RoofSnap API Documentation

Accessing the RoofSnap API

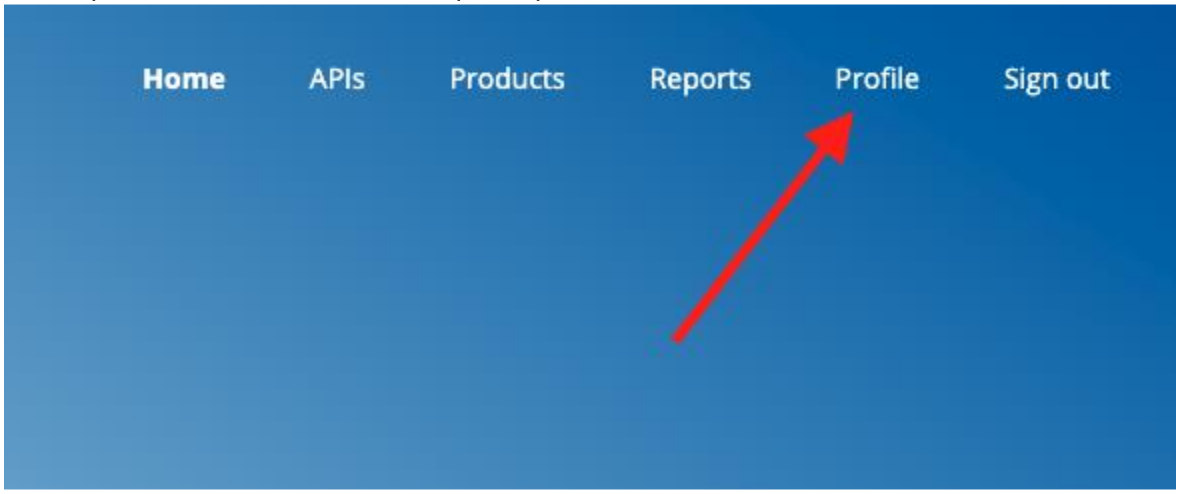
Generally, access to the RoofSnap API requires that two headers be set. The first, **Ocp-Apim-Subscription-Key**, identifies your organization to RoofSnap. Attempting to access the RoofSnap API without this key will result in a 401 response.

The second header, **Authorization**, should be set to a Bearer token acquired from the RoofSnap authentication server. This token requires RoofSnap account credentials to generate. The format of this header is simply `Authorization: Bearer` *`GeneratedToken`*

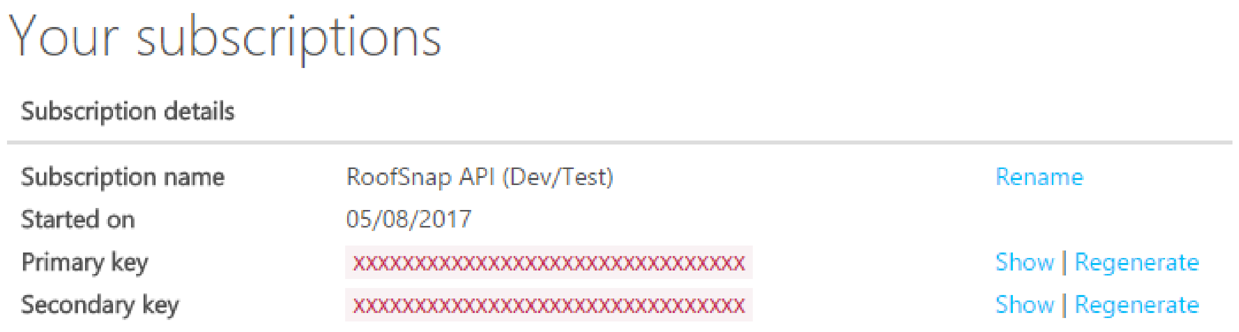
Acquiring your Ocp-Apim-Subscription-Key

1. Sign up for a RoofSnap API developer account at https://roofsnap.developer.azure-api.net/. Each organization integrating with RoofSnap should share a single developer account created here. The signup process will require that a full name be provided for the account. You may make this name a point-of-contact for your organization or fill in your organization name. The email address you use to sign up will be used to reset the developer account password.
2. Once you have completed the signup process, navigate to the "Products" tab of the RoofSnap API portal (https://roofsnap.developer.azure-api.net/products). You should see one product "RoofSnap API (Dev/Test)". Subscribing to this product will grant you access to the RoofSnap dev/test environment. A RoofSnap API administrator will need to approve your subscription request to finalize your access to the API.

3. After a RoofSnap Admin has approved your subscription request, navigate to your developer "Profile" on the RoofSnap API portal:



Your organization's assigned API access keys can be viewed here. Both Primary and Secondary keys provide equal access to the RoofSnap API. These access keys are unique to your organization and should be kept secret. If you need to regenerate your API key(s), you may do so through this interface.



Your subscriptions

Subscription details

Subscription name	RoofSnap API (Dev/Test)	Rename
Started on	05/08/2017	
Primary key	xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx	Show | Regenerate
Secondary key	xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx	Show | Regenerate

Acquiring a RoofSnap API Bearer token

RoofSnap uses Json Web Tokens (JWT) (https://jwt.io/) to secure access to the RoofSnap API at the account level. JWT tokens are generated with RoofSnap account credentials (for the dev/test environment, these credentials can be created at: https://dev-app.roofsnap.com/signup). RoofSnap recommends setting up a dedicated RoofSnap user account for API access.

1. Execute a POST to https://dev-auth2.roofsnap.com/oauth2/token
    a. Content-Type: application/x-www-form-urlencoded
    b. Request Body:

Key	Value
username	[RoofSnapAccountUsername]
password	[RoofSnapAccountPassword]
grant_type	password
client_id	1af9339e-2f1f-4963-8c00-c7093471b48f

    c. Sample Auth Request:

```
POST https://dev-auth2.roofsnap.com/oauth2/token HTTP/1.1
cache-control: no-cache
Content-Type: application/x-www-form-urlencoded
Accept: */*
Host: dev-auth2.roofsnap.com
accept-encoding: gzip, deflate
content-length: 122
Connection: close

username=user%40roofsnap.com&password=********&grant_type=password&client_id=1af
9339e-2f1f-4963-8c00-c7093471b48f
```

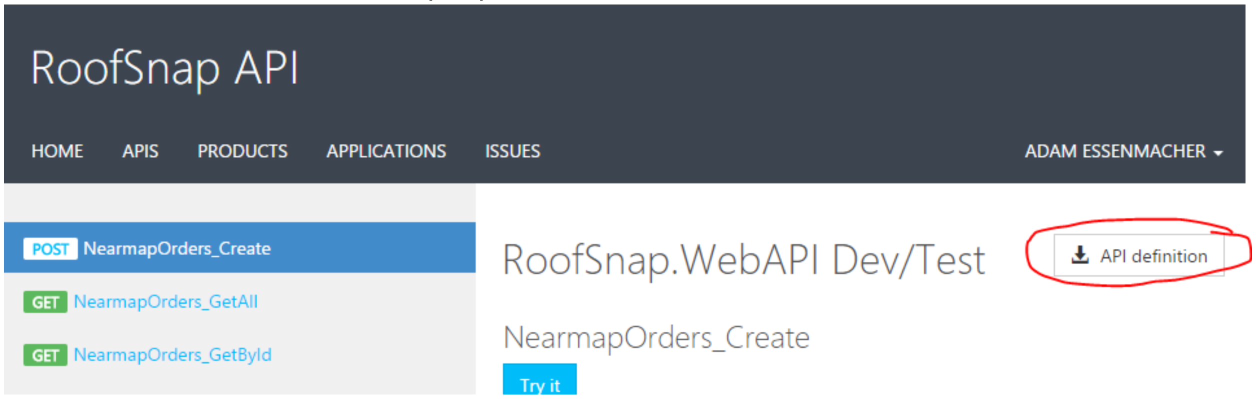
Sample Auth Response:

```
{
  "access_token":
"eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJuYW1lIjoiNiIsInN1YiI6ImFkbWluQHJvb2Zzbm
FwLmNvbSIsImh0dHA6Ly9zY2hlbWFzLm1pY3Jvc29mdC5jb20vd3MvMjAwOC8wNi9pZGVudGl0eS9jbG
FpbXMvcm9sZSI6WyJBY2NvdW50IEFkbWluIiwiQWRtaW4iLCJNQVgiXSwibmJmIjoxNDk0MjYyMDk2LC
JleHAiOjE0OTQ4NjY4OTYsImlzcyI6Imh0dHBzOi8vcm9vZnNuYXAtZGV2LWF1dGguYXp1cmV3ZWJzaX
Rlcy5uZXQiLCJhdWQiOiIxYWY5MzM5ZS0yZjFmLTQ5NjMtOGMwMC1jNzA5MzQ3MWI0OGYifQ.25DTKEl
nHCdWt6UdWhQff_N5tf-HmXJP-y8PdVXxSJQ",
  "token_type": "bearer",
  "expires_in": 604799
}
```

Navigating the RoofSnap API

The RoofSnap API uses the HyperText Application Language (HAL) to expose a discoverable, REST-ful API. The IEFT draft for HAL can be found at https://tools.ietf.org/html/draft-kelly-json-hal-08. A simple explanation of HAL is available at http://stateless.co/hal_specification.html

Available operations on the RoofSnap API are documented using the OpenAPI specification: https://github.com/OAI/OpenAPI-Specification. The RoofSnap API specification may be downloaded from the API developer portal:



Once downloaded, the specification may be uploaded to the online “Swagger” editor at http://editor.swagger.io/

Server response codes are not included in this specification. In general, the RoofSnap API returns the following response codes as appropriate: 200, 201, 401, 403, 404, 409, 428, and 500.

Playing nice with the RoofSnap API

The RoofSnap API is a shared resource. RoofSnap may impose rate limiting to prevent accidental abuse from degrading service to other customers. We ask that you do not ‘poll’ RoofSnap for data at intervals less than 5 minutes. We have a limited webhook platform established and may add additional webhooks as they’re requested. Submit your requests at support@roofsnap.com.

Additional Information

1. Get org ID
    a. Use /whoami to get the organizationID and userid of the currently logged in user
    b. You can also use https://roofsnap.azure-api.net/dev/v1/organizations/{{OrganizationId}}/userprofiles to get other user's userID in your Organization
2. Create a sketch order
    a. There are 2 options a bulk option to send a list of orders and a single option
    b. POST https://roofsnap.azure-api.net/dev/v1/sketchorders with a body of:

```
{
  "Organizationid": {{OrganizationId}},
  "Address": "4249 Easton Way",
  "City": "Columbus",
  "Region": "OH",
  "Postcode": "43202",
  "Country": "United States",
  "Notes": "Example Notes Optional",
  "Projectname": "Example Project Name Optional",
  "SketchReportType": "FullSnap",
  "RequesterUserId": {{UserId}},
  "OwnerUserId": {{UserId}},
  "SketchOptions": "MainBuildingOnly"
}
```

    c. You can also Optionally specify an ExternalID at this point if you would like to link this back to a Salesforce Opportunity
3. Get the ProjectId on a completed Order
    a. At this point the sketch order will just be in a pending status and does not have a projectID associated with it yet. It will not have a projectID until the order is actually completed. When testing in dev we may need to coordinate some for me to complete some of your orders so that you can get a projectID. You can check the response for
    b. "ProjectClonedForDelivery": true,
       "SketchOrderStatus": "Complete", or "Billed"
    c. If an order is rejected it will have status of "Incomplete"
    d. This is also the point where you can set your system to listen for a webhook that we will push to you when a project is created in your account.
5. Get the Measurements

    a. Use v1/projects/{{ProjectId}}/projectdrawing to get the project drawing - you can ignore most of the response except for the Measurements Object that is returned it should have the ActualSquares (measured square count without waste percentage applied), TotalSquares (with waste), Waste Percentage, And then linear measurements for Eaves, Rakes etc.

6. Get the Sketch Order Report Document
    a. First get a list of "renderings" using v1/projects/{{ProjectId}}/v2.1/documentrenderings
    b. This will return a documentID and rendering ID which you can use to get a SAS token do a POST to v2/documents/{{DocumentId}}/renderings/{{RenderingId}}/sharedaccesstokens
    c. The SAS token is a url that you can use to download the pdf - it can occasionally act weird based on your URL encoder/decoder should only replace the spaces with + symbols, and not URL encode the entire URL because the token section is already encoded.

About webhooks:
Our webhooks will send a request to whatever URL you specify that we should send it to and this is what they payload will look like:
{ "id": "PROJECTID", "subject": "/projects/PROJECTID, "data": { "ResourceId": "PROJECTID", "OrganizationId":{{OrganizationId}}.. }, "eventType": "RoofSnap.ProjectCreated", "dataVersion": "1", "metadataVersion": "1", "eventTime": "2022-06-22T16:11:56.5145884Z"...}