Digital Twin Operational Platform

Release 0.1

Linus

Mar 08, 2021

CONTENTS:

CHAPTER

ONE

DTOP

1.1 Get started

Updated version of the DTOP code with some plotting and front end code examples

Clone our repository using:

```
$ git clone git@github.com:Digital-Twin-Operational-Platform/dtop3.git
```

On MacOS/Linux:

```
$ python3 -m venv venv

$ source venv/bin/activate

(venv) % pip install -r requirements.txt

(venv) % export FLASK_APP=run.py

(venv) % export FLASK_DEBUG=1

(venv) % flask run
```

On Windows10:

```
% python -m venv venv

% venv\Scripts\activate

(venv) % pip install -r requirements.txt

(venv) % set FLASK_APP=run.py

(venv) % set FLASK_ENV=development

(venv) % flask run
```

Then in a web browser navigate to http://localhost:5000/

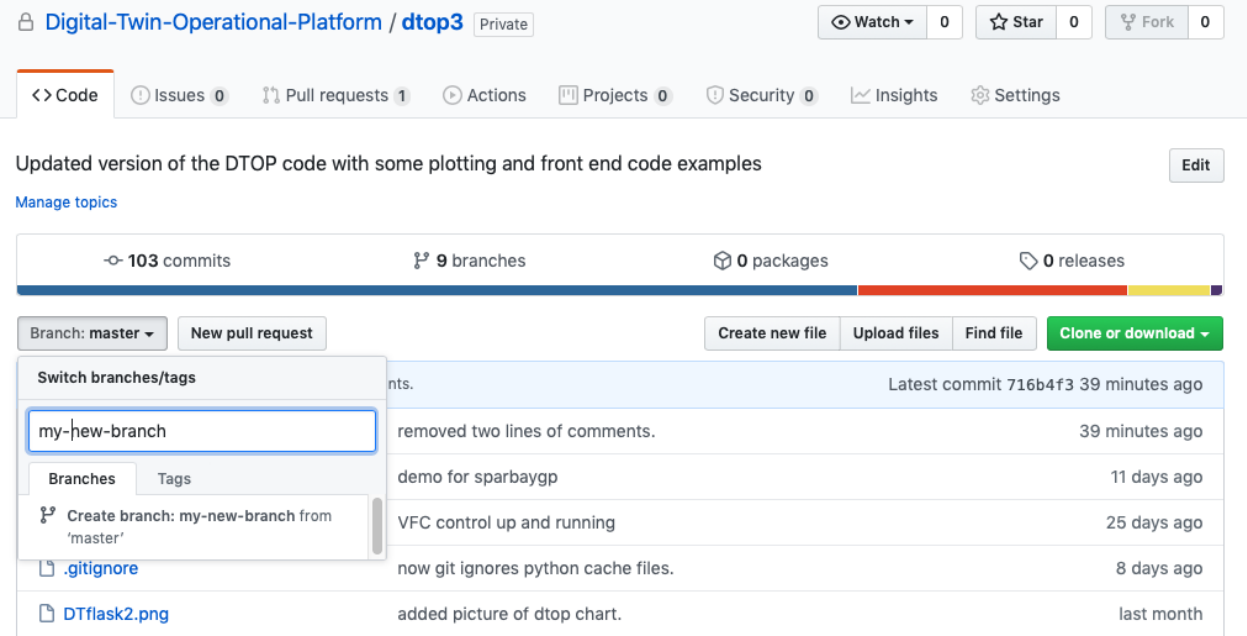
1.2 Project folders

```
'''.
├── README.md
├── config.py              # DTOP app configuration
├── dtApp                  # Flask code folder for the dtop app
│   ├── __init__.py
│   ├── dtCode
│   │   ├── __init__.py
│   │   ├── control.py
│   │   ├── dplot.py
│   │   ├── dplot2.py
│   │   ├── dplot3.py
│   │   ├── dtopstruc.py
│   │   └── prop.py
│   ├── dtData
│   │   ├── data_th_harmonic.csv
│   │   ├── data_th_impulse.csv
│   │   ├── data_th_noise.csv
│   │   └── soton_twin.py
│   ├── routes.py
│   ├── static             # Static web content
│   │   ├── css            # Style sheets for web content
│   │   ├── img            # Static images in this folder
│   │   └── txt            # Text only (like this file)
│   └── templates          # html pages
│       ├── base.html
│       ├── control.html
│       ├── dashboard.html
│       ├── docs.html
│       ├── dplot.html
│       ├── dplot2.html
│       ├── dtop3.html
│       ├── dtopstruc.html
│       ├── dtwin.html
│       ├── home.html
│       └── propagation.html
├── dtDoc                  # Documentation folder
├── dtLib                  # Scientific code library
│   ├── __init__.py
│   ├── general
│   │   ├── __init__.py
│   │   └── module1.py
│   ├── liverpool
│   │   ├── __init__.py
│   │   ├── msd.py
│   │   └── number.py
│   └── southampton
│       ├── __init__.py
│       ├── activeStructureVFCideal.py
│       └── passiveStructure.py
├── requirements.txt     # A list of all the required Python packages
└── run.py               # DTOP app launcher '''
```

1.3 Contribute to the DTOP project

This page explains how to create a page to display in the DTOP app.

1.3.1 Create your branch on GitHub



1.3.2 Clone our repository using:

```
$ git clone git@github.com:Digital-Twin-Operational-Platform/dtop3.git
```

If you have your branch already, you wanna update it:

```
$ git fetch origin
$ git pull
```

1.3.3 Follow the steps in the README.md

1.3.4 Open your text editor (e.g. VS Code)

1. Create new html file `./dtApp/templates/myinput.html` from the snippet below:

```
{% extends "base.html" %}
{% block title %}{% endblock %}
{% block head %}
{{ super() }}
{% endblock %}
{% block content %}

<h1>Digital Twin Operational Platform (DTOP) for a 3-storey structure</h1>
```

(continues on next page)

(continued from previous page)

```
<p>
    Welcome to the homepage for the Digital Twin Operational Platform (DTOP) for a 3-storey structure.
</p>

<p>These webpages are designed to demonstrate the ideas being developed in the DigiTwin research project. More details
    of this reseach project can be found on the <a href="http://digitwin.ac.uk//">DigiTwin website.</a></p>
{% endblock %}
```

2. Open `dtwin.html` in the editor, and write the following line of code

```
<div>
    <a href="new-file">Title of my new page</a>
</div>
```

3. Create new file `./dtApp/dtCode/myinput.py` from the snippet below:

```
'''
'''
from flask import render_template
from dtApp import app

@app.route('/myinput')
def myinput():
    return render_template('myinput.html')
```

4. Add line to the `./dtApp/routes.py` as follows

```
'''
routes.py
---------

The core script of the dtop project

Add to the list of import below your project module.
'''

from flask import render_template, request, redirect, Response, url_for
from dtApp import app
# At the moment each route file needs to be imported separately
# Better to change this in future
from .dtCode import dtopstruc
from .dtCode import dplot
from .dtCode import dplot2
from .dtCode import dplot3
from .dtCode import prop
from .dtCode import plotlyex
from .dtCode import dashboard
from .dtCode import sheffalg
from .dtCode import control
from .dtCode import cadmodel
from .dtCode import dplot4
from .dtCode import sparbaygp
from .dtCode import myinput
```

5. Go to `http://localhost:5000` and refresh the page.

Update your repository to current state of the project

1. Make a duplicate of your current repository and name it differently

```
$ cp dtop3 dtop3_DATE
```

2. Enter your repository (of which now there is a duplicate)

```
$ cd dtop3
```

3. Bring the changes to your current branch

```
$ git fetch origin
$ git pull origin master
$ git commit -m 'any change you had made to update your repo'
$ git push origin
```

4. You now have your branch in line with master.

1.4 Interactive web application

1.4.1 Flask code library

Control

dtApp.dtCode.control.**control**()

Dplot

dtApp.dtCode.dplot.**dplot**()

dtApp.dtCode.dplot.**draw1**(*ax*)
Draw a random scatterplot

dtApp.dtCode.dplot.**example1**()

dtApp.dtCode.dplot.**example2**()
Draw a hexbin with marginals From https://seaborn.pydata.org/examples/hexbin_marginals.html

dtApp.dtCode.dplot.**fig_response**(*fig*)
Turn a matplotlib Figure into Flask response

dtApp.dtCode.dplot.**nocache**(*response*)
Add Cache-Control headers to disable caching a response

Dplot2

This file is for plotting a time series.

dtApp.dtCode.dplot2.**dplot2**()

Dplot3

class dtApp.dtCode.dplot3.**InputForm**(**args*, ***kwargs*)
Bases: wtforms.form.Form

A = <UnboundField(FloatField, (), {'label': 'amplitude (m)', 'default': 1.0, 'valida

T = <UnboundField(FloatField, (), {'label': 'time interval (s)', 'default': 18, 'val

b = <UnboundField(FloatField, (), {'label': 'damping factor (kg/s)', 'default': 0, '

w = <UnboundField(FloatField, (), {'label': 'frequency (1/s)', 'default': 6.28318530

dtApp.dtCode.dplot3.**damped_vibrations**(*t*, *A*, *b*, *w*)

dtApp.dtCode.dplot3.**dplot3**()

Dtop Structure

Summary of this file.

dtApp.dtCode.dtopstruc.**create_figure**()

dtApp.dtCode.dtopstruc.**plot_png**()

dtApp.dtCode.dtopstruc.**q**()

Propagation

Module contents

1.4.2 Routes

1.4.3 Module contents

1.5 Scientific code library

1.5.1 List of sub-packages

General code

List of sub-modules

First module

class dtLib.general.module1.**DigitalTwin**(***kwargs*)
Bases: object

projectTree(**args*)

projectTreeArray(**args*)

Module contents

University of Liverpool

Mass-Spring-Damper

//
Created May 2020
@author: Marco De Angelis
github.com/marcodeangelis
University of Liverpool
Under LGPL v3 license.
//

This class is an API for the Single degree-of-freedom mass-spring-damper system.

class dtLib.liverpool.msd.**M_S_D**(***kwargs*)

Bases: object

centerNotation(*c*, *e*)

getC()

getDAMPING()

getDataPlot()

getDataPlotInnerBounds()

getDataPlotPrecise()

getExactBounds(*w*)

getInnerBound_MC(*w*)

getK()

getM()

getMASS()

getOMEGA0(**args*)

getOMEGAD()

getOmegaRange()

getProperties()

getPropertyTable()

getSTIFFNESS()

getW0()

getWD()

import_interval_library()

map2oz(**args*)

plotAllBounds()

plotExactBounds()

plotInnerBounds()

sdof_disp_analytic(*w*, *m*, *k*, *c*)

sdof_disp_analytic_2(*w*, *m*, *k*, *c*)

Number class

//
Created on Tue Dec 26 2017
@author: Marco De Angelis
University of Liverpool
Under LGPL v3 license.
//

class dtLib.liverpool.number.**I**(**args*)
Bases: *dtLib.liverpool.number.Interval*

superclass()

class dtLib.liverpool.number.**Interval**(**args*)
Bases: object

constructorNames()

contains(*other*)

hi()

inf()

inside(*other*)

lo()

mid()

numberTypes()

rad()

slider(*p*)

stradzero()

sup()

superclass()

width()

class dtLib.liverpool.number.**interval**(**args*)
Bases: *dtLib.liverpool.number.Interval*

superclass()

Module contents

University of Southampton

Active structure

Passive structure

dtLib.southampton.passiveStructure.**keoriginal**()

Module contents

1.5.2 Module contents

1.6 run.py

1.7 config.py

class config.**Config**
Bases: object

SECRET_KEY = 'you-will-never-guess'

1.8 Digital Twin Operational Platform

Get back to the DTOP (run Flask in background).

- Home page
- Dashboard
- Digital Twin

CHAPTER

TWO

INDICES AND TABLES

- genindex
- modindex
- search

PYTHON MODULE INDEX

c

d