Google Nest: Overview

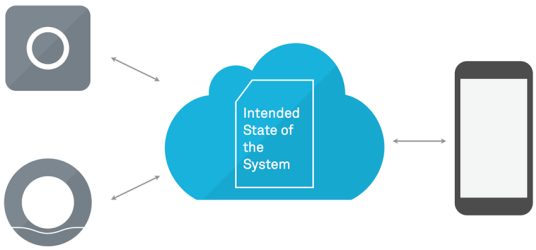


Table of Contents


```
{
  "devices": {
    "thermostats": {
      "peyiJNoTldAksjfAKQ": {
      }
    },
    "smoke_co_alarms": {
      "RTMSkdfjaskAskdjfQAskdjfS": {
      }
    }
  },
  "structures": {
```

```
        "Vqdj439sAjfjk00askAS": {
        }
    }
}
```

2.3 Structures object

2.3.1 Data values

- **Name**: The name of the structure defaults to **"Home"**
- **Metadata**: All data values are read only, unless otherwise specified.
  - `structure_id`: a string that uniquely represents this structure, every developer will see a different ID for the same structure, but multiple clients from the same developer will see the same ID
  - `country_code`: an ISO 3166-1 alpha-2 country code that maps to the registered location of the structure
  - `time_zone`: an IANA time zone string that maps to the structure's time zone
- **Devices**: Nest devices are listed by type as an array of IDs, which can be used to uniquely identify a device via the device path. So a thermostat ID of `peyiJNo0IldT2YlIVtYaGQ` means you could load the thermostats device model at `/devices/thermostats/peyiJNo0IldT2YlIVtYaGQ` via the Firebase API. The same would apply to smoke and CO alarms.
- **Away**: Access to this field requires Away read or Away read/write permission. With the Away read permission you can see if Nest thinks the structure is occupied. The possible states are:
  - `home`: there is someone in the house, either because Nest has high confidence from motion sensor data, or the user has explicitly set the structure to home via a Nest App, your client, or via ETA.
  - `away`: the user has explicitly set the structure to away, either via a Nest App or your client.
  - `auto-away`: Nest has determined algorithmically that no one is home. Currently there is no way to vote on entering this state via the Nest API. Clients are expected to use explicit controls or actions that will set away directly rather than algorithmically.

  Be aware that there are only two ways to change the state from away to home:
  - manually, when the user actively selects it
  - algorithmically, via ETA

  Conversely, the only way to change state from home to away is manually.
- **Energy programs**: If a user has paired their Nest account with a participating energy partner, Nest will know when an energy Rush Hour event is happening. These are peak demand periods predicted by the energy provider, during which time the customer is awarded a rebate if they reduce energy usage.

  If your product uses significant amounts of electricity, and has the ability to delay action or run in a low power mode, listening for Rush Hour Rewards Energy Event fields (`peak_period_start_time` and `peak_period_end_time`) will save your customers money. Notice of an event will range from just-in-time, to as much as 24 hours in advance, so consider multiple algorithms for energy savings, depending on your product's profile.
- **ETA**: Requires ETA permission, and is write only. ETA is an object, set on a structure. Use eta to give Nest information on when it should prepare a house for people arriving.
  - `trip_id`: a unique, client-generated identifier to organize a stream of eta estimates
  - `estimated_arrival_window_begin`: timestamp of the earliest time you expect the user to arrive, in ISO 8601 format
  - `estimated_arrival_window_end`: timestamp of the latest time you expect the user to arrive, in ISO 8601 format

  Because circumstances around trips can change due to traffic, altered user plans and other events, clients should update eta periodically as the trip progresses, providing Nest with a stream of estimated arrival times (ideally once every 5 minutes or so). The more information we receive, the more confident we can be in the reliability of eta inputs, which makes us more confident on taking action to prepare the home.

2.4 Thermostats object

2.4.1 Data values

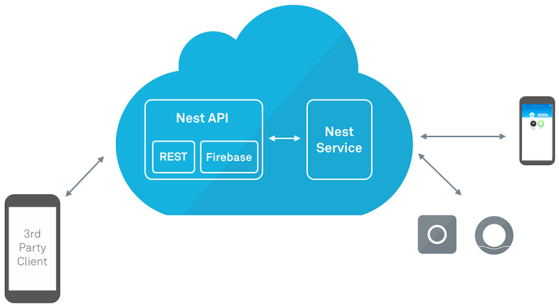
- **Metadata**: all values are read only, unless otherwise specified
  - `device_id`:
    - A string that uniquely represents this device
    - When a device is connected to multiple clients, each developer will see a different ID for that device; for a device that has installed multiple clients from the same developer, the developer will see the same ID
  - `structure_id`: string that uniquely represents this structure; this is the structure that the device is **paired with**
  - `last_connection`
  - `is_online`

2.5 User reports and oversight

3 Nest API

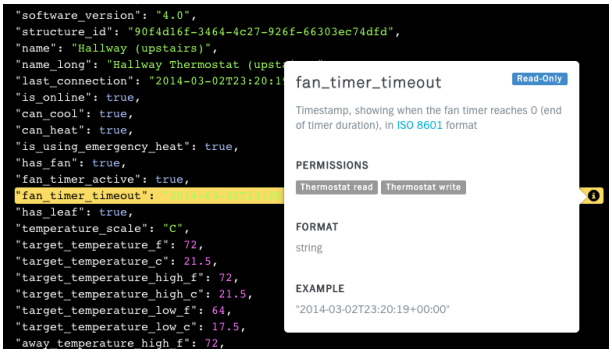
3.1 How Nest API works

- client uses **Firebase client libraries** to subscribe to data values via Nest API
- When a subscribed data value changes, the new value is updated in standard JSON document
- this change is delivered to client and client can update a display or trigger an action based on subscribed data



3.2 Example

- Client for the **Nest Learning Thermostat** that monitors two values: current temperature and target temperature
- When your customer (i.e. human) adjusts the temperature at the Nest Thermostat, the target temperature changes and the Nest service updates the JSON doc, which is synchronized in real time
- Your client **listens for** (HOW?) changes, then:
  - displays the new target temperature
  - updates the current temperature as it changes



4 Authentication and authorization

4.1 Three per-structure permissions

- **Away read**: grants read permission to most of the Structure data values
- **Away read/write**:
  - grants read permission to most of the Structure data values
  - grants read/write permission for **away**
- **ETA write**
  - grants read permission to most of the Structure data values
  - grants write permission for **away**

4.2 Two per-thermostat permissions

- **Thermostat read**
  - Grants read permission to most of the Thermostat and Structure data values
- **Thermostat read/write**
  - Grants read permission to all of the Thermostat data values
  - Grants write permission to these Thermostat data values: `is_fan_timer_active`, all `target_temperature fields` and `hvac_mode`
  - Grants read permission to most of the **Structure** data values

5 REST and REST Streaming Interfaces

5.1 REST Streaming

In addition to the standard REST interface, the Nest service also supports REST Streaming. You may want to use REST if your preferred development environment or programming language has no compatible helper library and does not support **WebSockets**. If your client needs to receive real time updates from the Nest service via REST, then you'll want to use REST Streaming.

Like the standard REST interface, Nest's REST Streaming implementation is based on Firebase. You can learn more about REST Streaming on the Firebase web site. As you read through the Firebase REST Streaming reference, keep in mind these differences with Nest's implementation:

- `cancel` event is not supported
- `patch` event is not supported
- Your REST clients will need to handle redirects with status code 307
- Data rate limiting may be applied to your clients

REST Streaming and rate limits In order to prevent overutilization of the Nest service, we limit the number of connections a client can make in a specific time period. If you're using the Firebase client you shouldn't hit these limits, as a single connection can handle all communication over a long period of time. For REST Streaming calls, each access token has a limited number of read calls.

REST Streaming and redirects Another consideration of using REST Streaming is that your client must also handle 307 redirects.

structures	devices	
	thermostats	smoke_co_alarms
structure_id	device_id	device_id
thermostats	locale	locale
smoke_co_alarms	software_version	software_version
away	structure_id	structure_id
name	name	name
country_code	name_long	name_long
peak_period_start_time	last_connection	last_connection
peak_period_end_time	is_online	is_online
time_zone	can_cool	battery_health
eta	can_heat	co_alarm_state
	can_heat	smoke_alarm_state
	is_using_emergency_heat	ui_color_state
	has_fan	
	fan_timer_active	
	fan_timer_timeout	
	has_leaf	
	temperature_scale	
	target_temperature	
	target_temperature_high	
	target_temperature_low	
	away_temperature_high	
	away_temperature_low	
	hvac_mode	
	ambient_temperature	

Figure 1: Current Nest data model – not sure how to extend this (e.g. add new field)