
Craft AI MLOps platform Python SDK

Release 0.6.0

Craft AI

Oct 04, 2022

Contents

1	craft_ai_sdk package	1
1.1	craft_ai_sdk.exceptions module	1
1.2	craft_ai_sdk.sdk module	1
1.3	craft_ai_sdk.utils module	1
2	CraftAiSdk class	3
2.1	Step	4
2.2	Pipeline	5
2.3	Pipeline Execution	6
2.4	Pipeline Endpoint	7
2.5	Data store	8
2.6	Environment Variables	9
	Python Module Index	11
	Index	13

craft_ai_sdk package

1.1 craft_ai_sdk.exceptions module

exception `craft_ai_sdk.exceptions.SdkException`(*message, status_code=None, name=None, stack_message=None, request_id=None, additional_data=None*)

Bases: `Exception`

General exception while using this package

1.2 craft_ai_sdk.sdk module

See *CraftAiSdk class* for more details.

1.3 craft_ai_sdk.utils module

CraftAiSdk class

class craft_ai_sdk.**CraftAiSdk**(*access_token=None, environment_url=None, verbose_log=None*)

Main class to instantiate

base_environment_url

Base URL to CraftAI Environment.

Type

str

base_api_url

Base URL to CraftAI API.

Type

str

verbose_log

If True, information during method execution will be printed.

Type

bool

__init__(*access_token=None, environment_url=None, verbose_log=None*)

Initiates CraftAiSdk.

Parameters

- **access_token** (str, optional) – CraftAI access token. Defaults to CRAFT_AI_ACCESS_TOKEN environment variable.
- **environment_url** (str, optional) – URL to CraftAI environment. Defaults to CRAFT_AI_ENVIRONMENT_URL environment variable.
- **verbose_log** (bool, optional) – If True, information during method execution will be printed. It is set to True, if the environment variable SDK_VERBOSE_LOG is set to true; False if it is set to false. Defaults to True in interactive mode; False otherwise.

Raises

ValueError – if the environment_url is not defined or when the corresponding environment variable is not set.

2.1 Step

`CraftAiSdk.create_step(repo, private_key, step_name, branch_name=None)`

Create pipeline step from a source code located on a remote repository.

Parameters

- **repo** (*str*) – Remote repository url.
- **private_key** (*str*) – Private SSH key related to the repository.
- **step_name** (*str*) – Step name (as defined in the `config.yaml` configuration file) to create.
- **branch_name** (*str*, optional) – Branch name. Defaults to None.

Returns

List of steps represented as dict (with either keys "id" and "name" if the creation succeeded or keys "name" and "error" if the creation failed).

Return type

list of dict[*str*, *str*]

`CraftAiSdk.get_step(step_name)`

Get a single step if it exists.

Parameters

step_name (*str*) – The name of the step to get.

Returns

The step informations (with keys "id", "name") or None if the step does not exist.

Return type

dict

`CraftAiSdk.update_step(step_name, repo, private_key=None, branch_name=None)`

Update a pipeline step from a source code located on a remote repository.

Parameters

- **step_name** (*str*) – Name of the step to update.
- **repo** (*str*) – Remote repository url.
- **private_key** (*str*) – Private SSH key related to the repository.
- **branch_name** (*str*, optional) – Branch name. Defaults to None.

Returns

Informations of the updated step represented as dict (with either keys "id" and "name" if the creation succeeded or keys "name" and "error" if the creation failed).

Return type

dict[*str*, *str*]

`CraftAiSdk.list_steps()`

Get the list of all steps.

Returns

List of steps represented as dict (with keys "id", "name").

Return type

list of dict

`CraftAiSdk.delete_step(step_name)`

Delete one step.

Parameters

step_name (*str*) – Name of the step to delete as defined in the `config.yaml` configuration file.

Returns

Deleted step represented as dict (with keys "id" and "name").

Return type

dict[str, str]

2.2 Pipeline

`CraftAiSdk.create_pipeline(pipeline_name, step_name)`

Create a pipeline containing a single step.

Parameters

- **pipeline_name** (*str*) – Name of the pipeline to be created.
- **step_name** (*str*) – Name of the step to be included in the pipeline.

Returns

Created pipeline represented as dict (with key "id").

Return type

dict[str, str]

`CraftAiSdk.get_pipeline(pipeline_name)`

Get a single pipeline if it exists.

Parameters

pipeline_name (*str*) – Name of the pipeline to get.

Returns

The pipeline informations or None if the pipeline does not exist.

- "id" (*str*): Pipeline id.
- "name" (*str*): Pipeline name.
- "created_at" (*str*): Pipeline date of creation.
- "steps" (*list*): List of step names.

Return type

dict

`CraftAiSdk.delete_pipeline(pipeline_name, force_endpoints_deletion=False)`

Delete a pipeline identified by its name and id.

Parameters

- **pipeline_name** (*str*) – Name of the pipeline.
- **force_endpoints_deletion** (*bool*, optional) – if True the associated endpoints will be deleted too. Defaults to False.

Returns

The deleted pipeline and associated deleted endpoints. The returned dict contains two keys:

- "pipeline" (dict): Deleted pipeline represented as dict (with keys "id" and "name").
- "endpoints" (list): List of deleted endpoints represented as dict (with keys "id", "name", "body_params" and allow_unknown_params).

Return type
dict

2.3 Pipeline Execution

`CraftAiSdk.execute_pipeline(pipeline_name)`

Execute a pipeline.

Parameters

pipeline_name (*str*) – Name of an existing pipeline.

Returns

Created pipeline execution represented as dict (with key "execution_id").

Return type

dict[str, str]

`CraftAiSdk.list_pipeline_executions(pipeline_name)`

Get a list of executions for the given pipeline

Parameters

pipeline_name (*str*) – Name of an existing pipeline.

Returns

A list of information on the pipeline execution represented as dict (with keys "execution_id", "status", "created_at", "steps", "pipeline_name"). In particular the keys:

- "steps" is a list of the execution steps represented as dict (with keys "name", "status").
- "pipeline_name" is the executed pipeline name.

Return type

list

`CraftAiSdk.get_pipeline_execution(pipeline_name, execution_id)`

Get the status of one pipeline execution identified by its name.

Parameters

- **pipeline_name** (*str*) – Name of an existing pipeline.
- **execution_id** (*str*) – Name of the pipeline execution.

Returns

Information on the pipeline execution with id execution_id represented as dict (with keys "execution_id", "status", "created_at", "steps", "pipeline_name"). In particular the keys:

- "steps" is a list of the execution steps represented as dict (with keys "name", "status").
- "pipeline_name" is the executed pipeline name.

Return type

dict

```
CraftAiSdk.get_pipeline_execution_logs(pipeline_name, execution_id,
                                       from_datetime=None, to_datetime=None,
                                       limit=None)
```

Get the logs of an executed pipeline identified by its name.

Parameters

- **pipeline_name** (*str*) – Name of an existing pipeline.
- **execution_id** (*str*) – ID of the pipeline execution.
- **from_datetime** (*datetime.time*, optional) – Datetime from which the logs are collected.
- **to_datetime** (*datetime.time*, optional) – Datetime until which the logs are collected.
- **limit** (*int*, optional) – Maximum number of logs that are collected.

Returns

List of collected logs represented as dict (with keys "message", "timestamp" and "stream").

Return type

list

2.4 Pipeline Endpoint

```
CraftAiSdk.create_endpoint(pipeline_name, endpoint_name, endpoint_params=None,
                           allow_unknown_params=None)
```

Create a custom endpoint associated to a given pipeline.

Parameters

- **pipeline_name** (*str*) – Name of the pipeline.
- **endpoint_name** (*str*) – Name of the endpoint.
- **endpoint_params** (*dict[str, dict]*, optional) – structure of the endpoint parameters. Each item defines a parameter which name is given by the key and which constraints (type and requirement) are given by the value. An item has the form:

```
[str: parameter name] : {
    "required": [bool],
    "type": [str in ["string", "number", "object", "array"]],
}
```

Defaults to None.

- **allow_unknown_params** (*bool*, optional) – if True the custom endpoint allows other parameters not specified in `endpoint_params`. Defaults to None.

Returns

Created endpoint represented as dict (with key "id" and "name").

Return type

dict[str, str]

```
CraftAiSdk.list_endpoints()
```

Get the list of all endpoints.

Returns

List of endpoints represented as dict (with keys "id", "name", "body_params", "allow_unknown_params" and "pipeline").

Return type

list of dict

`CraftAiSdk.get_endpoint(endpoint_name)`

Get information of an endpoint.

Parameters

endpoint_name (*str*) – Name of the endpoint.

Returns

Endpoint information represented as dict (with keys "id", "name", "body_params", "allow_unknown_params" and "pipeline").

Return type

dict

`CraftAiSdk.trigger_endpoint(**kwargs)`

`CraftAiSdk.delete_endpoint(endpoint_name)`

Delete an endpoint identified by its name.

Parameters

endpoint_name (*str*) – Name of the endpoint.

Returns

Deleted endpoint represented as dict (with keys "id", "name", "body_params", "allow_unknown_params").

Return type

dict

`CraftAiSdk.generate_new_endpoint_token(endpoint_name)`

Generate a new endpoint token for an endpoint.

Parameters

endpoint_name (*str*) – Name of the endpoint.

Returns

New endpoint token represented as dict (with keys "endpoint_token").

Return type

dict

2.5 Data store

`CraftAiSdk.upload_data_store_object(filepath_or_buffer, object_path_in_datastore)`

Upload a file as an object into the data store.

Parameters

- **filepath_or_buffer** (*str*, or file-like object) – String, path to the file to be uploaded ; or file-like object implementing a `read()` method (e.g. via builtin `open` function). The file object must be opened in binary mode, not text mode.
- **object_path_in_datastore** (*str*) – Destination of the uploaded file.

`CraftAiSdk.download_data_store_object(object_path_in_datastore, filepath_or_buffer)`

Download an object in the data store and save it into a file.

Parameters

- **object_path_in_datastore** (*str*) – Location of the object to download from the data store.
- **filepath_or_buffer** (*str* or file-like object) – String, filepath to save the file to ; or a file-like object implementing a `write()` method, (e.g. via builtin open function). The file object must be opened in binary mode, not text mode.

Returns

content of the file

Return type

`str`

`CraftAiSdk.list_data_store_objects()`

Get the list of the objects stored in the data store.

Returns

List of objects in the data store represented as dict (with keys "path", "last_modified", and "size").

Return type

list of dict

`CraftAiSdk.delete_data_store_object(object_path_in_datastore)`

Delete an object on the datastore.

Parameters

- **object_path_in_datastore** (*str*) – Location of the object to be deleted in the data store.

Returns

Deleted object represented as dict (with key "path").

Return type

dict

2.6 Environment Variables

`CraftAiSdk.create_or_update_environment_variable(environment_variable_name, environment_variable_value)`

Create or update an environment variable available for all pipelines executions.

Parameters

- **environment_variable_name** (*str*) – Name of the environment variable to create.
- **environment_variable_value** (*str*) – Value of the environment variable to create.

Returns

An object containing the id of environment variable (with keys "id")

Return type

dict

`CraftAiSdk.list_environment_variables()`

Get a list of all environments variables.

Returns

List of environment variable as dict (with keys "name" and "value")

Return type

list of dict

`CraftAiSdk.delete_environment_variable(environment_variable_name)`

Delete the specified environment variable

Parameters

environment_variable_name (*str*) – Name of the environment variable to delete.

Returns

dict (with keys "name" and "value") of the deleted environment variable

Python Module Index

C

`craft_ai_sdk.exceptions`, [1](#)
`craft_ai_sdk.sdk`, [1](#)
`craft_ai_sdk.utils`, [1](#)

Symbols

`__init__()` (*craft_ai_sdk.CraftAiSdk method*), 3

B

`base_api_url` (*craft_ai_sdk.CraftAiSdk attribute*), 3
`base_environment_url` (*craft_ai_sdk.CraftAiSdk attribute*), 3

C

`craft_ai_sdk.exceptions`
 module, 1
`craft_ai_sdk.sdk`
 module, 1
`craft_ai_sdk.utils`
 module, 1
`CraftAiSdk` (class in *craft_ai_sdk*), 3
`create_endpoint()` (*craft_ai_sdk.CraftAiSdk method*), 7
`create_or_update_environment_variable()`
 (*craft_ai_sdk.CraftAiSdk method*), 9
`create_pipeline()` (*craft_ai_sdk.CraftAiSdk method*), 5
`create_step()` (*craft_ai_sdk.CraftAiSdk method*), 4

D

`delete_data_store_object()`
 (*craft_ai_sdk.CraftAiSdk method*), 9
`delete_endpoint()` (*craft_ai_sdk.CraftAiSdk method*), 8
`delete_environment_variable()`
 (*craft_ai_sdk.CraftAiSdk method*), 10
`delete_pipeline()` (*craft_ai_sdk.CraftAiSdk method*), 5
`delete_step()` (*craft_ai_sdk.CraftAiSdk method*), 4
`download_data_store_object()`
 (*craft_ai_sdk.CraftAiSdk method*), 8

E

`execute_pipeline()` (*craft_ai_sdk.CraftAiSdk method*), 6

G

`generate_new_endpoint_token()`
 (*craft_ai_sdk.CraftAiSdk method*), 8
`get_endpoint()` (*craft_ai_sdk.CraftAiSdk method*), 8

`get_pipeline()` (*craft_ai_sdk.CraftAiSdk method*), 5

`get_pipeline_execution()`
 (*craft_ai_sdk.CraftAiSdk method*), 6

`get_pipeline_execution_logs()`
 (*craft_ai_sdk.CraftAiSdk method*), 6

`get_step()` (*craft_ai_sdk.CraftAiSdk method*), 4

L

`list_data_store_objects()`
 (*craft_ai_sdk.CraftAiSdk method*), 9
`list_endpoints()` (*craft_ai_sdk.CraftAiSdk method*), 7
`list_environment_variables()`
 (*craft_ai_sdk.CraftAiSdk method*), 9
`list_pipeline_executions()`
 (*craft_ai_sdk.CraftAiSdk method*), 6
`list_steps()` (*craft_ai_sdk.CraftAiSdk method*), 4

M

module
 `craft_ai_sdk.exceptions`, 1
 `craft_ai_sdk.sdk`, 1
 `craft_ai_sdk.utils`, 1

S

`SdkException`, 1

T

`trigger_endpoint()` (*craft_ai_sdk.CraftAiSdk method*), 8

U

`update_step()` (*craft_ai_sdk.CraftAiSdk method*), 4
`upload_data_store_object()`
 (*craft_ai_sdk.CraftAiSdk method*), 8

V

`verbose_log` (*craft_ai_sdk.CraftAiSdk attribute*), 3