ServiceX Dashboard for Jupyterlab

Jake Li

University of Illinois at Urbana-Champaign

Mentors: Oksana Shadura, Alexander Held, Benjamin Galewsky,

Gordon Watts, Mason Profitt

Background

- ServiceX is a data extraction and delivery service

- Users input datasets with specific instructions and ServiceX, with the given framework, extracts the data and stores it in an easily analyzable format

- ServiceX can be co-located with datasets to provide fast and efficient data reduction strategies, provide data as awkward arrays, and work as a data backend to Coffea and TRExFitter analysis tools

Motivation

- Currently, the ServiceX dashboard is only accessible from the original instance website

- To both have access to the ServiceX dashboard and a ServiceX notebook in Jupyterlab, one has to alternate between two tabs

- The goal of the project is to create a ServiceX dashboard within Jupyterlab

Dashboard Website

ServiceX

opendataaf-servicex.servicex.coffea-opendata.casa/servicex/transformation

ServiceX Docs Dashboard

Transformation Requests

Sort: **Finish (desc)**

Title	Start time	Finish time	Status	Files completed	Workers	Actions
Untitled	2022-08-04 18:28:49	2022-08-04 19:14:48	Complete	1,697 of 1,697	-	
Untitled	2022-07-26 09:24:50	2022-07-26 10:10:53	Complete	1,697 of 1,697	-	
Untitled	2022-06-16 15:39:24	2022-06-16 16:35:54	Complete	1,697 of 1,697	-	
Untitled	2022-04-23 13:28:49	2022-04-23 14:15:38	Complete	1,697 of 1,697	-	
Untitled	2022-04-20 17:17:47	2022-04-20 18:03:27	Complete	1,697 of 1,697	-	
Untitled	2022-04-20 10:28:58	2022-04-20 11:14:05	Complete	1,697 of 1,697	-	
Untitled	2022-04-19 16:52:01	2022-04-19 17:35:56	Complete	1,697 of 1,697	-	
Untitled	2022-04-19 16:01:20	2022-04-19 16:56:06	Complete	1,697 of 1,697	-	

All times in UTC.

« 1 »

64°F Cloudy

ENG 11:28 PM 9/24/2022

Project Overview

- The goal of the project was to create a Jupyterlab plugin that emulates the behavior of a ServiceX Dashboard with Jupyterlab

- The plugin will be able to monitor transforms on a certain ServiceX instance

- The plugin also has the ability to switch between different ServiceX instances, view successful and failed transforms, and cancel in-progress transform requests

ServiceX Dashboard in Jupyterlab

lab - JupyterLab

coffea-opendata.casa/user/jli301@illinois.edu/lab

File Edit View Run Kernel Git Tabs Settings Help

Select ServiceX Instance URL

▼ Select an instance

Transformation Requests

Current instance: https://opendataaf-servicex-aod.servicex.coffea-opendata-dev.casa/

Request	Start Time	Finish Time	Status	Files Completed	Workers	Actions
Link	2022-08-04 18:28:49	2022-08-04 19:14:48	Complete	1697 of 1697	-	
Link	2022-07-26 09:24:50	2022-07-26 10:10:53	Complete	1697 of 1697	-	
Link	2022-06-16 15:39:24	2022-06-16 16:35:54	Complete	1697 of 1697	-	
Link	2022-04-23 13:28:49	2022-04-23 14:15:38	Complete	1697 of 1697	-	
Link	2022-04-20	2022-04-20	Complete	1697 of	-	

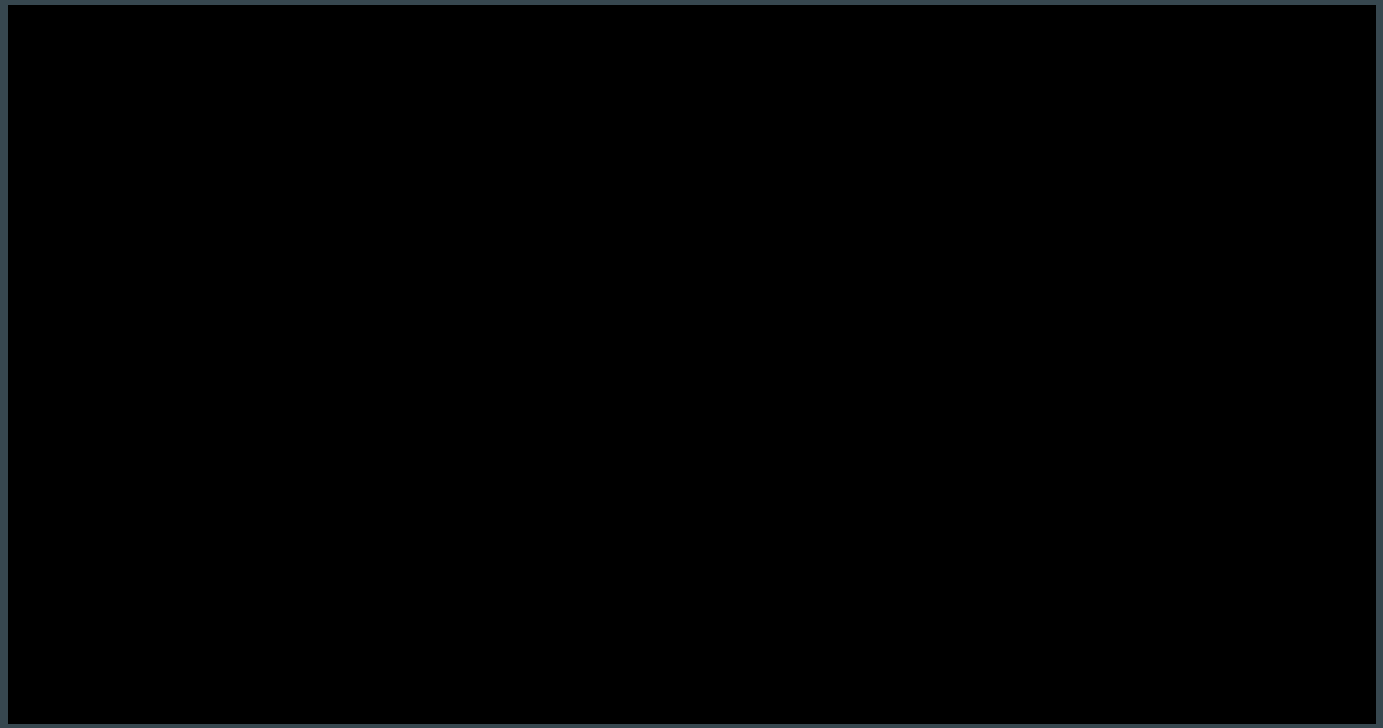
Terminal 1

```
groups: cannot find name for group ID 11265
cms-jovyan@jupyter-jli301-40illinois-2eedu:~$ pip install servicex-dashboard
Collecting servicex-dashboard
  Using cached servicex_dashboard-0.1.1-py3-none-any.whl (241 kB)
Installing collected packages: servicex-dashboard
Successfully installed servicex-dashboard-0.1.1
cms-jovyan@jupyter-jli301-40illinois-2eedu:~$ pip install servicex-dashboard
Requirement already satisfied: servicex-dashboard in /opt/conda/lib/python3.8/site-pack
ages (0.1.1)
cms-jovyan@jupyter-jli301-40illinois-2eedu:~$ pip install servicex-dashboard
Requirement already satisfied: servicex-dashboard in /opt/conda/lib/python3.8/site-pack
ages (0.1.1)
cms-jovyan@jupyter-jli301-40ill
cms-jovyan@jupyter-jli301-40illinois-2eedu:~$
```

Simple 1 0 Terminal 1

64°F Cloudy ENG 11:39 PM 9/24/2022

Project Demo



Project Timeline

- Week 1-2: Study background of project. Become familiar with JupyterHub and ServiceX. Give Lightning Talk.
- Weeks 3-5: Start to work on developing the Jupyterhub plug-in. Develop the framework for the plug-in using tools learned from weeks 1-2.
- Weeks 6-8: Work out any bugs that appear from initial development of the plug-in.
- Weeks 9-10: Test created plug-in with ServiceX instances, work out any bugs, tidy up look of the plug-in. Create and present the final presentation for the project.

High-level Code Overview (activate)

- The creation of the dashboard is contained within the activate function.

- Inside this activate function, there are numerous other functions, the primary one being the createTable function.

- The main body of the activate function creates the dropdown menu portion of the plug-in, and each link within the dropdown menu, when clicked, will call the createTable function and pass in the parameter of the instance url that inhabits that particular link.

Activate Function

```
async function activate(app: JupyterFrontEnd, palette: ICommandPalette) { //Activate function for plugin
  console.log('JupyterLab extension servicex-dashboard is activated!');

  let loop: any = null; //variable to set/reset polling (this was a bit of a rushed change, probably should get reworked in the future)

  let state = { //indicates current page, number of rows, order, and number of buttons on bar for table
    'page': 1,
    'rows': 7,
    'window': 4,
    'desc': true,
  }

  function pagination(querySet: any[], page: number, rows: number){ //returns page specific data for table and the total number of pages
    let trimStart = (page-1) * rows;
    let trimEnd = trimStart + rows;
    var trimmedData = querySet.slice(trimStart, trimEnd);
    var pages = Math.ceil(querySet.length / rows);
    return {
      'querySet': trimmedData,
      'pages': pages
    }
  }
```

High-Level Code Overview (createTable)

- The createTable function creates the table portion of the dashboard

- The first process that happens within the createTable function is the retrieval of json data from the passed in link. This json data is then parsed, sorted, and put into an array of objects.

- After this, the structure of the table and its various elements inside the dashboard is created using html and css. The table is then filled out by going through the previously mentioned array of objects. With this done, the whole process repeats itself on a certain interval.

Example of top level json


```
{"requests": [{"request_id": "a43f4da0-8a46-405d-815b-456f2ce32ab6", "did": "File List Provided in Request", "columns": null, "selection": "(call Select (call EventDataset
'bogus.root' 'Events') (lambda (list event) (attr event 'MET_pt')))", "tree-name": null, "image": "sslhep/servicex_func_adl_uproot_transformer:20220826-0614-stable",
"workers": 20, "result-destination": "object-store", "result-format": "root-file", "workflow-name": "selection_codegen", "generated-code-cm": "a43f4da0-8a46-405d-815b-
456f2ce32ab6-generated-source", "status": "Complete", "failure-info": null, "app-version": "1.0.0rc3", "code-gen-image": "sslhep/servicex_code_gen_func_adl_uproot:20220826-
0614-stable"}, {"request_id": "2eee32a4-f9ca-4a30-b162-40c89fc92ea5", "did": "File List Provided in Request", "columns": null, "selection": "(call Select (call EventDataset
'bogus.root' 'Events') (lambda (list event) (attr event 'MET_pt')))", "tree-name": null, "image": "sslhep/servicex_func_adl_uproot_transformer:20220826-0614-stable",
"workers": 20, "result-destination": "object-store", "result-format": "root-file", "workflow-name": "selection_codegen", "generated-code-cm": "2eee32a4-f9ca-4a30-b162-
40c89fc92ea5-generated-source", "status": "Complete", "failure-info": null, "app-version": "1.0.0rc3", "code-gen-image": "sslhep/servicex_code_gen_func_adl_uproot:20220826-
0614-stable"}, {"request_id": "40560f62-6bf4-4910-9155-2ff53071b286", "did": "File List Provided in Request", "columns": null, "selection": "(call Select (call EventDataset
'bogus.root' 'Events') (lambda (list event) (attr event 'Jet_pt')))", "tree-name": null, "image": "sslhep/servicex_func_adl_uproot_transformer:20220826-0614-stable",
"workers": 20, "result-destination": "object-store", "result-format": "root-file", "workflow-name": "selection_codegen", "generated-code-cm": "40560f62-6bf4-4910-9155-
2ff53071b286-generated-source", "status": "Complete", "failure-info": null, "app-version": "1.0.0rc3", "code-gen-image": "sslhep/servicex_code_gen_func_adl_uproot:20220826-
0614-stable"}, {"request_id": "16b0fdcf-12b4-4776-8320-19b23d3065a4", "did": "File List Provided in Request", "columns": null, "selection": "(call Select (call EventDataset
'bogus.root' 'Events') (lambda (list event) (attr event 'MET_pt')))", "tree-name": null, "image": "sslhep/servicex_func_adl_uproot_transformer:20220826-0614-stable",
"workers": 20, "result-destination": "object-store", "result-format": "root-file", "workflow-name": "selection_codegen", "generated-code-cm": "16b0fdcf-12b4-4776-8320-
19b23d3065a4-generated-source", "status": "Complete", "failure-info": null, "app-version": "1.0.0rc3", "code-gen-image": "sslhep/servicex_code_gen_func_adl_uproot:20220826-
0614-stable"}, {"request_id": "65f43548-00a9-4057-bb9e-7efa3cb22f21", "did": "cernopendata://cms:DYJetsToLL_M-50_TuneCP5_13TeV-amcatnloFXFX-pythia8/RunIIAutumn18NanoAODv7-
Nano02Apr2020_102X_upgrade2018_realistic_v21_ext2-v1/NANOAODSIM", "columns": null, "selection": "(call Select (call EventDataset 'bogus.root' 'Events') (lambda (list event)
(attr event 'MET_pt')))", "tree-name": null, "image": "sslhep/servicex_func_adl_uproot_transformer:20220826-0614-stable", "workers": 20, "result-destination": "object-store",
"result-format": "root-file", "workflow-name": "selection_codegen", "generated-code-cm": "65f43548-00a9-4057-bb9e-7efa3cb22f21-generated-source", "status": "Fatal", "failure-
info": "DID Request Failed for id 65f43548-00a9-4057-bb9e-7efa3cb22f21: CERNOpenData can only work with dataset numbers as names (e.g. 1507) - CERNOpenData can only work with
dataset numbers as names (e.g. 1507)", "app-version": "1.0.0rc3", "code-gen-image": "sslhep/servicex_code_gen_func_adl_uproot:20220826-0614-stable"}, {"request_id":
"2429695d-f00e-4ba6-8337-f81f88d6d753", "did": "cernopendata://cms:DYJetsToLL_M-50_TuneCP5_13TeV-amcatnloFXFX-pythia8/RunIIAutumn18NanoAODv7-
Nano02Apr2020_102X_upgrade2018_realistic_v21_ext2-v1/NANOAODSIM", "columns": null, "selection": "(call Select (call EventDataset 'bogus.root' 'Events') (lambda (list event)
(attr event 'MET_pt')))", "tree-name": null, "image": "sslhep/servicex_func_adl_uproot_transformer:20220826-0614-stable", "workers": 20, "result-destination": "object-store",
"result-format": "root-file", "workflow-name": "selection_codegen", "generated-code-cm": "2429695d-f00e-4ba6-8337-f81f88d6d753-generated-source", "status": "Fatal", "failure-
info": "DID Request Failed for id 2429695d-f00e-4ba6-8337-f81f88d6d753: CERNOpenData can only work with dataset numbers as names (e.g. 1507) - CERNOpenData can only work with
dataset numbers as names (e.g. 1507)", "app-version": "1.0.0rc3", "code-gen-image": "sslhep/servicex_code_gen_func_adl_uproot:20220826-0614-stable"}, {"request_id":
"fee5861e-5c9a-427c-bd96-f9717a0980ea", "did": "File List Provided in Request", "columns": null, "selection": "(call Select (call EventDataset 'bogus.root' 'events') (lambda
(list event) (attr event 'jet_pt')))", "tree-name": null, "image": "sslhep/servicex_func_adl_uproot_transformer:20220826-0614-stable", "workers": 20, "result-destination":
"object-store", "result-format": "root-file", "workflow-name": "selection_codegen", "generated-code-cm": "fee5861e-5c9a-427c-bd96-f9717a0980ea-generated-source", "status":
"Complete", "failure-info": null, "app-version": "1.0.0rc3", "code-gen-image": "sslhep/servicex_code_gen_func_adl_uproot:20220826-0614-stable"}, {"request_id": "1f0840e0-
```

CreateTable Function

```
async function createTable(SERVICEX_URL: string){ //Function that creates instance of dashboard
  //Code for retrieving live json result

  const start = Date.now(); //For run time testing purposes
  let response;
  try{ //Testing to see if fetch request to current instance works
    response = await fetch(SERVICEX_URL + 'servicex/transformation');
  } catch(error){ //If not alert an error
    alert('An error has occured: ' + error + '. This is most likely a CORS header issue with ' + SERVICEX_URL + '. Do cltr+shift+j to view the developer console for
    return;
  }
  let arr_1 = []; //Overall array for unsorted data
  if(response != null){ //If response is not null, proceed with the filling of arr_1. Else, exit.
    const data = await response.json(); //Getting json response for all requests
    let requests = data.requests;

    for(var i = requests.length - 1; i > -1; i--){ //creating array containing objects for all of requests
      const obj = {
          request_id: '',
          status: '',
          title_link: '',
```

Conclusion and Future Outlook

- Overall, the goal of creating a ServiceX dashboard plug-in in Jupyterlab was successful

- There are still areas where the plug-in can be improved and there are features that still need to be added

- In spite of this, the servicex-labextension looks to become a component in the future of ServiceX

Thank you!

- I would I like to personally thank all my mentors: Alex, Oksana, Ben, Gordon, and Mason for their help and support during this project. This couldn't have happened without your support!

- Github link: https://github.com/ssl-hep/servicex-labextension

- Pypi link: https://pypi.org/project/servicex-dashboard/