

---

# **ProKnow DS - Python SDK Documentation**

**Kyle Burnett**

**Jan 28, 2019**



---

## Contents:

---

<b>1</b>	<b>Usage</b>	<b>3</b>
1.1	Installation . . . . .	3
1.2	Basic Usage . . . . .	3
1.3	Troubleshooting . . . . .	4
<b>2</b>	<b>API Reference</b>	<b>5</b>
2.1	proknow.ProKnow – ProKnow . . . . .	5
2.2	proknow.Requestor – Requestor . . . . .	6
2.3	proknow.Exceptions – Exceptions . . . . .	8
2.4	proknow.CustomMetrics – Custom Metrics . . . . .	9
2.5	proknow.Workspaces – Workspaces . . . . .	13
2.6	proknow.Roles – Roles . . . . .	17
2.7	proknow.Users – Users . . . . .	21
2.8	proknow.Uploads – Uploads . . . . .	25
2.9	proknow.Patients – Patients . . . . .	29
<b>3</b>	<b>Release History</b>	<b>43</b>
3.1	v0.3.0 . . . . .	43
3.2	v0.2.0 . . . . .	43
3.3	v0.1.0 . . . . .	44
<b>4</b>	<b>Contributing Guide</b>	<b>45</b>
4.1	Requirements . . . . .	45
4.2	Accepted Contributions . . . . .	45
<b>5</b>	<b>Indices and tables</b>	<b>47</b>
	<b>Python Module Index</b>	<b>49</b>



Welcome to the ProKnow SDK documentation. Use the links below to access the usage page and the complete api documentation.



# CHAPTER 1

---

## Usage

---

Read this guide to learn how to install ProKnow SDK and get started with your first script.

### 1.1 Installation

Use `pip` to install:

```
pip install --upgrade proknow
```

### 1.2 Basic Usage

To use the SDK, you'll need two important items:

- **Base URL:** This is the URL that will be used to make API requests to ProKnow. It's the first part of the web address you see when you are signed in to your ProKnow DS account. The form of this URL is typically `https://your-domain.proknow.com`.
- **API Token:** You can generate an [API token](#) in the ProKnow DS user interface. Always keep your API Token secret. Once you have your `credentials.json`, make note of the file path.

The code snippet below can be used to test your installation of the `proknow` package. Please be sure to replace `https://example.proknow.com` with your organization's Base URL described above, and replace `./credentials.json` with the path to your credentials file:

```
from proknow import ProKnow

pk = ProKnow('https://example.proknow.com', credentials_file='./credentials.json')
workspaces = pk.workspaces.query()
print("The workspaces are as follows:")
for workspace in workspaces:
    print(workspace.name)
```

If an adaptation of the script above does not work for you see the troubleshooting section below.

### 1.3 Troubleshooting

#### Connection errors

These can be caused by the failure to replace `https://example.proknow.com` as your `base_url`. `https://example.proknow.com` is not a real endpoint and will not work. If you've already replaced the base URL, login to your ProKnow DS account and check to make sure it matches the base URL shown in the address bar.

If you are able to login to your ProKnow DS account and have verified that the base URLs match, there is also a chance that your organization's firewall is configured to block traffic originating from unknown sources like your Python script. If this is the case please contact your IT department to request special permission for your script to access the ProKnow API from inside your organization's network.

#### File not found errors

If you specified a `credentials_file` in the ProKnow constructor, but are receiving a message saying that the file or directory does not exist, you may have a typo in your credentials file path. Correct the file path, and try again. Note that relative file paths are usually relative to the directory from which the `python` executable is being invoked.

#### Type or Assertion errors

These indicate that you've provided invalid parameters to the ProKnow constructor. The error message can usually point you in the right direction. Correct the parameters, and try again.



This guide includes a complete API reference.

## 2.1 proknow.ProKnow – ProKnow

**class** proknow.ProKnow.**ProKnow**(*base\_url*, *credentials\_file*=None, *credentials\_id*=None, *credentials\_secret*=None)

This is the main class that should be instantiated at the beginning of your Python program with your base URL (which should include your account subdomain) and your API token credentials.

**requestor**

*proknow.Requestor.Requestor* – An instance of the Requestor class.

**custom\_metrics**

*ProKnow.CustomMetrics.CustomMetrics* – An instance of the Custom Metrics class.

**workspaces**

*proknow.Workspaces.Workspaces* – An instance of the Workspaces class.

**roles**

*proknow.Roles.Roles* – An instance of the Roles class.

**users**

*proknow.Users.Users* – An instance of the Users class.

**uploads**

*proknow.Uploads.Uploads* – An instance of the Uploads class.

**patients**

*proknow.Patients.Patients* – An instance of the Patients class.

**\_\_init\_\_**(*base\_url*, *credentials\_file*=None, *credentials\_id*=None, *credentials\_secret*=None)

Initializes the ProKnow class.

The *base\_url* must be provided as should either the *credentials\_file* or both the *credentials\_id* and *credentials\_secret*.

**Parameters**

- **base\_url** (*str*) – The base URL to use when making request to the ProKnow API.
- **credentials\_file** (*str*) – The path to the credentials file obtained by creating an API key in your ProKnow account. The file should be a JSON file containing an object with the fields *id* and *secret*.
- **credentials\_id** (*str*) – An API key id.
- **credentials\_secret** (*str*) – An API key secret.

**Raises** `AssertionError` – If the input parameters are invalid.

## 2.2 proknow.Requestor – Requestor

**class** proknow.Requestor.**Requestor** (*base\_url, username, password*)

A class used for issuing requests for the ProKnow API

**\_\_init\_\_** (*base\_url, username, password*)

Initializes the Requestor class.

**Parameters**

- **base\_url** (*str*) – The base URL to use when making request to the ProKnow API.
- **username** (*str*) – The string used in Basic Authentication as the user name.
- **password** (*str*) – The string used in Basic Authentication as the user password.

**delete** (*route, query=None, body=None*)

Issues an HTTP DELETE request.

**Parameters**

- **route** (*str*) – The API route to use in the request.
- **query** (*dict, optional*) – An optional dictionary of query parameters to use in the request.
- **body** (*dict, optional*) – An optional dictionary to be used as a JSON request body for the request.

**Returns**

A tuple (status\_code, msg).

1. status\_code (int): the response code
2. msg (str, dict): the text response or, if the response was JSON, the decoded JSON dictionary.

**Return type** tuple

**get** (*route, query=None*)

Issues an HTTP GET request.

**Parameters**

- **route** (*str*) – The API route to use in the request.
- **query** (*dict, optional*) – An optional dictionary of query parameters to use in the request.

**Returns**

A tuple (status\_code, msg).

1. status\_code (int): the response code
2. msg (str, dict): the text response or, if the response was JSON, the decoded JSON dictionary.

**Return type** tuple

**patch** (route, query=None, body=None)

Issues an HTTP PATCH request.

**Parameters**

- **route** (str) – The API route to use in the request.
- **query** (dict, optional) – An optional dictionary of query parameters to use in the request.
- **body** (dict, optional) – An optional dictionary to be used as a JSON request body for the request.

**Returns**

A tuple (status\_code, msg).

1. status\_code (int): the response code
2. msg (str, dict): the text response or, if the response was JSON, the decoded JSON dictionary.

**Return type** tuple

**post** (route, query=None, body=None)

Issues an HTTP POST request.

**Parameters**

- **route** (str) – The API route to use in the request.
- **query** (dict, optional) – An optional dictionary of query parameters to use in the request.
- **body** (dict, optional) – An optional dictionary to be used as a JSON request body for the request.

**Returns**

A tuple (status\_code, msg).

1. status\_code (int): the response code
2. msg (str, dict): the text response or, if the response was JSON, the decoded JSON dictionary.

**Return type** tuple

**put** (route, query=None, body=None)

Issues an HTTP PUT request.

**Parameters**

- **route** (str) – The API route to use in the request.
- **query** (dict, optional) – An optional dictionary of query parameters to use in the request.

- **body** (*dict*, *optional*) – An optional dictionary to be used as a JSON request body for the request.

**Returns**

A tuple (status\_code, msg).

1. status\_code (int): the response code
2. msg (str, dict): the text response or, if the response was JSON, the decoded JSON dictionary.

**Return type** tuple

**stream** (*route*, *path*)

Issues an HTTP PUT request.

**Parameters**

- **route** (*str*) – The API route to use in the request.
- **path** (*str*) – The file path to stream the request response
- **query** (*dict*, *optional*) – An optional dictionary of query parameters to use in the request.
- **body** (*dict*, *optional*) – An optional dictionary to be used as a JSON request body for the request.

## 2.3 proknow.Exceptions – Exceptions

**exception** proknow.Exceptions.**ProKnowError**

All errors involved in interacting with the ProKnow DS - Python SDK modules extend this.

**exception** proknow.Exceptions.**HttpError** (*status\_code*, *body*)

Errors produced at the HTTP layer.

**status\_code**

*int* – The status code.

**body**

*str* – The response body.

**\_\_init\_\_** (*status\_code*, *body*)

Initializes the WorkspaceLookupError class.

**Parameters**

- **status\_code** (*int*) – The status code.
- **body** (*str*) – The response body.

**exception** proknow.Exceptions.**WorkspaceLookupError** (*message*)

Indicates that there was a problem looking up the workspace by the given identifier.

**message**

*str* – An explanation of the error.

**\_\_init\_\_** (*message*)

Initializes the WorkspaceLookupError class.

**Parameters** **message** (*str*) – An explanation of the error.

**exception** `proknow.Exceptions.CustomMetricLookupError(message)`  
 Indicates that there was a problem looking up the custom metric by the given identifier.

**message**  
*str* – An explanation of the error.

**\_\_init\_\_**(*message*)  
 Initializes the CustomMetricLookupError class.

**Parameters** **message** (*str*) – An explanation of the error.

**exception** `proknow.Exceptions.InvalidPathError(message)`  
 Indicates that the provided path is not valid.

**message**  
*str* – An explanation of the error.

**\_\_init\_\_**(*message*)  
 Initializes the InvalidPathError class.

**Parameters** **message** (*str*) – An explanation of the error.

**exception** `proknow.Exceptions.TimeoutExceededError(message)`  
 Indicates that the current operation exceeded the timeout limits.

**message**  
*str* – An explanation of the error.

**\_\_init\_\_**(*message*)  
 Initializes the TimeoutExceededError class.

**Parameters** **message** (*str*) – An explanation of the error.

## 2.4 `proknow.CustomMetrics` – Custom Metrics

**class** `proknow.CustomMetrics.CustomMetrics(proknow, requestor)`  
 This class should be used to interact with the custom metrics in a ProKnow organization. It is instantiated for you as an attribute of the `proknow.ProKnow.ProKnow` class.

**\_\_init\_\_**(*proknow, requestor*)  
 Initializes the CustomMetrics class.

**Parameters**

- **proknow** (`proknow.ProKnow.ProKnow`) – The ProKnow instance that is instantiating the object.
- **requestor** (`proknow.Requestor.Requestor`) – An object used to make API requests.

**create**(*name, context, metric\_type*)  
 Creates a new custom metric.

**Parameters**

- **name** (*str*) – The custom metric name.
- **context** (*str*) – The custom metric context.
- **metric\_type** (*dict*) – The custom metric type.

**Returns** A representation of the created custom metric.

**Return type** `proknow.CustomMetrics.CustomMetricItem`

**Raises**

- `AssertionError` – If the input parameters are invalid.
- `proknow.Exceptions.HttpError` – If the HTTP request generated an error.

### Example

This example creates a three new custom metrics, demonstrating each of the available custom metric types:

```
from proknow import ProKnow

pk = ProKnow('https://example.proknow.com', credentials_file="./credentials.
↪json")
pk.custom_metrics.create('Genetic Type', "patient", {
    "enum": {
        "values": ["Type I", "Type II", "Type III", "Type IV"]
    }
})
pk.custom_metrics.create('Physician Name', "patient", {
    "string": {}
})
pk.custom_metrics.create('Toxicity of Larynx (1-4)', "patient", {
    "number": {}
})
```

**delete** (*custom\_metric\_id*)

Deletes a custom metric by id.

**Parameters** `custom_metric_id` (*str*) – The id of the custom metric to delete.

**Raises**

- `AssertionError` – If the input parameters are invalid.
- `proknow.Exceptions.HttpError` – If the HTTP request generated an error.

### Example

If you know the custom metric id, you can delete the custom metric directly using this method:

```
from proknow import ProKnow

pk = ProKnow('https://example.proknow.com', credentials_file="./credentials.
↪json")
pk.custom_metrics.delete('5c463a6c040040f1efda74db75c1b121')
```

**find** (*predicate=None, \*\*props*)

Finds the first custom metric that matches the input paramters.

---

**Note:** This method utilizes a cache of custom metrics. Once it has a cache of custom metrics, it will use that cache until the `proknow.CustomMetrics.CustomMetrics.query()` method is called to refresh the cache. If you wish to make your code resilient to custom metric changes (i.e., new custom metrics, renamed custom metrics, deleted custom metrics, etc.) while your script is running, you should

call the `proknow.CustomMetrics.CustomMetrics.query()` method before this method. In most use cases, this is not necessary.

#### Parameters

- **predicate** (*func*) – A function that is passed a metric as input and which should return a bool indicating whether the metric is a match.
- **\*\*props** – A dictionary of keyword arguments that may include any custom metric attribute. These arguments are considered in turn to find matching custom metrics.

**Returns** A representation of the matching custom metric.

**Return type** `proknow.CustomMetrics.CustomMetricItem`

**Raises** `proknow.Exceptions.HttpError` – If the HTTP request generated an error.

#### `query()`

Queries for custom metrics.

---

**Note:** This method refreshes the custom metric cache.

---

**Returns** A list of `proknow.CustomMetrics.CustomMetricItem` objects, each representing a custom metric in the organization.

**Return type** list

**Raises** `proknow.Exceptions.HttpError` – If the HTTP request generated an error.

#### Example

This example queries the custom metrics and prints the name of each custom metric:

```
from proknow import ProKnow

pk = ProKnow('https://example.proknow.com', credentials_file="./credentials.
↪ json")
for custom_metric in pk.custom_metrics.query():
    print(custom_metric.name)
```

#### `resolve(custom_metric)`

Resolves a metric by id or name.

**Parameters** **custom\_metric** (*str*) – The metric id or name.

**Returns** A representation of the resolved custom metric.

**Return type** `proknow.CustomMetrics.CustomMetricItem`

**Raises**

- `AssertionError` – If the input parameters are invalid.
- `proknow.Exceptions.HttpError` – If the HTTP request generated an error.
- `proknow.Exceptions.CustomMetricLookupError` – If the custom metric with the given id or name could not be found.

**resolve\_by\_id**(*custom\_metric\_id*)

Resolves a custom metric id to a custom metric.

**Parameters** **custom\_metric\_id** (*str*) – The custom metric id.

**Returns** A representation of the resolved custom metric.

**Return type** *proknow.CustomMetrics.CustomMetricItem*

**Raises**

- `AssertionError` – If the input parameters are invalid.
- *proknow.Exceptions.HttpError* – If the HTTP request generated an error.
- *proknow.Exceptions.CustomMetricLookupError* – If the custom metric with the given id could not be found.

**resolve\_by\_name**(*name*)

Resolves a custom metric name to a custom metric.

**Parameters** **name** (*str*) – The custom metric name.

**Returns** A representation of the resolved custom metric.

**Return type** *proknow.CustomMetrics.CustomMetricItem*

**Raises**

- `AssertionError` – If the input parameters are invalid.
- *proknow.Exceptions.HttpError* – If the HTTP request generated an error.
- *proknow.Exceptions.CustomMetricLookupError* – If the custom metric with the given name could not be found.

**class** `proknow.CustomMetrics.CustomMetricItem`(*custom\_metrics*, *custom\_metric*)

This class represents a custom metric. It's instantiated by the *proknow.CustomMetrics.CustomMetrics* class to represent each of the custom metrics in a query result and a created custom metric.

**id**

*str* – The id of the custom metric (readonly).

**data**

*dict* – The complete representation of the custom metric as returned from the API (readonly).

**name**

*str* – The name of the custom metric.

**context**

*str* – The context of the custom metric.

**type**

*dict* – The type of the custom metric (readonly).

**\_\_init\_\_**(*custom\_metrics*, *custom\_metric*)

Initializes the CustomMetricItem class.

**Parameters**

- **custom\_metrics** (*proknow.CustomMetrics.CustomMetrics*) – The CustomMetrics instance that is instantiating the object.
- **custom\_metric** (*dict*) – A dictionary of custom metric attributes.



**delete()**

Deletes the custom metric.

**Raises** `proknow.Exceptions.HttpError` – If the HTTP request generated an error.

**Example**

The following example shows how to find a custom metric by its name and delete it:

```
from proknow import ProKnow

pk = ProKnow('https://example.proknow.com', credentials_file="./credentials.
↪json")
metric = pk.custom_metric.find(name='Type')
metric.name = "Genetic Type"
metric.save()
```

**save()**

Saves the changes made to a custom metric.

**Raises** `proknow.Exceptions.HttpError` – If the HTTP request generated an error.

**Example**

The following example shows how to find a custom metric by its name, change the name, and save it:

```
from proknow import ProKnow

pk = ProKnow('https://example.proknow.com', credentials_file="./credentials.
↪json")
metric = pk.custom_metric.find(name='Type')
metric.name = "Genetic Type"
metric.save()
```

## 2.5 proknow.Workspaces – Workspaces

**class** `proknow.Workspaces.Workspaces` (*proknow, requestor*)

This class should be used to interact with the workspaces in a ProKnow organization. It is instantiated for you as an attribute of the `proknow.ProKnow.ProKnow` class.

**\_\_init\_\_** (*proknow, requestor*)

Initializes the Workspaces class.

**Parameters**

- **proknow** (`proknow.ProKnow.ProKnow`) – The ProKnow instance that is instantiating the object.
- **requestor** (`proknow.Requestor.Requestor`) – An object used to make API requests.

**create** (*slug, name, protected=True*)

Creates a new workspace.

**Parameters**

- **slug** (*str*) – The workspace slug.
- **name** (*str*) – The workspace name.
- **protected** (*bool, optional*) – Indicates whether the workspace should be protected from accidental deletion.

**Returns** A representation of the created workspace.

**Return type** `proknow.Workspaces.WorkspaceItem`

**Raises**

- `AssertionError` – If the input parameters are invalid.
- `proknow.Exceptions.HttpError` – If the HTTP request generated an error.

### Example

This example creates a new workspace with the slug “research” and the name “Research”. The protected argument is not provided, so it will default to True:

```
from proknow import ProKnow

pk = ProKnow('https://example.proknow.com', credentials_file="./credentials.
↪json")
pk.workspaces.create('research', 'Research')
```

**delete** (*workspace\_id*)

Deletes a workspace.

**Parameters** **workspace\_id** (*str*) – The id of the workspace to delete.

**Raises**

- `AssertionError` – If the input parameters are invalid.
- `proknow.Exceptions.HttpError` – If the HTTP request generated an error.

### Example

If you know the workspace id, you can delete the workspace directly using this method:

```
from proknow import ProKnow

pk = ProKnow('https://example.proknow.com', credentials_file="./credentials.
↪json")
pk.workspaces.delete('5c463a6c040040f1efda74db75c1b121')
```

**find** (*predicate=None, \*\*props*)

Finds the first workspace that matches the input paramters.

---

**Note:** This method utilizes a cache of workspaces. Once it has a cache of workspaces, it will use that cache until the `proknow.Workspaces.Workspaces.query()` method is called to refresh the cache. If you wish to make your code resilient to workspaces changes (i.e., new workspaces, renamed workspaces, deleted workspaces, etc.) while your script is running, you should call the `proknow.Workspaces.Workspaces.query()` method before this method. In most use cases, this is not necessary.

---

**Parameters**

- **predicate** (*func*) – A function that is passed a workspace as input and which should return a bool indicating whether the workspace is a match.
- **\*\*props** – A dictionary of keyword arguments that may include any workspace attribute. These arguments are considered in turn to find matching workspaces.

**Returns** A representation of the matching workspace.

**Return type** `proknow.Workspaces.WorkspaceItem`

**Raises** `proknow.Exceptions.HttpError` – If the HTTP request generated an error.

**query()**

Queries for workspaces.

---

**Note:** This method refreshes the workspaces cache.

---

**Returns** A list of `proknow.Workspaces.WorkspaceItem` objects, each representing a workspace in the organization.

**Return type** list

**Raises** `proknow.Exceptions.HttpError` – If the HTTP request generated an error.

**Example**

This example queries the workspaces and prints the name of each workspace:

```
from proknow import ProKnow

pk = ProKnow('https://example.proknow.com', credentials_file="./credentials.
↪ json")
for workspace in pk.workspaces.query():
    print(workspace.name)
```

**resolve(workspace)**

Resolves a workspace by id or name.

**Parameters** **workspace** (*str*) – The workspace id or name.

**Returns** A representation of the resolved workspace.

**Return type** `proknow.Workspaces.WorkspaceItem`

**Raises**

- `AssertionError` – If the input parameters are invalid.
- `proknow.Exceptions.HttpError` – If the HTTP request generated an error.
- `proknow.Exceptions.WorkspaceLookupError` – If the workspace with the given name could not be found.

**resolve\_by\_id(workspace\_id)**

Resolves a workspace id to a workspace.

**Parameters** **workspace\_id** (*str*) – The workspace id.

**Returns** A representation of the resolved workspace.

**Return type** `proknow.Workspaces.WorkspaceItem`

**Raises**

- `AssertionError` – If the input parameters are invalid.
- `proknow.Exceptions.HttpError` – If the HTTP request generated an error.
- `proknow.Exceptions.WorkspaceLookupError` – If the workspace with the given name could not be found.

**resolve\_by\_name** (*name*)

Resolves a workspace name to a workspace.

**Parameters** **name** (*str*) – The workspace name.

**Returns** A representation of the resolved workspace.

**Return type** `proknow.Workspaces.WorkspaceItem`

**Raises**

- `AssertionError` – If the input parameters are invalid.
- `proknow.Exceptions.HttpError` – If the HTTP request generated an error.
- `proknow.Exceptions.WorkspaceLookupError` – If the workspace with the given name could not be found.

**class** `proknow.Workspaces.WorkspaceItem` (*workspaces, workspace*)

This class represents a workspace. It's instantiated by the `proknow.Workspaces.Workspaces` class to represent each of the workspaces in a query result and a created workspace.

**id**

*str* – The id of the workspace (readonly).

**data**

*dict* – The complete representation of the workspace as returned from the API (readonly).

**slug**

*str* – A string used in the URL that uniquely identifies the workspace.

**name**

*str* – The name of the workspace.

**protected**

*bool* – Indicates whether the workspace should be protected from accidental deletion.

**\_\_init\_\_** (*workspaces, workspace*)

Initializes the WorkspaceItem class.

**Parameters**

- **workspaces** (`proknow.Workspaces.Workspaces`) – The Workspaces instance that is instantiating the object.
- **workspace** (*dict*) – A dictionary of workspace attributes.

**delete** ()

Deletes the workspace.

**Raises** `proknow.Exceptions.HttpError` – If the HTTP request generated an error.

## Example

The following example shows how to find a workspace by its slug and delete it:

```
from proknow import ProKnow

pk = ProKnow('https://example.proknow.com', credentials_file="./credentials.
↪json")
research = pk.workspaces.find(slug='research')
research.delete()
```

**save()**

Saves the changes made to a workspace.

**Raises** `proknow.Exceptions.HttpError` – If the HTTP request generated an error.

## Example

The following example shows how to find a workspace by its slug, modify the name, and save it:

```
from proknow import ProKnow

pk = ProKnow('https://example.proknow.com', credentials_file="./credentials.
↪json")
clinical = pk.workspaces.find(slug='clinical')
clinical.name = "Clinical Patients"
clinical.save()
```

## 2.6 proknow.Roles – Roles

**class** `proknow.Roles.Roles` (*proknow, requestor*)

This class should be used to interact with the roles in a ProKnow organization. It is instantiated for you as an attribute of the `proknow.ProKnow.ProKnow` class.

**\_\_init\_\_** (*proknow, requestor*)

Initializes the Roles class.

### Parameters

- **proknow** (`proknow.ProKnow.ProKnow`) – The ProKnow instance that is instantiating the object.
- **requestor** (`proknow.Requestor.Requestor`) – An object used to make API requests.

**create** (*name, permissions*)

Creates a new role.

### Parameters

- **name** (*str*) – The name of the role.
- **permissions** (*dict*) – A dictionary of permissions.

**Returns** A representation of the created item.

**Return type** `proknow.Roles.RoleItem`

**Raises**

- `AssertionError` – If the input parameters are invalid.
- `proknow.Exceptions.HttpError` – If the HTTP request generated an error.

**Example**

This example creates a new role called “Researchers” with full permissions on the “Research” workspace:

```
from proknow import ProKnow

pk = ProKnow('https://example.proknow.com', credentials_file="./credentials.
↪json")
pk.roles.create("Researchers", {
    "create_api_keys": False,
    "manage_access": False,
    "manage_custom_metrics": False,
    "manage_template_metric_sets": False,
    "manage_renaming_rules": False,
    "manage_template_checklists": False,
    "organization_read": False,
    "organization_view_phi": False,
    "organization_download_dicom": False,
    "organization_write_collections": False,
    "organization_write_patients": False,
    "organization_contour_patients": False,
    "organization_delete_collections": False,
    "organization_delete_patients": False,
    "workspaces": [{
        "id": pk.workspaces.find(name="Research").id,
        "read": True,
        "view_phi": True,
        "download_dicom": True,
        "write_collections": True,
        "write_patients": True,
        "contour_patients": True,
        "delete_collections": True,
        "delete_patients": True,
    }],
})
```

**delete (role\_id)**

Deletes a role.

**Parameters** `role_id (str)` – The id of the role to delete.

**Raises**

- `AssertionError` – If the input parameters are invalid.
- `proknow.Exceptions.HttpError` – If the HTTP request generated an error.

**Example**

If you know the role id, you can delete the role directly using this method:

```
from proknow import ProKnow

pk = ProKnow('https://example.proknow.com', credentials_file="./credentials.
↪json")
pk.roles.delete('5c463a6c040068100c7f665acad17ac4')
```

**find** (*predicate=None, \*\*props*)

Finds the first role that matches the input parameters.

**Parameters**

- **predicate** (*func*) – A function that is passed a role as input and which should return a bool indicating whether the role is a match.
- **\*\*props** – A dictionary of keyword arguments that may include any role attribute. These arguments are considered in turn to find matching roles.

**Returns** A representation of the matching role.

**Return type** *proknow.Roles.RoleItem*

**Raises** *proknow.Exceptions.HttpError* – If the HTTP request generated an error.

**get** (*role\_id*)

Gets a role.

**Parameters** **role\_id** (*str*) – The id of the role to get.

**Returns** an object representing a role in the organization

**Return type** *proknow.Roles.RoleItem*

**Raises** *proknow.Exceptions.HttpError* – If the HTTP request generated an error.

## Example

If you know the role id, you can get the role directly using this method:

```
from proknow import ProKnow

pk = ProKnow('https://example.proknow.com', credentials_file="./credentials.
↪json")
researchers = pk.roles.get('5c463a6c040068100c7f665acad17ac4')
```

**query** ()

Queries for roles.

**Returns** A list of *proknow.Roles.RoleSummary* objects, each representing a summarized role in the organization.

**Return type** list

**Raises** *proknow.Exceptions.HttpError* – If the HTTP request generated an error.

## Example

This example queries the roles and prints the name of each role:

```
from proknow import ProKnow

pk = ProKnow('https://example.proknow.com', credentials_file="./credentials.
↪json")
for role in pk.roles.query():
    print(role.name)
```

**class** proknow.Roles.**RoleSummary**(roles, role)

This class represents a summary view of a role. It's instantiated by the `proknow.Roles.Roles.query()` method to represent each of the roles returned in a query result.

**id**

*str* – The id of the role (readonly).

**name**

*str* – The name of the role (readonly).

**data**

*dict* – The summary representation of the role as returned from the API (readonly).

**\_\_init\_\_**(roles, role)

Initializes the RoleSummary class.

**Parameters**

- **roles** (proknow.Roles.Roles) – The Roles instance that is instantiating the object.
- **role** (*dict*) – A dictionary of role attributes.

**get()**

Gets the complete representation of the role.

**Returns** an object representing a role in the organization

**Return type** `proknow.Roles.RoleItem`

**Raises** `proknow.Exceptions.HttpError` – If the HTTP request generated an error.

## Example

The following example shows how to turn a list of RoleSummary objects into a list of RoleItem objects:

```
from proknow import ProKnow

pk = ProKnow('https://example.proknow.com', credentials_file="./credentials.
↪json")
roles = [role.get() for role in pk.roles.query()]
```

**class** proknow.Roles.**RoleItem**(roles, role)

This class represents a role. It's instantiated by the `proknow.Roles.Roles` class as a complete representation of a role.

**id**

*str* – The id of the role (readonly).

**data**

*dict* – The complete representation of the role as returned from the API (readonly).

**name**

*str* – The name of the role.



**permissions***dict* – The dictionary of role permissions.**\_\_init\_\_** (*roles, role*)

Initializes the RoleItem class.

**Parameters**

- **roles** (`proknow.Roles.Roles`) – The Role instance that is instantiating the object.
- **role** (*dict*) – A dictionary of role attributes.

**delete** ()

Deletes the role.

**Raises** `proknow.Exceptions.HttpError` – If the HTTP request generated an error.**Example**

The following example shows how to find a role by its name and delete it:

```
from proknow import ProKnow

pk = ProKnow('https://example.proknow.com', credentials_file="./credentials.
↪ json")
researchers = pk.roles.find(name='researchers').get()
researchers.delete()
```

**save** ()

Saves the changes made to a role.

**Raises** `proknow.Exceptions.HttpError` – If the HTTP request generated an error.**Example**

The following example shows how to find a role by its name, modify a permission, and save it:

```
from proknow import ProKnow

pk = ProKnow('https://example.proknow.com', credentials_file="./credentials.
↪ json")
researchers = pk.roles.find(name='researchers').get()
researchers.permissions["organization_read"] = True
researchers.save()
```

## 2.7 proknow.Users – Users

**class** `proknow.Users.Users` (*proknow, requestor*)This class should be used to interact with the users in a ProKnow organization. It is instantiated for you as an attribute of the `proknow.ProKnow.ProKnow` class.**\_\_init\_\_** (*proknow, requestor*)

Initializes the Users class.

**Parameters**

- **proknow** (`proknow.ProKnow.ProKnow`) – The ProKnow instance that is instantiating the object.
- **requestor** (`proknow.Requestor.Requestor`) – An object used to make API requests.

**create** (*email, name, role\_id, password=None*)

Creates a new user.

**Parameters**

- **email** (*str*) – The email of the user.
- **name** (*str*) – The name of the user.
- **role\_id** (*str*) – The id of the role for the user.
- **password** (*str, optional*) – The password of the user.

**Returns** A representation of the created item.

**Return type** `proknow.Users.UserItem`

**Raises**

- `AssertionError` – If the input parameters are invalid.
- `proknow.Exceptions.HttpError` – If the HTTP request generated an error.

## Example

This example creates a new user under the Admin role:

```
from proknow import ProKnow

pk = ProKnow('https://example.proknow.com', credentials_file="./credentials.
↪ json")
role_id = pk.roles.find(name="Admin").id
pk.users.create("jsmith@example.com", "John Smith", role_id)
```

**delete** (*user\_id*)

Deletes a user.

**Parameters** **user\_id** (*str*) – The id of the user to delete.

**Raises**

- `AssertionError` – If the input parameters are invalid.
- `proknow.Exceptions.HttpError` – If the HTTP request generated an error.

## Example

If you know the user id, you can delete the user directly using this method:

```
from proknow import ProKnow

pk = ProKnow('https://example.proknow.com', credentials_file="./credentials.
↪ json")
pk.users.delete('5c463a6c04005a992cc16b29f9a7637b')
```

**find** (*predicate=None*, *\*\*props*)

Finds a user by id, email, or name.

**Parameters**

- **predicate** (*func*) – A function that is passed a user as input and which should return a bool indicating whether the user is a match.
- **\*\*props** – A dictionary of keyword arguments that may include any user attribute. These arguments are considered in turn to find matching users.

**Returns** A representation of the matching user.

**Return type** *proknow.Users.UserItem*

**Raises** *proknow.Exceptions.HttpError* – If the HTTP request generated an error.

**get** (*user\_id*)

Gets a user.

**Parameters** **user\_id** (*str*) – The id of the user to get.

**Returns** an object representing a user in the organization

**Return type** *proknow.Users.UserItem*

**Raises** *proknow.Exceptions.HttpError* – If the HTTP request generated an error.

## Example

If you know the user id, you can get the user directly using this method:

```
from proknow import ProKnow

pk = ProKnow('https://example.proknow.com', credentials_file="./credentials.
↪json")
admin = pk.users.get('5c463a6c04005a992cc16b29f9a7637b')
```

**query** ()

Queries for users.

**Returns** A list of *proknow.Users.UserSummary* objects, each representing a summarized user in the organization.

**Return type** list

**Raises** *proknow.Exceptions.HttpError* – If the HTTP request generated an error.

## Example

This example queries the users and prints the name of each user:

```
from proknow import ProKnow

pk = ProKnow('https://example.proknow.com', credentials_file="./credentials.
↪json")
for user in pk.users.query():
    print(user.name)
```

**class** `proknow.Users.UserSummary` (*users, user*)

This class represents a summary view of a user. It's instantiated by the `proknow.Users.Users.query()` method to represent each of the users returned in a query result.

**id**

*str* – The id of the user (readonly).

**email**

*str* – The email of the user (readonly).

**name**

*str* – The name of the user (readonly).

**data**

*dict* – The summary representation of the user as returned from the API (readonly).

**\_\_init\_\_** (*users, user*)

Initializes the UserSummary class.

**Parameters**

- **users** (`proknow.Users.Users`) – The Users instance that is instantiating the object.
- **user** (*dict*) – A dictionary of user attributes.

**get** ()

Gets the complete representation of the user.

**Returns** an object representing a user in the organization

**Return type** `proknow.Users.UserItem`

**Raises** `proknow.Exceptions.HttpError` – If the HTTP request generated an error.

## Example

The following example shows how to turn a list of UserSummary objects into a list of UserItem objects:

```
from proknow import ProKnow

pk = ProKnow('https://example.proknow.com', credentials_file="./credentials.
↪ json")
users = [user.get() for user in pk.users.query()]
```

**class** `proknow.Users.UserItem` (*users, user*)

This class represents a user. It's instantiated by the `proknow.Users.Users` class as a complete representation of a user.

**id**

*str* – The id of the user (readonly).

**data**

*dict* – The complete representation of the user as returned from the API (readonly).

**name**

*str* – The name of the user.

**email**

*str* – The email of the user.

**active**

*bool* – Indicates whether the user is active.

**role\_id***str* – The id of the role for the user.**\_\_init\_\_** (*users, user*)

Initializes the UserItem class.

**Parameters**

- **users** (`proknow.Users.Users`) – The User instance that is instantiating the object.
- **user** (*dict*) – A dictionary of user attributes.

**delete** ()

Deletes the user.

**Raises** `proknow.Exceptions.HttpError` – If the HTTP request generated an error.**Example**

The following example shows how to find a user by its email and delete it:

```
from proknow import ProKnow

pk = ProKnow('https://example.proknow.com', credentials_file='./credentials.
↪json')
jsmith = pk.users.find(email='jsmith@example.com')
jsmith.delete()
```

**save** ()

Saves the changes made to a user.

**Raises** `proknow.Exceptions.HttpError` – If the HTTP request generated an error.**Example**

The following example shows how to find a user by its email, set it to inactive, and save it:

```
from proknow import ProKnow

pk = ProKnow('https://example.proknow.com', credentials_file='./credentials.
↪json')
jsmith = pk.users.find(email='jsmith@example.com')
jsmith.active = False
jsmith.save()
```

## 2.8 proknow.Uploads – Uploads

**class** `proknow.Uploads.Uploads` (*proknow, requestor*)This class should be used to interact with the uploads in a ProKnow organization. It is instantiated for you as an attribute of the `proknow.ProKnow.ProKnow` class.**\_\_init\_\_** (*proknow, requestor*)

Initializes the Uploads class.

**Parameters**

- **proknow** (`proknow.ProKnow.ProKnow`) – The ProKnow instance that is instantiating the object.
- **requestor** (`proknow.Requestor.Requestor`) – An object used to make API requests.

**upload** (*workspace, path\_or\_paths*)

Initiates an upload or series of uploads to the API.

#### Parameters

- **workspace** (*str*) – An id or name of the workspace in which to create the uploads.
- **path\_or\_paths** (*str or list*) – A path or list of paths such that each path is a directory of files to upload or a path to a file to upload.

**Returns** An object used to manage a batch of uploads.

**Return type** `proknow.Uploads.UploadBatch`

#### Raises

- `AssertionError` – If the input parameters are invalid.
- `proknow.Exceptions.HttpError` – If the HTTP request generated an error.
- `proknow.Exceptions.InvalidPathError` – If the provided file path is invalid.
- `proknow.Exceptions.WorkspaceLookupError` – If the workspace with the given name or id could not be found.
- `proknow.Exceptions.TimeoutExceededError` – If the timeout was exceeded while waiting for the uploads to complete.

## Example

This example shows how to upload a directory of files:

```
from proknow import ProKnow
pk = ProKnow('https://example.proknow.com', credentials_file="./credentials.
↪ json")
pk.uploads.upload("Upload Test", "./DICOM")
```

You may also provide the path to each file:

```
from proknow import ProKnow
pk = ProKnow('https://example.proknow.com', credentials_file="./credentials.
↪ json")
pk.uploads.upload("Upload Test", [
    "./DICOM/img000001.dcm",
    "./DICOM/img000002.dcm",
    "./DICOM/img000003.dcm",
    "./DICOM/img000004.dcm",
    "./DICOM/img000005.dcm",
])
```

Lists containing both directories and file paths are also permitted:

```
from proknow import ProKnow
pk = ProKnow('https://example.proknow.com', credentials_file="./credentials.
↪ json")
```

(continues on next page)

(continued from previous page)

```
pk.uploads.upload("Upload Test", [
    "./DICOM/CT/",
    "./DICOM/structures.dcm",
    "./DICOM/plan.dcm",
    "./DICOM/dose.dcm",
])
```

**class** `proknow.Uploads.UploadBatch(uploads, workspace_id, files)`

This class is instantiated by the `proknow.Uploads.Uploads.upload()` method and is used as an interface for looking up patients and entities within a batch of resolved uploads.

**\_\_init\_\_** (*uploads, workspace\_id, files*)

Initializes the UploadBatch class.

#### Parameters

- **uploads** (`proknow.Uploads.Uploads`) – The Uploads instance that is instantiating the object.
- **workspace\_id** (*str*) – The id of the workspace to which the uploads belong.
- **files** (*list*) – A list of files.

**find\_entity** (*path*)

Find the entity associated with the given file.

**Parameters** **path** (*str*) – An absolute file path.

#### Raises

- `AssertionError` – If the input parameters are invalid.
- `proknow.Exceptions.InvalidPathError` – If the provided file path is invalid.

## Example

This example shows how to find an entity associated with a given file upload:

```
from proknow import ProKnow
import os

pk = ProKnow('https://example.proknow.com', credentials_file="./credentials.
↪json")
batch = pk.uploads.upload("Upload Test", "./DICOM")
path = os.path.abspath("./DICOM/plan.dcm")
entity_summary = batch.find_entity(path)
```

**find\_patient** (*path*)

Find the patient associated with the given file.

**Parameters** **path** (*str*) – An absolute file path.

#### Raises

- `AssertionError` – If the input parameters are invalid.
- `proknow.Exceptions.InvalidPathError` – If the provided file path is invalid.

### Example

This example shows how to find a patient associated with a given file upload:

```
from proknow import ProKnow
import os

pk = ProKnow('https://example.proknow.com', credentials_file="./credentials.
↪json")
batch = pk.uploads.upload("Upload Test", "./DICOM")
path = os.path.abspath("./DICOM/plan.dcm")
patient_summary = batch.find_patient(path)
```

**class** `proknow.Uploads.UploadPatientSummary` (*uploads, workspace\_id, patient*)

This class provides a summary view of a patient as returned as part of an upload.

**id**

*str* – The id of the patient (readonly).

**data**

*dict* – The summary representation of the patient as returned from the API for an upload (readonly).

**entities**

*list* – A list of `proknow.Uploads.UploadEntitySummary` items.

**\_\_init\_\_** (*uploads, workspace\_id, patient*)

Initializes the UploadPatientSummary class.

#### Parameters

- **uploads** (`proknow.Uploads.Uploads`) – The Uploads instance that is instantiating the object.
- **workspace\_id** (*str*) – The id of the workspace to which the patient belongs.
- **patient** (*dict*) – Patient summary information.

**add\_entity** (*entity*)

Adds an entity to patient summary.

**Parameters** **entity** (`proknow.Uploads.UploadEntitySummary`) – An entity summary to add to the patient summary.

**get** ()

Gets the complete representation of the patient.

**Returns** an object representing a patient in the organization.

**Return type** `proknow.Patients.PatientItem`

**Raises** `proknow.Exceptions.HttpError` – If the HTTP request generated an error.

### Example

This example shows how to get a list of patient items associated with a batch of uploads:

```
from proknow import ProKnow

pk = ProKnow('https://example.proknow.com', credentials_file="./credentials.
↪json")
```

(continues on next page)



(continued from previous page)

```
batch = pk.uploads.upload("Upload Test", "./DICOM")
patients = [patient.get() for patient in batch.patients]
```

**class** `proknow.Uploads.UploadEntitySummary` (*uploads, workspace\_id, patient\_id, entity*)

This class provides a summary view of a entity as returned as part of an upload.

**id**

*str* – The id of the entity (readonly).

**data**

*dict* – The summary representation of the entity as returned from the API for an upload (readonly).

**\_\_init\_\_** (*uploads, workspace\_id, patient\_id, entity*)

Initializes the UploadEntitySummary class.

#### Parameters

- **uploads** (`proknow.Uploads.Uploads`) – The Uploads instance that is instantiating the object.
- **workspace\_id** (*str*) – The id of the workspace to which the entity belongs.
- **patient\_id** (*str*) – The id of the patient to which the entity belongs
- **entity** (*dict*) – entity summary information.

**get** ()

Gets the complete representation of the entity.

**Returns** One of (`proknow.Patients.ImageSetItem`, `proknow.Patients.StructureSetItem`, `proknow.Patients.PlanItem`, `proknow.Patients.DoseItem`) based on the entity summary type.

**Raises** `proknow.Exceptions.HttpError` – If the HTTP request generated an error.

### Example

This example shows how to get a list of entity items associated with a batch of uploads:

```
from proknow import ProKnow

pk = ProKnow('https://example.proknow.com', credentials_file="./credentials.
↪ json")
batch = pk.uploads.upload("Upload Test", "./DICOM")
entities = [entity.get() for patient in batch.patients for entity in patient.
↪ entities]
```

## 2.9 proknow.Patients – Patients

**class** `proknow.Patients.Patients` (*proknow, requestor*)

This class should be used to interact with the patients in a ProKnow organization. It is instantiated for you as an attribute of the `proknow.ProKnow.ProKnow` class.

**\_\_init\_\_** (*proknow, requestor*)

Initializes the Patients class.

#### Parameters

- **proknow** (`proknow.ProKnow.ProKnow`) – The ProKnow instance that is instantiating the object.
- **requestor** (`proknow.Requestor.Requestor`) – An object used to make API requests.

**create** (*workspace, mrn, name, birth\_date=None, birth\_time=None, sex=None*)

Creates a new patient.

#### Parameters

- **workspace** (*str*) – An id or name of the workspace in which to create the patient.
- **mrn** (*str*) – The MRN of the patient.
- **name** (*str*) – The name of the patient.
- **birth\_date** (*str, optional*) – The birth date of the patient.
- **birth\_time** (*str, optional*) – The birth time of the patient.
- **sex** (*str, optional*) – The sex of the patient.

**Returns** A representation of the created patient.

**Return type** `proknow.Patients.PatientItem`

#### Raises

- `AssertionError` – If the input parameters are invalid.
- `proknow.Exceptions.HttpError` – If the HTTP request generated an error.
- `proknow.Exceptions.WorkspaceLookupError` – If the workspace with the given name or id could not be found.

### Example

This example shows how to create a patient in the workspace called “Clinical”:

```
from proknow import ProKnow

pk = ProKnow('https://example.proknow.com', credentials_file="./credentials.
↪json")
patient = pk.patients.create("Clinical", "12345", "Becker^Matthew")
```

**delete** (*workspace\_id, patient\_id*)

Deletes a patient.

#### Parameters

- **workspace\_id** (*str*) – The id of the workspace to which the patient belongs.
- **patient\_id** (*str*) – The id of the patient to delete.

#### Raises

- `AssertionError` – If the input parameters are invalid.
- `proknow.Exceptions.HttpError` – If the HTTP request generated an error.

## Example

If you know the workspace id and patient id, you can delete a patient directly using this method:

```
from proknow import ProKnow

pk = ProKnow('https://example.proknow.com', credentials_file='./credentials.
↪ json')
pk.patients.delete('5c463a6c040040f1efda74db75c1b121',
↪ '5c4b4c52a5c058c3d1d98ac194d0200f')
```

**find** (*workspace*, *predicate=None*, *\*\*props*)

Finds the first patient that matches the input paramters.

### Parameters

- **workspace** (*str*) – An id or name of the workspace in which to query for patients.
- **predicate** (*func*) – A function that is passed a metric as input and which should return a bool indicating whether the metric is a match.
- **\*\*props** – A dictionary of keyword arguments that may include any patient attribute. These arguments are considered in turn to find matching patients.

**Returns** A representation of the matching patient.

**Return type** *proknow.Patients.PatientSummary*

### Raises

- *proknow.Exceptions.HttpError* – If the HTTP request generated an error.
- *proknow.Exceptions.WorkspaceLookupError* – If the workspace with the given name or id could not be found.

**get** (*workspace\_id*, *patient\_id*)

Gets a patient from the given workspace.

### Parameters

- **workspace\_id** (*str*) – The id of the workspace to which the patient belongs.
- **patient\_id** (*str*) – The id of the patient to get.

**Returns** an object representing a patient in the organization

**Return type** *proknow.Patients.PatientItem*

### Raises

- *AssertionError* – If the input parameters are invalid.
- *proknow.Exceptions.HttpError* – If the HTTP request generated an error.

## Example

If you know the workspace id and patient id, you can get the patient directly using this method:

```
from proknow import ProKnow
```

(continues on next page)

(continued from previous page)

```
pk = ProKnow('https://example.proknow.com', credentials_file="./credentials.
↪json")
pk.patients.get('5c463a6c040040f1efda74db75c1b121',
↪'5c4b4c52a5c058c3d1d98ac194d0200f')
```

**lookup** (*workspace*, *mrns*)

Looks up a collection of patients matching the given list of MRNs.

**Parameters**

- **workspace** (*str*) – An id or name of the workspace in which to query for patients.
- **mrns** (*list*) – A list of MRN string values.

**Returns** A list of *proknow.Patients.PatientSummary* objects that match the given MRNs.

**Return type** list

**Raises**

- *proknow.Exceptions.HttpError* – If the HTTP request generated an error.
- *proknow.Exceptions.WorkspaceLookupError* – If the workspace with the given name or id could not be found.

**Example**

Use this method to resolve a list of patient MRNs. Just provide the ID as a list in the second argument:

```
from proknow import ProKnow

pk = ProKnow('https://example.proknow.com', credentials_file="./credentials.
↪json")
pk.patients.lookup('Clinical', ['HNC-0522c0009', 'HNC-0522c0013'])
```

**query** (*workspace*)

Queries for patients.

**Parameters** **workspace** (*str*) – An id or name of the workspace in which to query for patients.

**Returns** A list of *proknow.Patients.PatientSummary* objects, each representing a summarized patient in the given workspace.

**Return type** list

**Raises**

- *AssertionError* – If the input parameters are invalid.
- *proknow.Exceptions.HttpError* – If the HTTP request generated an error.

**Example**

This example queries the patients belonging to the Clinical workspace and prints the name of each patient:

```

from proknow import ProKnow

pk = ProKnow('https://example.proknow.com', credentials_file="./credentials.
↪json")
for patient in pk.patients.query("Clinical"):
    print(patient.name)

```

**class** `proknow.Patients.PatientSummary` (*patients, workspace\_id, patient*)

This class represents a summary view of a patient. It's instantiated by the `proknow.Patients.Patients.query()` method to represent each of the patients returned in a query result.

**id**

*str* – The id of the patient (readonly).

**mrn**

*str* – The patient medical record number or MRN (readonly). In the Proknow interface, this is referred to as the Patient ID.

**name**

*str* – The name of the patient (readonly).

**birth\_date**

*str* – The birth\_date of the patient (readonly).

**birth\_time**

*str* – The birth\_time of the patient (readonly).

**sex**

*str* – The sex of the patient (readonly).

**data**

*dict* – The summary representation of the patient as returned from the API (readonly).

**\_\_init\_\_** (*patients, workspace\_id, patient*)

Initializes the PatientSummary class.

#### Parameters

- **patients** (`proknow.Patients.Patients`) – The Patients instance that is instantiating the object.
- **workspace\_id** (*str*) – The id of the workspace to which the patient belongs.
- **patient** (*dict*) – A dictionary of patient attributes.

**get** ()

Gets the complete representation of the patient.

**Returns** an object representing a patient in the organization.

**Return type** `proknow.Patients.PatientItem`

**Raises** `proknow.Exceptions.HttpError` – If the HTTP request generated an error.

## Example

The following example shows how to turn a list of PatientSummary objects into a list of PatientItem objects:

```
from proknow import ProKnow

pk = ProKnow('https://example.proknow.com', credentials_file="./credentials.
↪json")
patients = [patient.get() for patient in pk.patients.query("Clinical")]
```

**class** proknow.Patients.PatientItem(patients, workspace\_id, patient)

This class represents a patient. It's instantiated by the `proknow.Patients.Patients` class as a complete representation of a patient.

**id**

*str* – The id of the patient (readonly).

**data**

*dict* – The complete representation of the patient as returned from the API (readonly).

**mrn**

*str* – The MRN of the patient.

**name**

*str* – The name of the patient.

**birth\_date**

*str* – The birth\_date of the patient.

**birth\_time**

*str* – The birth\_time of the patient.

**sex**

*str* – The sex of the patient.

**metadata**

*dict* – The metadata of the patient.

**\_\_init\_\_**(patients, workspace\_id, patient)

Initializes the PatientItem class.

#### Parameters

- **patients** (`proknow.Patients.Patients`) – The Patients instance that is instantiating the object.
- **workspace\_id** (*str*) – The id of the workspace to which the patient belongs.
- **patient** (*dict*) – A dictionary of patient attributes.

**delete()**

Deletes the patient.

**Raises** `proknow.Exceptions.HttpError` – If the HTTP request generated an error.

### Example

The following example shows how to find a patient by its MRN and delete it:

```
from proknow import ProKnow

pk = ProKnow('https://example.proknow.com', credentials_file="./credentials.
↪json")
patients = pk.patients.lookup("Clinical", ["HNC-0522c0009"])
```

(continues on next page)

(continued from previous page)

```
patient = patients[0].get()
patient.delete()
```

**find\_entities** (*predicate=None, \*\*props*)

Finds the entities for the patient matching the input paramters.

**Parameters**

- **predicate** (*func*) – A function that is passed a entity as input and which should return a bool indicating whether the entity is a match.
- **\*\*props** – A dictionary of keyword arguments that may include any entity attribute to match. These arguments are considered in turn to find matching entities.

**Returns** A list of matching *proknow.Patients.EntitySummary* objects.

**Return type** list

**Example**

Use this example to find the patient entities matching the predicate function (returns all plan and dose entities):

```
from proknow import ProKnow

pk = ProKnow('https://example.proknow.com', credentials_file="./credentials.
↪json")
patients = pk.patients.lookup("Clinical", ["HNC-0522c0009"])
patient = patients[0].get()
entities = patient.find_entities(lambda entity: entity.data["type"] == "dose"
↪or entity.data["type"] == "plan")
```

Use this example to find the patient entities matching the predicate function (returns entities with the modality value of “MR”):

```
from proknow import ProKnow

pk = ProKnow('https://example.proknow.com', credentials_file="./credentials.
↪json")
patients = pk.patients.lookup("Clinical", ["HNC-0522c0009"])
patient = patients[0].get()
entities = patient.find_entities(modality="MR")
```

**get\_metadata** ()

Gets the metadata dictionary and decodes the ids into metrics names.

**Returns** The dictionary of custom metric key-value pairs where the keys are the decoded custom metric names.

**Return type** dict

**Raises** *proknow.Exceptions.CustomMetricLookupError* – If a custom metric could not be resolved.

**Example**

Use this example to print the metadata values for a patient:

```
from proknow import ProKnow

pk = ProKnow('https://example.proknow.com', credentials_file="./credentials.
↪json")
patients = pk.patients.lookup("Clinical", ["HNC-0522c0009"])
patient = patients[0].get()
for key, value in patient.get_metadata():
    print(key + ": " + value)
```

**save()**

Saves the changes made to a patient.

**Raises** *proknow.Exceptions.HttpError* – If the HTTP request generated an error.

## Example

The following example shows how to find a patient by its MRN, update the name, and save it:

```
from proknow import ProKnow

pk = ProKnow('https://example.proknow.com', credentials_file="./credentials.
↪json")
patients = pk.patients.lookup("Clinical", ["HNC-0522c0009"])
patient = patients[0].get()
patient.name = "ANON-1234"
patient.save()
```

**set\_metadata(metadata)**

Sets the metadata dictionary to an encoded version of the given metadata dictionary.

**Params:** metadata (dict): A dictionary of custom metric key-value pairs where the keys are the names of the custom metric.

**Raises** *proknow.Exceptions.CustomMetricLookupError* – If a custom metric could not be resolved.

## Example

Use this example to set the metadata value for “Genetic Type” for a patient before saving:

```
from proknow import ProKnow

pk = ProKnow('https://example.proknow.com', credentials_file="./credentials.
↪json")
patients = pk.patients.lookup("Clinical", ["HNC-0522c0009"])
patient = patients[0].get()
meta = patient.get_metadata()
meta["Genetic Type"] = "Type II"
patient.set_metadata(meta)
patient.save()
```

**class proknow.Patients.StudySummary(patients, workspace\_id, patient\_id, study)**

This class represents a summary view of a study. It's instantiated by the *proknow.Patients.PatientItem()* class to represent each of the studies that belong to the patient.



**id***str* – The id of the study (readonly).**data***dict* – The summary representation of the study as returned from the API (readonly).**\_\_init\_\_** (*patients, workspace\_id, patient\_id, study*)

Initializes the PatientItem class.

**Parameters**

- **patients** (*proknow.Patients.Patients*) – The Patients instance that is instantiating the object.
- **workspace\_id** (*str*) – The id of the workspace to which the patient belongs.
- **patient\_id** (*str*) – The id of the patient to which the study belongs.
- **study** (*dict*) – A dictionary of study attributes.

**class** *proknow.Patients.EntitySummary* (*patients, workspace\_id, patient\_id, entity*)

This class represents a summary view of an entity. It's instantiated by the *proknow.Patients.StudySummary()* class to represent each of the entities that belong to the study.

**id***str* – The id of the entity (readonly).**data***dict* – The summary representation of the entity as returned from the API (readonly).**\_\_init\_\_** (*patients, workspace\_id, patient\_id, entity*)

Initializes the PatientItem class.

**Parameters**

- **patients** (*proknow.Patients.Patients*) – The Patients instance that is instantiating the object.
- **workspace\_id** (*str*) – The id of the workspace to which the patient belongs.
- **patient\_id** (*str*) – The id of the patient to which the entity belongs.
- **entity** (*dict*) – A dictionary of entity attributes.

**get()**

Gets the entity item for the entity summary type.

**Returns** One of (*proknow.Patients.ImageSetItem*, *proknow.Patients.StructureSetItem*, *proknow.Patients.PlanItem*, *proknow.Patients.DoseItem*) based on the entity summary type.

**Raises** *proknow.Exceptions.HttpError* – If the HTTP request generated an error.

**Example**

Use this example to find all of the patient's entities and construct a list of the type-specific object types:

```
from proknow import ProKnow

pk = ProKnow('https://example.proknow.com', credentials_file="./credentials.
↪json")
patients = pk.patients.lookup("Clinical", ["HNC-0522c0009"])
```

(continues on next page)

(continued from previous page)

```
patient = patients[0].get()
entities = [entity.get() for entity in patient.find_entities(lambda entity:
↪ True)]
```

**class** `proknow.Patients.EntityItem` (*patients, workspace\_id, patient\_id, entity*)

This class is a base class for specific entity types.

**id**

*str* – The id of the entity (readonly).

**data**

*dict* – The summary representation of the entity as returned from the API (readonly).

**\_\_init\_\_** (*patients, workspace\_id, patient\_id, entity*)

Initializes the PatientItem class.

**Parameters**

- **patients** (`proknow.Patients.Patients`) – The Patients instance that is instantiating the object.
- **workspace\_id** (*str*) – The id of the workspace to which the patient belongs.
- **patient\_id** (*str*) – The id of the patient to which the entity belongs.
- **entity** (*dict*) – A dictionary of entity attributes.

**class** `proknow.Patients.ImageSetItem` (*patients, workspace\_id, patient\_id, entity*)

This class represents a an image set. It's instantiated by the `proknow.Patients.EntitySummary` class as a complete representation of an image set entity.

**id**

*str* – The id of the entity (readonly).

**data**

*dict* – The summary representation of the entity as returned from the API (readonly).

**\_\_init\_\_** (*patients, workspace\_id, patient\_id, entity*)

Initializes the ImageSetItem class.

**Parameters**

- **patients** (`proknow.Patients.Patients`) – The Patients instance that is instantiating the object.
- **workspace\_id** (*str*) – The id of the workspace to which the patient belongs.
- **patient\_id** (*str*) – The id of the patient to which the entity belongs.
- **entity** (*dict*) – A dictionary of entity attributes.

**download** (*path*)

Download the image set as a directory of images.

**Parameters** **path** (*str*) – A path to a directory in which a directory of images should be downloaded.

**Returns** The absolute path to the downloaded image set directory.

**Return type** *str*

**Raises**

- `AssertionError` – If the input parameters are invalid.

- `proknow.Exceptions.HttpError` – If the HTTP request generated an error.
- `proknow.Exceptions.InvalidPathError` – If the provided path is invalid.

### Example

This example shows how to download an image set into the current directory:

```
from proknow import ProKnow

pk = ProKnow('https://example.proknow.com', credentials_file="./credentials.
↪json")
patients = pk.patients.lookup("Clinical", ["HNC-0522c0009"])
patient = patients[0].get()
entities = patient.find_entities(type="image_set")
image_set = entities[0].get()
image_set.download("./")
```

**class** `proknow.Patients.StructureSetItem` (*patients, workspace\_id, patient\_id, entity*)

This class represents a structure set. It's instantiated by the `proknow.Patients.EntitySummary` class as a complete representation of a structure set entity.

**id**

*str* – The id of the entity (readonly).

**data**

*dict* – The summary representation of the entity as returned from the API (readonly).

**\_\_init\_\_** (*patients, workspace\_id, patient\_id, entity*)

Initializes the StructureSetItem class.

#### Parameters

- **patients** (`proknow.Patients.Patients`) – The Patients instance that is instantiating the object.
- **workspace\_id** (*str*) – The id of the workspace to which the patient belongs.
- **patient\_id** (*str*) – The id of the patient to which the entity belongs.
- **entity** (*dict*) – A dictionary of entity attributes.

**download** (*path*)

Download the currently approved structure set file.

#### Parameters

- **path** (*str*) – A path to a directory or file to which the structure set file should be
- **streamed**. –

**Returns** The absolute path to the downloaded file.

**Return type** *str*

#### Raises

- `AssertionError` – If the input parameters are invalid.
- `proknow.Exceptions.HttpError` – If the HTTP request generated an error.
- `proknow.Exceptions.InvalidPathError` – If the provided path is invalid.

## Example

This example shows how to download a plan file for a patient to the current directory:

```
from proknow import ProKnow

pk = ProKnow('https://example.proknow.com', credentials_file="./credentials.
↪json")
patients = pk.patients.lookup("Clinical", ["HNC-0522c0009"])
patient = patients[0].get()
entities = patient.find_entities(type="structure_set")
structure_set = entities[0].get()
structure_set.download("./")
```

**class** proknow.Patients.PlanItem(*patients, workspace\_id, patient\_id, entity*)

This class represents a plan. It's instantiated by the `proknow.Patients.EntitySummary` class as a complete representation of a plan entity.

**id**

*str* – The id of the entity (readonly).

**data**

*dict* – The summary representation of the entity as returned from the API (readonly).

**\_\_init\_\_**(*patients, workspace\_id, patient\_id, entity*)

Initializes the PlanItem class.

### Parameters

- **patients** (`proknow.Patients.Patients`) – The Patients instance that is instantiating the object.
- **workspace\_id** (*str*) – The id of the workspace to which the patient belongs.
- **patient\_id** (*str*) – The id of the patient to which the entity belongs.
- **entity** (*dict*) – A dictionary of entity attributes.

**download**(*path*)

Download the plan file.

**Parameters** **path** (*str*) – A path to a directory or file to which the plan file should be streamed.

**Returns** The absolute path to the downloaded file.

**Return type** *str*

### Raises

- `AssertionError` – If the input parameters are invalid.
- `proknow.Exceptions.HttpError` – If the HTTP request generated an error.
- `proknow.Exceptions.InvalidPathError` – If the provided path is invalid.

## Example

This example shows how to download a plan file for a patient to the current directory:

```

from proknow import ProKnow

pk = ProKnow('https://example.proknow.com', credentials_file="./credentials.
↪json")
patients = pk.patients.lookup("Clinical", ["HNC-0522c0009"])
patient = patients[0].get()
entities = patient.find_entities(type="plan")
plan = entities[0].get()
plan.download("./")

```

**class** proknow.Patients.DoseItem(*patients, workspace\_id, patient\_id, entity*)

This class represents a dose. It's instantiated by the `proknow.Patients.EntitySummary` class as a complete representation of a dose entity.

**id**

*str* – The id of the entity (readonly).

**data**

*dict* – The summary representation of the entity as returned from the API (readonly).

**\_\_init\_\_**(*patients, workspace\_id, patient\_id, entity*)

Initializes the DoseItem class.

#### Parameters

- **patients** (`proknow.Patients.Patients`) – The Patients instance that is instantiating the object.
- **workspace\_id** (*str*) – The id of the workspace to which the patient belongs.
- **patient\_id** (*str*) – The id of the patient to which the entity belongs.
- **entity** (*dict*) – A dictionary of entity attributes.

**download**(*path*)

Downloads the dose file.

**Parameters** **path** (*str*) – A path to a directory or file to which the dose file should be streamed.

**Returns** The absolute path to the downloaded file.

**Return type** *str*

**Raises**

- `AssertionError` – If the input parameters are invalid.
- `proknow.Exceptions.HttpError` – If the HTTP request generated an error.
- `proknow.Exceptions.InvalidPathError` – If the provided path is invalid.

## Example

This example shows how to download a dose file for a patient to the current directory:

```

from proknow import ProKnow

pk = ProKnow('https://example.proknow.com', credentials_file="./credentials.
↪json")
patients = pk.patients.lookup("Clinical", ["HNC-0522c0009"])
patient = patients[0].get()

```

(continues on next page)

(continued from previous page)

```
entities = patient.find_entities(type="dose")
dose = entities[0].get()
dose.download("./")
```

### 3.1 v0.3.0

#### Disclaimer

All releases in the v0.x.x series are subject to breaking changes from one version to another. After the release of v1.0.0, this project will be subject to [semantic versioning](#).

#### New Features & Enhancements

- Changed the following method names:
  - `resolveById` to `resolve_by_id`
  - `resolveByName` to `resolve_by_name`
- Improved documentation throughout
- Uploads module for initiating new uploads
- Implemented testing
- Implemented classes for `ImageSetItem` and `StructureSetItem` in the `Patients` module.

#### Bug Fixes

- Fixed bug in `Workspaces.resolve_by_id` method
- Fixed bug in `CustomMetricItem.save` method
- Fixed bug in `Patients.create` method

### 3.2 v0.2.0

#### Disclaimer

All releases in the v0.x.x series are subject to breaking changes from one version to another. After the release of v1.0.0, this project will be subject to [semantic versioning](#).

### New Features & Enhancements

- Added `resolve`, `resolveById`, and `resolveByName` methods to `Workspace` class
- Added `stream` method to `Requestor` class
- New `Exceptions` module for errors specific to the ProKnow DS - Python SDK.
- `CustomMetrics` module for interacting with organization custom metrics. This module is also used by the `Patients` module when getting and setting metadata.
- `Patients` module for interacting with patient data (including studies and entities).
- Change implementation of `.find` method throughout to use the signature `(self, predicate=None, **props)`.
- Renamed identifier arguments as `workspace_id`, `role_id`, `user_id`, etc.

## 3.3 v0.1.0

### New Features & Enhancements

- Base `ProKnow` module that may be initialized with API credentials and used to access ProKnow services
- `Requestor` helper module for constructing and issuing API requests
- Identity and Access Management (IAM) Modules
  - `Workspaces`
  - `Roles`
  - `Users`



This is an open source repository, and we appreciate your contributions. Use this guide to learn how you can be involved.

### 4.1 Requirements

1. You should discuss your change with a ProKnow developer before you begin. We know your time is valuable, and submitting a proposal of your change on our [GitHub issues page](#) is a great way to ensure that you will be working on something that we will likely merge once you submit a pull request.
2. You should follow the existing coding styles. We try to follow the guidelines presented in **PEP 8**. Should you find code that not comply with these guidelines, please bring it to our attention so that we may address it.
3. You should have a ProKnow DS account to run basic tests on your changes. If you do not have an account but still wish to contribute, please contact us at [support@proknow.com](mailto:support@proknow.com).

### 4.2 Accepted Contributions

1. New features.
2. Bug fixes.
3. Documentation.
4. Tests.

We are unlikely to accept pull requests that modify the license or change the way we publish and deploy. For example, we would not accept a change that separated the source and build directories because it would change the way we published the documentation. Tests are the exception to this because we do not yet have a testing strategy in place.



## CHAPTER 5

---

### Indices and tables

---

- `genindex`
- `modindex`
- `search`



### p

- `proknow.CustomMetrics`, 9
- `proknow.Exceptions`, 8
- `proknow.Patients`, 29
- `proknow.ProKnow`, 5
- `proknow.Requestor`, 6
- `proknow.Roles`, 17
- `proknow.Uploads`, 25
- `proknow.Users`, 21
- `proknow.Workspaces`, 13



## Symbols

- `__init__()` (proknow.CustomMetrics.CustomMetricItem method), 12
  - `__init__()` (proknow.CustomMetrics.CustomMetrics method), 9
  - `__init__()` (proknow.Exceptions.CustomMetricLookupError method), 9
  - `__init__()` (proknow.Exceptions.HttpError method), 8
  - `__init__()` (proknow.Exceptions.InvalidPathError method), 9
  - `__init__()` (proknow.Exceptions.TimeoutExceededError method), 9
  - `__init__()` (proknow.Exceptions.WorkspaceLookupError method), 8
  - `__init__()` (proknow.Patients.DoseItem method), 41
  - `__init__()` (proknow.Patients.EntityItem method), 38
  - `__init__()` (proknow.Patients.EntitySummary method), 37
  - `__init__()` (proknow.Patients.ImageSetItem method), 38
  - `__init__()` (proknow.Patients.PatientItem method), 34
  - `__init__()` (proknow.Patients.PatientSummary method), 33
  - `__init__()` (proknow.Patients.Patients method), 29
  - `__init__()` (proknow.Patients.PlanItem method), 40
  - `__init__()` (proknow.Patients.StructureSetItem method), 39
  - `__init__()` (proknow.Patients.StudySummary method), 37
  - `__init__()` (proknow.ProKnow.ProKnow method), 5
  - `__init__()` (proknow.Requestor.Requestor method), 6
  - `__init__()` (proknow.Roles.RoleItem method), 21
  - `__init__()` (proknow.Roles.RoleSummary method), 20
  - `__init__()` (proknow.Roles.Roles method), 17
  - `__init__()` (proknow.Uploads.UploadBatch method), 27
  - `__init__()` (proknow.Uploads.UploadEntitySummary method), 29
  - `__init__()` (proknow.Uploads.UploadPatientSummary method), 28
  - `__init__()` (proknow.Uploads.Uploads method), 25
  - `__init__()` (proknow.Users.UserItem method), 25
  - `__init__()` (proknow.Users.UserSummary method), 24
  - `__init__()` (proknow.Users.Users method), 21
  - `__init__()` (proknow.Workspaces.WorkspaceItem method), 16
  - `__init__()` (proknow.Workspaces.Workspaces method), 13
- ## A
- active (proknow.Users.UserItem attribute), 24
  - add\_entity() (proknow.Uploads.UploadPatientSummary method), 28
- ## B
- birth\_date (proknow.Patients.PatientItem attribute), 34
  - birth\_date (proknow.Patients.PatientSummary attribute), 33
  - birth\_time (proknow.Patients.PatientItem attribute), 34
  - birth\_time (proknow.Patients.PatientSummary attribute), 33
  - body (proknow.Exceptions.HttpError attribute), 8
- ## C
- context (proknow.CustomMetrics.CustomMetricItem attribute), 12
  - create() (proknow.CustomMetrics.CustomMetrics method), 9
  - create() (proknow.Patients.Patients method), 30
  - create() (proknow.Roles.Roles method), 17
  - create() (proknow.Users.Users method), 22
  - create() (proknow.Workspaces.Workspaces method), 13
  - custom\_metrics (proknow.ProKnow.ProKnow attribute), 5
  - CustomMetricItem (class in proknow.CustomMetrics), 12
  - CustomMetricLookupError, 8
  - CustomMetrics (class in proknow.CustomMetrics), 9
- ## D
- data (proknow.CustomMetrics.CustomMetricItem attribute), 12
  - data (proknow.Patients.DoseItem attribute), 41

data (proknow.Patients.EntityItem attribute), 38  
data (proknow.Patients.EntitySummary attribute), 37  
data (proknow.Patients.ImageSetItem attribute), 38  
data (proknow.Patients.PatientItem attribute), 34  
data (proknow.Patients.PatientSummary attribute), 33  
data (proknow.Patients.PlanItem attribute), 40  
data (proknow.Patients.StructureSetItem attribute), 39  
data (proknow.Patients.StudySummary attribute), 37  
data (proknow.Roles.RoleItem attribute), 20  
data (proknow.Roles.RoleSummary attribute), 20  
data (proknow.Uploads.UploadEntitySummary attribute), 29  
data (proknow.Uploads.UploadPatientSummary attribute), 28  
data (proknow.Users.UserItem attribute), 24  
data (proknow.Users.UserSummary attribute), 24  
data (proknow.Workspaces.WorkspaceItem attribute), 16  
delete() (proknow.CustomMetrics.CustomMetricItem method), 12  
delete() (proknow.CustomMetrics.CustomMetrics method), 10  
delete() (proknow.Patients.PatientItem method), 34  
delete() (proknow.Patients.Patients method), 30  
delete() (proknow.Requestor.Requestor method), 6  
delete() (proknow.Roles.RoleItem method), 21  
delete() (proknow.Roles.Roles method), 18  
delete() (proknow.Users.UserItem method), 25  
delete() (proknow.Users.Users method), 22  
delete() (proknow.Workspaces.WorkspaceItem method), 16  
delete() (proknow.Workspaces.Workspaces method), 14  
DoseItem (class in proknow.Patients), 41  
download() (proknow.Patients.DoseItem method), 41  
download() (proknow.Patients.ImageSetItem method), 38  
download() (proknow.Patients.PlanItem method), 40  
download() (proknow.Patients.StructureSetItem method), 39

## E

email (proknow.Users.UserItem attribute), 24  
email (proknow.Users.UserSummary attribute), 24  
entities (proknow.Uploads.UploadPatientSummary attribute), 28  
EntityItem (class in proknow.Patients), 38  
EntitySummary (class in proknow.Patients), 37

## F

find() (proknow.CustomMetrics.CustomMetrics method), 10  
find() (proknow.Patients.Patients method), 31  
find() (proknow.Roles.Roles method), 19  
find() (proknow.Users.Users method), 22  
find() (proknow.Workspaces.Workspaces method), 14  
find\_entities() (proknow.Patients.PatientItem method), 35

find\_entity() (proknow.Uploads.UploadBatch method), 27  
find\_patient() (proknow.Uploads.UploadBatch method), 27

## G

get() (proknow.Patients.EntitySummary method), 37  
get() (proknow.Patients.Patients method), 31  
get() (proknow.Patients.PatientSummary method), 33  
get() (proknow.Requestor.Requestor method), 6  
get() (proknow.Roles.Roles method), 19  
get() (proknow.Roles.RoleSummary method), 20  
get() (proknow.Uploads.UploadEntitySummary method), 29  
get() (proknow.Uploads.UploadPatientSummary method), 28  
get() (proknow.Users.Users method), 23  
get() (proknow.Users.UserSummary method), 24  
get\_metadata() (proknow.Patients.PatientItem method), 35

## H

HttpError, 8

## I

id (proknow.CustomMetrics.CustomMetricItem attribute), 12  
id (proknow.Patients.DoseItem attribute), 41  
id (proknow.Patients.EntityItem attribute), 38  
id (proknow.Patients.EntitySummary attribute), 37  
id (proknow.Patients.ImageSetItem attribute), 38  
id (proknow.Patients.PatientItem attribute), 34  
id (proknow.Patients.PatientSummary attribute), 33  
id (proknow.Patients.PlanItem attribute), 40  
id (proknow.Patients.StructureSetItem attribute), 39  
id (proknow.Patients.StudySummary attribute), 36  
id (proknow.Roles.RoleItem attribute), 20  
id (proknow.Roles.RoleSummary attribute), 20  
id (proknow.Uploads.UploadEntitySummary attribute), 29  
id (proknow.Uploads.UploadPatientSummary attribute), 28  
id (proknow.Users.UserItem attribute), 24  
id (proknow.Users.UserSummary attribute), 24  
id (proknow.Workspaces.WorkspaceItem attribute), 16  
ImageSetItem (class in proknow.Patients), 38  
InvalidPathError, 9

## L

lookup() (proknow.Patients.Patients method), 32

## M

message (proknow.Exceptions.CustomMetricLookupError attribute), 9



message (proknow.Exceptions.InvalidPathError attribute), 9

message (proknow.Exceptions.TimeoutExceededError attribute), 9

message (proknow.Exceptions.WorkspaceLookupError attribute), 8

metadata (proknow.Patients.PatientItem attribute), 34

mrn (proknow.Patients.PatientItem attribute), 34

mrn (proknow.Patients.PatientSummary attribute), 33

## N

name (proknow.CustomMetrics.CustomMetricItem attribute), 12

name (proknow.Patients.PatientItem attribute), 34

name (proknow.Patients.PatientSummary attribute), 33

name (proknow.Roles.RoleItem attribute), 20

name (proknow.Roles.RoleSummary attribute), 20

name (proknow.Users.UserItem attribute), 24

name (proknow.Users.UserSummary attribute), 24

name (proknow.Workspaces.WorkspaceItem attribute), 16

## P

patch() (proknow.Requestor.Requestor method), 7

PatientItem (class in proknow.Patients), 34

Patients (class in proknow.Patients), 29

patients (proknow.ProKnow.ProKnow attribute), 5

PatientSummary (class in proknow.Patients), 33

permissions (proknow.Roles.RoleItem attribute), 20

PlanItem (class in proknow.Patients), 40

post() (proknow.Requestor.Requestor method), 7

ProKnow (class in proknow.ProKnow), 5

proknow.CustomMetrics (module), 9

proknow.Exceptions (module), 8

proknow.Patients (module), 29

proknow.ProKnow (module), 5

proknow.Requestor (module), 6

proknow.Roles (module), 17

proknow.Uploads (module), 25

proknow.Users (module), 21

proknow.Workspaces (module), 13

ProKnowError, 8

protected (proknow.Workspaces.WorkspaceItem attribute), 16

put() (proknow.Requestor.Requestor method), 7

Python Enhancement Proposals

PEP 8, 45

## Q

query() (proknow.CustomMetrics.CustomMetrics method), 11

query() (proknow.Patients.Patients method), 32

query() (proknow.Roles.Roles method), 19

query() (proknow.Users.Users method), 23

query() (proknow.Workspaces.Workspaces method), 15

## R

Requestor (class in proknow.Requestor), 6

requestor (proknow.ProKnow.ProKnow attribute), 5

resolve() (proknow.CustomMetrics.CustomMetrics method), 11

resolve() (proknow.Workspaces.Workspaces method), 15

resolve\_by\_id() (proknow.CustomMetrics.CustomMetrics method), 11

resolve\_by\_id() (proknow.Workspaces.Workspaces method), 15

resolve\_by\_name() (proknow.CustomMetrics.CustomMetrics method), 12

resolve\_by\_name() (proknow.Workspaces.Workspaces method), 16

role\_id (proknow.Users.UserItem attribute), 24

RoleItem (class in proknow.Roles), 20

Roles (class in proknow.Roles), 17

roles (proknow.ProKnow.ProKnow attribute), 5

RoleSummary (class in proknow.Roles), 20

## S

save() (proknow.CustomMetrics.CustomMetricItem method), 13

save() (proknow.Patients.PatientItem method), 36

save() (proknow.Roles.RoleItem method), 21

save() (proknow.Users.UserItem method), 25

save() (proknow.Workspaces.WorkspaceItem method), 17

set\_metadata() (proknow.Patients.PatientItem method), 36

sex (proknow.Patients.PatientItem attribute), 34

sex (proknow.Patients.PatientSummary attribute), 33

slug (proknow.Workspaces.WorkspaceItem attribute), 16

status\_code (proknow.Exceptions.HttpError attribute), 8

stream() (proknow.Requestor.Requestor method), 8

StructureSetItem (class in proknow.Patients), 39

StudySummary (class in proknow.Patients), 36

## T

TimeoutExceededError, 9

type (proknow.CustomMetrics.CustomMetricItem attribute), 12

## U

upload() (proknow.Uploads.Uploads method), 26

UploadBatch (class in proknow.Uploads), 27

UploadEntitySummary (class in proknow.Uploads), 29

UploadPatientSummary (class in proknow.Uploads), 28

Uploads (class in proknow.Uploads), 25

uploads (proknow.ProKnow.ProKnow attribute), 5

UserItem (class in proknow.Users), 24

[Users \(class in proknow.Users\)](#), [21](#)  
[users \(proknow.ProKnow.ProKnow attribute\)](#), [5](#)  
[UserSummary \(class in proknow.Users\)](#), [23](#)

## W

[WorkspaceItem \(class in proknow.Workspaces\)](#), [16](#)  
[WorkspaceLookupError](#), [8](#)  
[Workspaces \(class in proknow.Workspaces\)](#), [13](#)  
[workspaces \(proknow.ProKnow.ProKnow attribute\)](#), [5](#)