hbp_archive Documentation

Release 0.8.0

Andrew Davison and Shailesh Appukuttan

Sep 17, 2020

Contents

A high-level API for interacting with the Human Brain Project archival storage at CSCS.

Author: Andrew Davison and Shailesh Appukuttan, CNRS

Documentation: https://hbp-archive.readthedocs.io

Installation:

```
pip install hbp_archive
```

CHAPTER 1

Example Usage

```
from hbp_archive import Container, PublicContainer, Project, Archive


# Working with a public container

container = PublicContainer("https://object.cscs.ch/v1/AUTH_id/my_container")
files = container.list()
local_file = container.download("README.txt")
print(container.read("README.txt"))
number_of_files = container.count()
size_in_MB = container.size("MB")

# Working with a private container

container = Container("MyContainer", username="xyzabc")  # you will be prompted for␣
↪your password
files = container.list()
local_file = container.download("README.txt", overwrite=True)  # default is not to␣
↪overwrite existing files
print(container.read("README.txt"))
number_of_files = container.count()
size_in_MB = container.size("MB")

container.move("my_file.dat", "a_subdirectory", "new_name.dat")  # move/rename file␣
↪within a container

# Reading a file directly, without downloading it

with container.open("my_data.txt") as fp:
    data = np.loadtxt(fp)

# Working with a project

my_proj = Project('MyProject', username="xyzabc")
```

(continues on next page)

(continued from previous page)

```
container = my_proj.get_container("MyContainer")

# Listing all your projects

archive = Archive(username="xyzabc")
projects = archive.projects
container = archive.find_container("MyContainer")  # will search through all projects
```

CHAPTER 2

Regarding CSCS Authentication

The Python Client attempts to simplify the CSCS authentication process. The users have the following options (in order of priority):

1. Setting an environment variable named `CSCS_PASS` with your CSCS password. On Linux, this can be done as:

   ```
   export CSCS_PASS='putyourpasswordhere'
   ```

   Environment variables set like this are only stored temporally. When you exit the running instance of bash by exiting the terminal, they get discarded. To save this permanentally, write the above command into *~/.bashrc* or *~/.profile* (you might need to reload these files by, for example, `source ~/.bashrc`)

2. Enter your CSCS password when prompted by the Python Client.

CHAPTER 3

File

class hbp_archive.**File**(*name*, *bytes*, *content_type*, *hash*, *last_modified*, *container=None*)

A representation of a file in a container.

The following actions can be performed:

Action	Method
Get directory name	*dirname*
Get file name	*basename*
Download a file	*download()*
Read contents of a file	*read()*
Move a file	*move()*
Rename a file	*rename()*
Copy a file	*copy()*
Delete a file	*delete()*
Get size of file	*size()*

basename

Returns the file name from file path.

Returns Name of file.

Return type string

copy(*target_directory*, *new_name=None*, *overwrite=False*)

Copy this file to specified directory.

Parameters

- **target_directory** (*string*) – Target directory where the file is to be copied.
- **new_name** (*string, optional*) – New name to be assigned to file (including extension, if any).
- **overwrite** (*boolean, optional*) – Specify if any already existing file at target location should be overwritten.

delete()
Delete this file.

dirname
Returns the directory name from file path.

Returns Directory path of file.

Return type string

download(*local_directory*, *with_tree=True*, *overwrite=False*)
Download this file to a local directory.

Parameters

- **local_directory** (*string*) – Local directory path where file is to be saved.
- **with_tree** (*boolean, optional*) – Specify if directory structure of file is to be retained.
- **overwrite** (*boolean, optional*) – Specify if any already existing file should be overwritten.

Returns Path of file created inside specified local directory.

Return type string

move(*target_directory*, *new_name=None*, *overwrite=False*)
Move this file to the specified directory.

Parameters

- **target_directory** (*string*) – Target directory where the file is to be moved.
- **new_name** (*string, optional*) – New name to be assigned to file (including extension, if any).
- **overwrite** (*boolean, optional*) – Specify if any already existing file should be overwritten.

read(*decode='utf-8'*, *accept=[]*)
Read and return the contents of this file in the container.

Parameters

- **file_path** (*string*) – Path of file to be retrieved.
- **decode** (*string, optional*) – Files containing text will be decoded using specified encoding (default: 'utf-8'). To prevent any attempt at decoding, set *decode=False*.
- **accept** (*boolean, optional*) – To force decoding, put the expected content type in *accept*.

Returns Contents of the specified file.

Return type string (unicode)

rename(*new_name*, *overwrite=False*)
Rename this file within the source directory.

Parameters

- **new_name** (*string*) – New name to be assigned to file (including extension, if any).
- **overwrite** (*boolean, optional*) – Specify if any already existing file should be overwritten.

size(*units='bytes'*)

Return the size of this file in the requested unit (default bytes).

Parameters **units** (*string*) – Requested units for output. Options: 'bytes' (default), 'kB', 'MB', 'GB', 'TB'

Returns Size of specified file in requested units.

Return type float

CHAPTER 4

Container

class hbp_archive.**Container**(*container*, *username*, *token=None*, *project=None*)

A representation of a CSCS storage container. Can be used to operate both public and private CSCS containers. A CSCS account is needed to use this class.

The following actions can be performed:

Action	Method
Get metadata about the container	*metadata*
Get url if container is public	*public_url*
List all files in container	*list()*
Return a file from given path	*get()*
Get number of files in container	*count()*
Get total size of data in container	*size()*
Upload file(s) to container	*upload()*
Download a file from container	*download()*
Read contents of file in container	*read()*
Copy a file in container	*copy()*
Move a file in container	*move()*
Delete a file in container	*delete()*
Copy a directory in container	*copy_directory()*
Move a directory in container	*move_directory()*
Delete a directory in container	*delete_directory()*
List users with access to container	*access_control()*
Grant container access to user	*grant_access()*
Revoke container access from user	*revoke_access()*

access_control(*show_usernames=True*)

List the users that have access to this container.

Parameters **show_usernames** (*boolean, optional*) – default is *True*

Returns Dictionary with keys 'read' and 'write'; each having a value in the form of a list of usernames

Return type dict

copy (*file_path*, *target_directory*, *new_name=None*, *overwrite=False*)
Copy a file to the specified directory.

Parameters

- **file_path** (*string*) – Path of file to be copied.
- **target_directory** (*string*) – Target directory where the file is to be copied.
- **new_name** (*string, optional*) – New name to be assigned to file (including extension, if any).
- **overwrite** (*boolean, optional*) – Specify if any already existing file should be overwritten.

copy_directory (*directory_path*, *target_directory*, *new_name=None*, *overwrite=False*)

Copy a directory to the specified directory location. The original tree structure of the directory will be maintained at the target location.

Parameters

- **directory_path** (*string*) – Path of directory to be copied.
- **target_directory** (*string*) – Path of target directory where specified directory is to be copied.
- **new_name** (*string, optional*) – New name to be assigned to directory.
- **overwrite** (*boolean, optional*) – Specify if any already existing files at target location should be overwritten. If False (default value), then only non-conflicting files will be copied over.

count ()
Number of files in the container

Returns Count of number of files in the container.

Return type int

delete (*file_path*)
Delete the specified file.

Parameters **file_path** (*string*) – Path of file to be deleted.

delete_directory (*directory_path*)
Delete the specified directory (and its contents).

Parameters **directory_path** (*string*) – Path of directory to be deleted.

download (*file_path*, *local_directory='.'*, *with_tree=True*, *overwrite=False*)
Download a file from the container.

Parameters

- **file_path** (*string*) – Path of file to be downloaded.
- **local_directory** (*string, optional*) – Local directory path where file is to be saved.
- **with_tree** (*boolean, optional*) – Specify if directory structure of file is to be retained.

- **overwrite** (*boolean, optional*) – Specify if any already existing file should be overwritten.

Returns Path of file created inside specified local directory.

Return type string

get(*file_path*)
Return a File object for the file at the given path.

Parameters **file_path** (*string*) – Path of file to be retrieved.

Returns Requested *hbp_archive.File* object from container.

Return type *hbp_archive.File*

grant_access(*username*, *mode='read'*)
Give read or write access to the given user.

Parameters

- **username** (*string*) – username of user to be granted access; set to 'PUBLIC' to give public read-only access (no password required)
- **mode** (*string, optional*) – the access permission to be granted: 'read'/'write'; default = 'read'

Note: Use restricted to Superusers/Operators.

---

list(*dir_path=None*, *content_type=None*, *newer_than=None*, *older_than=None*, *contains_substring=None*, *extension=None*)
List all files in the container.

Parameters

- **dir_path** (*string*) – base directory of files to be listed, default is set to root directory.
- **content_type** (*string*) – content_type of files to be listed.
- **newer_than** (*datetime*) – start timestamp for files to be listed.
- **older_than** (*datetime*) – end timestamp for files to be listed.
- **contains_substring** (*string*) – substring to be matched for files to be listed.
- **extension** (*string*) – extension to be matched for files to be listed.

Returns List of *hbp_archive.File* objects existing in container.

Return type list

metadata
Metadata about the container.

Returns Dictionary with metadata about the container.

Return type dict

move(*file_path*, *target_directory*, *new_name=None*, *overwrite=False*)
Move a file to the specified directory.

Parameters

- **file_path** (*string*) – Path of file to be moved.
- **target_directory** (*string*) – Target directory where the file is to be moved.

- **new_name** (*string, optional*) – New name to be assigned to file (including extension, if any).
- **overwrite** (*boolean, optional*) – Specify if any already existing file should be overwritten.

move_directory(*directory_path*, *target_directory*, *new_name=None*, *overwrite=False*)

Move a directory to the specified directory location. Can also be used to rename a directory. The original tree structure of the directory will be maintained at the target location.

Parameters

- **directory_path** (*string*) – Path of directory to be copied.
- **target_directory** (*string*) – Path of target directory where specified directory is to be copied.
- **new_name** (*string, optional*) – New name to be assigned to directory.
- **overwrite** (*boolean, optional*) – Specify if any already existing files at target location should be overwritten. If False (default value), then only non-conflicting files will be copied over.

public_url

Get url if container is public.

Returns URL to access public container; returns None for private containers.

Return type string

read(*file_path*, *decode='utf-8'*, *accept=[]*)

Read and return the contents of a file in the container.

Parameters

- **file_path** (*string*) – Path of file to be retrieved.
- **decode** (*string, optional*) – Files containing text will be decoded using specified encoding (default: 'utf-8'). To prevent any attempt at decoding, set *decode=False*.
- **accept** (*boolean, optional*) – To force decoding, put the expected content type in *accept*.

Returns Contents of the specified file.

Return type string (unicode)

revoke_access(*username*, *mode='read'*)

Remove read or write access from the given user.

Parameters

- **username** (*string*) – username of user to be revoked access; set to 'PUBLIC' to make a container private
- **mode** (*string, optional*) – the access permission to be revoked: 'read'/'write'; default = 'read'

Note: Use restricted to Superusers/Operators.

---

size(*units='bytes'*)

Total size of all data in the container

Parameters **units** (*string*) – Requested units for output. Options: 'bytes' (default), 'kB', 'MB', 'GB', 'TB'

Returns Total size of all data in the container in requested units.

Return type float

upload(*local_paths*, *remote_directory=''*, *overwrite=False*)

Upload file(s) to the container.

Parameters

- **local_paths** (*string, list of strings*) – Local path of file(s) to be uploaded.
- **remote_directory** (*string, optional*) – Remote directory path where data is to be uploaded. Default is root directory.
- **overwrite** (*boolean, optional*) – Specify if any already existing file at target should be overwritten.

Returns List of strings indicating file paths created on container.

Return type list

Note: Using the command-line "swift upload" will likely be faster since it uses a pool of threads to perform multiple uploads in parallel. It is thus recommended for bulk uploads.

CHAPTER 5

PublicContainer

class hbp_archive.**PublicContainer**(*url*)

A representation of a public CSCS storage container. Can be used to operate only public CSCS containers. A CSCS account is not needed to use this class.

The following actions can be performed:

Action	Method
List all files in container	*list()*
Return a file from given path	*get()*
Get number of files in container	*count()*
Get total size of data in container	*size()*
Download a file from container	*download()*
Read contents of file in container	*read()*

Note: This class only permits read-only operations. For other features, you may access a public container via the *Container* class.

count()

Number of files in the container.

Returns Count of number of files in the container.

Return type int

download(*file_path*, *local_directory='.'*, *with_tree=True*, *overwrite=False*)

Download a file from the container.

file_path [string] Path of file to be downloaded.

local_directory [string, optional] Local directory path where file is to be saved.

with_tree [boolean, optional] Specify if directory structure of file is to be retained.

overwrite [boolean, optional] Specify if any already existing file should be overwritten.

Returns Path of file created inside specified local directory.

Return type string

get (*file_path*)
Return a File object for the file at the given path.

Parameters **file_path** (*string*) – Path of file to be retrieved.

Returns Requested *hbp_archive.File* object from container.

Return type *hbp_archive.File*

list (*dir_path=None*, *content_type=None*, *newer_than=None*, *older_than=None*, *contains_substring=None*, *extension=None*, *refresh=False*)
List all files in the container.

Parameters

- **dir_path** (*string*) – base directory of files to be listed, default is set to root directory.
- **content_type** (*string*) – content_type of files to be listed.
- **newer_than** (*datetime*) – start timestamp for files to be listed.
- **older_than** (*datetime*) – end timestamp for files to be listed.
- **contains_substring** (*string*) – substring to be matched for files to be listed.
- **extension** (*string*) – extension to be matched for files to be listed.
- **refresh** (*boolean*) – to force refreshing, in case contents have changed.

Returns List of *hbp_archive.File* objects existing in container.

Return type list

read (*file_path*, *decode='utf-8'*, *accept=[]*)
Read and return the contents of a file in the container.

Parameters

- **file_path** (*string*) – Path of file to be retrieved.
- **decode** (*string, optional*) – Files containing text will be decoded using specified encoding (default: 'utf-8'). To prevent any attempt at decoding, set *decode=False*.
- **accept** (*boolean, optional*) – To force decoding, put the expected content type in *accept*.

Returns Contents of the specified file.

Return type string (unicode)

size (*units='bytes'*)
Total size of all data in the container.

Parameters **units** (*string*) – Requested units for output. Options: 'bytes' (default), 'kB', 'MB', 'GB', 'TB'

Returns Total size of all data in the container in requested units.

Return type float

CHAPTER 6

Project

class hbp_archive.**Project**(*project*, *username*, *token=None*, *archive=None*)

A representation of a CSCS Project.

The following actions can be performed:

Action	Method / Property
Create a container inside project	*create_container()*
Rename a container inside project	*rename_container()*
Delete a container inside project	*delete_container()*
Get a container from project	*get_container()*
List containers that you can access	*containers*
Get names of containers in project	*container_names*
Get mapping of usernames to user ids	*users*

container_names

Returns a list of container names

Returns List of strings indicating container names in Project.

Return type list

containers

Containers you have access to in this project.

Returns Dictionary with keys as names of containers and their values being the corresponding 'hbp_archive.Container' object.

Return type dict

create_container(*container_name*, *public=False*)

Create a container inside the current project

Parameters

- **container_name** (*string*) – name to be assigned to container

- **public** (*boolean, optional*) – specify if container is to be made public; default is private

Note: Use restricted to Superusers/Operators.

delete_container(*container_name*)
Delete a container from the current project

Parameters **container_name** (*string*) – name of container to be deleted

Note: Use restricted to Superusers/Operators.

get_container(*name*)
Get a container from project.

Parameters **name** (*string*) – name of the container to be retrieved.

Returns Requested Container object from Project.

Return type 'hbp_archive.Container'

rename_container()
Rename a container inside the current project

Note: Use restricted to Superusers/Operators.

users
Return a mapping from usernames to user ids

Returns dict of mapping from usernames to user ids.

Return type dict

CHAPTER 7

Archive

class hbp_archive.**Archive**(*username*, *token=None*)

A representation of the Human Brain Project archival storage (Pollux SWIFT) at CSCS.

The following actions can be performed:

Action	Method / Property
List projects that you can access	*projects*
Search for container in all projects	*find_container()*

find_container(*container*)

Search through all projects for the container with the given name.

Parameters **name** (*string*) – name of the container to be searched

Returns Requested Container object from Project.

Return type 'hbp_archive.Container'

projects

Projects you have access to

Returns Dictionary with keys as names of projects and their values being the corresponding 'hbp_archive.Project' object.

Return type dict

CHAPTER 8

Misc

hbp_archive.**scale_bytes**(*value*, *units*)

Convert a value in bytes to a different unit.

Parameters

- **value** (*int*) – Value (in bytes) to be converted.
- **units** (*string*) – Requested units for output. Options: 'bytes', 'kB', 'MB', 'GB', 'TB'

Returns Value in requested units.

Return type float

hbp_archive.**set_logger**(*location='screen'*, *level='INFO'*)

Set the logging specifications for this module.

Parameters

- **location** (*string / None, optional*) – Can be set to following options: - 'screen' (case insensitive; default) : display log messages on screen - None : disable logging - Any other input will be considered as filename for logging to a file
- **level** (*string, option*) – Specify the logging level. Options: 'DEBUG'/'INFO'/'WARNING'/'ERROR'/'CRITICAL'

Python Module Index

h

Index

U