TION IOT CORE. СПЕЦИФИКАЦИЯ REST API

Версия документа: 2.0 (14.07.2016)

Содержание

История версий

Версия	Дата	Автор	Содержание изменений
1.0	30.06.2016	Егор Беликов, Павел Мельников	Первая версия документа. Описаны запросы getPackets, postPacket, getCommandBlock, postCommandBlock.
2.0	14.07.2016	Павел Мельников, Егор Беликов	Добавлены запросы status, removeCommandBlock, уточнено содержание пакетов от станций

Введение

Данный документ описывает протокол взаимодействия станции, сервера (back-end) и клиентов (front-end) и содержит описание возможных запросов к сервису TION IoT Core.

Документ предполагает, что читающие знакомы с принципами работы протокола HTTP, а также с форматом JSON.

Используемы определения:

- **Станция, узел** – устройство с набором сенсоров и модулем передачи данных, способное периодически отправлять информацию через сеть Интернет по протоколу HTTP.
- **Пакет данных** – показания сенсоров одной станции за одну временную точку. Состоит из полезной нагрузки (собственно показания датчиков) и сопроводительной метаинформации (идентификатор станции и т.д.)
- **Блок настроек** – набор параметров станции, которые она может запросить у сервера и сохранить в своей памяти для дальнейшего использования. Состоит из полезной нагрузки (собственно настроек) и метаинформации (для какой станции предназначаются настройки)
- **Сервер, сервис, бэк-енд, back-end** – серверное программное обеспечение, принимающее запросы от станций на сохранение пакетов с данными, хранящее их и принимающее запросы на просмотр существующих пакетов, а также на сохранение и чтение блоков настроек. Представляет из себя HTTP-сервер с базой данных.
- **Клиенты, внешние системы, front-end** – программы/устройства, запрашивающие от сервера сохраненные на нем пакеты данных для целей обработки, визуализации и т.д.

Пакеты данных

Пример пакета данных в виде JSON-объекта представлен на врезке справа. Поле values – полезная нагрузка пакета, остальные поля – метаданные пакета.

```
{
  "protocol": "cityair",
  "type": "mobile",
  "node": 30,
  "status": "ok",
  "timestamp":
       "2016-07-14T19:25:47.324Z",
  "values": {
    "LAT": 54.9445,
    "LNG": 83.0943,
    "ALT": 153.967,
    "TMP": 25.9753,
    "HUM": 42.121,
    "PRS": 754.552,
    "SND": 51.766,
    "LGT": 24197,
    "PM2": 39.224,
    "PM10": 10.8455
  }
}
```

Идентификатор узла (поле «node») представляет собой натуральное число и идентифицирует узел в пределах своего протокола.

Протокол (поле «protocol») однозначно определяет набор сенсоров в измерительном узле и набор параметров в “values”. Примеры: «cityair», «cityair-alphasense», «kns». Набор сенсоров и полей полезной нагрузки пакета данных един для всех узлов в пределах одного протокола. Пара «протокол + идентификатор» однозначно определяет станцию; пакеты с одинаковыми значениями протокола и идентификатора считаются пришедшими от одной станции.

Тип (поле «type») определяется протоколом и характеризует логику работы измерительного узла. Для протокола «cityair» определены следующие типы: «static», «mobile». Эти типы характеризуют, в первую очередь, частоту, с которой станция посылает пакеты на сервер.

Поле «timestamp» хранит время сохранения пакета сервером и с достаточной точностью совпадает со временем регистрации данных датчиков. Формат поля – строка, представляющая дату и время в соответствии со стандартом ISO 8601.

***Замечание**. В пакете, посылаемом со станции на сервер, отсутствует поле «timestamp» (т.к. получение точного времени на станции сопряжено с техническими сложностями) и поле заполняется сервером в момент сохранения пакета. При запросе пакета с сервера поле в пакете присутствует.*

Пакеты сохраняются на сервере станциями с помощью запроса postPacket и считываются клиентскими приложениями с помощью запроса getPackets.

Блоки настроек

Пример блока настроек в формате JSON приведен на врезке справа.

```
{
  "protocol": "cityair",
  "node": 30,
  "timestamp": "2016-07-
15T17:26:41.650Z",
  "commands": [
    {
      "c": "changeParameter",
      "p": [ "name", "newValue"  ]
    },
    {
      "c": "writeByte2EEPROM",
      "p": [1024, 127 ]
    }
  ]
}
```

Поля «protocol» и «node» определяют, к какому узлу относятся настройки и имеют тот же формат, что и в пакете данных. Поле «timestamp» содержит время регистрации блока на сервере. *Это поле заполняется сервером при сохранении блока, поэтому может не присутствовать в блоке, отправляющемся на сервер с помощью запроса* postCommandBlock.

Поле «commands» содержит, собственно, настройки для станции. Поле является массивом из объектов, каждый из объектов описывает отдельное действие станции по изменению собственной конфигурации. Формат и семантика этих объектов в данной версии спецификации никак не определена. Пример двух возможных команд по изменению параметра или по записи байтов в энергонезависимую память станции приведены во врезке.

Порядок работы с блоками настроек

Блоки настроек отправляются клиентскими приложениями серверу при помощи HTTP-запроса postCommandBlock и считываются станциями с помощью запроса getCommandBlock. На этот запрос сервер отправляет самый ранний (старый) блок настроек, предназначенный для запрашивающей станции. После того, как станция обработала этот блок (внесла необходимые изменения в конфигурацию), блок настроек удаляется с сервера с помощью запроса removeCommandBlock.

Разграничение прав доступа

Большинство запросов к серверу должны иметь в URL-параметр authkey. Значение этого параметра представляет собой ключ доступа, каждый ключ имеет права, приведенные в таблице ниже. Несоответствие ключа и запроса приведет к тому, что запрос не будет выполнен.

Ключ	Описание	Необходимо указывать в запросах
7JElOsKIpD6s0oDm2nd7	чтение пакетов данных	getPackets
J3UK0ex9RUVhzgFi9BeI	сохранение пакетов на сервере	postPacket
oH9Wcvkp116UQ41b0A0x	чтение блоков настроек	getCommandBlock, removeCommandBlock
9XYpgKdR7zCfSWbQoU16	сохранение блоков настроек на сервере	postCommandBlock

Как видно, станция должна хранить в своей памяти ключ на сохранение пакетов данных и ключ на чтение блоков настроек.

Запросы

Работа с пакетами данных

POSTPACKET

Описание	Запрос на сохранение пакета данных на сервере
Тип	POST
Путь	/postPacket
Параметры запроса	authkey – ключ доступа
MIME-тип содержимого POST-запроса	application/json
MIME-тип ответа	application/json

Запрос выполняется станцией с заданной на станции регулярностью. Ключ доступа для этого запроса «J3UK0ex9RUVhzgFi9BeI».

Телом запроса является пакет данных в формате JSON.

Ответом является JSON-объект со следующими полями:

- result – строка «ok», если запрос выполнен успешно, либо строка «err», если произошла ошибка.
- desc – строка с описанием ошибки (если запрос завершен успешно, поле отсутствует), для отладки. Возможные варианты поля смотри в коде.

Пример:

```
POST /postPacket?authkey=J3UK0ex9RUVhzgFi9BeI HTTP/1.1
Content-Length: 305
Content-Type: application/json
```

{"protocol":"cityair","type":"mobile","node":30,"status":"ok","values":{"LAT":54.9840,"LNG":83.0392,"ALT":156.0593,"TMP":26,"HUM":30,"PRS":764,"SND":54,"LGT":24812,"PM2":32.03,"PM10":10.73}}

```
HTTP/1.1 200 OK
content-type: application/json; charset=utf-8
content-length: 15

{"result":"ok"}
```

GETPACKETS

Описание	Запрос пакетов данных, сохраненных на сервере, отвечающих заданным критериям
Тип	GET
Путь	/getPackets
Параметры запроса	authkey – ключ доступа остальные параметры – см. таблицу ниже.
MIME-тип ответа	application/json

Ключ доступа для данного запроса – «7JElOsKIpD6s0oDm2nd7».

Поддерживаются следующие типы критериев:

Критерий	Параметры запроса	Пример
Протокол	protocol – строка	protocol=cityair
Попадание в заданную область на местности	lat1, lon1, lat2, lon2 – широта и долгота верхнего правого и нижнего левого углов прямоугольника, в котором должны находиться пакеты	lat1=55.1569&lat2=54.8418&lon1=83.4809&lon2=82.5196
Тип узла	type – строка	type=mobile
Идентификатор узла	node – число	node=30
Временной отрезок	before, after – строки, представляющие собой времена в формате ISO8601	before=2016-07-14T18%3A29%3A43.883Z&after=2016-07-14T12

Если в запросе отсутствует временной критерий, то в ответе будут находиться только самые новые пакеты от каждого узла, удовлетворяющего остальным критериям.

Допустимы следующие 6 комбинаций критериев:

1. протокол + попадание в заданную область,
2. протокол + тип узла + попадание в заданную область,
3. протокол + идентификатор узла,
4. протокол + попадание в заданную область + временной отрезок,
5. протокол + тип узла + попадание в заданную область + временной отрезок,
6. протокол + идентификатор узла + временной отрезок.

Ответом является JSON-объект со следующими полями:

- result – строка «ok», если запрос выполнен успешно, либо строка «err», если произошла ошибка.
- desc – строка с описанием ошибки (если запрос завершен успешно, поле отсутствует), для отладки. Возможные варианты поля смотри в коде.
- packets – массив из объектов, описывающих пакеты данных (если запрос не был завершен успешно, поле отсутствует). Может быть пустым, если не было найдено пакетов, удовлетворяющих критериям.

Примеры:

```
GET /getPackets?authkey=7JElOsKIpD6s0oDm2nd7&protocol=cityair&before=2016-07-14T18%3A42%3A53.116Z&after=2016-07-14T12%3A42%3A53.116Z&node=30 HTTP/1.1
```

```
HTTP/1.1 200 OK
content-type: application/json; charset=utf-8
content-length: 715

{"result":"ok","packets":[
{"protocol":"cityair","type":"mobile","node":30,"status":"ok","values":{"LAT":54.9832,"LNG":83.038,"ALT":159.2,"TMP":21,"HUM":11.6,"PRS":752.9,"SND":50.1,"LGT":24960,"PM2":31.7,"PM10":13 }, "timestamp":"2016-07-14T18:01:59.527Z"},
{"protocol":"cityair","type":"mobile","node":30,"status":"ok","values":{"LAT":54.9840,"LNG":83.0392,"ALT":156,"TMP":26,"HUM":30.2,"PRS":764.3,"SND":54.4,"LGT":24812,"PM2":32.03,"PM10":10.7}, "timestamp":"2016-07-14T18:08:29.687Z"}
]}
```

```
GET /getPackets?authkey=7JElOsKIpD6s0oDm2nd7&protocol=cityair&before=2016-07-14T18%3A49%3A23.456Z&after=2016-07-14T12%3A49%3A23.456Z HTTP/1.1
```

```
HTTP/1.1 200 OK
content-type: application/json; charset=utf-8
content-length: 53

{"result":"err","desc":"wrong_set_of_request_fields"}
```

Работа с командами настройки станций

POSTCOMMANDBLOCK

Описание	Запрос на сохранение нового блока настроек на сервере
Тип	POST
Путь	/postCommandBlock
Параметры запроса	authkey – ключ доступа

MIME-тип содержимого POST-запроса	application/json
MIME-тип ответа	application/json

Ключ доступа для данного запроса – «9XYpgKdR7zCfSWbQoU16».

Телом запроса является блок настроек в формате JSON.

Ответом является JSON-объект со следующими полями:

- result – строка «ok», если запрос выполнен успешно, либо строка «err», если произошла ошибка.
- desc – строка с описанием ошибки (если запрос завершен успешно, поле отсутствует), для отладки. Возможные варианты поля смотри в коде.
- registered_as – строка, содержащая протокол, идентификатор и время регистрации данного блока. Предназначено для отладки.

Пример:

```
POST /postCommandBlock?authkey=9XYpgKdR7zCfSWbQoU16 HTTP/1.1
Content-Length: 133
Content-Type: application/json

{"protocol":"cityair","node":30,"commands":[{"c":"changeParameter","p":["name","newValue"]},{
"c":"writeByte2EEPROM","p":[1024,127]}]}
HTTP/1.1 200 OK
content-type: application/json; charset=utf-8
content-length: 70

{"result":"ok", "registered_as":"cityair/30/2016-07-15T17:25:28.511Z"}
```

GETCOMMANDBLOCK

Описание	Запрос блока настроек
Тип	GET
Путь	/getCommandBlock
Параметры запроса	authkey – ключ доступа, protocol – протокол node – идентификатор узла
MIME-тип ответа	application/json

Ключ доступа для данного запроса – «oH9Wcvkp116UQ41b0A0x».

В запросе указывается, для какой станции запрашиваются настройки (с помощью параметров protocol и node).

Ответом является JSON-объект со следующими полями:

- result – строка “ok”, если запрос выполнен успешно, либо строка “err”, если произошла ошибка.

- desc – строка с описанием ошибки (если запрос завершен успешно, поле отсутствует), для отладки.
- block – объект, представляющий блок настроек, сохраненный ранее на сервере. Поле равно null, если для станции нет сохраненных настроек.

Пример.

```
GET /getCommandBlock?authkey=oH9Wcvkp116UQ41b0A0x&protocol=cityair&node=31
HTTP/1.1
Content-Type: application/json

HTTP/1.1 200 OK
content-type: application/json; charset=utf-8
content-length: 196

{"result":"ok","block":{"protocol":"cityair","node":31,"commands":[{"c":"changeParameter","p":["
name","newValue"]},{"c":"writeByte2EEPROM","p":[1024,127]}],"timestamp":"2016-07-
15T17:52:46.640Z"}}
```

REMOVECOMMANDBLOCK

Описание	Запрос удаления самого раннего блока настроек с сервера
Тип	GET
Путь	/removeCommandBlock
Параметры запроса	authkey – ключ доступа, protocol – протокол node – идентификатор узла timestamp – время, указанное блоке настроек.
MIME-тип ответа	application/json

Ключ доступа для данного запроса – «oH9Wcvkp116UQ41b0A0x».

В запросе указывается, для какой станции необходимо удалить настройки (с помощью параметров protocol и node). В качестве дополнительной меры предосторожности, в запросе также должно содержаться время, указанное в блоке настроек, который необходимо удалить. Если значение этого параметра не совпадает со временем, указанном в самом раннем хранящимся на сервере блоке настроек для заданной станции, запрос не будет выполнен (такое возможно, например, если станция попытается дважды удалить один и тот же блок. Без проверки времени возможно удаление блока, который еще не был обработан).

Ответом является JSON-объект со следующими полями:

- result – строка «ok», если запрос выполнен успешно, либо строка «err», если произошла ошибка.
- desc – строка с описанием ошибки (если запрос завершен успешно, поле отсутствует), для отладки. Одно из возможных значений: «date_not_oldest»

Примеры.

```
GET
/removeCommandBlock?authkey=oH9Wcvkp116UQ41b0A0x&protocol=cityair&node=30&timesta
mp=2016-07-05T09:29:46.935Z HTTP/1.1
```

```
HTTP/1.1 200 OK
content-type: application/json; charset=utf-8
content-length: 41

{"result":"err","desc":"date_not_oldest"}
```

```
GET
/removeCommandBlock?authkey=oH9Wcvkp116UQ41b0A0x&protocol=cityair&node=31&timesta
mp=2016-07-15T17:52:46.640Z HTTP/1.1
```

```
HTTP/1.1 200 OK
content-type: application/json; charset=utf-8
content-length: 15

{"result":"ok"}
```

Вспомогательные запросы

STATUS

Описание	Запрос статуса сервера
Тип	GET
Путь	/status
Параметры запроса	нет
MIME-тип ответа	application/json

Применяется для мониторинга сервера.

Ответом является JSON-объект со следующими полями:

- result – строка «ok»
- uptime – вещественное число, время в секундах со старта сервера.

Пример:

```
GET /status HTTP/1.1
```

```
HTTP/1.1 200 OK
content-type: application/json; charset=utf-8
content-length: 33
```

{"result":"ok","uptime":1719.373}