signac-dashboard Documentation

Release 0.1.6

Bradley Dice

Oct 10, 2018

Contents

CHAPTER 1

About

`signac-dashboard` is a data visualization, analysis, and "dashboard" monitoring tool as part of the signac framework. The signac-dashboard interface allows users to rapidly view data managed in a signac project.

Important: This is documentation for the **signac-dashboard** *package*, which is part of the signac framework. See the signac-docs for a comprehensive introduction to the **signac** *framework*.

CHAPTER 2

Contents

2.1 Quickstart

2.1.1 Installation

The `signac-dashboard` app requires at least Python version 3.4! To install this package, first clone the repository and install its submodules.

```
$ git clone https://bitbucket.org/glotzer/signac-dashboard.git
$ cd signac-dashboard
$ git submodule update --init --recursive
```

and then install using pip:

```
$ pip install .
```

2.1.2 Usage

You can start a dashboard to visualize *signac* project data in the browser, by importing the `Dashboard` class and calling its `main()` method.

```
from signac_dashboard import Dashboard
Dashboard().main()
```

2.1.3 Start a Dashboard

The code below will open a dashboard for an newly-initialized (empty) project, with no jobs and one module loaded. Write the file *dashboard.py* with these contents:

```
from signac_dashboard import Dashboard
from signac_dashboard.modules import ImageViewer

if __name__ == '__main__':
    dashboard = Dashboard(modules=[ImageViewer()])
    dashboard.main()
```

Then launch the dashboard with *python dashboard.py run*.

2.1.4 Specifying a custom job title

By creating a class that inherits from `Dashboard` (which we'll call `MyDashboard`), we can begin to customize some of the functions that make up the dashboard, like `job_title()`, which gives a human-readable title to each job.

```
class MyDashboard(Dashboard):

    def job_title(self, job):
        return 'Concentration(A) = {}'.format(job.sp['conc_A'])

if __name__ == '__main__':
    MyDashboard().main()
```

2.1.5 Running dashboards on a remote host

To use dashboards hosted by a remote computer, open an SSH tunnel to the remote computer and forward the port where the dashboard is hosted. For example, connect to the remote computer with

```
ssh username@remote.server.org -L 8888:localhost:8888
```

to forward port 8888 on the host to port 8888 on your local computer.

2.1.6 Dissecting the Dashboard Structure

- *Jobs* are how signac manages data. Each job has a statepoint (which contains job metadata) and a document (for persistent storage of key-value pairs). Jobs can be displayed in *list view* or *grid view*. The list view provides quick descriptions and status information from many jobs, while the grid view is intended to show text and media content from one or more jobs.
- *Templates* provide the HTML structure of the dashboard's pages, written in Jinja template syntax for rendering content on the server
- *Modules* are server-side Python code that interface with your signac data to display content. Generally, a module will render content from a specific *job* into a *card template*.
- *Cards* are a type of template that is shown in *grid view* and contains content rendered by a *module*.

2.1.7 Included Modules

Defining a module requires a *name* for display, a *context* to determine when the module should be shown (currently only `'JobContext'` is supported), and a *template* (written in HTML/Jinja-compatible syntax) where the content will be rendered. An optional `enabled` argument can be set to `False` to disable the module until it is selected by the user. A module must be a subclass of `Module` and define the function `get_cards()` which returns an array of dictionaries with properties `'name'` and `'content'`, like so:

```
class MyModule(Module):

    def get_cards(self):
        return [{'name': 'My Module', 'content': render_template('path/to/template.
↪html')}]
```

Statepoint Parameters

The `StatepointList` module shows the key-value pairs in the statepoint.

```
from signac_dashboard.modules.statepoint_list import StatepointList
sp_mod = StatepointList()
```

Job Document

The `DocumentList` module shows the key-value pairs in the job document, with long values optionally truncated (default is no truncation).

```
from signac_dashboard.modules.document_list import DocumentList
doc_mod = DocumentList(max_chars=140)  # Output will be truncated to one tweet length
```

File List

The `FileList` module shows a listing of the job's workspace directory with links to each file. This can be very slow since it has to read the disk for every job displayed, use with caution in large signac projects.

```
from signac_dashboard.modules.file_list import FileList
file_mod = FileList(enabled=False)  # Recommended to disable this module by default
```

Image Viewer

The `ImageViewer` module displays images in any format that works with a standard HTML `<img>` tag. The module defaults to showing all images of PNG, JPG, or GIF types. A filename or glob can be defined to select specific filenames. Multiple Image Viewer modules can be defined with different filenames or globs to enable/disable cards individually.

```
from signac_dashboard.modules.image_viewer import ImageViewer
img_mod = ImageViewer()  # Shows all PNG/JPG/GIF images
img_mod = ImageViewer(name='Bond Order Diagram', img_globs=['bod.png'])
```

Video Viewer

The `VideoViewer` module displays videos using a standard HTML `<video>` tag. The module defaults to showing all videos of MP4 or M4V types. A filename or glob can be defined to select specific filenames, which may be of any format supported by your browser with the `<video>` tag. A "poster" can be defined, which shows a thumbnail with that filename before the video is started. Videos do not preload by default, since file sizes can be large and there may be many videos on a page. To enable preloading, use the argument `preload='auto'` or `preload='metadata'`. Multiple Video Viewer modules can be defined with different filenames or globs to enable/disable cards individually.

```
from signac_dashboard.modules.video_viewer import VideoViewer
video_mod = VideoViewer()  # Shows all MP4/M4V videos
video_mod = VideoViewer(name='Cool Science Video',
                        video_globs=['cool_science.mp4'],
                        poster='cool_science_thumbnail.jpg',
                        preload='none')
```

Notes

The *Notes* module uses the `'notes'` key in the job document to store plain text, perhaps human-readable descriptions of a job that may be useful in later analysis.

```
from signac_dashboard.modules.notes import Notes
notes_mod = Notes()
```

2.1.8 Searching jobs

The search bar accepts JSON-formatted queries in the same way as the `signac find` command-line tool. For example, using the query `{"key": "value"}` will return all jobs where the job statepoint `key` is set to `value`. To search jobs by their document key-value pairs, use `doc:` before the JSON-formatted query, like `doc:{"key": "value"}`.

2.1.9 Tips for Developers

During continuous integration, the code is checked with `flake8`. Run the following commands to set up a pre-commit hook that will ensure your code is compliant before pushing.

```
flake8 --install-hook git
git config --bool flake8.strict true
```

2.2 API

The complete **signac-dashboard** application interface (API).

2.2.1 Command Line Interface

Some core **signac-dashboard** functions are – in addition to the Python interface – accessible directly via the `$ signac-dashboard` command.

For more information, please see `$ signac-dashboard --help`.

2.2.2 Module contents

class signac_dashboard.**Dashboard**(*config={}*, *project=None*, *modules=[]*)

Bases: `object`

create_app(*config={}*)

create_assets()

classmethod decode_modules(*json_modules*, *enabled_modules=None*)

classmethod encode_modules(*modules*, *target='dict'*)

encoded_modules

get_all_jobs

get_job_details(*jobs*)

job_search

job_sorter(*job*)

job_subtitle(*job*)

job_title(*job*)

main()

Call this function to use the dashboard command line interface.

prepare()

register_module_asset(*asset*)

register_routes()

run(*host='localhost'*, *port=8888*, **args*, ***kwargs*)

url(*import_name*, *url_rules=[]*, *import_file='signac_dashboard'*, ***options*)

class signac_dashboard.**Module**(*name*, *context*, *template*, *enabled=True*)

Bases: `object`

disable()

enable()

get_cards()

register_assets(*dashboard*)

register_routes(*dashboard*)

toggle()

2.3 Modules

class signac_dashboard.modules.**ImageViewer**(*name='Image Viewer'*, *context='JobContext'*, *template='cards/image_viewer.html'*, *img_globs=['*.png'*, *'*.jpg'*, *'*.gif']*, ***kwargs*)

Bases: `signac_dashboard.module.Module`

get_cards(*job*)

class signac_dashboard.modules.**VideoViewer**(*name='Video Viewer'*, *context='JobContext'*, *template='cards/video_viewer.html'*, *video_globs=['*.mp4'*, *'*.m4v']*, *preload='none'*, *poster=None*, ***kwargs*)

Bases: `signac_dashboard.module.Module`

get_cards(*job*)

class signac_dashboard.modules.**StatepointList**(*name='Statepoint Parameters'*, *context='JobContext'*, *template='cards/statepoint_list.html'*, ***kwargs*)

Bases: signac_dashboard.module.Module

get_cards(*job*)

class signac_dashboard.modules.**DocumentList**(*name='Job Document'*, *context='JobContext'*, *template='cards/document_list.html'*, *max_chars=None*, ***kwargs*)

Bases: signac_dashboard.module.Module

get_cards(*job*)

class signac_dashboard.modules.**FileList**(*name='File List'*, *context='JobContext'*, *template='cards/file_list.html'*, *prefix_jobid=True*, ***kwargs*)

Bases: signac_dashboard.module.Module

download_name(*job*, *filename*)

get_cards(*job*)

class signac_dashboard.modules.**Notes**(*name='Notes'*, *context='JobContext'*, *template='cards/notes.html'*, ***kwargs*)

Bases: signac_dashboard.module.Module

get_cards(*job*)

register_assets(*dashboard*)

register_routes(*dashboard*)

CHAPTER 3

Indices and tables

- genindex
- modindex
- search

Python Module Index

Index