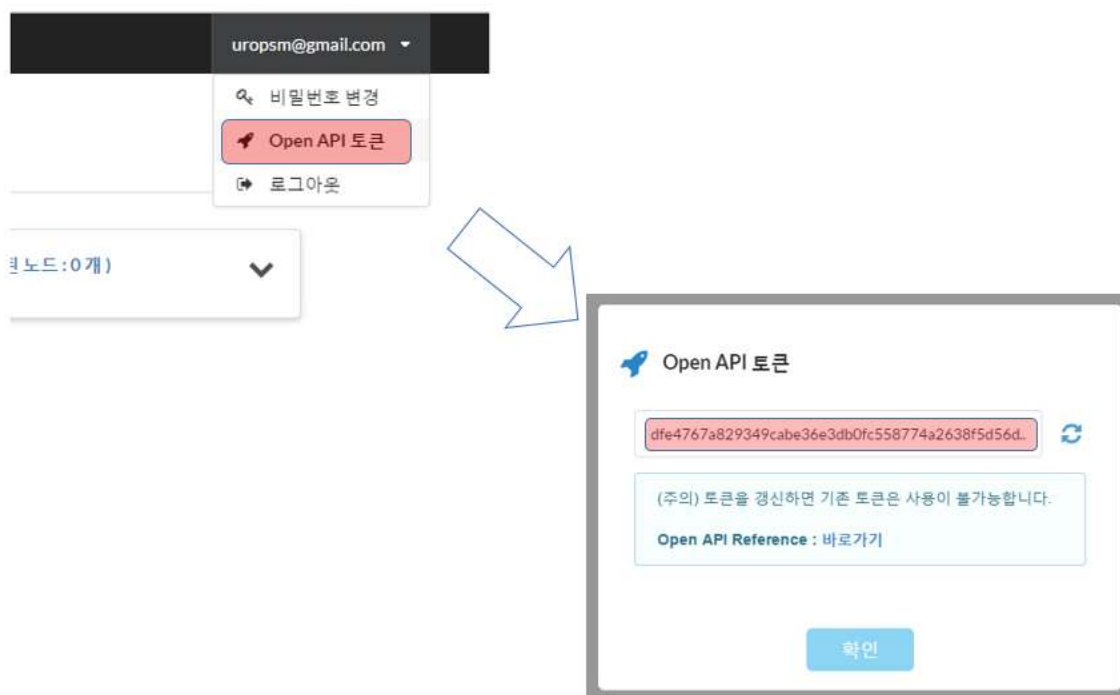
Open API

Open API Reference

1. Introduction

IoT.own 은 타 시스템에서 연동하기 위하여 RESTful API 를 제공합니다.

API 를 사용하려면 토큰이 필요하며 IoT.own 에 로그인 후 Open API 토큰 메뉴에서 얻을 수 있습니다.



2. API List

- [\[GET\] gateways](#)
- [\[GET\] gateway/{gateway_id}](#)
- [\[POST\] gateway](#) (2018/10/17 이후 버전에서만 지원)
- [\[DELETE\] gateway](#) (2018/10/17 이후 버전에서만 지원)
- [\[GET\] gateway/{gateway_id}/connected-nodes](#)
- [\[GET\] gateway/{gateway_id}/stats](#) (2019/01/08 이후 버전에서만 지원)
- [\[GET\] nodes](#)
- [\[GET\] node/{node_id}](#)
- [\[POST\] node](#) (2018/10/17 이후 버전에서만 지원)
- [\[DELETE\] node](#) (2018/10/17 이후 버전에서만 지원)
- [\[POST\] storage/search](#) (deprecated)
- [\[GET\] storage](#) (2018/11/18 이후 버전에서만 지원)
- [\[PUT\] callback](#)
- [\[GET\] callback](#)
- [\[DELETE\] callback](#)
- [\[POST\] command](#)
- [\[DELETE\] command](#)
- [\[POST\] data](#)

3. Error Descriptions

HTTP request 에 대한 response code 가 200 인 경우 성공을 의미합니다. 반면, 200 이 아닌 경우 오류로 인해 정상 처리되지 않았음을 의미합니다. Status code 와 더불어 모든 response 는 JSON 포맷의 payload 가 반환되며, 여기에는 **errorCode** 와 **errorMsg** 을 반환합니다. Status code 에 대한 설명은 아래 정리되어 있습니다. 자세한 설명은 **errorMsg** 를 참고하시기 바랍니다.

```
{  
  
  "errorCode": 200,  
  
  "errorMsg": "OK"
```

```
}
```

errorCode 는 HTTP response code 와 동일한 값입니다.

Response code	Description
200	성공
400	Bad Request
401	Invalid Token
500	서버에 문제가 발생했거나 처리가 불가능한 요청인 경우

(주의사항) 시간 표기법

API 에서 리턴되는 모든 시간 정보는 표준 UTC 시간이므로 원하는 시간대에 맞게 변형해서 사용해야 합니다.

4. API Descriptions

[GET] gateways

URL: <https://town.coxlab.kr/api/v1.0/gateways>

이 API 를 통해서 IoT.own 과 연동된 게이트웨이 목록을 가져올 수 있습니다.

Header

- **Token**: API 토큰

Response

- **gateways[]**: 게이트웨이 객체 배열
- **gateway_id**: 게이트웨이 아이디
- **gateway_eui**: 게이트웨이 EUI
- **gateway_type**: IoT.own 등록시 사용자가 임의 입력하는 정보
- **gateway_desc**: IoT.own 등록시 사용자가 임의 입력하는 정보
- **created_at**: IoT.own 에 등록된 시간(UTC)
- **recent_act** (optional): 게이트웨이가 마지막으로 동작한 시간(UTC)
- **bootup_time** (optional): 마지막으로 부팅된 시간(UTC). 부팅 메시지가 누락된 경우, 필드가 없을 수 있음.
- **latitude** (optional): 위도
- **longitude** (optional): 경도
- **altitude** (optional): 고도
- **errorCode**: 에러 코드
- **errorMsg**: 에러 내용

Example

cURL

```
1  curl -X GETW
2  --header 'Accept: application/json'W
3  --header 'Token: your_own_token'W
4  'https://town.coxlab.kr/api/v1.0/gateways'
```

Return

```
{
  "gateways": [
1    {
2      "created_at": "2016-05-30T05:54:59.995Z",
3      "gateway_id": "GW002",
4      ...
5    },
6    {
7      "created_at": "2016-05-30T05:27:14.808Z",
8      "gateway_id": "GW001",
9      ...
10   }
11  ],
12  "errorCode": 200,
13  "errorMsg": "Success"
14 }
15
16
17
```

[GET] gateway/{gateway_id}

URL: https://town.coxlab.kr/api/v1.0/gateway/{gateway_id}

이 API 를 통해서 IoT.own 과 연동된 특정 게이트웨이의 정보를 가져올 수 있습니다.

Header

- **Token**: API 토큰

Parameters

- **gateway_id**: 게이트웨이 아이디

Response

- **gateway**: 게이트웨이 객체
- **gateway_id**: 게이트웨이 아이디
- **gateway_eui**: 게이트웨이 EUI
- **gateway_type**: IoT.own 등록시 사용자가 임의 입력하는 정보
- **gateway_desc**: IoT.own 등록시 사용자가 임의 입력하는 정보
- **created_at**: IoT.own 에 등록된 시간(UTC)
- **recent_act** (optional): 게이트웨이가 마지막으로 동작한 시간(UTC)
- **bootup_time** (optional): 마지막으로 부팅된 시간(UTC). 부팅 메시지가 누락된 경우, 필드가 없을 수 있음.
- **latitude** (optional): 위도. 만약 Gateway Management 페이지에서 별도로 설정한 좌표 정보가 있거나 게이트웨이로부터 실시간 위치 정보가 제공되는 경우에만 이 필드가 존재합니다.
- **longitude** (optional): 경도. 만약 Gateway Management 페이지에서 별도로 설정한 좌표 정보가 있거나 게이트웨이로부터 실시간 위치 정보가 제공되는 경우에만 이 필드가 존재합니다.
- **altitude** (optional): 고도. 만약 Gateway Management 페이지에서 별도로 설정한 좌표 정보가 있거나 게이트웨이로부터 실시간 위치 정보가 제공되는 경우에만 이 필드가 존재합니다.
- **errorCode**: 에러 코드
- **errorMsg**: 에러 내용

Example

cURL

```
1 curl -X GET --headerW
2   'Accept: application/json' --headerW
3   'Token: your_own_token'W
4   'https://town.coxlab.kr/api/v1.0/gateway/GW001'
```

Return

```
{
  1   "gateway":{
  2       "bootup_time":"2016-06-01T07:03:21.445Z",
  3       "created_at":"2016-05-30T05:27:14.808Z",
  4       "gateway_desc":"GW001 설명",
  5       "gateway_id":"GW001"
  6       "gateway_type":"GW001 종류",
  7       "gateway_eui":"AA555A00000000101",
  8       "recent_act":"2016-06-04T02:11:16.902Z",
  9       "latitude":"36.49197",
 10       "longitude":"127.25633",
 11       "altitude":"10.5"
 12   },
 13   "errorCode":200,
 14   "errorMsg":"Success"
 15 }
 16 }
```

[POST] gateway

URL: <https://town.coxlab.kr/api/v1.0/gateway>

이 API 를 통해서 IoT.own 에 게이트웨이를 등록할 수 있습니다.

Header

- **Token**: API 토큰

Parameters

- **gid**: 게이트웨이 아이디
- **eui**: 게이트웨이 EUI
- **type**: 게이트웨이 타입
- **desc**(optional): 게이트웨이 설명
- **lora**: LoRaWAN 용도 여부 (Boolean)

Response

- **errorCode**: 에러 코드
- **errorMsg**: 에러 내용

Example

cURL

```
curl -X POSTW
1  --header 'Content-Type: application/json'W
2  --header 'Accept: application/json'W
3  --header 'Token: your_own_token'W
4  -d '{W
5      "gid":"G001",W
6      "eui":"1111222233334444",W
7      "type":"gateway type",W
8      "desc":"gateway desc",W
9      "lora":trueW
10 }'W
11
12  'https://town.coxlab.kr/api/v1.0/gateway'
```

Return

```
1  {
2    "errorCode":200,
3    "errorMsg":"Success"
4  }
```

[DELETE] gateway

URL: <https://town.coxlab.kr/api/v1.0/gateway>

이 API 를 통해서 IoT.own 에 등록된 게이트웨이를 삭제할 수 있습니다.

Header

- **Token**: API 토큰

Parameters

- **gid**: 게이트웨이 아이디

Response

- **errorCode**: 에러 코드
- **errorMsg**: 에러 내용

Example

cURL

```
curl -X DELETEW
1  --header 'Content-Type: application/json'W
2  --header 'Accept: application/json'W
3  --header 'Token: your_own_token'W
4  -d '{W
5      "gid": "G001"W
6      }'W
7  'https://town.coxlab.kr/api/v1.0/gateway'
```

Return

```
1  {
2      "errorCode": 200,
3      "errorMsg": "Success"
4  }
```

[GET] gateway/{gateway_id}/connected-nodes

URL: https://town.coxlab.kr/api/v1.0/gateway/{gateway_id}/connected-nodes

이 API 를 통해서 IoT.own 과 연동된 특정 게이트웨이의 하위 노드 목록을 가져올 수 있습니다.

Header

- **Token**: API 토큰

Parameters

- **gateway_id**: 게이트웨이 아이디

Response

- **nodes[]**: 게이트웨이에 연결된 노드 리스트 객체
- **node_id**: 노드 아이디
- **errorCode**: 에러 코드
- **errorMsg**: 에러 내용

Example

cURL

```
1  curl -X GETW
2      --header 'Accept: application/json'W
3      --header 'Token: your_own_token'W
4      'https://town.coxlab.kr/api/v1.0/gateway/G001/connected-nodes'
```

Return

```
1  {
2  "nodes" : [
3      {"node_id" : "N001"},
4      {"node_id" : "N002"},
5  ]
6  }
```

```
7      ...
8    ],
9    "errorCode":200,
    "errorMsg":"Success"
  }
```

[GET] gateway/{gateway_id}/stats

URL: https://town.coxlab.kr/api/v1.0/gateway/{gateway_id}/stats

이 API 를 통해서 IoT.own 과 연동된 특정 게이트웨이의 통계 정보를 가져올 수 있습니다.

Header

- **Token**: API 토큰

Parameters

- **gateway_id**: 게이트웨이 아이디
- **interval**: 정보의 표시 시간 단위 (hour / day)
- **from**: 검색 시간 범위(시작시간)
- **to**: 검색 시간 범위(끝시간)

Response

- **stats[]**: 통계 정보
- **errorCode**: 에러 코드
- **errorMsg**: 에러 내용

Example

cURL

```
1  curl -X GETW
2  --header 'Accept: application/json'W
3  --header 'Token: your_own_token'W
4  'https://town.coxlab.kr/api/v1.0/gateway/G001/stats'
```

Return

```
    {
1    "stats" : [
2      {
3        "timestamp": "2019-01-01T12:00:00Z",
4        "rxPacketsReceived": 103,
5        "rxPacketsReceivedOK": 23,
6        "txPacketsReceived": 0,
7        "txPacketsEmitted": 0
8      }
9    ],
10   "...",
11   ],
12   "errorCode":200,
13   "errorMsg":"Success"
14   }
```

[GET] nodes

URL: `https://town.coxlab.kr/api/v1.0/nodes`

이 API 를 통해서 IoT.own 에 등록된 노드 목록을 가져올 수 있습니다.

Header

- **Token**: API 토큰

Response

- **nodes[]**: 노드 객체 배열
- **node_id**: 노드 아이디
- **node_type**: IoT.own 등록시 사용자가 임의 입력하는 정보
- **node_desc**: IoT.own 등록시 사용자가 임의 입력하는 정보
- **created_at**: IoT.own 에 등록된 시간(UTC)
- **connected_gw_id** (optional): 노드가 연결된 게이트웨이 아이디. 부팅 메시지가 누락된 경우 제외
- **bootup_time** (optional): 마지막으로 부팅된 시간(UTC). 부팅 메시지가 누락된 경우 제외
- **last_data** (optional): 마지막에 기록된 데이터 객체. 데이터가 한번도 기록되지 않은 경우 제외
- **last_timestamp** (optional): 마지막 데이터가 기록된 시간(UTC). 데이터가 한번도 기록되지 않은 경우 제외
- **errorCode**: 에러 코드
- **errorMsg**: 에러 내용

Example

cURL

```
1 curl -X GETW
2   --header 'Accept: application/json'W
3   --header 'Token: your_own_token'W
4   'https://town.coxlab.kr/api/v1.0/nodes'
```

Return

```
{
  "nodes" : [
1    {
2      "node_id": "N001",
3      "node_type": "N001 종류",
4      "node_desc": "N001 설명",
5      "created_at": "2016-05-27T08:13:31.633Z"
6    },
7    {
8      "node_id": "N002",
9      "node_type": "N002 종류",
10     "node_desc": "N002 설명",
11     "bootup_time": "2016-05-26T08:07:05.893Z",
12     "connected_gw_id": "GW001",
13     "created_at": "2016-05-27T08:13:31.633Z",
14     "last_data": {"CC": "33", "BB": "11", "AA": "00"},
15     "last_timestamp": "2016-05-30T07:50:10.848Z"
16   },
17   ...
18 ]
19 },
20 "errorCode": 200,
21 "errorMsg": "Success"
22 }
23
24
```

[GET] node/{node_id}

URL: https://town.coxlab.kr/api/v1.0/node/{node_id}

이 API 를 통해서 IoT.own 에 등록된 노드 정보를 가져올 수 있습니다.

Header

- **Token**: API 토큰

Parameters

- **node_id**: 노드 아이디

Response

- **node**: 노드 객체
- **node_id**: 노드 아이디
- **node_type**: IoT.own 등록시 사용자가 임의 입력하는 정보
- **node_desc**: IoT.own 등록시 사용자가 임의 입력하는 정보
- **created_at**: IoT.own 에 등록된 시간(UTC)
- **connected_gw_id** (optional): 노드가 연결된 게이트웨이 아이디. 부팅 메시지가 누락된 경우 제외
- **bootup_time** (optional): 마지막으로 부팅된 시간(UTC). 부팅 메시지가 누락된 경우 제외
- **last_data** (optional): 마지막에 기록된 데이터 객체. 데이터가 한번도 기록되지 않은 경우 제외
- **last_timestamp** (optional): 마지막 데이터가 기록된 시간(UTC). 데이터가 한번도 기록되지 않은 경우 제외
- **errorCode**: 에러 코드
- **errorMsg**: 에러 내용

Example

cURL

```
1  curl -X GETW
2      --header 'Accept: application/json'W
3      --header 'Token: your_own_token'W
4      'https://town.coxlab.kr/api/v1.0/node/N001'
```

Return

```
{
  "node": {
1    "node_id": "N001",
2    "node_type": "N001 종류",
3    "node_desc": "N001 설명",
4    "created_at": "2016-05-27T08:13:31.633Z",
5    "last_data": {
6      "A": "0",
7      "B": "1"
8    },
9    "last_timestamp": "2016-05-30T07:50:52.033Z"
10  },
11  "errorCode": 200,
12  "errorMsg": "Success"
13  }
14  }
15  }
```

[POST] node

URL: <https://town.coxlab.kr/api/v1.0/node>

이 API 를 통해서 IoT.own 에 노드를 등록할 수 있습니다.

Header

- **Token**: API 토큰

Parameters

- **nid**: 노드 아이디
- **type**(optional): 노드 타입
- **desc**(optional): 노드 설명
- **profile**(optional) : LoRa Device Profile 이름 (**deprecated**)
- **lorawan**(optional) : Object 형태로 LoRa Device Profile 과 App key 를 지정할 수 있습니다.
LoRa 용 노드인 경우 필수이며 IoT.own 의 노드 등록에서 Device Profile 목록을 확인
가능합니다. (**2019/1/10 이후 버전부터 가능**)

Response

- **errorCode**: 에러 코드
- **errorMsg**: 에러 내용

Example

cURL

```
1  curl -X POSTW
2      --header 'Content-Type: application/json'W
3      --header 'Accept: application/json'W
4      --header 'Token: your_own_token'W
5      -d '{W
```

```
7      "nid": "LW1111222233334444", W
8      "type": "node type", W
9      "desc": "node desc", W
10     "lorawan": { W
11         "profile": "KR920 1.0.2 A", W
12         "appKey": "11112222333344445555666677778888" W
13     } W
14     } ' W
      'https://town.coxlab.kr/api/v1.0/gateway'
```

Return

```
1  {
2      "errorCode": 200,
3      "errorMsg": "Success"
4  }
```

[DELETE] node

URL: <https://town.coxlab.kr/api/v1.0/node>

이 API 를 통해서 IoT.own 에 등록된 노드를 삭제할 수 있습니다.

Header

- **Token**: API 토큰

Parameters

- **nid**: 노드 아이디

Response

- **errorCode**: 에러 코드
- **errorMsg**: 에러 내용

Example

cURL

```
curl -X DELETEW
1  --header 'Content-Type: application/json'W
2  --header 'Accept: application/json'W
3  --header 'Token: your_own_token'W
4  -d '{W
5      "nid": "LW1111222233334444"W
6  }'W
8  'https://town.coxlab.kr/api/v1.0/node'
```

Return

```
1  {
2    "errorCode": 200,
3    "errorMsg": "Success"
4  }
```

[POST] storage/search (deprecated)

URL: `https://town.coxlab.kr/api/v1.0/storage/search`

이 API 를 통해서 IoT.own 에 저장된 데이터를 가져올 수 있습니다.

Header

- **Token**: API 토큰

Parameters

- **nid** (optional): 노드 아이디. 콤마(,)로 구분. (예 : **nid:"N001,N002,N003"**) **nid** 가 없을 경우 모든 노드를 대상으로 검색
- **from** (optional): 검색할 시간 범위의 시작값 UTC 형태만 가능. (예: **from:2016-06-01T13:45+09:00**)
- **to** (optional): 검색할 시간 범위의 끝값. UTC 형태만 가능. (예: **to:2016-06-25T13:45+09:00**)
- **lastKey** (optional): 검색 결과가 5000 개를 초과할 경우, 5000 개와 함께 **lastKey** 가 리턴됨. 동일 검색 조건에 **lastKey** 를 추가하여 다시 호출하면 다음 5000 개를 리턴함.(반복적으로 5000 개씩 로딩 가능)

Response

- **data**: 검색된 데이터 객체 배열
- **node_id**: 노드 아이디
- **name**: 데이터 이름
- **value**: 데이터 값
- **timestamp**: 데이터가 기록된 시간(UTC)

- **lastKey** (optional): 검색 결과가 5000 개를 초과할 경우, 5000 개와 함께 **lastKey** 가 리턴됨. 동일 검색 조건에 **lastKey** 를 추가하여 다시 호출하면 다음 5000 개를 리턴함.(반복적으로 5000 개씩 로딩 가능)
- **errorCode**: 에러 코드
- **errorMsg**: 에러 내용

Example

cURL

```
curl -X POSTW
1   --header 'Content-Type: application/json'W
2   --header 'Accept: application/json'W
3   --header 'Token: your_own_token'W
4   -d '{W
5       "nid": "N001",W
6       "from": "2016-02-01T13:45+09:00",W
7       "to": "2016-05-30T13:55+09:00"W
8   }'W
9   'https://town.coxlab.kr/api/v1.0/storage/search'
```

Return

```
1   {
2       "data":[
3           {
4               "node_id":"N001",
5               "name":"B",
6               "value":"1",
7               "timestamp":"2016-02-30T07:50:52.033Z"
8           },
9       ],
10      ...
11  }
```

```
12      {
13        "node_id": "N001",
14        "name": "B",
15        "value": "1",
16        "timestamp": "2016-05-30T09:50:21.748Z"
17      }
18    ],
19    "lastKey": "574bf0dbabe178ec1f776758",
20    "errorCode": 200,
    "errorMsg": "Success"
  }
```

위의 예제처럼 **lastKey** 가 리턴된 경우, 아래와 같이 **lastKey** 를 포함하여 아래와 같이 다시 호출하면 다음 5000 개를 불러옴.

```
1  {
2    "nid": "001",
3    "from": "2016-02-01T13:45+09:00",
4    "to": "2016-05-30T13:55+09:00",
5    "lastKey": "574bf0dbabe178ec1f776758"
6  }
```

[GET] storage

URL: <https://town.coxlab.kr/api/v1.0/storage>

이 API 를 통해서 IoT.own 에 저장된 데이터를 가져올 수 있습니다.

Header

- **Token**: API 토큰

Parameters

- **nid** (optional): 노드 아이디. 콤마(,)로 구분. (예 : "**nid=N001,N002,N003**") **nid** 가 없을 경우 모든 노드를 대상으로 검색
- **from** (optional): 검색할 시간 범위의 시작값 UTC 형태만 가능. (예: **from:2018-11-01T13:00:00Z**)
- **to** (optional): 검색할 시간 범위의 끝값. UTC 형태만 가능. (예: **to:2018-11-02T13:00:00Z**)
- **lastKey** (optional): 검색 결과가 5000 개를 초과할 경우, 5000 개와 함께 **lastKey** 가 리턴됨. 동일 검색 조건에 **lastKey** 를 추가하여 다시 호출하면 다음 5000 개를 리턴함.(반복적으로 5000 개씩 로딩 가능)

Response

- **data**: 검색된 데이터 객체 배열
- **node_id**: 노드 아이디
- **name**: 데이터 이름
- **value**: 데이터 값
- **timestamp**: 데이터가 기록된 시간(UTC)
- **lastKey** (optional): 검색 결과가 5000 개를 초과할 경우, 5000 개와 함께 **lastKey** 가 리턴됨. 동일 검색 조건에 **lastKey** 를 추가하여 다시 호출하면 다음 5000 개를 리턴함.(반복적으로 5000 개씩 로딩 가능)
- **errorCode**: 에러 코드
- **errorMsg**: 에러 내용

Example

cURL

```
1  curl -X GETW
2  --header 'Accept: application/json'W
3  --header 'Token: your_own_token'W
4  'https://town.coxlab.kr/api/v1.0/storage?nid=N001&from=2018-11-01T13:00:00Z&
```

Return

```
    {
      "data":[
1      {
2        ...
3        "node_id":"N001",
4        "value": { a : "1" },
5        "timestamp":"2018-11-01T07:50:52.033Z"
6        ...
7      },
8      ...
9      {
10       ...
11       "node_id":"N001",
12       "value": { a : "2" },
13       "timestamp":"2018-11-01T09:50:21.748Z"
14       ...
15     }
16   ],
17   "lastKey":"574bf0dbabe178ec1f776758",
18   "errorCode":200,
19   "errorMsg":"Success"
20 }
21
22
```

위의 예제처럼 **lastKey** 가 리턴된 경우, 아래와 같이 URL 에 **lastKey** 를 포함하여 다시 호출하면 다음 5000 개를 불러옵니다.

```
curl -X GETW
1 --header 'Accept: application/json'W
2 --header 'Token: your_own_token'W
3 'https://town.coxlab.kr/api/v1.0/storage?nid=N001&from=2018-11-01T13:00:00Z&
4 02T13:00:00Z&lastKey=574bf0dbabe178ec1f776758'
```

[PUT] callback

URL: <https://town.coxlab.kr/api/v1.0/callback>

이 API 를 통해서 IoT.own 으로 데이터가 들어올 때마다 특정 URL 로 POST 하도록 설정할 수 있습니다. 자세한 내용은 [How to use callback](#) 을 참고하시기 바랍니다.

Header

- **Token**: API 토큰

Parameters

- **url**: callback URL

Response

- **errorCode**: 에러 코드

- **errorMsg**: 에러 내용

Example

cURL

```
curl -X PUTW
1  --header 'Content-Type: application/json'W
2  --header 'Accept: application/json'W
3  --header 'Token: your_own_token'W
4  -d '{W
5      "url": "http://your.server.url"W
6  }'W
7  'https://town.coxlab.kr/api/v1.0/callback'
```

Return

```
1  {
2    "errorCode":200,
3    "errorMsg":"Success"
4  }
```

[GET] callback

URL: <https://town.coxlab.kr/api/v1.0/callback>

이 API 를 통해서 [\[PUT\] callback](#) 으로 지정한 콜백 URL 을 확인할 수 있습니다.

Header

- **Token**: API 토큰

Response

- **callbackUrl**: 등록된 URL
- **callbackStatus**: URL 의 상태 코드
 - **0**: 등록된 URL 이 없음
 - **1**: 등록된 URL 이 있음
 - **-1**: 콜백 호출이 실패함
- **errorCode**: 에러 코드
- **errorMsg**: 에러 내용

Example

cURL

```
1 curl -X GETW
2   --header 'Accept: application/json'W
3   --header 'Token: your_own_token'W
4   'https://town.coxlab.kr/api/v1.0/callback'
```

Return

```
1 {
2   "callbackUrl": "http://your.server.url",
3   "callbackStatus": "1",
4   "errorCode": 200,
5   "errorMsg": "Success"
6 }
```

[DELETE] callback

URL: `https://town.coxlab.kr/api/v1.0/callback`

이 API 를 통해서 [\[PUT\] callback](#) 으로 지정한 콜백 URL 을 삭제할 수 있습니다.

Header

- **Token**: API 토큰

Response

- **errorCode**: 에러 코드
- **errorMsg**: 에러 내용

Example

cURL

```
1 curl -X DELETEW
2   --header 'Accept: application/json'W
3   --header 'Token: your_own_token'W
4   'https://town.coxlab.kr/api/v1.0/callback'
```

Return

```
1 {
2   "errorCode":200,
3
```

```
4    "errorMsg": "Success"
    }
```

[POST] command

URL: <https://town.coxlab.kr/api/v1.0/command>

이 API 를 통해서 HTTP 를 지원하는 IoT 장치로 명령을 전송할 수 있습니다.

주의 : LoRaWAN 타입의 노드에만 사용 가능합니다.

Header

- **Token**: API 토큰

Parameters

- **nid**: 노드 ID
- **type**: 명령 종류 ("**hex**" or "**string**")
- **command**: 명령 내용
- **lorawan** (LoRaWAN only): LoRaWAN 관련 설정
 - **f_port** (2020/03/30 이후 버전에서만 지원): Downlink 전송시 사용할 fPort 설정. 생략시 1 로 전송. 허용 가능한 범위는 1~222. 범위 밖의 값 입력시 400 반환.

Response

- **errorCode**: 에러 코드
- **errorMsg**: 에러 내용

- **fCnt** (LoRaWAN only): 전송된 downlink 프레임의 fCnt 값

Example

cURL

type 이 "string"인 경우,

```
curl -X POSTW
1  --header 'Content-Type: application/json'W
2  --header 'Accept: application/json'W
3  --header 'Token: your_own_token'W
4  -d '{W
5      "nid":"LW1111222233334444",W
6      "type":"string",W
7      "command":"command_string_here"W
8  }'W
9  'https://town.coxlab.kr/api/v1.0/command'
```

type 이 "hex"인 경우,

```
curl -X POSTW
1  --header 'Content-Type: application/json'W
2  --header 'Accept: application/json'W
3  --header 'Token: your_own_token'W
4  -d '{W
5      "nid":"LW1111222233334444",W
6      "type":"hex",W
7      "command":"FF1234"W
8  }'W
9  'https://town.coxlab.kr/api/v1.0/command'
```


Return

```
1  {  
2    "errorCode":200,  
3    "errorMsg":"Success"  
4  }
```

[DELETE] command

URL: <https://town.coxlab.kr/api/v1.0/command>

이 API 를 통해서 LoRa 서버에 쌓여있는 command queue 를 비울 수 있습니다.

주의 : LoRaWAN 노드에만 사용 가능합니다.

Header

- **Token**: API 토큰

Parameters

- **nid**: 노드 아이디

Response

- **errorCode**: 에러 코드
- **errorMsg**: 에러 내용

Example

cURL

```
curl -X DELETEW
1  --header 'Content-Type: application/json'W
2  --header 'Accept: application/json'W
3  --header 'Token: your_own_token'W
4  -d '{W
5      "nid":"LW1111222233334444",W
6  }'W
7  'https://town.coxlab.kr/api/v1.0/command'
```

Return

```
1  {
2    "errorCode":200,
3    "errorMsg":"Success"
4  }
```

[POST] data

URL: <https://town.coxlab.kr/api/v1.0/data>

이 API 를 통해서 HTTP 를 지원하는 IoT 장치에서 IoT.own 으로 데이터를 전송할 수 있습니다.

Header

- **Token**: API 토큰

Parameters

- **type**: 메시지 형식 (문자열 "2"로 고정)
- **nid**: 노드 아이디
- **data**: 전송할 데이터 내용 (JSON object)

Response

- **errorCode**: 에러 코드
- **errorMsg**: 에러 내용

Example

cURL

```
1  curl -X POSTW
2  --header 'Content-Type: application/json'W
3  --header 'Accept: application/json'W
4  --header 'Token: your_own_token'W
5  -d '{W
6      "type" : "2",W
7      "nid" : "N001",W
9      "data" : {W
10         "temperature":26.1,W
11         "humidity":43W
12     }W
13 }'W
```

'https://town.coxlab.kr/api/v1.0/data'

Return

```
1  {  
2    "errorCode":200,  
3    "errorMsg": "Success"  
4  }
```