WDL Workflows

March 23, 2023

Table of Contents

# Overview

This book introduces WDL Workflows on AnVIL. After introducing several concepts, including basic WDL syntax, we present hands-on exercises to run a workflow, write a WDL, localize a file, customize a Docker image, and join the Discourse. No local software installation is required as each exercise leverages web-based resources.



## Skills Level

*Genetics*  
**Novice**: No genetics knowledge needed

*Programming skills*  
**Novice**: No programming experience needed

## Learning Objectives

* Understand when WDL Workflows are the right tool
* Run a Workflow on AnVIL
* Write a WDL using Broad Methods Repository
* Bring your own data to analyze
* Customize your Docker environment
* Join the conversation

## AnVIL Collection

Please check out our full collection of AnVIL and related resources: <https://hutchdatascience.org/AnVIL_Collection/>

# 1 Introduction

This lecture module introduces several basic concepts when working with WDL Workflows. It highlights several considerations when using Docker container technology and provides an overview of the main sections in an example WDL workflow. These slides are adapted from the [WDL 101 Workshop](https://support.terra.bio/hc/en-us/articles/8693717360411). More information can be found in [this presentation](https://zenodo.org/record/7062397#.YyM-vy1h1qu) hosted by Bioinformática UFMG.

**Learning Objectives**

1. Four parts of WDL Workflows on Terra
2. How Docker improves reproducibility
3. Overview of basic WDL syntax
4. Find and configure Workflows

You can view and download the Google Slides [here](https://docs.google.com/presentation/d/1Dioe_Qd3hdJ0UAP4LA6gJsm07Lr1wRpax5FLYr6xU3M).

# 2 Run Workflow

This tutorial demonstrates how to run a WDL Workflow on AnVIL. It uses the fun and accessible [WDL puzzles](https://support.terra.bio/hc/en-us/articles/360056599991) workspace to run a “Hello, World!” style workflow. By showcasing how input parameters are specified and where output files can be found, this exercise helps provide context before writing your own WDL Workflow.

**Learning Objectives**

1. Import WDL by cloning a Workspace
2. Run Workflow using AnVIL
3. Examine Output to appreciate data flow

You can view and download the Google Slides [here](https://docs.google.com/presentation/d/1FbTgCnAZTgVzp-lPwOVqOqU4Kw19nu825dMngfyc-Nw).

# 3 Write WDL

Now that you’ve successfully run a Workflow on AnVIL, this tutorial demonstrates how you can create and edit a WDL using the [Broad Methods Repository](https://portal.firecloud.org/?return=anvil#methods). While this “legacy” Methods repository does not have many of the features present in the open-source [Dockstore](https://anvilproject.org/overview#platform-interoperability) platform, it does offer a convenient web-based editor for demonstration purposes. This material is adapted from the [WDL 101 Workshop](https://support.terra.bio/hc/en-us/articles/8693717360411); you can read about other ways the Broad Methods Repository can be used in [this Terra Support article](https://support.terra.bio/hc/en-us/articles/360031366091).

**Learning Objectives**

1. Access Broad Methods Repository
2. Write WDL101 Training Example
3. Export to Terra and run

## 3.1 Access Broad Methods Repository

Let’s start by navigating to the WDL-puzzles workspace that we previously cloned. Please double check your workspace name to ensure that this is the copy that you made rather than the original as you will not be able to use the original workspace to create a new WDL or run a workflow.

## Warning: replacing previous import 'ellipsis::check\_dots\_unnamed' by  
## 'rlang::check\_dots\_unnamed' when loading 'tibble'

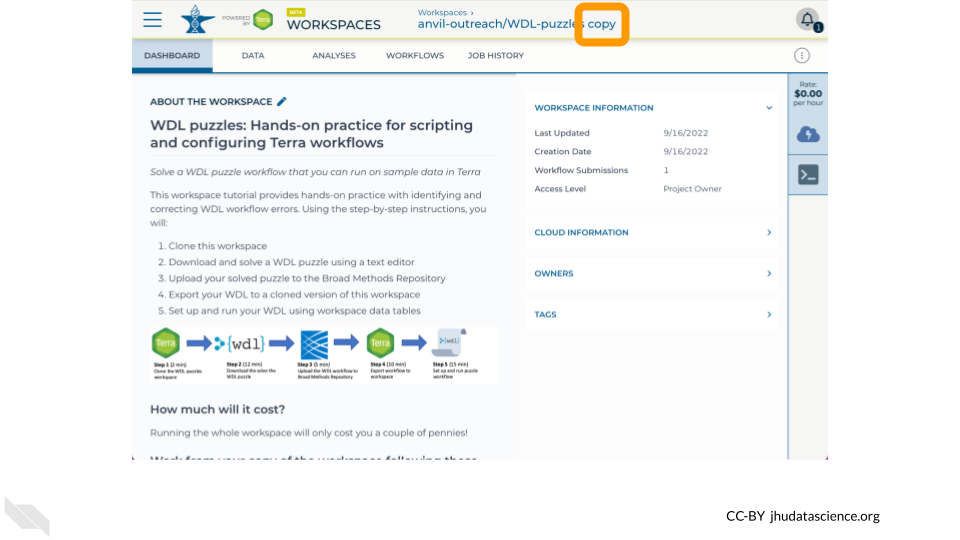
## Warning: replacing previous import 'ellipsis::check\_dots\_used' by  
## 'rlang::check\_dots\_used' when loading 'tibble'

## Warning: replacing previous import 'ellipsis::check\_dots\_empty' by  
## 'rlang::check\_dots\_empty' when loading 'tibble'

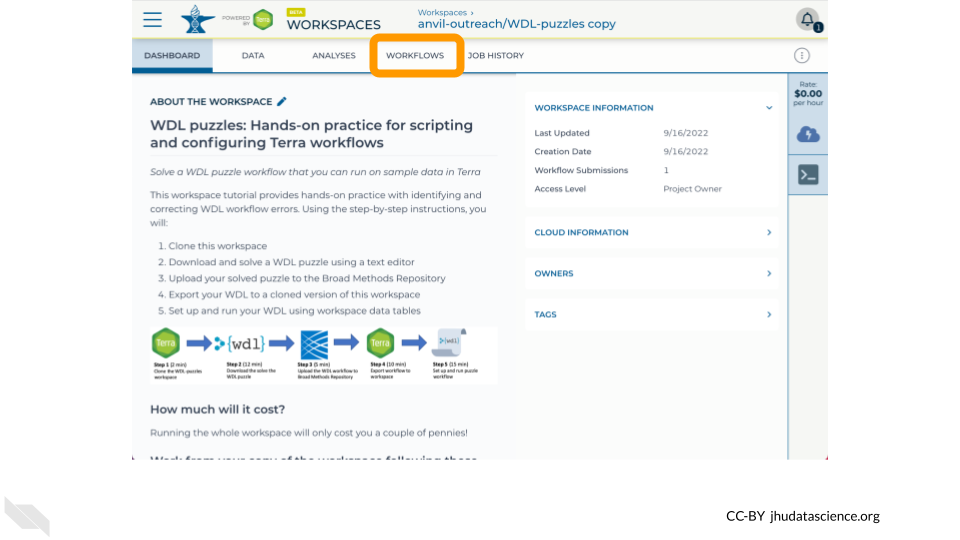
## Warning: replacing previous import 'ellipsis::check\_dots\_unnamed' by  
## 'rlang::check\_dots\_unnamed' when loading 'pillar'

## Warning: replacing previous import 'ellipsis::check\_dots\_used' by  
## 'rlang::check\_dots\_used' when loading 'pillar'

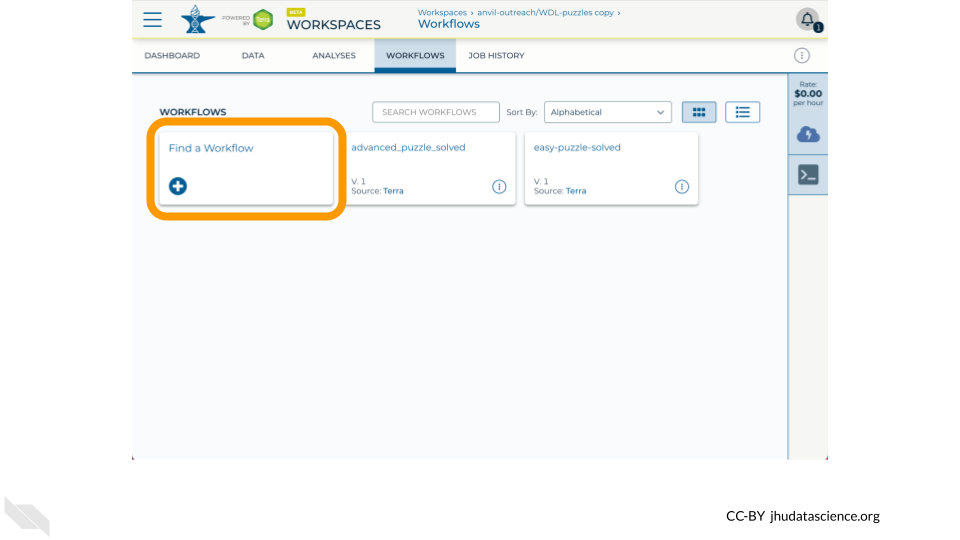
## Warning: replacing previous import 'ellipsis::check\_dots\_empty' by  
## 'rlang::check\_dots\_empty' when loading 'pillar'



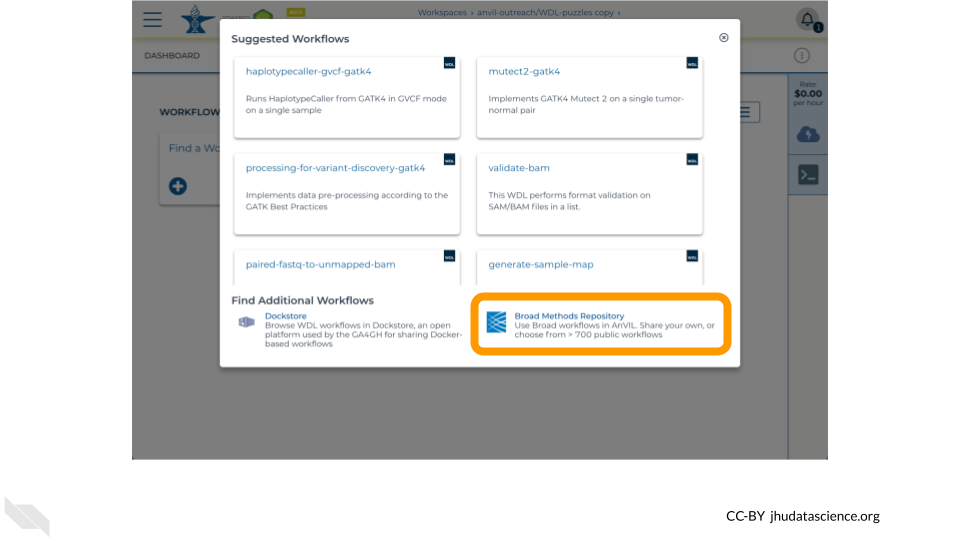
Once you’ve double checked that you are in a workspace that you can modify and compute, click on the Workflows tab.



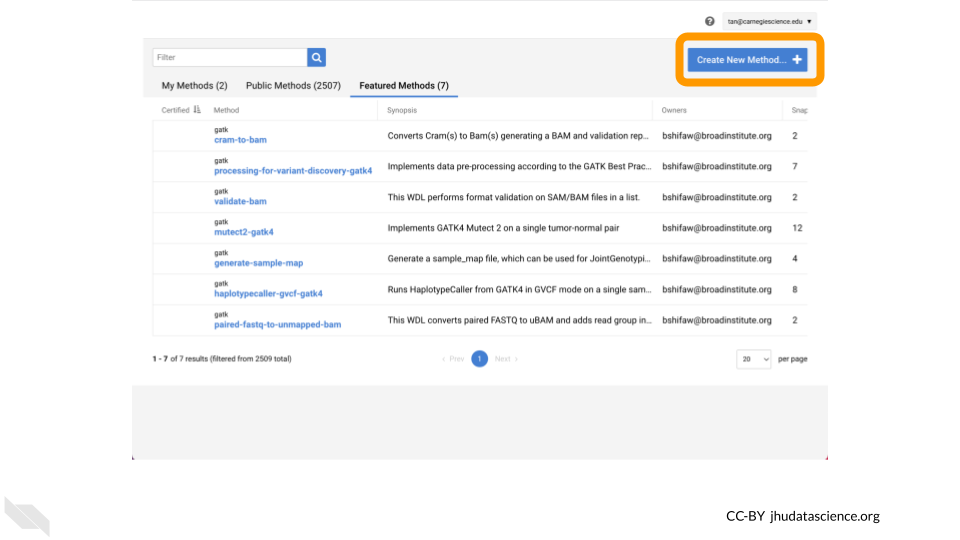
Click on the Find a Workflow card.



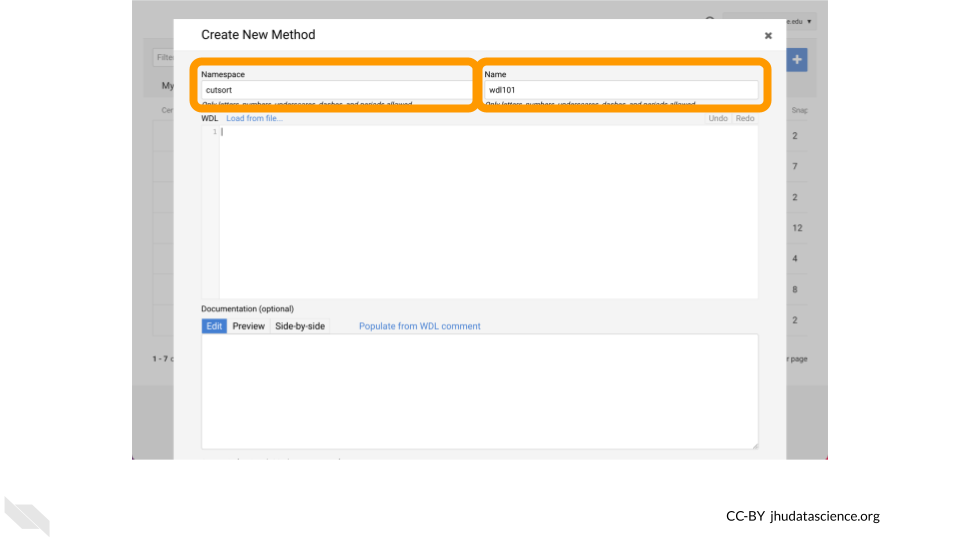
Select the Broad Methods Repository option.



Click Create New Method.



Add a namespace to the first text box to organize your WDLs. Your username (prepended with your lab name) is a reasonable namespace as this must be unique across all of Broad Methods Repository. Afterwards, add a name such as wdl101 to name your WDL.



## 3.2 Write WDL101 Training Example

Let’s now create a basic WDL! This simple “Hello, World!” style workflow will take as input a string, call a single task, and save the output of that task to your workspace bucket. The task that is called will run the [Bash](https://swcarpentry.github.io/shell-novice/01-intro/index.html) echo command to print the input string to stdout.

First note that we are using the [WDL 1.0 spec](https://github.com/openwdl/wdl/tree/main/versions).

version 1.0

Let’s add a workflow HelloInput that calls a single task WriteGreeting.

version 1.0  
workflow HelloInput {  
}  
  
task WriteGreeting{  
}

To create the task, we will define input, command, output, and runtime blocks. Note that the command block is defined as a “[here doc](https://en.wikipedia.org/wiki/Here_document)” and prints the input string to stdout.

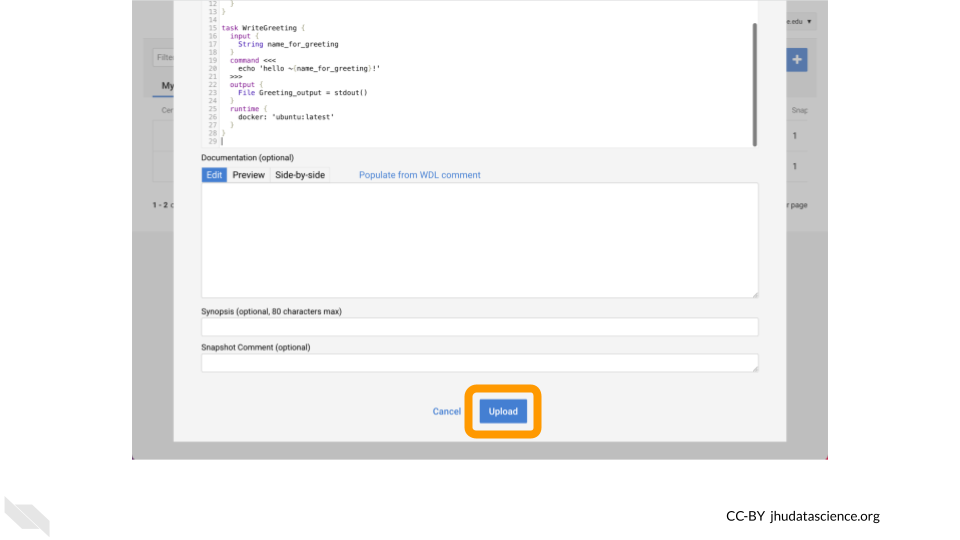
version 1.0  
workflow HelloInput {  
}  
  
task WriteGreeting {  
 input {  
 String name\_for\_greeting  
 }  
 command <<<  
 echo 'hello ~{name\_for\_greeting}!'  
 >>>  
 output {  
 File Greeting\_output = stdout()  
 }  
 runtime {  
 docker: 'ubuntu:latest'  
 }  
}

Putting it all together, we now create the workflow by defining an input string stored in a variable named name\_input, calling the task by passing name\_input to name\_for\_greeting, and storing what is returned by the task in a File labeled final\_output.

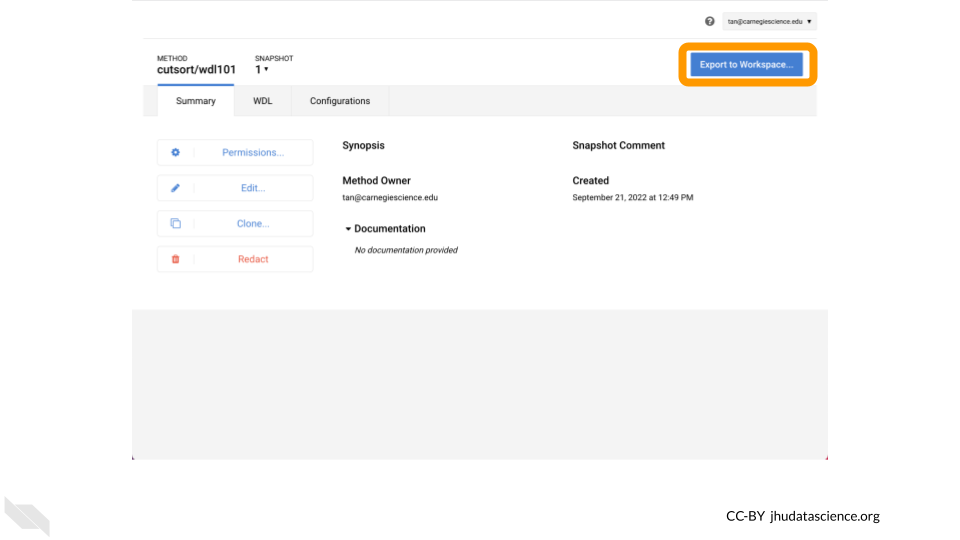
version 1.0  
workflow HelloInput {  
 input {  
 String name\_input  
 }  
 call WriteGreeting {  
 input:   
 name\_for\_greeting = name\_input  
 }  
 output {  
 File final\_output = WriteGreeting.Greeting\_output  
 }  
}  
  
task WriteGreeting {  
 input {  
 String name\_for\_greeting  
 }  
 command <<<  
 echo 'hello ~{name\_for\_greeting}!'  
 >>>  
 output {  
 File Greeting\_output = stdout()  
 }  
 runtime {  
 docker: 'ubuntu:latest'  
 }  
}

## 3.3 Export to AnVIL and run

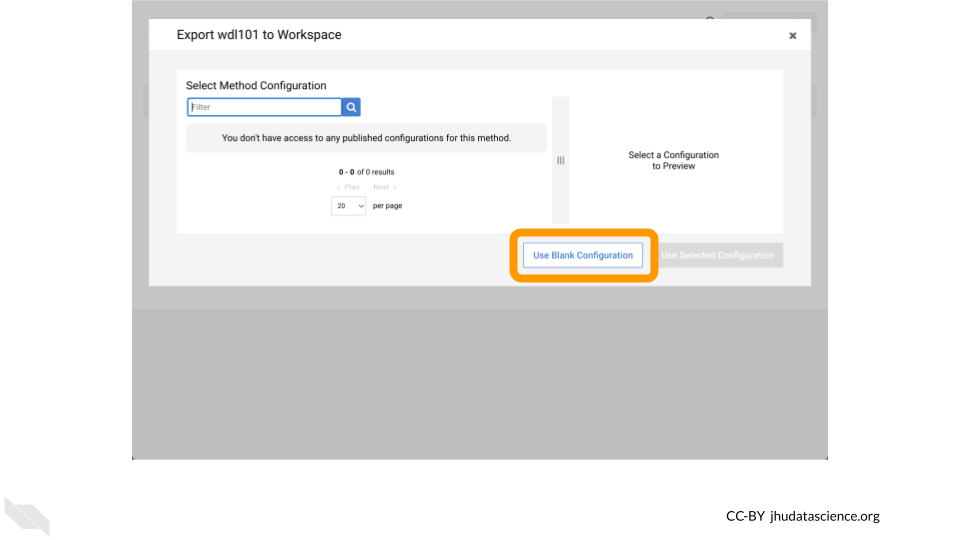
Once your WDL is complete, click on Upload.



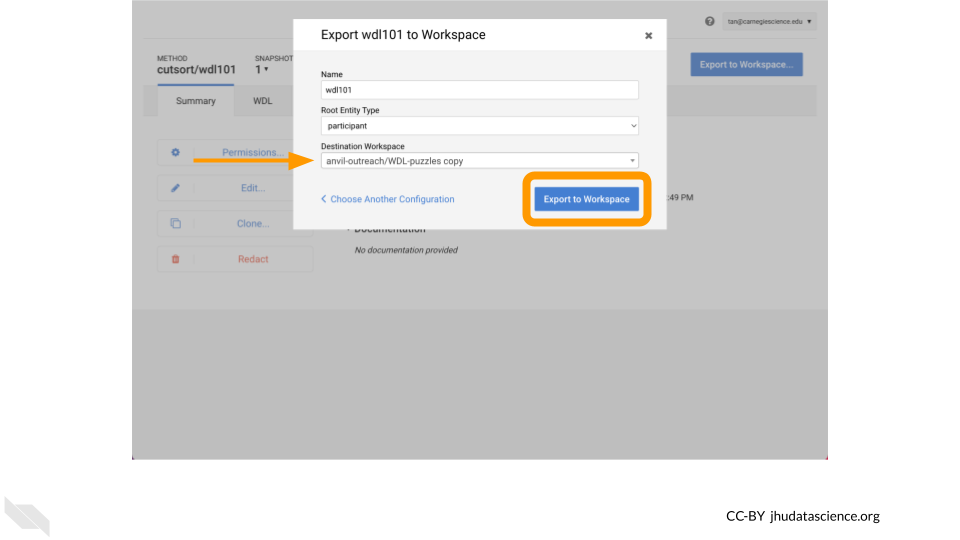
Now click on Export to Workspace.



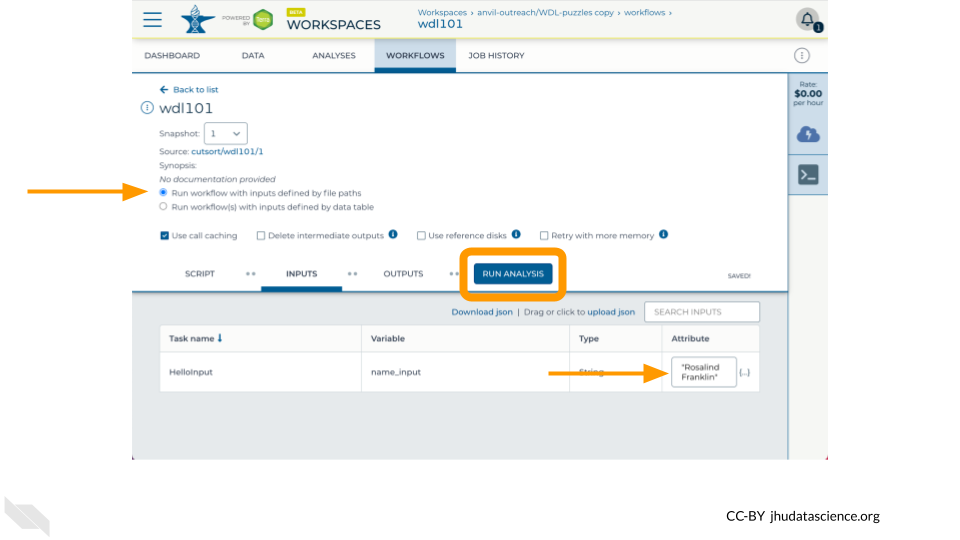
Select Use Blank Configuration.



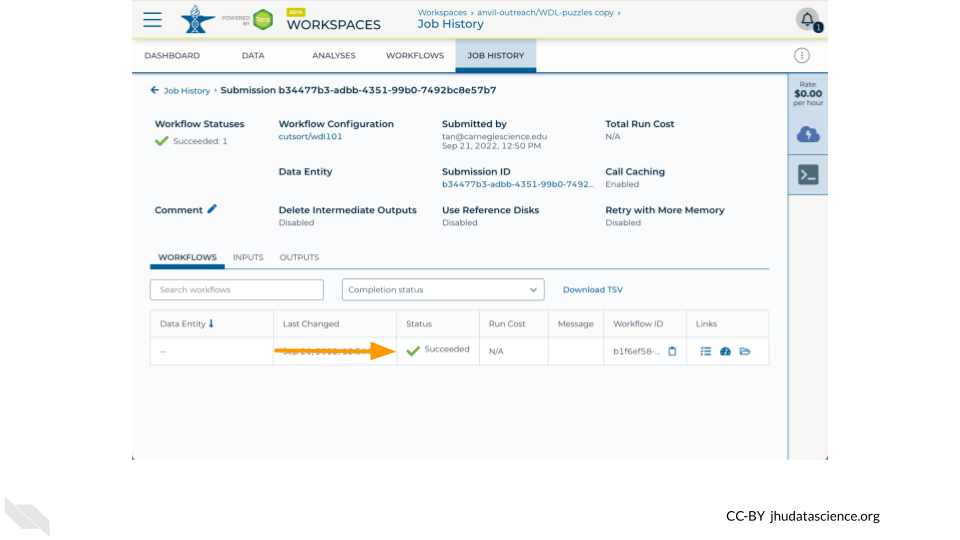
Select a Destination Workspace such as your clone of WDL-puzzles. Afterwards, click Export to Workspace.



Lastly, configure your Workflow as your did previously (e.g. inputs defined by file paths, name in double quotes), click Save, and then click Run Analysis.



Voila! Here’s what you hopefully see after successfully running your WDL101 Training Example !



# 4 Localize Files

# 5 Customize Docker

# 6 Join Discourse

# About the Authors

These credits are based on our [course contributors table guidelines](https://github.com/jhudsl/DaSL_Course_Template_Bookdown/wiki/How-to-give-credits).

| Credits | Names |
| --- | --- |
| **Pedagogy** |  |
| Lead Content Instructor(s) | [FirstName LastName](link%20to%20personal%20website) |
| Lecturer(s) (include chapter name/link in parentheses if only for specific chapters) - make new line if more than one chapter involved | Delivered the course in some way - video or audio |
| Content Author(s) (include chapter name/link in parentheses if only for specific chapters) - make new line if more than one chapter involved | If any other authors besides lead instructor |
| Content Contributor(s) (include section name/link in parentheses) - make new line if more than one section involved | Wrote less than a chapter |
| Content Editor(s)/Reviewer(s) | Checked your content |
| Content Director(s) | Helped guide the content direction |
| Content Consultants (include chapter name/link in parentheses or word “General”) - make new line if more than one chapter involved | Gave high level advice on content |
| Acknowledgments | Gave small assistance to content but not to the level of consulting |
| **Production** |  |
| Content Publisher(s) | Helped with publishing platform |
| Content Publishing Reviewer(s) | Reviewed overall content and aesthetics on publishing platform |
| **Technical** |  |
| Course Publishing Engineer(s) | Helped with the code for the technical aspects related to the specific course generation |
| Template Publishing Engineers | [Candace Savonen](https://www.cansavvy.com/), [Carrie Wright](https://carriewright11.github.io/) |
| Publishing Maintenance Engineer | [Candace Savonen](https://www.cansavvy.com/) |
| Technical Publishing Stylists | [Carrie Wright](https://carriewright11.github.io/), [Candace Savonen](https://www.cansavvy.com/) |
| Package Developers ([Leanbuild](https://github.com/jhudsl/leanbuild)) | [John Muschelli](https://johnmuschelli.com/), [Candace Savonen](https://www.cansavvy.com/), [Carrie Wright](https://carriewright11.github.io/) |
| **Art and Design** |  |
| Illustrator(s) | Created graphics for the course |
| Figure Artist(s) | Created figures/plots for course |
| Videographer(s) | Filmed videos |
| Videography Editor(s) | Edited film |
| Audiographer(s) | Recorded audio |
| Audiography Editor(s) | Edited audio recordings |
| **Funding** |  |
| Funder(s) | Institution/individual who funded course including grant number |
| Funding Staff | Staff members who help with funding |

## ─ Session info ───────────────────────────────────────────────────────────────  
## setting value   
## version R version 4.0.2 (2020-06-22)  
## os Ubuntu 20.04.5 LTS   
## system x86\_64, linux-gnu   
## ui X11   
## language (EN)   
## collate en\_US.UTF-8   
## ctype en\_US.UTF-8   
## tz Etc/UTC   
## date 2023-03-23   
##   
## ─ Packages ───────────────────────────────────────────────────────────────────  
## package \* version date lib source   
## assertthat 0.2.1 2019-03-21 [1] RSPM (R 4.0.5)   
## bookdown 0.24 2023-03-10 [1] Github (rstudio/bookdown@88bc4ea)   
## cachem 1.0.7 2023-02-24 [1] CRAN (R 4.0.2)   
## callr 3.5.0 2020-10-08 [1] RSPM (R 4.0.2)   
## cli 3.6.0 2023-01-09 [1] CRAN (R 4.0.2)   
## crayon 1.3.4 2017-09-16 [1] RSPM (R 4.0.0)   
## desc 1.2.0 2018-05-01 [1] RSPM (R 4.0.3)   
## devtools 2.3.2 2020-09-18 [1] RSPM (R 4.0.3)   
## digest 0.6.25 2020-02-23 [1] RSPM (R 4.0.0)   
## ellipsis 0.3.1 2020-05-15 [1] RSPM (R 4.0.3)   
## evaluate 0.20 2023-01-17 [1] CRAN (R 4.0.2)   
## fastmap 1.1.1 2023-02-24 [1] CRAN (R 4.0.2)   
## fs 1.5.0 2020-07-31 [1] RSPM (R 4.0.3)   
## glue 1.4.2 2020-08-27 [1] RSPM (R 4.0.5)   
## htmltools 0.5.4 2022-12-07 [1] CRAN (R 4.0.2)   
## knitr 1.33 2023-03-10 [1] Github (yihui/knitr@a1052d1)   
## magrittr 2.0.3 2022-03-30 [1] CRAN (R 4.0.2)   
## memoise 2.0.1 2021-11-26 [1] CRAN (R 4.0.2)   
## pkgbuild 1.1.0 2020-07-13 [1] RSPM (R 4.0.2)   
## pkgload 1.1.0 2020-05-29 [1] RSPM (R 4.0.3)   
## prettyunits 1.1.1 2020-01-24 [1] RSPM (R 4.0.3)   
## processx 3.4.4 2020-09-03 [1] RSPM (R 4.0.2)   
## ps 1.4.0 2020-10-07 [1] RSPM (R 4.0.2)   
## R6 2.4.1 2019-11-12 [1] RSPM (R 4.0.0)   
## remotes 2.2.0 2020-07-21 [1] RSPM (R 4.0.3)   
## rlang 1.0.6 2022-09-24 [1] CRAN (R 4.0.2)   
## rmarkdown 2.10 2023-03-10 [1] Github (rstudio/rmarkdown@02d3c25)  
## rprojroot 2.0.3 2022-04-02 [1] CRAN (R 4.0.2)   
## sessioninfo 1.1.1 2018-11-05 [1] RSPM (R 4.0.3)   
## stringi 1.5.3 2020-09-09 [1] RSPM (R 4.0.3)   
## stringr 1.4.0 2019-02-10 [1] RSPM (R 4.0.3)   
## testthat 3.0.1 2023-03-10 [1] Github (R-lib/testthat@e99155a)   
## usethis 1.6.3 2020-09-17 [1] RSPM (R 4.0.2)   
## withr 2.3.0 2020-09-22 [1] RSPM (R 4.0.2)   
## xfun 0.26 2023-03-10 [1] Github (yihui/xfun@74c2a66)   
## yaml 2.2.1 2020-02-01 [1] RSPM (R 4.0.3)   
##   
## [1] /usr/local/lib/R/site-library  
## [2] /usr/local/lib/R/library

# References