Introduction to Workflow Management Solutions for Public Health Bioinformatics

Model for Distributed Public Health Bioinformatics: Week 1 – Introduction to Workflow Management Using WDL

Wednesday April 27th, 2022

Kevin G. Libuit, MS | Theiagen Genomics

Training Workshop Overview

Communication and Support

- Slack workspaces:
  - **Terra-US-PHL; #wdl-writing**
  - **StaPH-B; #workflow-management, #cromwell_noobz**
- Weekly Office Hours:
  - **Mountain Region - Friday 9-10AM (PT)**
  - **North East Region - Friday 10-11AM (PT)**

Training Workshop Overview

Main Course Objective

Learn how to use **workflow management systems** to **develop accessible, interoperable, and reproducible** public health bioinformatics solutions

Training Workshop Overview

This Workshop is an Introductory Course

Great resources for more information regarding Workflow Management:

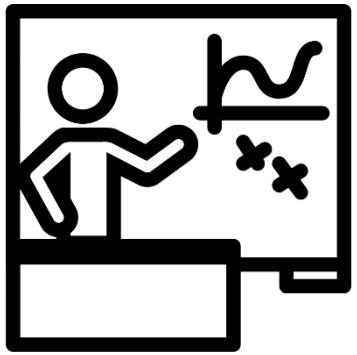
- **Compute Workflow Management**
  - Danny Park's StaPH-B Talk
- **Genomics in the Cloud: Using Docker, GATK, and WDL in Terra**
  - Geraldine A. Van der Auwera & Brian D. O'Connor
- **Lynn Langit YouTube Channel**

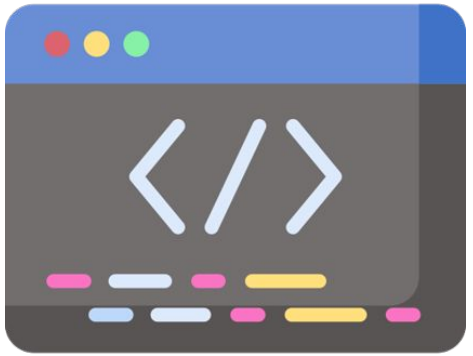
Training Workshop Overview

Course Layout

120m sessions broken up into two sections each:

1. Lectures to **provide conceptual background information**
2. Hands-on exercise to **demonstrate how to use workflow management solutions** and apply lecture materials





GCP VMs have been provisioned to each trainee to carry out hands-on exercises

Training Workshop Overview

Week 1

Lecture Material: Intro to Workflow Management Using WDL

Hands-on Exercise: Writing a Single-Task WDL Workflow

Week 2

Lecture Material : Deeper Dive into WDL Workflows

Hands-on Exercise: Writing a Multi-Task WDL Workflow

Training Workshop Overview

Week 3

Lecture Material: Making WDL Workflows Accessible through Dockstore and Terra.Bio

Hands-on Exercise: Publishing a WDL Workflow on Dockstore and Launching through Terra.Bio

Week 4

Lecture Material: Workflow Management with Nextflow & Nextflow Tower

Hands-on Exercise: Writing a Nextflow Workflow

Training Workshop Overview

Week 1

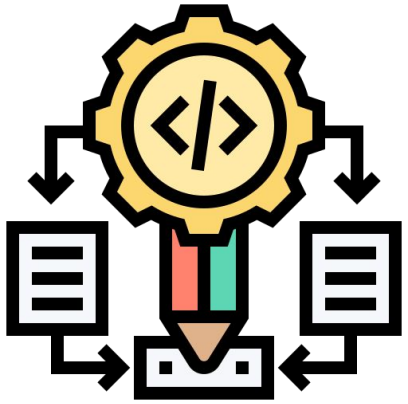
Lecture Material: Intro to Workflow Management Using WDL

Hands-on Exercise: Writing a Single-Task WDL Workflow

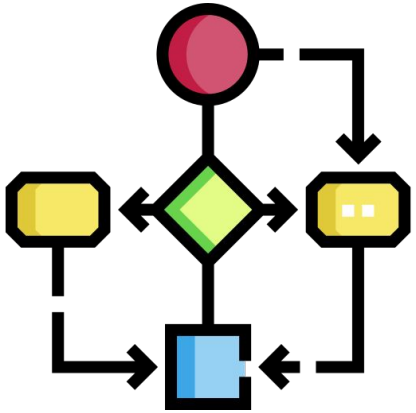
Introduction to Workflow Managers

Workflow managers provide a **standardized framework** for creating reproducible and interoperable **bioinformatics pipelines**, especially when **containerized software** are **utilized**

Workflow managers help to **standardize the process** of writing bioinformatics pipelines



Analytical Module
Individual step performing a single and specific bioinformatics job

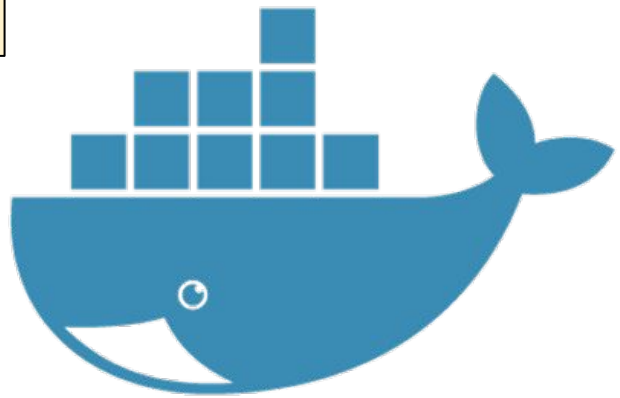


Bioinformatics Pipeline
Software that strings together multiple analytical modules

Containerized Software

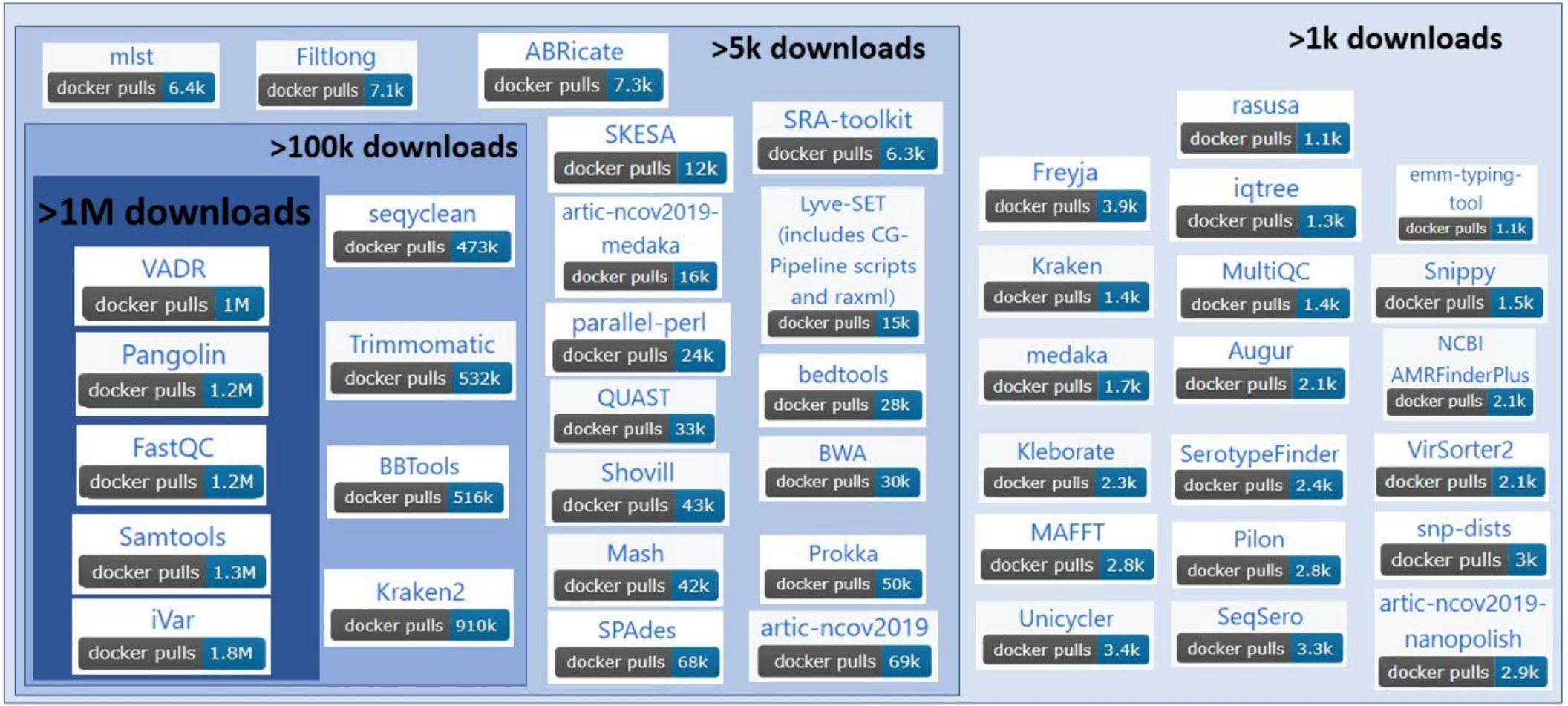
*"Containerization involves **bundling an application** together with all of the necessary configuration files, libraries, and dependencies to **ensure the software can run in a reproducible fashion** across a diversity of computing environments."*
StaPH-B Docker Builds User Guide

Address the challenge of **complex dependency libraries** for accessing bioinformatics software



StaPH-B/docker-builds

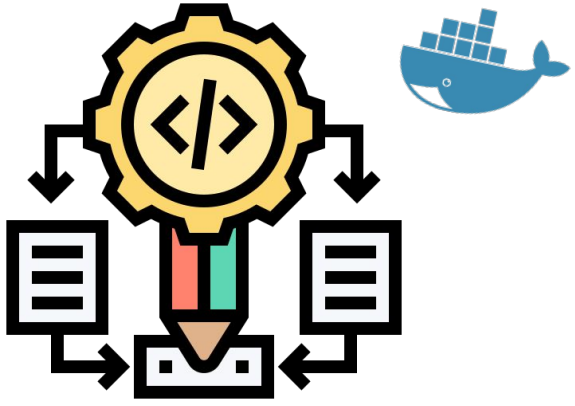
Most downloaded Docker images
of pulls reported by DockerHub as of 2022-04-17



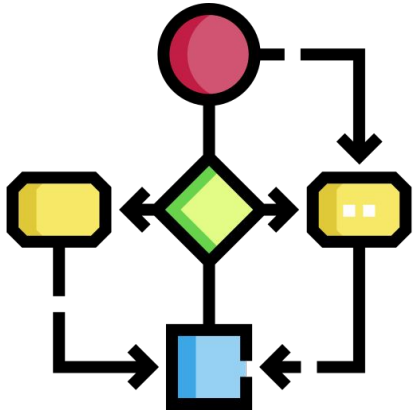
Twitter, @CurtisKapsak, 2022-04-26

Introduction to Workflow Managers

Workflow managers provide a **standardized framework** for creating reproducible and interoperable **bioinformatics pipelines**, especially when **containerized software are utilized**

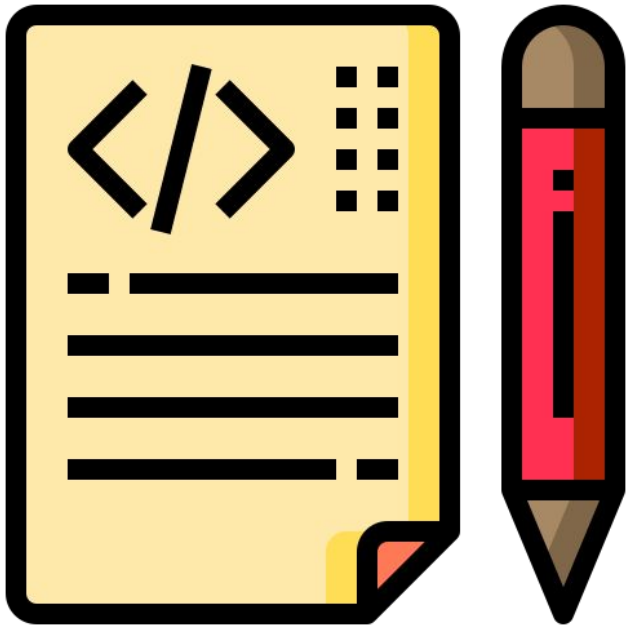


Analytical Module
Individual step performing a single and specific bioinformatics job



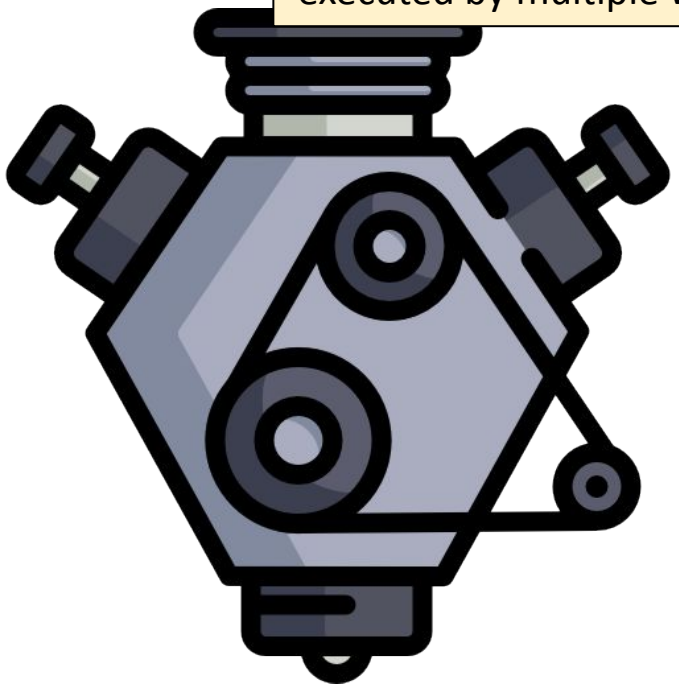
Bioinformatics Pipeline
Software that strings together multiple analytical modules

Workflow Languages and Engines



Workflow Language
Programming language used to describe the bioinformatics pipeline

Some workflow languages can be executed by multiple workflow engines



Workflow Engine
Software to interpret and execute the workflow itself

WDL Workflows

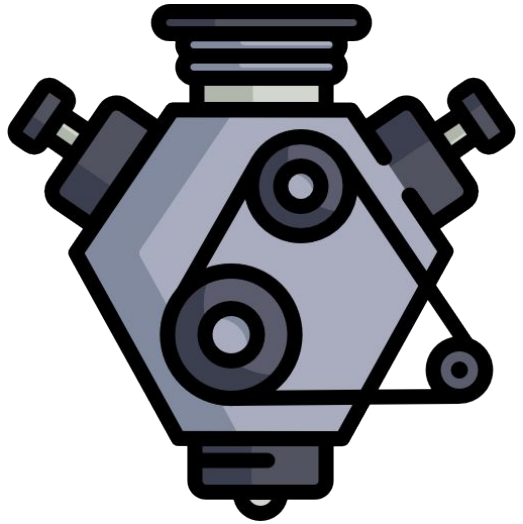
Workflow Description Language (WDL)

- Widely-utilized workflow manager for writing reproducible and interoperable bioinformatics pipelines
  - WDL language specifications available on open/wdl GitHub repository
- WDL workflows can be **accessed and executed on the Terra.Bio platform**
  - Web application that connects users to WDL workflows & dynamic cloud computing resources (GCP) through a clean and intuitive user interface

WDL Workflows

WDL Execution Engines

- *WDL workflows require an execution engine to run in and of itself, but requires an execution engine to run*
- Compliant execution engines:
  - Cromwell
    - Utilized by the Terra.Bio platform
  - MiniWDL
    - Recommended for local testing and development
  - dxWDL

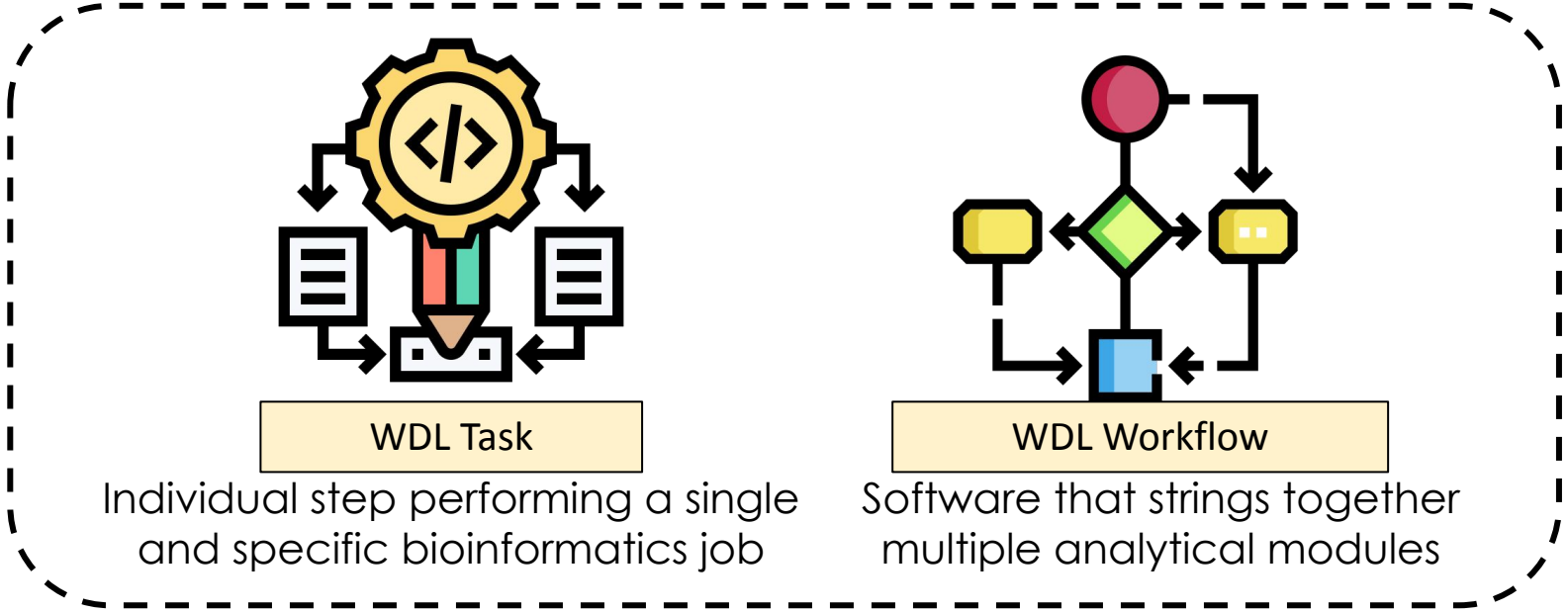


See WDL Specification documentation for additional information

WDL Workflows

WDL Task and Workflow Files

- WDL Tasks define individual analytical modules to perform a specific bioinformatics job
- WDL Workflows define the bioinformatics pipeline itself
  - Made up of individual WDL Tasks



WDL Task

WDL Task Elements

- Input
  - Designates the task input data and parameters
- Command
  - Defines the executables that are evaluated and executed when the task is called
- Output
  - Defines outputs of the task after a call to the task completes successfully
- Runtime
  - Defines the runtime environment in which the task in executed

```
version 1.0

task shovill_pe {
  input {
    File read1_cleaned
    File read2_cleaned
    String samplename
    String docker = "quay.io/staphb/shovill:1.1.0"
    Int min_contig_length = 200
  }
  command <<<
    shovill --version | head -1 | tee VERSION
    shovill \
    --outdir out \
    --R1 ~{read1_cleaned} \
    --R2 ~{read2_cleaned} \
    --minlen ~{min_contig_length}
    mv out/contigs.fa out/~{samplename}_contigs.fasta
    mv out/contigs.gfa out/~{samplename}_contigs.gfa
  >>>
  output {
    File assembly_fasta = "out/~{samplename}_contigs.fasta"
    File contigs_gfa = "out/~{samplename}_contigs.gfa"
    String shovill_version = read_string("VERSION")
  }
  runtime {
    docker: "~{docker}"
    memory: "16 GB"
    cpu: 4
    disks: "local-disk 100 SSD"
    preemptible: 0
  }
}
```

`task_shovill.wdl` file from Theiagen's Public Health Bacterial Genomics (PHBG) Repository

```
version 1.0

task pmga {
    meta {
        description: "Serogrouping and serotyping of all Neisseria species and Haemophilus influenzae"
    }

    input {
        File assembly
        String samplename
        String docker = "quay.io/staphb/pmga:3.0.2"
        Int? cpu = 4
    }

    command <<<
        echo $(pmga --version 2>&1) | sed 's/.*pmga //; s/ .*\$//' | tee VERSION
        pmga \
            ~{assembly} \
            --blastdir /data/blastdbs \
            --threads ~{cpu} \
            --prefix ~{samplename}

        # Parse pmga TSV
        # https://github.com/rpetit3/pmga#pmga-output-files
        cut -f 2 pmga/~{samplename}.txt | tail -n 1 | tee PMGA_SPECIESDB
        cut -f 3 pmga/~{samplename}.txt | tail -n 1 | tee PMGA_SEROTYPE
        cut -f 4 pmga/~{samplename}.txt | tail -n 1 | tee PMGA_GENES
        cut -f 5 pmga/~{samplename}.txt | tail -n 1 | tee PMGA_NOTES
    >>>

    output {
        String version = read_string("VERSION")
        String docker = "~{docker}"
        String pmga_speciesdb = read_string("PMGA_SPECIESDB")
        String pmga_serotype = read_string("PMGA_SEROTYPE")
        String pmga_genes = read_string("PMGA_GENES")
        String pmga_notes = read_string("PMGA_NOTES")
        File pmga_results = "./pmga/~{samplename}.txt"
        File pmga_allele_matrix = "./pmga/~{samplename}-allele-matrix.txt"
        File pmga_blast_final = "./pmga/~{samplename}-blast-final-results.json.gz"
        File pmga_blast_raw = "./pmga/~{samplename}-blast-raw-results.json.gz"
        File pmga_loci_counts = "./pmga/~{samplename}-loci-counts.txt"
        File pmga_gff = "./pmga/~{samplename}.gff.gz"
    }
```

```
    runtime {
        docker: "~{docker}"
        memory: "8 GB"
        cpu: 4
        disks: "local-disk 50 SSD"
        preemptible: 0
    }
}
```

task_pmga.wdl file from
Theiagen's Public Health Bacterial Genomics (PHBG) Repository

WDL Workflow

WDL Workflow Elements

- Input
  - Designates the workflow input data and parameters
- Call Statements
  - Defines the WDL tasks to execute as part of the WDL Workflow
- Output
  - Defines outputs of the workflow after a call to the workflow completes successfully

```
version 1.0

import "../tasks/tools/task_pmga.wdl" as pmga
import "../tasks/task_versioning.wdl" as versioning

workflow pmga_wf {
    input {
        File assembly
        String samplename
    }
    call pmga.pmga {
        input:
            assembly = assembly,
            samplename = samplename
    }
    call versioning.version_capture{
        input:
    }
    output {
        String pmga_wf_version = version_capture.phbg_version
        String pmga_wf_analysis_date = version_capture.date
        String pmga_version = pmga.version
        String pmga_docker = pmga.docker
        String pmga_speciesdb = pmga.pmga_speciesdb
        String pmga_serotype = pmga.pmga_serotype
        String pmga_genes = pmga.pmga_genes
        String pmga_notes = pmga.pmga_notes
        File pmga_results = pmga.pmga_results
        File pmga_allele_matrix = pmga.pmga_allele_matrix
        File pmga_blast_final = pmga.pmga_blast_final
        File pmga_blast_raw = pmga.pmga_blast_raw
        File pmga_loci_counts = pmga.pmga_loci_counts
        File pmga_gff = pmga.pmga_gff
    }
}
```

`wf_pmga.wdl` file from Theiagen's Public Health Bacterial Genomics (PHBG) Repository

Introduction to Workflow Management Using WDL

Major Takeaways:

- **Workflow managers** provide a **standardized framework** for creating reproducible and interoperable **bioinformatics pipelines**, especially when **containerized software are utilized**
  - Containerized software address the challenge of complex dependency libraries for accessing bioinformatics software
- **Workflow languages** are used to describe the workflow
- **Workflow engines** are used to execute and run the workflow

Introduction to Workflow Management Using WDL

Major Takeaways:

- **Workflow Description Language (WDL)** - Widely-utilized workflow manager for writing reproducible and interoperable bioinformatics pipelines
  - **WDL Task** files define individual analytical modules to perform a specific bioinformatics job
  - **WDL Workflow** define the bioinformatics pipeline itself
    - Made up of individual WDL Tasks

Workflow Management Training

10 Minute Session Break Before Hands-On Exercise

We will begin again at _____

Introduction to Workflow Management Using WDL

Major Takeaways:

- Workflow Description Language (WDL) - Widely-utilized workflow manager for writing reproducible and interoperable bioinformatics pipelines
  - **WDL Task** files define individual analytical modules to perform a specific bioinformatics job
  - **WDL Workflow** define the bioinformatics pipeline itself
    - Made up of individual WDL Tasks

Training Workshop Overview

Week 1

Lecture Material: Intro to Workflow Management Using WDL

Hands-on Exercise: Writing a Single-Task WDL Workflow

- To follow along with today's exercise, access your GCP VM and navigate to the wm_training GitHub repository

Workflow Management Training

15m to complete Exercise 1.1

We will begin again at _____

Workflow Management Training

5m to complete Exercise 2.1

We will begin again at _____

Workflow Management Training

30m to complete Exercise 2.3

We will begin again at _____