# RENKU CHEATSHEET

## Getting Started

```
$ pip install renku
```

Install with pip.

```
$ brew tap swissdatasciencecenter/renku
$ brew install renku
```

Install with Homebrew.

```
$ cd my_project
$ renku init
```

Starting a Renku project.

## Workflows

```
$ renku run --name <name> <command>
  [--input <in_file>...]
  [--output <out_file>...]
```

Execute a <command> with Renku tracking inputs and outputs. Input and output files are automatically detected. A workflow template(Plan) called <name> is created. --input/--output: specify inputy/outputs manually.

```
$ renku run <command> --no-output
```

Run a <command> that produces no output.

```
$ renku rerun <path>...
```

Recreate the file(s) <path> by rerunning the commands that created them.

```
$ renku update [--all] [<path>...]
```

Update outdated output files created by renku run.
With <path>'s: Only recreate these files.
With --all: Update all outdated output files.

```
$ renku workflow ls
```

List recorded workflow templates (Plans)

```
$ renku workflow edit <plan_name>
```

Edit Plan <plan_name> to set names/descriptions.

```
$ renku workflow execute <plan_name>
```

Execute a recorded Plan <plan_name> with new values or a different backend.

```
$ renku workflow compose <name> [<plan>...]
```

Create a meta-workflow by composing multiple plans.

```
$ renku workflow visualize [<path>...]
```

Visualize a graph of workflows that were involved in creating <path>. Use -i for interactive mode.

```
$ renku workflow export <plan_name>
```

Export a recorded Plan to a workflow language.

```
$ renku workflow show <plan_name>
```

Show details about a recorded Plan.

## Datasets

```
$ renku dataset create <dataset>
```

Create a new dataset.

```
$ renku dataset ls
```

List all datasets in the project.

```
$ renku dataset rm <dataset>
```

Remove a dataset.

```
$ renku dataset add <dataset> <url>
```

Add data from <url> to a dataset
<url> can be a local file path, an http(s) address or a Git git+http or git+ssh repository.

```
$ renku dataset add <dataset>
  --target <path> [--relative-to <rel-path>]
  <github-url>
```

Add only data contained in <path> from Git. With --relative-to: Trim <rel-path> from <path> locally.

```
$ renku dataset ls-files
```

List all dataset files in the project.

```
$ renku dataset unlink <dataset>
  [--include <path_or_pattern>]
```

Unlink (all) files from a dataset. With --include: Specify single path or pattern of files to unlink.

```
$ renku dataset import <uri>
```

Import a dataset. <uri> can be a Zenodo or Dataverse URL or DOI.

```
$ renku dataset export <dataset> <provider>
```

Export the dataset <dataset> to <provider>.
Providers:Zenodo, Dataverse.

```
$ renku dataset tag <dataset> <tag>
  [-d description]
```

Add a tag for the dataset at the current point in time.

```
$ renku dataset ls-tags <dataset>
```

Show all tags for a dataset.

```
$ renku dataset rm-tags <dataset> <tags..>
```

Remove tags from a dataset.

ETH zürich SDSC EPFL

# RENKU CHEATSHEET

## Config

```
$ renku config show [--global] <key>
```

Show Renku config value for <key>. With --global: Read from global settings.

```
$ renku config set [--global] <key> <value>
```

Set Renku config <key> to <value>. With --global: Read from global settings.

```
$ renku config remove [--global] <key>
```

Delete <key> from Renku config. With --global: Delete from global settings.

Available config options:

- **registry:** Docker image registry.
- **zenodo.access_token:** Token for Zenodo export.
- **dataverse.access_token**: Token for Dataverse export.
- **dataverse.server_url**: Url of the Dataverse instance.
- **show_lfs_message**: Whether to show the LFS warning message.
- **lfs_threshold**: Threshold below which files aren't added to git LFS. Accepts values like *100kb, 1mb or 0.5gb*

## Past Activities

```
$ renku log [--workflows]
```

Show past Renku workflow executions and dataset activities.

## Knowledge Graph

```
$ renku graph export [--revision <revision>]
```

Export Renku metadata as JSON-LD. With --revision: only export metadata modified in a Git revision.

## Login/Logout

```
$ renku login [--git] <endpoint>
```

Login to a Renku instance, e.g. renkulab.io. With --git: Also login to Gitlab.

```
$ renku logout[<endpoint>]
```

Login from all logged in Renku instances. With <endpoint>: Only logout from this instance.

## Misc

```
$ renku save [-m <message>]
```

Save (commit) and push all local changes.
With -m: Set a message for the changes made.

```
$ renku rm [<path>...]
```

Remove <path>'s from a renku repository. Removes from datasets and Git.

```
$ renku mv [<path>...] <destination>
```

Move <path>'s  to <destination>.

```
$ renku migrate
```

Migrate metadata to the current Renku version.

```
$ renku doctor
```

Check system and repository for potential problems.

```
$ renku githooks install [--force]
```

Install Renku Git hooks. With --force: Override existing hooks.

```
$ renku githooks uninstall
```

Uninstall Renku Git hooks.

```
$ renku storage pull <path>...
```

Pull <path>'s from external storage.

```
$ renku status
```

Show status of the Renku repository.

## Resources

Public instance of Renku:
https://renkulab.io/

Github:
https://github.com/SwissDataScienceCenter/renku

Documentation:
https://renku.readthedocs.io/en/latest/

Renku-Python Documentation:
https://renku-python.readthedocs.io/en/latest/

SDSC:
https://datascience.ch/

ETH zürich SDSC EPFL