

Tutorial: Design Patterns for Apps and Workflows

Contents

Introduction	2
Learning Objectives	2
Download and Install the pfda CLI	3
Windows	3
Linux.....	5
First App: <i>docker_pull_and_save</i>	5
Create the docker_pull_and_save App	6
Run the docker_pull_and_save App	7
Second App: <i>worker_layout_inspection</i>	9
Create an asset with code and data	9
Layout your asset directories and files	9
Create the asset using the CLI	11
Manually deploying an asset tar file	11
Create the App and Specify the I/O Spec	12
Specify the VM Environment	13
Specify the Script	14
Specify the Readme	15
Run the app	16
Inspect the execution logs and the inspection results file	17
Utility Apps: <i>untar_files</i> , <i>tar_files_from_manifest</i>	21
<i>tar_files_from_manifest</i> : Tar a manifest of fileIDs	22
<i>untar_files</i> : Untar archive to files	24
First Workflow: <i>simple_workflow</i>	27
Add the workflow stages	27
Configure the workflow stages	28
Run the workflow	31
Second Workflow: <i>workflow_from_wdl</i>	32

Database App: <i>worker_database</i>	34
Create the Database	34
Fork <i>workstation_layout_inspection</i> to <i>workstation_database</i> app	36
Specify the I/O Spec	36
Specify the VM Environment	37
Specify the Script	37
Specify the Readme	38
Run the app	39
Inspect the execution logs	39

Introduction

This hands-on tutorial presents design patterns for collaborative bioinformatics using precisionFDA Apps and Workflows. There is a considerable range of design patterns for deploying code, marshalling data, and producing results, within precisionFDA's file and transient worker-driven (i.e. batch, non-interactive) application framework. This course will present this framework, including:

- Apps, their input/out specifications for files and other data types, virtual environment specification, and scripting
- Assets (i.e. tar archives) that can be bundled with assets and overlaid onto the worker's root volume
- Incorporation of Docker images into apps from file inputs and from assets
- Workflow creation using the web interface and WDL

Through the development of the tutorial artifacts, precisionFDA's powerful capabilities for secure collaborative development of bioinformatics tools, and sharing of -omics and RWD, are clearly demonstrated, and users will be empowered to develop their own bioinformatics use cases.

Learning Objectives

Through this hands-on tutorial you will:

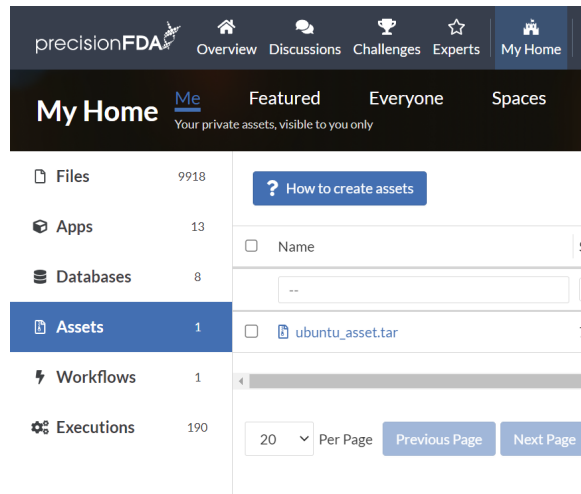
- Install the precisionFDA command line utility and use it upload and download files.
- Explore the multiple types of app inputs and outputs, assets, and their presentation to the app script as shell variables and files.
- Build a bioinformatics app from a biocontainers Docker image.
- Run the latest Ubuntu OS as a Docker image in an app.
- Use the precisionFDA Command Line Interface asset to process an arbitrary number of inputs or produce an arbitrary number of outputs to make up for the lack of an array data type.
- Create a workflow through the web interface.
- Create a workflow through by importing WDL.

- Access a database cluster from an app.

Download and Install the pfda CLI

The precisionFDA CLI is useful for programmatic uploading and downloading of files from Windows, MacOS, and Linux clients (e.g. your laptop) to and from precisionFDA. Installation and testing for Windows and Linux OS are described below.

Under My Home Assets, click on the How to create assets button to find links to the precisionFDA CLI, and the button to generate the temporary authorization key that you'll use with the CLI.



STEP 3 Download the precisionFDA CLI



STEP 4 Get your Authorization Key

Generate Authorization Key

Your authorization key

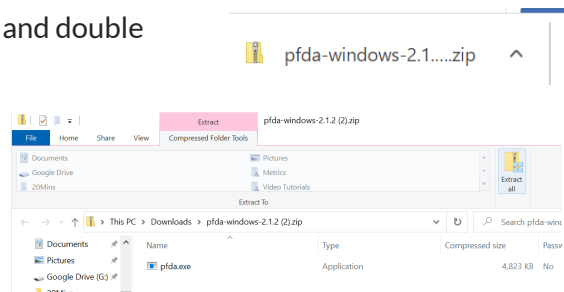
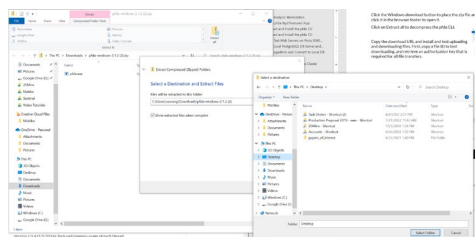
This key is only valid for the next 24 hours. Please keep it safe:

```
Yy9reHJEVjVoVnR0eTRLQ3h6MzRhMVZRdjJyVFMvenFEZDZlYk1EbG1PdGJYwXAwU1FNO  
HV6b2ZNaJv3wXBkZmZkVMzYjI2MVoyd1h3U1ZjTWc2bWFnUjc2MWZoZW90WHB3M0MxRk  
RNeEjV1NsYkFZUTdTaXd1OUFnWtdLQ25mWEJVZVVeJ1Kd3NsQTRvukVBK1NxYj1PL3A  
1NGMyZ096Wd5MzRjamI1UE5VeXpVjZxNmtadDB6aFZtcUFMLS10ZTkvcjB3LzBLNGMw  
TGh4aTFUOWZ3PT0=-88ea73b01fc4f0817d668ef07780a244169edef8
```

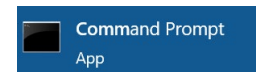
Windows

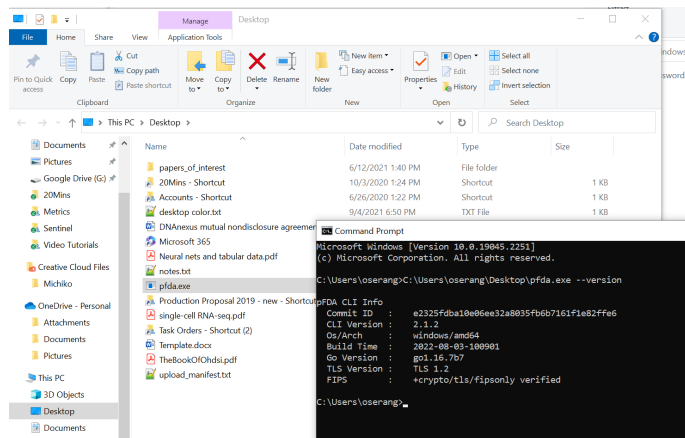
Click the Windows download button to place the zip file and double click it in the browser footer to open it.

Click on Extract all to decompress the pfda CLI and browse to select the Desktop as the destination for the pfda.exe file.

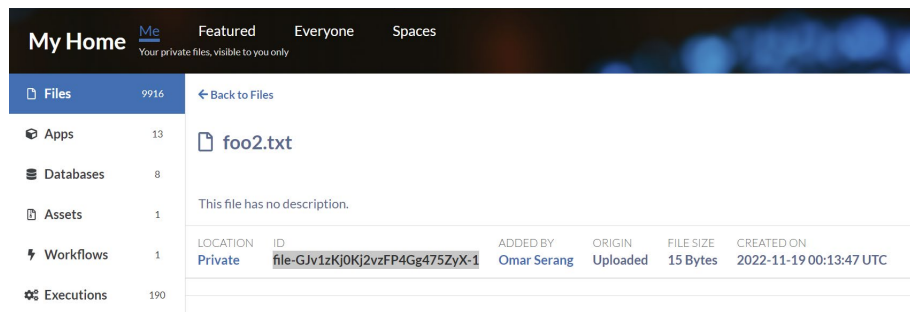


Using the Windows start menu, bring up a Command Prompt window with the pfda.exe file visible in a file explorer side-by-side. Drag pfda.exe onto the Command Prompt window to expand the full path the executable and add -version and hit return.





First, copy a file ID to test downloading, and retrieve an authorization key that is required for all file transfers.



:: Provide the auth key

:: Download the selected file

```
C:\Users\osering\Desktop\pfda.exe download -key <key> -file-id file-GJv1zKj0Kj2vzFP4Gg475ZyX-1
```

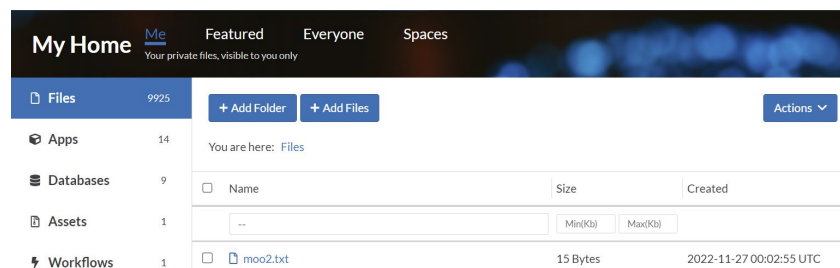
:: Copy it to a new file and upload it

```
copy foo2.txt moo2.txt
```

```
C:\Users\osering\Desktop\pfda.exe upload-file -file moo2.txt
```

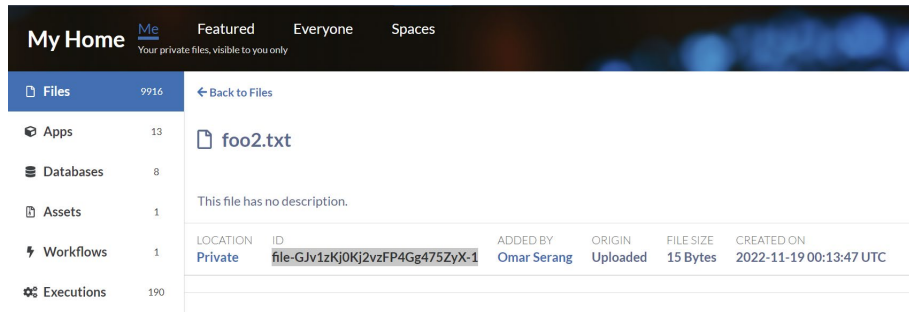
```
dir *.txt
```

```
11/26/2022 04:26 PM 15 foo2.txt
11/26/2022 04:26 PM 15 moo2.txt
```



Linux

Copy the download URL and install and test uploading and downloading files. First, copy a file ID to test downloading, and retrieve an authorization key that is required for all file transfers.



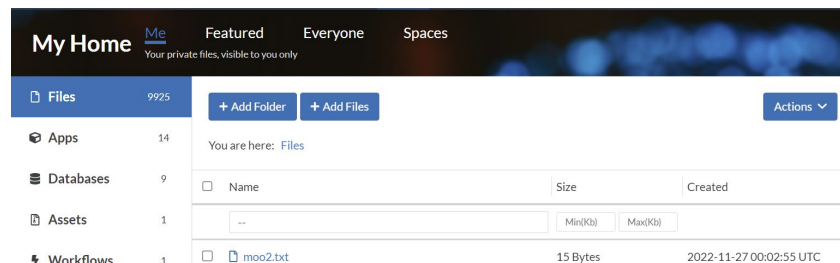
```
-- Install pfda CLI
wget https://pfda-production-static-files.s3.amazonaws.com/cli/pfda-
linux-2.2.tar.gz
tar xf pfda-linux-2.2.tar.gz
mv pfda /usr/bin/
pfda --version

-- Provide the auth key
key="...."

-- Download the selected file
pfda download -key $key -file-id file-GJv1zKj0Kj2vzFP4Gg475ZyX-1

-- Copy it to a new file and upload it
cp foo2.txt moo2.txt
pfda upload-file -key $key -file moo2.txt

ls *.txt
foo2.txt  moo2.txt
```

First App: *docker_pull_and_save*

The first app demonstrates the use of string input variables, allowing the app to access the internet, how to use the emit shell function to output a named file. This app pulls a docker image and saves it to a .tar file suitable for use in precisionFDA apps. We'll create three image files in our Files:

	docker_source	docker_image_filename
1	ubuntu:latest	ubuntu_latest.tar
2	biocontainers/samtools:v1.9-4-deb_cv1	samtools_biocontainers.tar
3	postgres:13.4-buster	postgres_13.4-buster.tar

Create the docker_pull_and_save App

From My Home / Apps, click on Create App to create the *docker_pull_and_save* app. In the I/O Spec tab, add the input and output fields.

Create App Pull a specified docker image and save it to a tar

Import from *.cwl file Import from *.wdl file

APP NAME: docker_pull_and_save TITLE: Pull a specified docker image and save it to a tar + Create

I/O SPEC VM ENVIRONMENT SCRIPT README
Configure Input & Output Fields Configure your resources Write your shell script Describe your app

Need help? [Learn more about app inputs and outputs](#)

INPUTS + Add Input Field

Class	Name	Label	Help text	Default Value	Choices Limit what values can be set	Optional?
string	docker_source	Docker pull source name	Enter help text for this field	ubuntu:latest	Optional comma separated :	☐
string	docker_image_filename	Filename for saved docker	Enter help text for this field	ubuntu_latest.tar	Optional comma separated :	☐

OUTPUTS + Add Output Field

Class	Name	Label	Help text	Optional?
file	docker_image_file	Docker image file	Enter help text for this output	☐

Select the VM Environment tab, enable internet access, and select Baseline 2 as the default instance type.

Create App Pull a specified docker image and save it to a tar

Import from *.cwl file Import from *.wdl file

APP NAME: docker_pull_and_save TITLE: Pull a specified docker image and save it to a tar + Create

I/O SPEC **VM ENVIRONMENT** SCRIPT README
Configure Input & Output Fields Configure your resources Write your shell script Describe your app

Need help? [Learn more about the virtual machine environment](#)

Internet Access? ☒ Enable access

Default Instance [See full list](#) Baseline 2 0.286\$/hour

Assets [Learn more](#) Select assets... Manage your assets

Ubuntu Packages Package name + TIP: Find packages within the distribution using Ubuntu Package Search

Select the Script tab and enter the following shell script:

```
set -euxo pipefail
sudo docker pull "$docker_source"
sudo docker save "$docker_source" > "$docker_image_filename"
```

```
emit "docker_image_file" "$docker_image_filename"
```

The set -euxo pipefail set is advisable at the start of all your app scripts. The docker source string is resolved and used to pull the image and to save it to the docker image filename. Lastly the resulting image is saved as a precisionFDA file with the specified image filename.

Create App Pull a specified docker image and save it to a tar

Import from *.cwl file Import from *.wdl file

APP NAME TITLE

I/O SPEC **VM ENVIRONMENT** **</> SCRIPT** **README**
Configure Input & Output Fields Configure your resources Write your shell script Describe your app

Need help? [Learn more about the app script](#)

```
1 set -euxo pipefail
2 sudo docker pull "$docker_source"
3 sudo docker save "$docker_source" > "$docker_image_filename"
4 emit "docker_image_file" "$docker_image_filename"
```

Select the Readme tab and describe your app, then hit the Create button to create your app.

Create App Pull a specified docker image and save it to a tar

Import from *.cwl file Import from *.wdl file

APP NAME TITLE

I/O SPEC **VM ENVIRONMENT** **</> SCRIPT** **README**
Configure Input & Output Fields Configure your resources Write your shell script Describe your app

Need help? [Learn how to format Markdown](#)

Edit README Preview

```
1 Pulls a docker image and saves into a .tar image suitable for docker load -i.
```

Run the docker_pull_and_save App

Run this app three times with the inputs listed above. You don't have to wait for one to finish to start the others as they will run in parallel.

Run Pull a specified docker image and save it to a tar

Need help? [Learn more about running an app](#)

CONFIGURE

Job Name: Pull a specified docker image and save it to a tar

Instance Type: Baseline 2 0.2865/hour

Execution Cost Limit (\$): 20

INPUTS

Docker pull source name:

Filename for saved docker tar:

My Home **Me** Featured Everyone Spaces

Your private executions, visible to you only

Files 9930

Apps 15

Databases 9

Assets 1

Workflows 1

Executions 197

Back to Executions

idle Pull a specified docker image and save it to a tar

Re-Run Execution Actions

LOCATION	APP	LAUNCHED BY	CREATED ON	INSTANCE TYPE
Private	Pull a specified docker image and save it to a tar	Omar Serang	2022-11-27 01:50:33 UTC	baseline-2

DURATION	COST	APP REVISION
N/A	\$	2

Select one of the completed executions My Home / Executions for the app and View Logs using the Action dropdown menu and we can see the desired actions took place and the image .tar files appear in My Home / Files.

precisionFDA Overview Discussions Challenges Experts My Home Notes Comparisons Spaces GSRS Support Get Started Omar Serang

My Home **Me** Featured Everyone Spaces

Your private executions, visible to you only

Files 9930

Apps 15

Databases 9

Assets 1

Workflows 1

Executions 197

State	Execution Name	App Title	Instance Type
terminated	Pull a specified docker image and save it to a tar	Pull a specified docker image baseline-2	
done	Pull a specified docker image and save it to a tar	Pull a specified docker image baseline-2	1 minutes 55 seconds
done	Pull a specified docker image and save it to a tar	Pull a specified docker image baseline-2	2 minutes 0 seconds
done	samtools_demo	samtools_demo baseline-8	1 minutes 35 seconds

Actions

- View Logs
- Terminate
- Track
- Copy to space
- Make Public
- Attach to...
- Comments
- Edit tags

```
Downloading files using 2 threads++ set -euxo pipefail
++ sudo docker pull postgres:13.4-buster
13.4-buster: Pulling from library/postgres
```

```
.
.
.
```

```
Status: Downloaded newer image for postgres:13.4-buster
++ sudo docker save postgres:13.4-buster
++ emit docker_image_file postgres_13.4-buster.tar
```

The screenshot shows the 'My Home' section of the precisionFDA interface. The sidebar on the left lists categories: Files (9930), Apps (15), Databases (9), Assets (1), Workflows (1), and Executions (197). The main content area shows a table of files with columns for Name, Size, Created, and Origin. The files listed are postgres_13.4-buster.tar (307 MB), samtools_biocontainers.tar (654 MB), and ubuntu_latest.tar (76.6 MB). Each file has a 'Pull a specified doc' button next to it.

Name	Size	Created	Origin
postgres_13.4-buster.tar	307 MB	2022-11-27 01:24:23 UTC	Pull a specified doc
samtools_biocontainers.tar	654 MB	2022-11-27 01:16:21 UTC	Pull a specified doc
ubuntu_latest.tar	76.6 MB	2022-11-27 01:05:13 UTC	Pull a specified doc

Second App: *worker_layout_inspection*

Since assets are such a great app developer convenience, our second app, *worker_layout_inspection*, will illustrate the use of assets and inputs for presenting code and data to the app. First let's create some assets that will be incorporated into the app. Additionally, since Docker is also such a great app developer tool, the incorporation of Docker images into the app by packaging them in an asset or providing them as an input file, are both illustrated.

Create an asset with code and data

Layout your asset directories and files

The file system structure that you layout in your asset will be overlaid on the worker / (root) mount when the app is run. If you've include code in the asset's /usr/bin, it will be available to your app running on the worker. The app framework uses /work as the directory to present input files to apps so any files placed in the asset's /work directory will be available with other input files, though you could place files in whatever asset directory structure you want.

We going to construct our asset in a Linux shell, downloading from precisionFDA the following for inclusion in the asset:

- ubuntu_latest.tar (file-GK1FP9j05gK8z93Y1xGQpf5B-1)
- countries.txt (file-GK1F6j80Kj2XbJzx29f25y42-1)

Create your asset directory structure.

```
apt install tree
mkdir -p ~/fakeroot/work
mkdir -p ~/fakeroot/usr/bin
tree ~/fakeroot/
/home/dnanexus/fakeroot/
|
|— usr
|   |
|   |— bin
|   |
|   |— work
```

Create a shell script to install tree and run it, then move the script into the asset directory structure.

```
cat > tree_script.sh
    sudo apt install tree
    tree $1
```

CTRL-D

```

chmod ugo+x tree_script.sh
./tree_script.sh ~
/home/dnanexus
├── EHR_Sample_backup_nodbccreate_postgres.sql
├── datafiles
│   ├── manifest.txt
│   ├── observations.txt
│   └── patients.txt
└── db_backups
mv tree_script.sh ~/fakeroot/usr/bin

```

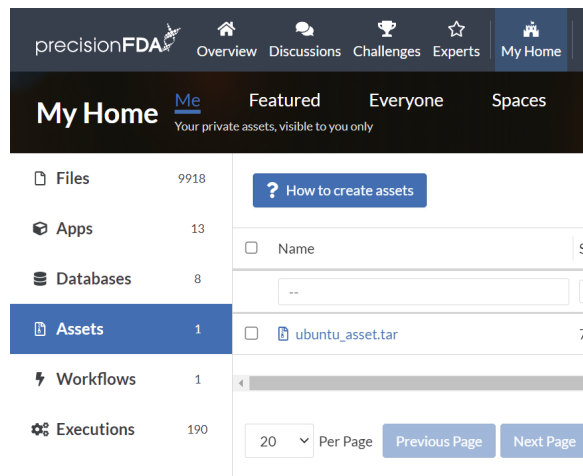
Create a Readme file as required for the asset. This doesn't need to reside in the asset /fakeroot directory structure.

```

echo "Assets for the worker layout inspection tutorial app" >
~/readme.txt

```

Download the Docker and data files and move them into the asset directory structure. Under My Home Assets, click on the How to create assets button to find links to the precisionFDA CLI, and the button to generate the temporary authorization key that you'll use with the CLI.



STEP 4 Get your Authorization Key

[Generate Authorization Key](#)

Your authorization key

This key is only valid for the next 24 hours. Please keep it safe:

```

Yy9reHJEVjVoVnROeTRLQ3h6MzRhMVZRdjJyVFMvenFEZDZlYk1EbG1PdG3YwXAwU1FNO
HV6b2ZNaJv3wXbkZmZkVVMzYjI2Mvoyd1h3U1ZjTwc2bWfHujc2MWZoZW9OWHB3M0MxRk
RNeE3jV1NsYkFZUTdTxd1OUFnWtdLQ25mWEJvZVVFfeJlKd3NsQTRvWkVBK1NxYj1PL3A
1NGMyZ096WHD5MzRjamI1UE5VeXpVjZxNmtadDB6aFZtcUFMLS10ZTkvcjB3LzBLNGMw
TGh4aTFUOWZ3PT0=-88ea73b01fc4f0817d668ef07780a244169edef8

```

```
key="..."
```

```

pfda download -key $key -file-id file-GK1FP9j05gK8z93Y1xGQpf5B-1
pfda download -key $key -file-id file-GK1F6j80Kj2XbJzx29f25y42-1
ls *.tar *.txt
countries.txt  foo2.txt  moo2.txt  ubuntu_latest.tar

```

```

mv countries.txt postgres_13.4-buster.tar ubuntu_latest.tar
~/fakeroot/work
tree ~/fakeroot
/home/dnanexus/fakeroot
├── usr
└── bin

```

```

├── tree_script.sh
├── work
│   ├── countries.txt
│   └── ubuntu_latest.tar

```

Create the asset using the CLI

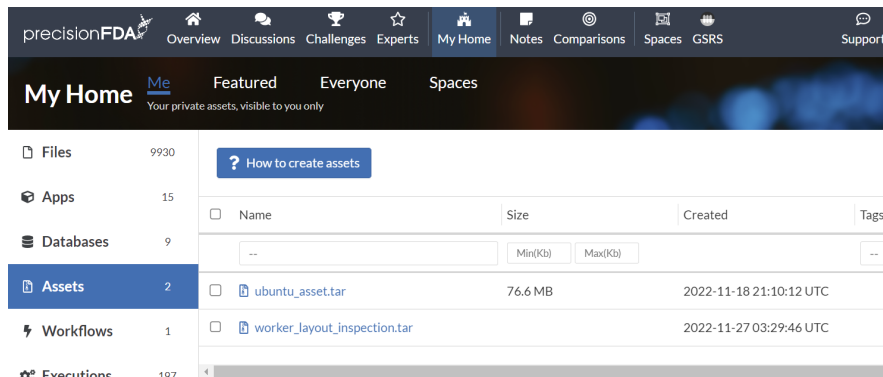
Now that the asset contents have been laid out, creating the asset on precisionFDA is a straightforward process using the precision FDA CLI.

key="..."

```

pfda upload-asset --key $key --name worker_layout_inspection.tar --root
~/fakeroot --readme ~/readme.txt
>> Archiving asset...
>> Finalizing asset...
>> Done! Access your asset at
https://precision.fda.gov/home/assets/file-GK1JJB00Kj2fkz464yxB5zY2-1

```



Manually deploying an asset tar file

While this workflow should not be required, it is worth knowing to understand how assets work. Upload the asset tarball (e.g. *worker_layout_inspection.tar*) as a file, then in your app, add a file to the I/O Spec (e.g. "asset_tarball") and select this tarball as the default. Add the following line to the Script, right after the `set -euxo pipefail` command:

```
tar zxvf ${asset_tarball_path} -C / --strip-components=1 --no-same-owner
```

Everything that was installed in the `fake_root/` in the tarball will be placed into the root directory of the worker, including any executable you may have. For example,

```
fake_root/usr/local/bin/sambamba
```

from the `asset_tarball` input, will be available in:

```
/usr/local/bin/sambamba
```

after the tar command above is run in the app script.

Create the App and Specify the I/O Spec

In My Home / Apps, click the Create App button to create the *worker_layout_inspection* app. In the I/O Spec tab, add the input and output fields.

APP NAME **worker_layout_inspection**
TITLE
UBUNTU RELEASE
Revision 24 → [Save Revision 25](#)

I/O SPEC
Configure Input & Output Fields

VM ENVIRONMENT
Configure your resources

<> SCRIPT
Write your shell script

README
Describe your app

App revisions are private

Need help? [Learn more about app inputs and outputs](#)

INPUTS
Add Input Field

Class	Name	Label	Help text	Default Value	Choices Limit what values can be set	Optional?
string	string_input	example string input	Enter help text for this:	this is a string	Optional comma separat	☐
int	integer_input	example integer input	Enter help text for this:	54321	Optional comma separat	☐
float	float_input	example float input	Enter help text for this:	1.234	Optional comma separat	☐
boolean	boolean_input	example boolean input	Enter help text for this:	<input type="radio" value="True"/> True <input checked="" type="radio" value="False"/> False		☐
file	observation_data	Delimited observation	Enter help text for this:	observations.txt		☐
file	patient_data	Delimited patient dat	Enter help text for this:	patients.txt		☐
file	samtools_docker_imag	Docker image for Sam	Enter help text for this:	samtools_biocontainers.tar		☐
string	inspection_results_file	Inspection results filer	Enter help text for this:	inspection_results.txt	Optional comma separat	☐
file	postgres_docker_imag	Docker image for post	Enter help text for this:	postgres_13.4-buster.tar		☐
string	url_to_fetch	URL to pass to next ap	Enter help text for this:	https://pfda-production-static-files.s3.amazor	Optional comma separat	☐
file	worker_layout_inspec	Bash script for the wo	Enter help text for this:	workstation_layout_inspection.bash		☐

OUTPUTS
Add Output Field

Class	Name	Label	Help text	Optional?
file	inspection_results	Worker layout and variable inspec	Enter help text for this output	☐
string	url_to_fetch	URL to pass to next app in workflo	Enter help text for this output	☐

Class	Input Name	Label	Default Value
string	string_input	example string input	this is a string
int	integer_input	example integer input	54321
float	float_input	example float input	1.234
boolean	boolean_input	example boolean input	False
file	observation_data	Delimited observation data	observations.txt

file	patient_data	Delimited patient data	patients.txt
file	samtools_docker_image	Docker image for Samtools	samtools_biocontainers.tar
file	postgres_docker_image	Docker image for postgres server	postgres_13.4-buster.tar
string	inspection_results_filename	Inspection results filename	inspection_results.txt
string	url_to_fetch	URL to pass to next app in workflow	https://pfda-production-static-files.s3.amazonaws.com/cli/pfda-linux-2.2.tar.gz
Class	Output Name	Label	
file	Inspection_results	Worker layout and variable inspection results	
string	url_to_fetch	URL to pass to next app in workflow	

Specify the VM Environment

In the VM Environment tab, enable internet access, select default instance Baseline 2, and add the following assets: worker_layout_inspection, pfda_cli_2.2, and ubuntu_asset.

APP NAME **worker_layout_inspection** TITLE UBUNTU RELEASE

[I/O SPEC](#)
Configure Input & Output Fields

[VM ENVIRONMENT](#)
Configure your resources

[SCRIPT](#)
Write your shell script

[README](#)
Describe your app

Need help? [Learn more about the virtual machine environment](#)

Internet Access? ☒ Enable access

Default Instance [See full list](#)

Assets [Learn more](#)

[Select assets...](#)

[Manage your assets](#)

[pfda_cli_2.1.2](#)
[ubuntu_asset](#)
[worker_layout_inspection2](#)

Ubuntu Packages

[+](#)

TIP: Find packages within the distribution using [Ubuntu Package Search](#)

Specify the Script

Add the following code to the script tab:

```
set -euxo pipefail

# Append the worker OS information to the specified inspection results
# file.
cat /etc/os-release | tee -a "$inspection_results_filename"

# Install tree
sudo apt update
sudo apt install tree

# Inspect the scalar input variables.
echo "string_input" "$string_input"
echo "integer_input" "$integer_input"
echo "float_input" "$float_input"
echo "url_to_fetch" "$url_to_fetch"
echo ""

# Inspect the file input variables and the different
# operators for accessing them on the worker FS.
echo "patient_data" "$patient_data"
echo "patient_data_name" "$patient_data_name"
echo "patient_data_path" "$patient_data_path"
echo "patient_data_prefix" "$patient_data_prefix"
echo ""
echo "observation_data" "$observation_data"
```

```
echo "observation_data_name" "$observation_data_name"
echo "observation_data_path" "$observation_data_path"
echo "observation_data_prefix" "$observation_data_prefix"
echo ""
echo "samtools_docker_image" "$samtools_docker_image"
echo "samtools_docker_image_name" "$samtools_docker_image_name"
echo "samtools_docker_image_path" "$samtools_docker_image_path"
echo "samtools_docker_image_prefix" "$samtools_docker_image_prefix"

# Use the pfda CLI that was loaded with the pfda_cli_2.2 asset.
ls -al /usr/bin/pfda*
pfda --version

# Use the simple script that was loaded with the
worker_layout_inspection asset.
ls -al /usr/bin/tree_script.sh
tree_script.sh /work

# Using the Docker image that was loaded with the ubuntu_asset asset,
# we can run an up-to-date Ubuntu OS in a docker container on the
worker.
# Append the container OS information to the specified inspection
results file.
docker load -i /ubuntu_latest.tar
docker run --rm -v /work:/work -w /work ubuntu:latest cat /etc/os-
release | tee -a "$inspection_results_filename"

# Using the Docker image that was provided as an input file,
# we can run samtools. Add inputs and outputs to this script and
# pass them to the samtools invocation to create your own
# samtools app.
docker load -i "$samtools_docker_image_path"
docker run --rm biocontainers/samtools:v1.9-4-deb_cv1 samtools --version

docker images
docker ps

# Pass the URL to fetch input to the same output variable to pass
# to the next app in the tutorial workflow (i.e. url fetcher).
emit "url_to_fetch" "$url_to_fetch"

# Save the inspection results file with specified name.
emit "inspection_results" "$inspection_results_filename"
```

Specify the Readme

Add the following in the Readme tab:

```
Produce an inspection report of the worker filesystem and file inputs
from the context of the app.
Also provide the multiple representations available for file inputs.
Echo the non-file input variables.
```

Run the app

In My Home / Apps, select the *worker_layout_inspection* app and click Run App to launch the latest version of the app with all default inputs.

Run Worker layout inspection tutorial app

Run App

Need help? Learn more about running an app

CONFIGURE

Job Name

Worker layout inspection tutorial app

Instance Type

Baseline 2 0.286\$/hour

Execution Cost Limit (\$)

10

INPUTS

example string input

this is a string

example integer input

54321

example float input

1.234

example boolean input

True

False (default)

Delimited observation data

observations.txt

Delimited patient data

patients.txt

Docker image for Samtools

samtools_biocontainers.tar

Inspection results filename

inspection_results.txt

Docker image for postgres server

postgres_13.4-buster.tar

URL to pass to next app in workflow

https://pfda-production-static-files.s3.amazonaws.com/cli/pfda-linux-2.1.2.tar.gz

My Home

Featured

Everyone

Spaces

Files 9937

Apps 16

Databases 9

Assets 4

Workflows 2

Executions 219

Back to Executions

idle Worker layout inspection tutorial app

Re-Run Execution

Actions

LOCATION	APP	LAUNCHED BY	CREATED ON	INSTANCE TYPE
Private	Worker layout inspection tutorial app	Omar Serang	2022-11-28 23:18:01 UTC	baseline-2
DURATION	COST	APP REVISION		
N/A	\$	25		

Refresh the execution status using the



button until the job is first idle, runnable, running, and done, (or failed).

LOCATION	APP	LAUNCHED BY	CREATED ON	INSTANCE TYPE
Private	Worker layout inspection tutorial app	Omar Serang	2022-11-28 23:46:40 UTC	baseline-2

DURATION	COST	APP REVISION
N/A	\$	29

Inputs and Outputs

INPUTS	OUTPUTS
example string input	this is a string
example integer input	54321
example float input	1.234
example boolean input	false
Delimited observation data	observations.txt
Delimited patient data	patients.txt
Docker image for Samtools	samtools_biocontainers.tar
Inspection results filename	inspection_results.txt
Docker image for postgres server	postgres_13.4-buster.tar
URL to pass to next app in workflow	https://pfda-production-static-files.s3.amazonaws.com/cli/pfda-linux-2.1.2.tar.gz

Inspect the execution logs and the inspection results file

Note that the execution of the app took just over a minute and produced inspection_results.txt file and URL string outputs.

LOCATION	APP	LAUNCHED BY	CREATED ON	INSTANCE TYPE
Private	Worker layout inspection tutorial app	Omar Serang	2022-11-28 23:32:47 UTC	baseline-2

DURATION	COST	APP REVISION
1 minutes 8 seconds	\$0.0	28

- View Logs
- Terminate
- Track
- Copy to space
- Make Public
- Attach to...
- Comments
- Edit tags

Files 9931
Apps 16
Databases 9
Assets 4
Workflows 2
Executions 223

Back to Executions

done

Worker layout inspection tutorial app

Re-Run Execution

Actions

LOCATION	APP	LAUNCHED BY	CREATED ON	INSTANCE TYPE
Private	Worker layout inspection tutorial app	Omar Serang	2022-11-28 23:46:40 UTC	baseline-2
DURATION	COST	APP REVISION		
2 minutes 15 seconds	\$0.01	29		

Inputs and Outputs

INPUTS	OUTPUTS
example string input this is a string	Worker layout and variable inspection results inspection_results.txt
example integer input 54321	
example float input 1.234	
example boolean input false	URL to pass to next app in workflow https://pfda-production-static-files.s3.amazonaws.com/cli/pfda-linux-2.1.2.tar.gz
Delimited observation data observations.txt	
Delimited patient data patients.txt	
Docker image for Samtools samtools_biocontainers.tar	
Inspection results filename inspection_results.txt	
Docker image for postgres server postgres_13.4-buster.tar	
URL to pass to next app in workflow https://pfda-production-static-files.s3.amazonaws.com/cli/pfda-linux-2.1.2.tar.gz	

Select View Logs from the the Actions dropdown menu. Let's look at a condensed and annotated version of the log output to understand what our app just did.

Worker initialization

Logging initialized (priority)

CPU: 20% (2 cores) * Memory: 779/3720MB * Storage: 32GB free * Net: 0 ↓ / 0 ↑ MBps

umount: /proc/diskstats: not mounted

dxpy/0.331.0 (Linux-5.4.0-1088-aws-x86_64-with-Ubuntu-16.04-xenial)

/usr/sbin/rsyslogd already running.

bash running (job ID job-GK2ZQj00Kj2z9q76KbZbkG3G)

Fetch the app's assets.

Fetching asset ubuntu_asset.tar (file-GJqz9J00Kj2zZQPGKVZBg436)

Fetching asset pfda_cli_2.2.tar (file-GJv0kFQ0Kj2YzFb86g4B8px5)

Fetching asset worker_layout_inspection2.tar (file-GK1xxbQ0Kj2gBZjxF244F21k)

Download the input files.

downloading file: file-GK1F6jj0Kj2gyF8jG8jPjGpV to filesystem: /work/in/patient_data/patients.txt

downloading file: file-GK1F6jQ0Kj2v90jxPV0ZBQ5G to filesystem: /work/in/observation_data/observations.txt

downloading file: file-GK1FXK80pyBj68X3K6jVX55b to filesystem: /work/in/samtools_docker_image/samtools_biocontainers.tar

downloading file: file-GK1Fg68072kf3ZXYGJvxPGPj to filesystem: /work/in/postgres_docker_image/postgres_13.4-buster.tar

The worker's OS release information.

```
cat /etc/os-release
NAME="Ubuntu"
VERSION="16.04.7 LTS (Xenial Xerus)"
ID=ubuntu
ID_LIKE=debian
PRETTY_NAME="Ubuntu 16.04.7 LTS"
VERSION_ID="16.04"
HOME_URL="http://www.ubuntu.com/"
SUPPORT_URL="http://help.ubuntu.com/"
BUG_REPORT_URL="http://bugs.launchpad.net/ubuntu/"
VERSION_CODENAME=xenial
UBUNTU_CODENAME=xenial
```

Install tree

```
sudo apt update
apt install tree
Unpacking tree (1.7.0-3) ...
Processing triggers for man-db (2.7.5-1) ...
Setting up tree (1.7.0-3) ...
```

Resolution of each scalar input variable

```
string_input this is a string
integer_input 54321
float_input 1.234
url_to_fetch https://pfda-production-static-
files.s3.amazonaws.com/cli/pfda-linux-2.2.tar.gz
```

**# Resolution of each file input variable into the four
types of references available**

```
patient_data file-GK1F6jj0Kj2gyF8jG8jPjGpV
patient_data_name patients.txt
patient_data_path /work/in/patient_data/patients.txt
patient_data_prefix patients
```

```
observation_data file-GK1F6jQ0Kj2v90jxPV0ZBQ5G
observation_data_name observations.txt
observation_data_path /work/in/observation_data/observations.txt
observation_data_prefix observations
```

```
samtools_docker_image file-GK1FXK80pyBj68X3K6jVX55b
samtools_docker_image_name samtools_biocontainers.tar
samtools_docker_image_path
/work/in/samtools_docker_image/samtools_biocontainers.tar
samtools_docker_image_prefix samtools_biocontainers
```

**# View the pfda CLI that was installed with the pfda CLI asset and run
it.**

```
ls -al /usr/bin/pfda
-rwxr-xr-x 1 root root 11810776 Aug  3 08:07 /usr/bin/pfda
```

```
pfda --version
```

pFDA CLI Info

```
Commit ID      : e2325fdb10e06ee32a8035fb6b7161f1e82ffe6
CLI Version    : 2.2
Os/Arch       : linux/amd64
Build Time    : 2022-08-03-100606
Go Version     : go1.16.7b7
TLS Version    : TLS 1.2
FIPS          : +crypto/tls/fipsonly verified
```

View the tree script that was installed with the workstation asset and run it.

Note the countries.txt file in /work as it was packaged in the worker_layout_inspection asset.

Note the directory layout of the file input types.

```
ls -al /usr/bin/tree_script.sh
-rwxr-xr-x 1 root root 30 Nov 27 21:32 /usr/bin/tree_script.sh
```

```
tree_script.sh /work
/work
|___ countries.txt
|___ in
|   |___ observation_data
|   |   |___ observations.txt
|   |___ patient_data
|   |   |___ patients.txt
|   |___ postgres_docker_image
|   |   |___ postgres_13.4-buster.tar
|   |___ samtools_docker_image
|   |   |___ samtools_biocontainers.tar
|___ inspection_results.txt
```

Load the docker image that was installed with ubuntu asset and run it, # presenting the container OS release information.

```
docker load -i /ubuntu_latest.tar
```

5 directories, 6 files

Loaded image: ubuntu:latest

```
docker run --rm -v /work:/work -w /work ubuntu:latest cat /etc/os-
release | tee -a inspection_results.txt
```

```
PRETTY_NAME="Ubuntu 22.04.1 LTS"
NAME="Ubuntu"
VERSION_ID="22.04"
VERSION="22.04.1 LTS (Jammy Jellyfish)"
VERSION_CODENAME=jammy
ID=ubuntu
ID_LIKE=debian
HOME_URL="https://www.ubuntu.com/"
```

```
SUPPORT_URL="https://help.ubuntu.com/"
BUG_REPORT_URL="https://bugs.launchpad.net/ubuntu/"
PRIVACY_POLICY_URL="https://www.ubuntu.com/legal/terms-and-
policies/privacy-policy"
UBUNTU_CODENAME=jammy

# Load the samtools docker image that was provided as an input file and
run it.
docker load -i /work/in/samtools_docker_image/samtools_biocontainers.tar
Loaded image: biocontainers/samtools:v1.9-4-deb_cv1

docker run --rm biocontainers/samtools:v1.9-4-deb_cv1 samtools --version
samtools 1.9
Using htlib 1.9
Copyright (C) 2018 Genome Research Ltd.

# Docker images and running containers on the worker
docker images

```

REPOSITORY	TAG	IMAGE ID	CREATED
ubuntu	latest	a8780b506fa4	3 weeks ago
77.8MB			
biocontainers/samtools	v1.9-4-deb_cv1	f210eb625ba6	3 years ago
666MB			

```

docker ps

```

CONTAINER ID	IMAGE	COMMAND	CREATED
STATUS	PORTS	NAMES	

```

# Emit the URL string output variable.
emit url_to_fetch https://pfda-production-static-
files.s3.amazonaws.com/cli/pfda-linux-2.2.tar.gz

# Emit the inspection results file.
emit inspection_results inspection_results.txt

```

Utility Apps: *untar_files, tar_files_from_manifest*

We are going to use the `pfda_cli_2.2.tar` asset to create two very useful apps that demonstrate a design pattern that is readily extensible. The precisionFDA app I/O specification supports scalar and file and output types but does not support arrays for input or output variables. This means that using the app input variable and output emit framework, you cannot create apps with an arbitrary number of inputs or outputs. For instance, you really can't even create an untar app due to this limitation. These two apps demonstrate a design pattern to overcome this limitation.

Note that these apps require a temporary authorization key that you'll use with the CLI and this key will appear in the execution log files. Thus you should only run this app in your My Home or Private Space contexts so as not to expose the key to other users.

Your authorization key

This key is only valid for the next 24 hours. Please keep it safe:

```
Yy9reHJEVjVoVnR0eTRLQ3h6MzRhMVZRdjJyVFMvenFEZDZ1Yk1EbG1PdGjYwXAwU1FNO
HV6b2ZNaJv3WXBkZmZkVWmZyJIZMVoYd1h3U1ZjTwc2bWfHuJc2MMZoZW9OWHBJM0MxRk
RNeEjV1NsYkFZUTdTxd1OUFnWtdLQ25mmEjVZVVFej1Kd3NsQTRvWkV8K1NxYj1PL3A
1NGMyZ096WHd5MzRjamI1UE5VeXJpVjZxNmtadDB6aFZtcUFMLS10ZTKvcjB3Lz8LNGMw
TGh4aTFUOWZ3PT0=-88ea73b01fc4f0817d668ef07780a244169edef8
```

tar_files_from_manifest: Tar a manifest of fileIDs

From My Home / Files, select the detail page and copy the file ID for a number of files. Create and upload *manifest.txt* with the list to be incorporated into an archive file (e.g.):

```
cat manifest.txt
file-GJv1zKj0Kj2vzFP4Gg475ZyX-1
file-GJv1zKj0Kj2XY800GgJY2f4G-1
file-GJv1zKQ0Kj2vfZkVFxp436B9-1
```

```
key="..."
```

```
pfda upload-file -key $key -file manifest.txt
```

Create a *tar_files_from_manifest* app titled "Tar a manifest of fileIDs", with the following I/O Spec.

Class	Input Name	Label	Default Value
file	manifest	Text manifest of fileIDs	manifest.file
string	tar_filename	Filename for tar archive	archive.tar
string	pfda_key	Token for pfda CLI	
Class	Output Name	Label	
file	tarfile	Tar archive file	

I/O SPEC
Configure Input & Output Fields
VM ENVIRONMENT
Configure your resources
SCRIPT
Write your shell script
README
Describe your app

Need help? [Learn more about app inputs and outputs](#)

INPUTS [+ Add Input Field](#)

Class	Name	Label	Help text	Default Value	Choices Limit what values can be set	Optional?
file	manifest	Text manifest of fileIDs	Enter help text for this field	manifest.txt		☐
string	tar_filename	Filename for tar archive	Enter help text for this field	archive.tar	Optional comma separated st	☐
string	pfda_key	Token for pfda CLI	Enter help text for this field	Optional default string	Optional comma separated st	☐

OUTPUTS [+ Add Output Field](#)

Class	Name	Label	Help text	Optional?
file	tarfile.tar	Tar archive file	Enter help text for this output	☐

Setup the VM environment with internet access enabled, Baseline 2 default instance type, and select the pfda_cli_2.2 asset. Note that it can take minutes for the list of assets to loaded before they can be searched.

Enter the following Script:

```
set -euxo pipefail
echo "$pfda_key"
echo "$tar_filename"
echo "$manifest"
sudo apt-get update
sudo apt-get install -y dos2unix
pfda -version
mkdir temp
cd temp
dos2unix "$manifest_path"
for file in $(cat "$manifest_path"); do pfda download -key "$pfda_key" -
file-id $file; done
cd ..
tar cvf "$tar_filename" temp/*
emit "tarfile" "$tar_filename"
```

Enter a Readme and Create the app.

Input a manifest file containing a list of fileIDs and the name of tar archive file. You'll need to provide a temporary authorization key for use with the pfda CLI and thus you should only run this app in your My Home or Private Spaces.

Run the app with the default inputs and a fresh authorization token for the pfda CLI and observe the new archive.tar file.

Run Tar a manifest of fileIDs

Run App

Need help? [Learn more about running an app](#)

CONFIGURE

Job Name Tar a manifest of fileIDs

Instance Type Baseline 2 0.286\$/hour

Execution Cost Limit (\$) 10

INPUTS

Text manifest of fileIDs manifest.txt

Filename for tar archive archive.tar

Token for pfda CLI UWNhMHdLNk55eEV3akhJOUZqQUdsSDExNVBxNDV3UW9lYWV2S0ZuY

When the execution is done, download the archive.tar file to verify its contents.

```
tar tvf archive.tar
-rw-r--r-- root/root      15 2022-11-29 02:22 temp/foo.txt
-rw-r--r-- root/root      15 2022-11-29 02:22 temp/foo2.txt
-rw-r--r-- root/root      15 2022-11-29 02:22 temp/foo3.txt
```

untar_files: Untar archive to files

Create a *untar_files* app titled “Untar archive to files”, with the following I/O Spec.

Class	Input Name	Label	Default Value
string	pfda_key	CLI access authorization token	
file	input_tarfile	Tar file to extract	archive.tar
string	extracted_list_filename	List of extracted files	extracted_list.txt
Class	Output Name	Label	
file	tarfile_contents	Listing of the tarfile contents	

APP NAME TITLE [+ Create](#)

[I/O SPEC](#) [VM ENVIRONMENT](#) [SCRIPT](#) [README](#)
 Configure Input & Output Fields Configure your resources Write your shell script Describe your app

Need help? [Learn more about app inputs and outputs](#)

INPUTS [+ Add Input Field](#)

Class	Name	Label	Help text	Default Value	Choices Limit what values can be set	Optional?
string	pfda_key	CLI access authorization to	Enter help text for this field	Optional default string	Optional comma separated st	☐ ×
file	input_tarfile	Tar file to extract	Enter help text for this field	archive.tar		☐
string	extracted_list_filename	List of extracted files	Enter help text for this field	extracted_list.txt	Optional comma separated st	☐

OUTPUTS [+ Add Output Field](#)

Class	Name	Label	Help text	Optional?
file	tarfile_contents	Listing of the tarfile contents	Enter help text for this output	☐

Setup the VM environment with internet access enabled, Baseline 2 default instance type, and select the pfda_cli_2.2 asset. Note that it can take minutes for the list of assets to loaded before they can be searched.

Enter the following Script:

```
set -euxo pipefail
echo "$pfda_key"
echo "$input_tarfile"
echo "$extracted_list_filename"
pfda -version
tar tvf "$input_tarfile_path" > "$extracted_list_filename"
mkdir temp
tar xvf "$input_tarfile_path" --directory temp
ls temp
for FILE in $(find ./temp -type f -print); do echo $FILE; done
emit "tarfile_contents" "$extracted_list_filename"
```

Enter a Readme and Create the app.

Input a tar archive file to extract into individual files. You'll need to provide a temporary authorization key for use with the pfda CLI and thus you should only run this app in your My Home or Private Spaces.

Run the app with the default inputs and a fresh authorization token for the pfda CLI.

Files9939

Apps18

Databases9

Assets4

Workflows2

Executions229

+ Add Folder

+ Add Files

Actions

You are here: Files

☐	Name	Size	Created	Origin
	--	Min(Kb)Max(Kb)		
☐	extracted_list.txt	185 Bytes	2022-11-29 02:58:37 UTC	Untar archive to files
☐	foo3.txt	15 Bytes	2022-11-29 02:56:34 UTC	Uploaded
☐	foo2.txt	15 Bytes	2022-11-29 02:56:32 UTC	Uploaded
☐	foo.txt	15 Bytes	2022-11-29 02:56:31 UTC	Uploaded
☐	archive.tar	10 KB	2022-11-29 02:22:57 UTC	Tar a manifest of fileIDs

Run Untar archive to files

Run App

Need help? [Learn more about running an app](#)

CONFIGURE

Job Name

Untar archive to files

Instance Type

Baseline 2 0.286\$/hour

Execution Cost Limit (\$)

10

INPUTS

CLI access authorization token

L1F1ZW5UWUdiN2YrR1QwVDFWOWZhZ3d4VWWhoWctiNGxEbXlpdEFmY

Tar file to extract

[archive.tar](#)

List of extracted files

[extracted_list.txt](#)

When the execution is done, observe the new extracted files, and open the `extracted_list.txt` file to see the list of extracted files.

Files9939

Apps18

Databases9

Assets4

Workflows2

Executions229

+ Add Folder

+ Add Files

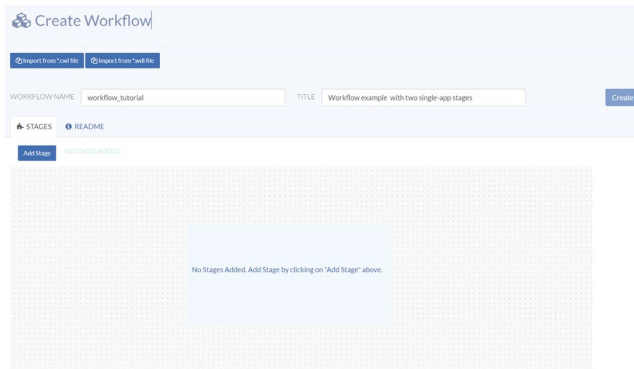
Actions

You are here: Files

☐	Name	Size	Created	Origin
	--	Min(Kb)Max(Kb)		
☐	extracted_list.txt	185 Bytes	2022-11-29 02:58:37 UTC	Untar archive to files
☐	foo3.txt	15 Bytes	2022-11-29 02:56:34 UTC	Uploaded
☐	foo2.txt	15 Bytes	2022-11-29 02:56:32 UTC	Uploaded
☐	foo.txt	15 Bytes	2022-11-29 02:56:31 UTC	Uploaded
☐	archive.tar	10 KB	2022-11-29 02:22:57 UTC	Tar a manifest of fileIDs

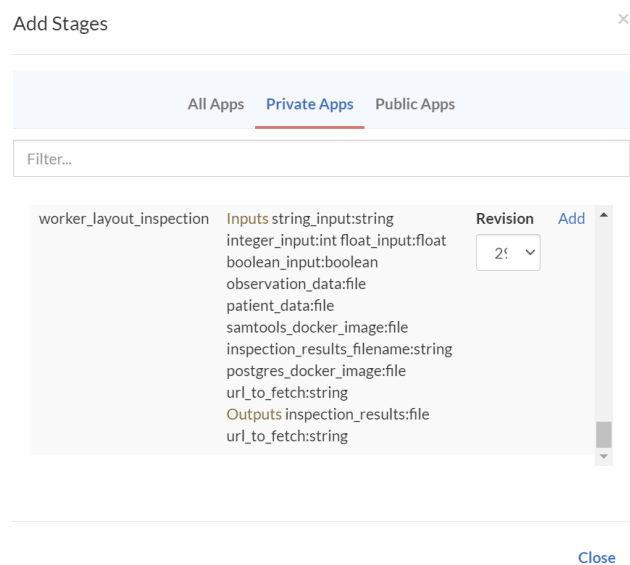
First Workflow: *simple_workflow*

Workflows enable you to string together multiple stages, each running on its own instance type and its own app(s). Outputs from a given stage can be routed to inputs in the next stage. We will create this workflow using the web interface. In My Home / Workflows, click Create Workflow and name it *simple_workflow* with a title “Workflow example with two single-app stages”.

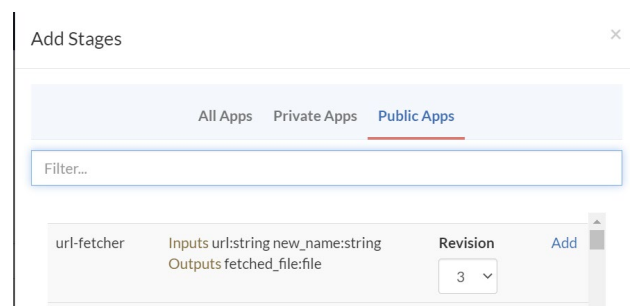


Add the workflow stages

Click the Add Stage button and add the private workflow_layout_inspection app to the stage.

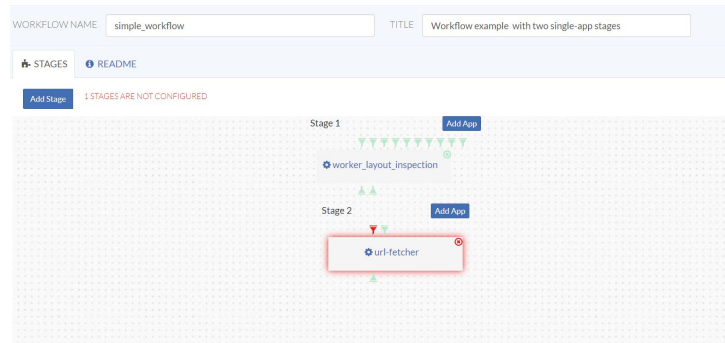


Add a second stage with the public url-fetcher app.



Configure the workflow stages

Now we are ready to configure the stages.



Click on Stage 1 to display the `worker_layout_inspection` app's configuration; we will accept all the default values provided in the app so there is nothing more that needs to be configured for Stage 1.

worker_layout_inspection

Revision: 29

INPUTS

example string input - Required Type: string

this is a string

✕

☐ Set as required workflow input

example integer input - Required Type: int

54321

✕

☐ Set as required workflow input

example float input - Required Type: float

1.234

✕

☐ Set as required workflow input

example boolean input - Required Type: boolean

false

✕

☐ Set as required workflow input

Delimited observation data - Required Type: file

file-GK1F6jQOKj2v90jxPV0ZBQ5G-1

✕

☐ Set as required workflow input

Delimited patient data - Required Type: file

file-GK1F6jQOKj2gyF8jG8jPjGpV-1

✕

☐ Set as required workflow input

Docker image for Samtools - Required Type: file

file-GK1FXK80pyBj68X3K6jVX55b-1

✕

☐ Set as required workflow input

Inspection results filename - Required Type: string

inspection_results.txt

✕

☐ Set as required workflow input

Docker image for postgres server - Required Type: file

file-GK1Fg68072kf3ZXYGJvxPGPJ-1

✕

☐ Set as required workflow input

URL to pass to next app in workflow - Required Type: string

https://pfda-production-static-files.s3.amazonaws.com/cli/pfda-linux-2.1.2.tar.gz

✕

☐ Set as required workflow input

OUTPUTS

Worker layout and variable Type: file

Inspection results

URL to pass to next app in workflow Type: string

CONFIG

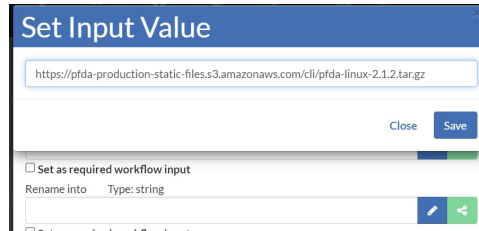
Instance Type

Baseline 2 0.286\$/hour

Close

Click on Stage 2 to display the `url-fetcher` app's configuration. Note the red color indicating that a required input has not been specified either from the output of a previous stage, or explicitly in the workflow stage's input spec.

Click the  button to enter an explicit URL for this Stage.



Set Input Value

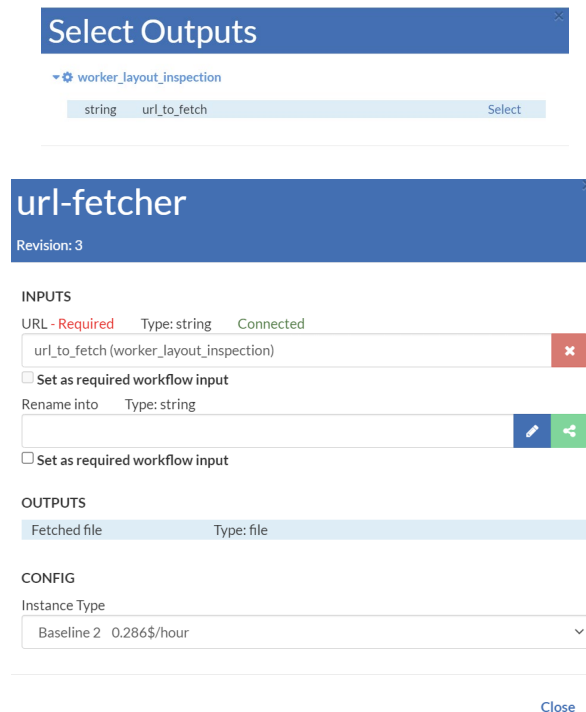
Close Save

☐ Set as required workflow input

Rename into Type: string

☐ Set as required workflow input

However, we want to specify this input field using the output field from the previous stage by clicking on the  button.



Select Outputs

worker_layout_inspection

string url_to_fetch Select

url-fetcher

Revision: 3

INPUTS

URL - Required Type: string Connected

☐ Set as required workflow input

Rename into Type: string

☐ Set as required workflow input

OUTPUTS

Fetchd file Type: file

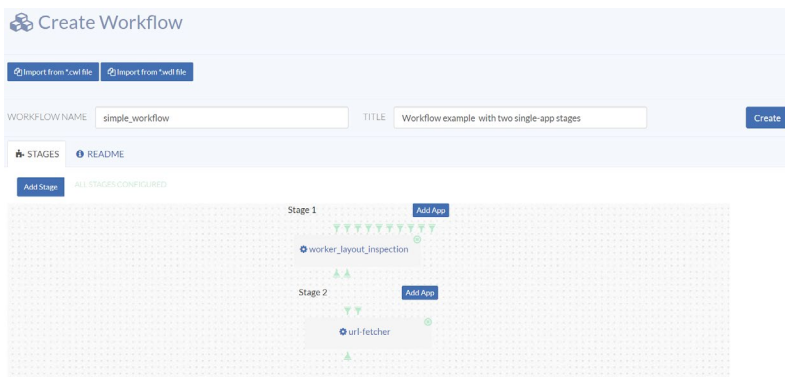
CONFIG

Instance Type

Baseline 2 0.286\$/hour

Close

Select the Baseline 2 instance type to override the app default value and close the stage. Note that everything is green now and ready to Create.



You can view the two stages of your new workflow.

Apps16

Databases9

Assets4

Workflows2

Executions223

Workflow example with two single-app stages

Run Workflow

Run Batch Workflow

Actions

Revision: 1

Label

LOCATION	NAME	ID	ADDED BY	CREATED ON
Private	simple_workflow	workflow-GK2bQ600KJ2z2vfy371JYpJK-1	Omar Serang	2022-11-29 00:53:44 UTC

Spec

Executions (0)

Diagram

Readme

STAGE

NAME

DEFAULT INSTANCE TYPE

1

worker_layout_inspection

baseline-2

INPUTS

OUTPUTS

string

example string input

Default: this is a string

REQUIRED

int

example integer input

Default: 54321

REQUIRED

float

example float input

Default: 1.234

REQUIRED

boolean

example boolean input

REQUIRED

file

Delimited observation data

Default file: GK1fGQ0KQ2vR9q4PVZ8QSG-1

REQUIRED

file

Delimited patient data

Default file: GK1fGQ0KQ2vR9q4PVZ8QSG-1

REQUIRED

file

Docker image for Santools

Default file: GK3FX00Qy4f640X3K6jX059-1

REQUIRED

string

Inspection results filename

Default: inspection_results.txt

REQUIRED

file

Docker image for postgres server

Default file: GK3Jg58072fK2Z0YGAnePGj-1

REQUIRED

string

URL to pass to next app in workflow

Default: https://fda.production-static-files3.amazonaws.com/ids/fda/linux-2.1.2.target

REQUIRED

file

Worker layout and variable inspection results

REQUIRED

string

URL to pass to next app in workflow

REQUIRED

STAGE

NAME

DEFAULT INSTANCE TYPE

2

url-fetcher

baseline-2

INPUTS

OUTPUTS

string

URL

REQUIRED

string

Rename into

REQUIRED

file

Fetches file

REQUIRED

Run the workflow

Click on the Run Workflow button, accept all the default values for the workflow_layout_inspection stage, but enter *pfdacli.tar.gz* in the *Rename into* field in the second stage, then run the workflow.

Run Workflow example with two single-app stages Run Workflow

CONFIGURE

Analysis Name

INPUTS

worker_layout_inspection

example string input

example integer input

example float input

example boolean input

Delimited observation data

Delimited patient data

Docker image for Samtools

Inspection results filename

Docker image for postgres server

URL to pass to next app in workflow

url-fetcher

Rename into

In the Executions tab, expand the simple_workflow listing to see the two executions associated with the workflow.

Files	9931	Actions				
Apps	16					
Databases	9					
Assets	4					
Workflows	2					
Executions	225					
		State	Execution Name	App Title	Instance Type	Duration
		waiting_on_inpx	simple_workflow			N/A
		runnable	Worker layout inspection tutorial app	Worker layout inspection tut baseline-2		
		waiting_on_inpx	Fetch file from URL	Fetch file from URL	baseline-2	
		done	Worker layout inspection tutorial app	Worker layout inspection tut baseline-2		2 minutes 15 seconds

Once the stages have completed, the workflow is done and we can open the executions associated with the stages and inspect the logs. Also note the inspection results file from stage 1 and the renamed fetched URL file from stage 2.

Files9933

Apps16

Databases9

Assets4

Workflows2

Executions225

+ Add Folder

+ Add Files

Actions

You are here: Files

☐	Name	Size	Created	Origin
	--	Min(KB) Max(KB)		
☐	pfdacli-2.1.2.tar.gz	5.61 MB	2022-11-29 01:07:42 UTC	Fetch file from URL
☐	inspection_results.txt	684 Bytes	2022-11-29 01:07:29 UTC	Worker layout inspection tutorial

Second Workflow: *workflow_from_wdl*

Importing a workflow from a WDL specification based on Docker images provides convenient way to express precisionFDA workflows in a serialized textual format. From My Home / Workflows, click the Create Workflow button, and click the Import from *.wdl file button. Enter the following WDL in the text input and click Import.

version 1.0

```

workflow bwa {
  Int threads
  Int min_seed_length
  Int min_std_max_min
  File reference
  File reads

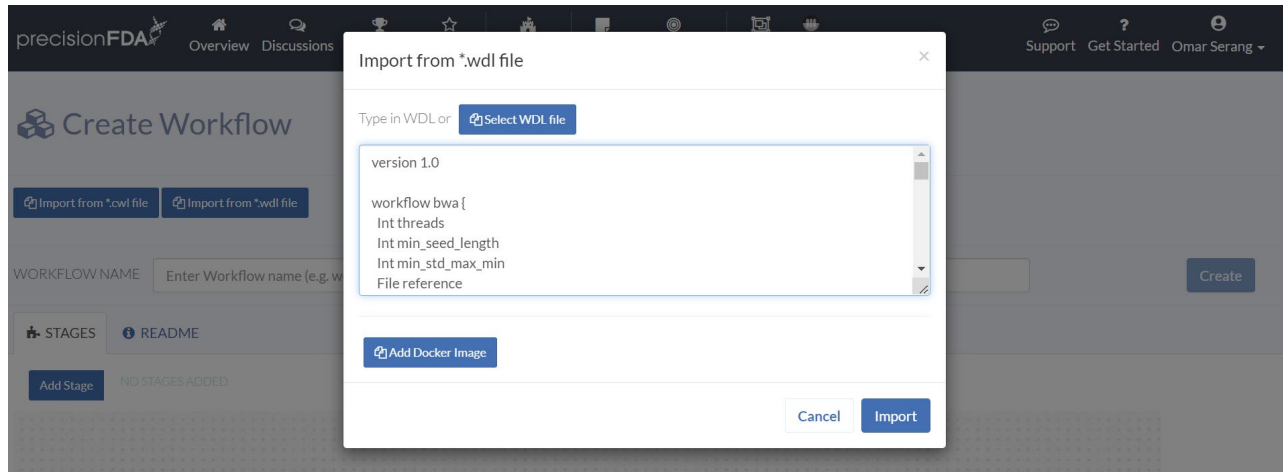
  call bwa_mem_tool {
    threads = threads,
    min_seed_length = min_seed_length,
    min_std_max_min = min_std_max_min,
    reference = reference,
    reads = reads
  }
}

task bwa_mem_tool {
  Int threads
  Int min_seed_length
  Int min_std_max_min
  File reference
  File reads

  command {
    bwa mem -t ${threads} \
    -k ${min_seed_length} \
    -I ${sep=', ' min_std_max_min+} \
    ${reference} \
    ${sep=' ' reads+} > output.sam
  }
}

```

```
output {  
  File sam = "output.sam"  
}  
  
runtime {  
  docker: "broadinstitute/baseimg"  
}  
}
```



Files 9939 [Back to Workflows](#)

Apps 19

Databases 9

Assets 4

Workflows 1

Executions 229

bwa [Run Workflow](#) [Run Batch Workflow](#) [Actions](#)

Revision: 2 [Latest](#)

LOCATION	NAME	ID	ADDED BY	CREATED ON
Private	bwa	workflow-GK2jz00KjZZG0V95694Q4pF-1	Omar Serang	2022-11-29 04:52:12 UTC

Spec Executions (0) Diagram **Readme**

STAGE	NAME	DEFAULT INSTANCE TYPE
1	bwa_mem_tool	baseline-6

INPUTS		OUTPUTS	
Int	threads REQUIRED	File	sam REQUIRED
Int	min_seed_length REQUIRED		
Int	min_std_max_min REQUIRED		
File	reference REQUIRED		
File	reads REQUIRED		

Like workflows created through the web interface, WDL-based workflows can be updated and run.

The image shows two screenshots of the precisionFDA web interface. The top screenshot is a modal window titled 'bwa_mem_tool' (Revision: 3). It contains sections for 'INPUTS', 'OUTPUTS', and 'CONFIG'. Under 'INPUTS', there are six required fields: 'threads' (Type: int), 'min_seed_length' (Type: int), 'min_std_max_min' (Type: int), 'reference' (Type: file), 'reads' (Type: file), and an unchecked checkbox 'Set as required workflow input'. Each input field has a red 'x' icon. Under 'OUTPUTS', there is one field 'sam' (Type: file). Under 'CONFIG', there is a dropdown menu for 'Instance Type' with the value 'Baseline 8 1.1445/hour'. A 'Close' button is at the bottom right. The bottom screenshot is the 'Run Workflow created by importing WDL' page. It has a 'Run Workflow' button at the top right. Below is a 'CONFIGURE' section with an 'Analysis Name' field containing 'Workflow created by importing WDL'. Below that is an 'INPUTS' section with a dropdown menu showing 'bwa_mem_tool'. Under this dropdown, there are six input fields: 'threads', 'min_seed_length', 'min_std_max_min', 'reference', and 'reads'. Each field has a 'Select file...' button and a 'Clear' button.

Database App: *worker_database*

This app demonstrates the use of a precisionFDA Database cluster (PostgreSQL), a convenient and very power resource for RDBMS-based analytics. You will need to be authorized for DB Clusters in order to create the database for this app.

Create the Database

Select the Databases tab in My Home and click the Create Database button.

precisionFDA Overview Discussions Challenges Experts My Home Notes Comparisons Spaces GSRS Support Get Started Omar Serang

My Home Me Featured Everyone Spaces
Your private databases, visible to you only

- Files 9918
- Apps 13
- Databases 8**
- Assets 1
- Workflows 1
- Executions 191

[+ Create Database](#)

Status	Name	Type	Instance	Created	Tags
☐	--	--	--		--
☐	terminated DB Tester	MySQL	db_std1_x1	2022-03-21 20:25:15 UTC	
☐	terminated DB Tester	MySQL	db_std1_x2	2022-03-22 19:37:04 UTC	1.1
☐	terminated test	MySQL	db_std1_x2	2022-03-24 20:14:48 UTC	1.1
☐	terminated SmileCDRDev	PostgreSQL	db_std1_x2	2022-05-03 22:39:14 UTC	2

Create a “Workstations and Databases Tutorial” database, “password”, PostgreSQL 11.16 on the smallest available database instance type, and click the Submit button.

precisionFDA Overview Discussions Challenges Experts My Home Notes Comparisons Spaces GSRS Support

My Home Me Featured Everyone Spaces
Your private databases, visible to you only

- Files 9918
- Apps 13
- Databases 8**
- Assets 1
- Workflows 1
- Executions 191

[← Back to Databases](#)

Name (required)

Description

DB SQL file

DB admin password (required):

This password must be at least 8 characters in length and is not changeable once set

Retype DB admin password (required):

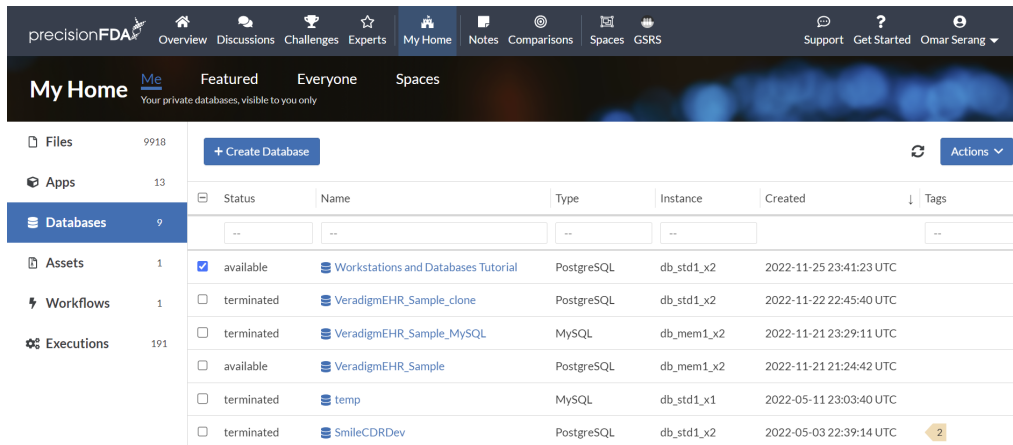
Database type (required):
☐ MySQL
☒ PostgreSQL

Instance (required):

Version (required):

[Submit](#)

Refresh the database status using the  button until the database is available.



Status	Name	Type	Instance	Created	Tags
☒ available	Workstations and Databases Tutorial	PostgreSQL	db_std1_x2	2022-11-25 23:41:23 UTC	
☐ terminated	VeradigmEHR_Sample_clone	PostgreSQL	db_std1_x2	2022-11-22 22:45:40 UTC	
☐ terminated	VeradigmEHR_Sample_MySQL	MySQL	db_mem1_x2	2022-11-21 23:29:11 UTC	
☐ available	VeradigmEHR_Sample	PostgreSQL	db_mem1_x2	2022-11-21 21:24:42 UTC	
☐ terminated	temp	MySQL	db_std1_x1	2022-05-11 23:03:40 UTC	
☐ terminated	SmileCDRDev	PostgreSQL	db_std1_x2	2022-05-03 22:39:14 UTC	2

Fork *workstation_layout_inspection* to *workstation_database* app

Using My Home / Apps, select the *worker_layout_inspection* app and select Fork. Name the new application *worker_database* titled “Database cluster app example”.

Specify the I/O Spec

Update the I/O spec to the following:

Class	Input Name	Label	Default Value
file	observation_data	Delimited data file for ETL into OBSERVATION table.	observations.txt
file	patient_data	Delimited data file for ETL into PATIENT table.	patients.txt
file	db_and_table_ddl	Database and table creation DDL	
file	query_sql	Query to run	(optional)
string	db_endpoint_url	Database host endpoint	
string	query_results_filename	Query results file name	query_results.txt
Class	Output Name	Label	
file	query_results	SQL query results	

APP NAME: **worker_database** TITLE: Database cluster app example UBUNTU RELEASE: 16.04 Revision 11 → Save Revision 12

I/O SPEC: Configure Input & Output Fields VM ENVIRONMENT: Configure your resources SCRIPT: Write your shell script README: Describe your app App revisions are private

Need help? Learn more about app inputs and outputs

INPUTS + Add Input Field

Class	Name	Label	Help text	Default Value	Choices Limit what values can be set	Optional?
file	observation_data	Delimited observation	Enter help text for this	observations.txt		☐
file	patient_data	Delimited patient data	Enter help text for this	patients.txt		☐
file	db_and_table_ddl	Database and table cr	Enter help text for this	apps_and_workflows_tutorial_DDL.sql		☐
file	query_sql	Query to run	Enter help text for this	Select file...		☒
string	query_results_filename	Query results file name	Enter help text for this	query_results.txt	Optional comma separat	☐
string	db_host_endpoint	Database host endpoint	Enter help text for this	dbcluster-gk0b58j0kj2y4v1k899bq0x6.cluster-	Optional comma separat	☐

OUTPUTS + Add Output Field

Class	Name	Label	Help text	Optional?
file	query_results	SQL query results	Enter help text for this output	☐

Specify the VM Environment

Update the VM environment to keep internet access enabled, Baseline 2 default instance type, and remove the pfda_cli_2.2 and ubuntu_asset assets so that only the worker_layout_inspection asset remains. Note that it can take minutes for the list of assets to loaded before they can be searched.

Specify the Script

Enter the following Script:

```
set -euxo pipefail
```

```
# Install postgres client
sudo apt install -y postgresql-client
psql --version
```

```
# Inspect the database and table DDL
# and the three data files for ETL.
# Two data files specified as inputs.
#cat "$db_and_table_ddl_path"
#cat "$observation_data_path"
#cat "$patient_data_path"
```

```
# Third data file installed with the worker_layout_inspection asset
#cat /work/countries.txt
```

```
# Connect to the DB cluster and list the databases
PGPASSWORD="password" psql -h "$db_host_endpoint" -U root -d postgres -c
'\1'
```

```
# Create the apps_and_workflows_tutorial_db and three tables
PGPASSWORD="password" psql -h "$db_host_endpoint" -U root -d postgres -f
"$db_and_table_ddl_path"
```

```
PGPASSWORD="password" psql -h "$db_host_endpoint" -U root -d
apps_and_workflows_tutorial_db -c '\d'

# ETL the OBSERVATION table data
PGPASSWORD="password" psql -h "$db_host_endpoint" -U root -d
apps_and_workflows_tutorial_db -c "\copy public.\"OBSERVATION\" from
'/work/in/observation_data/observations.txt' delimiter '|' NULL ''"

PGPASSWORD="password" psql -h "$db_host_endpoint" -U root -d
apps_and_workflows_tutorial_db -c "select * from public.\"OBSERVATION\""

# ETL the PATIENT table data
PGPASSWORD="password" psql -h "$db_host_endpoint" -U root -d
apps_and_workflows_tutorial_db -c "\copy public.\"PATIENT\" from
'/work/in/patient_data/patients.txt' delimiter '|' NULL ''"

PGPASSWORD="password" psql -h "$db_host_endpoint" -U root -d
apps_and_workflows_tutorial_db -c "select * from public.\"PATIENT\""

# ETL the COUNTRY table data
PGPASSWORD="password" psql -h "$db_host_endpoint" -U root -d
apps_and_workflows_tutorial_db -c "\copy public.\"COUNTRY\" from
'/work/countries.txt' delimiter '|' NULL ''"

PGPASSWORD="password" psql -h "$db_host_endpoint" -U root -d
apps_and_workflows_tutorial_db -c "select * from public.\"COUNTRY\""

# Run the specified query and put the results into the specified
filename.
PGPASSWORD="password" psql -h "$db_host_endpoint" -U root -d
apps_and_workflows_tutorial_db -f "$query_sql_path" | tee -a
"$query_results_filename"

emit "query_results" "$query_results_filename"
```

[Specify the Readme](#)

Enter a Readme and Create the app.

Connect to a database cluster and create a database and three tables. Load two tables with data provided as input files, and the third table with data from an asset. Present a specified query file and retrieve the query results file.

Run the app

Click on the Workstations and Databases Tutorial database to open the detail page and copy the host endpoint URL.

The screenshot shows the 'My Home' page with a sidebar on the left containing links to Files (9918), Apps (13), Databases (9), Assets (1), Workflows (1), and Executions (191). The main content area is titled 'Workstations and Databases Tutorial' and includes a table with database details:

LOCATION	ID	ADDED BY	CREATED ON
Private	dbcluster-GK0b58j0Kj2Y4V1k899BQ0X6	Omar Serang	2022-11-25 23:41:23 UTC

Below the table, there is a section for 'DB STATUS' and 'HOST ENDPOINT'.

DB STATUS	DB PORT	ENGINE	VERSION	INSTANCE
available	5432	aurora-postgresql	11.16	db_std1_x2

The 'HOST ENDPOINT' is: `dbcluster-gk0b58j0Kj2Y4V1k899BQ0X6.cluster-cqy4cenhebvus-east-1.rds.amazonaws.com`

Run the app providing the database host endpoint URL copied above, and leave the remaining inputs at their default values.

The screenshot shows the 'Run Database cluster app example' configuration page. It includes a sidebar with links to Files (9947), Apps (21), Databases (9), Assets (4), Workflows (2), and Executions (248). The main content area is titled 'Database cluster app example' and includes a 'Revision: 20 Latest' dropdown. Below this is a table with database details:

LOCATION	NAME	ID	ADDED BY	CREATED ON
Private	worker_database	app-GK387KQ0j5gX6bgz5113X0X4-1	Omar Serang	2022-11-29 22:12:10 UTC

Below the table, there is a 'CONFIGURE' section with fields for 'Job Name' (Database cluster app example), 'Instance Type' (Baseline 2 0.286\$/hour), and 'Execution Cost Limit (\$)' (1000). There is also an 'INPUTS' section with fields for 'Delimited observation data' (observations.txt), 'Delimited patient data' (patients.txt), 'Database and table creation DDL' (apps_and_workflows_tutorial_DDL.sql), 'Query to run' (query.sql), 'Query results file name' (query_results.txt), and 'Database host endpoint' (dbcluster-gk0b58j0Kj2Y4V1k899BQ0X6.cluster-cqy4cenhebvus-east-1.rds.amazonaws.com).

Inspect the execution logs

View the logs for the *Database cluster app example* execution using the Actions dropdown menu. Let's look at a condensed and annotated version of the log output to understand what our app just did.

```
# Install the assets and file inputs on the worker filesystem
Fetching asset worker_layout_inspection2.tar (file-
GK1xxbQ0Kj2gBZjxF244F21k)
downloading file: file-GK2xgy80Kj2x48j54fXGqF1V to filesystem:
/work/in/query_sql/query.sql
```

```

downloading file: file-GK1F6jj0Kj2gyF8jG8jPjGpV to filesystem:
/work/in/patient_data/patients.txt
downloading file: file-GK2v2Zj0Kj2gk9GB1920BYP3 to filesystem:
/work/in/db_and_table_ddl/apps_and_workflows_tutorial_DDL.sql
downloading file: file-GK1F6jQ0Kj2v90jxPV0ZBQ5G to filesystem:
/work/in/observation_data/observations.txt

```

Install postgres CLI client and display version

```

++ sudo apt install -y postgresql-client
++ psql --version
psql (PostgreSQL) 9.5.25

```

Providing the password, connect to the specified host, # user root, database postgres, and list the databases on the cluster.

```

++ PGPASSWORD=password
++ psql -h dbcluster-gk0b58j0kj2y4v1k899bq0x6.cluster-cqy4cenhebv.us-
east-1.rds.amazonaws.com -U root -d postgres -c '\l'

```

```

                                List of databases
Collate  |      Name      | Owner  | Encoding |
-----+-----+-----+-----+-----+
en_US.UTF-8 | en_US.UTF-8 |
postgres  | root        | UTF8    |
en_US.UTF-8 | en_US.UTF-8 |
rdsadmin  | rdsadmin   | UTF8    |
en_US.UTF-8 | en_US.UTF-8 | rdsadmin=CTc/rdsadmin
template0  | rdsadmin   | UTF8    |
en_US.UTF-8 | en_US.UTF-8 | =c/rdsadmin  +
|              | rdsadmin=CTc/rdsadmin

```

Create a new database and tables

```

++ PGPASSWORD=password
++ psql -h dbcluster-gk0b58j0kj2y4v1k899bq0x6.cluster-cqy4cenhebv.us-
east-1.rds.amazonaws.com -U root -d postgres -f
/work/in/db_and_table_ddl/apps_and_workflows_tutorial_DDL.sql
template1  | root      | UTF8    |
en_US.UTF-8 | en_US.UTF-8 | =c/root  +
|              | root=CTc/root
workstations_and_databases_tutorial_db | root      | UTF8    |
en_US.UTF-8 | en_US.UTF-8 |
(6 rows)

```

```

pg_terminate_backend
-----
(0 rows)

```

```
DROP DATABASE
```

```
CREATE DATABASE
```

```
You are now connected to database "apps_and_workflows_tutorial_db" as
user "root".
```

```
CREATE TABLE
```

```
CREATE TABLE
```

```
CREATE TABLE
```

Connect to the new database and list the new tables

```
++ PGPASSWORD=password
```

```
++ psql -h dbcluster-gk0b58j0kj2y4v1k899bq0x6.cluster-cqy4cenhebv.us-
east-1.rds.amazonaws.com -U root -d apps_and_workflows_tutorial_db -c
'\d'
```

```
                List of relations
 Schema |      Name      | Type  | Owner
-----+-----+-----+-----
 public | COUNTRY        | table | root
 public | OBSERVATION    | table | root
 public | PATIENT        | table | root
(3 rows)
```

ETL the OBSERVATION table from the specified file.

```
++ PGPASSWORD=password
```

```
++ psql -h dbcluster-gk0b58j0kj2y4v1k899bq0x6.cluster-cqy4cenhebv.us-
east-1.rds.amazonaws.com -U root -d apps_and_workflows_tutorial_db -c
'\copy public."OBSERVATION" from
```

```
'\''/work/in/observation_data/observations.txt'\'' delimiter '\''|\'\'
NULL '\''\''\''
```

```
COPY 5
```

```
++ PGPASSWORD=password
```

```
++ psql -h dbcluster-gk0b58j0kj2y4v1k899bq0x6.cluster-cqy4cenhebv.us-
east-1.rds.amazonaws.com -U root -d apps_and_workflows_tutorial_db -c
'select * from public."OBSERVATION"
```

```
 observation_id | patient_id | observation_name | loinc |
created_date
-----+-----+-----+-----+-----
---
          9870 |      12345 | Annual check up | 66678-4 | 2022-11-01
          9871 |      12345 | Emergency       | LG32756-5 | 2022-11-02
          9872 |      12346 | Clinic visit    | 66678-4 | 2022-11-03
          9873 |      12347 | Lab results     | 74418-5 | 2022-11-04
          9874 |      12347 | Post-op checkup | 65375-8 | 2022-11-05
(5 rows)
```

ETL the PATIENT table from the specified file.

```
++ PGPASSWORD=password
```

```
++ psql -h dbcluster-gk0b58j0kj2y4v1k899bq0x6.cluster-cqy4cenhebv.us-
east-1.rds.amazonaws.com -U root -d apps_and_workflows_tutorial_db -c
'\copy public."PATIENT" from '\''/work/in/patient_data/patients.txt'\''
delimiter '\''|\'\' NULL '\''\''\''
```

```
COPY 3
```

```
++ PGPASSWORD=password
```

```
++ psql -h dbcluster-gk0b58j0kj2y4v1k899bq0x6.cluster-cqy4cenhebv.us-east-1.rds.amazonaws.com -U root -d apps_and_workflows_tutorial_db -c 'select * from public."PATIENT"'
patient_id |      name      | gender | zip  | country_id | created_date
-----+-----+-----+-----+-----+-----
-
      12345 | Fred Foobar    | M      | 94040 |      1001 | 2022-10-25
      12346 | Mary Merry     | F      | 94040 |      1002 | 2022-09-24
      12347 | Barney Rubble  | M      | 94040 |      1003 | 2022-08-23
(3 rows)
```

ETL the COUNTRY table as installed from the asset.

```
++ PGPASSWORD=password
++ psql -h dbcluster-gk0b58j0kj2y4v1k899bq0x6.cluster-cqy4cenhebv.us-east-1.rds.amazonaws.com -U root -d apps_and_workflows_tutorial_db -c '\copy public."COUNTRY" from '\''/work/countries.txt'\'' delimiter '\''|'\'' NULL '\''\|'\''
COPY 3
++ PGPASSWORD=password
++ psql -h dbcluster-gk0b58j0kj2y4v1k899bq0x6.cluster-cqy4cenhebv.us-east-1.rds.amazonaws.com -U root -d apps_and_workflows_tutorial_db -c 'select * from public."COUNTRY"'
country_id | country_name
-----+-----
      1001 | USA
      1002 | CAN
      1003 | EU
(3 rows)
```

Run the specified query.

```
++ PGPASSWORD=password
++ psql -h dbcluster-gk0b58j0kj2y4v1k899bq0x6.cluster-cqy4cenhebv.us-east-1.rds.amazonaws.com -U root -d apps_and_workflows_tutorial_db -f /work/in/query_sql/query.sql
++ tee -a query_results.txt
Expanded display is on.
-[ RECORD 1 ]-----+-----
patient_id      | 12345
name            | Fred Foobar
gender          | M
zip             | 94040
country_id      | 1001
created_date    | 2022-10-25
observation_id  | 9870
observation_name | Annual check up
loinc           | 66678-4
created_date    | 2022-11-01
country_id      | 1001
country_name    | USA
-[ RECORD 2 ]-----+-----
patient_id      | 12345
```

```
name          | Fred Foobar
gender        | M
zip           | 94040
country_id    | 1001
created_date  | 2022-10-25
observation_id | 9871
observation_name | Emergency
loinc         | LG32756-5
created_date  | 2022-11-02
country_id    | 1001
country_name  | USA
-[ RECORD 3 ]-----+-----
patient_id    | 12346
name          | Mary Merry
gender        | F
zip           | 94040
country_id    | 1002
created_date  | 2022-09-24
observation_id | 9872
observation_name | Clinic visit
loinc         | 66678-4
created_date  | 2022-11-03
country_id    | 1002
country_name  | CAN
-[ RECORD 4 ]-----+-----
patient_id    | 12347
name          | Barney Rubble
gender        | M
zip           | 94040
country_id    | 1003
created_date  | 2022-08-23
observation_id | 9873
observation_name | Lab results
loinc         | 74418-5
created_date  | 2022-11-04
country_id    | 1003
country_name  | EU
-[ RECORD 5 ]-----+-----
patient_id    | 12347
++ emit query_results query_results.txt
name          | Barney Rubble
gender        | M
zip           | 94040
country_id    | 1003
created_date  | 2022-08-23
observation_id | 9874
observation_name | Post-op checkup
loinc         | 65375-8
created_date  | 2022-11-05
country_id    | 1003
country_name  | EU
```