SISG Module 12 - **C**ancer **G**enomics **C**loud Platform

CGC - Practice 1 - gzip App

1. **Go** to CGC - https://cgc.sbgenomics.com
2. **Sign up** for the CGC - https://docs.cancergenomicscloud.org/docs/sign-up-for-the-cgc
3. **Create a project** on CGC
   - Click the **Projects tab** on the top navigation bar and click **Create a project** button from the drop-down menu.
   - **Name** your project.
   - Click **Create** to finish.
4. **Download and run** Rabix Composer - http://rabix.io/
5. **Connect** Rabix Composer with CGC
   To add a CGC account from the **Settings tab** in Rabix Composer:
   - Click **Add an Account.**
   - In the popup, select the **Cancer Genomics Cloud** platform from the dropdown menu and specify your platform authentication token. Access your token here.
   - Click **Add**. Notice a new folder for the platform in the left-hand navigation pane. You can now open existing projects from CGC and add and edit apps in those projects.
6. Create the **gzip** tool in Rabix Composer in your project on CGC
   - **Create** the new tool
     - Click **Home**, then select **New Command Line Tool**
       - Choose **Platform**
       - App name: **gzip**
       - CWL Version: **v1.0**
       - Pick the **destination** project where app will be created
       - Click **Create**
   - The tool editor opens
   - Specify the **Docker** Repository: *alpine*
   - Specify the **base command:** *gzip*
   - Specify the **arguments:** *-k* and *-c* (shellQuote - **No**)
   - Specify the **input:**
     - Add an Input (shellQuote - **No**)
     - Edit input **ID** to *input_file*
     - Edit input Label and Description in something more human readable
       - *(Label: Input file, Description: Input file to gzip)*
   - **Command line** preview should look like this:
     ```
     gzip -k -c /path/to/input.ext
     ```
   - Specify the **output:**
     - Add an Output
     - Set **Glob** to **.gz*
     - Also, edit Label and Description into something intuitive
   - Specify the **Computational Resources** - Memory to *1 GB* and CPU to *single-thread*

- **Redirect** the output from stdout to file
  - In **OTHER** section find **Stdout redirect** and click on **</>** symbol
    - This is where you can write JavaScript expressions
  - Write the following JavaScript code in **Expression Editor**:

```
${
    return inputs.input_file.path.split('/').pop() + ".gz"
}
```

  - Click **Save**
  - *> input.ext* should append on your command line
- **Save** your tool by clicking the Disk icon in the upper right corner of the Composer
  - Write some intuitive **revision note** and click **Push**
- Now your tool is in your project on CGC, let's run it!

7. **Run task** on gzip app on CGC using Public Reference Files
   - Open CGC, click on Projects and navigate to the project you created in step 3
   - Click on **Apps**
   - Find **gzip** app and click **Run**
   - Click on **Select file(s)** button and choose **Public Reference Files** tab
   - Search for *NA12878_GRCh38_exome_subset.vcf* file and select it
   - Click **Save selection** and **Copy**
   - You are now back on the **Task** page, make sure that Spot Instances are **On**
   - Click **Run** to run the task

Cancer Genomics Cloud - Practice 2 - vcf2gds App

1. Create the **vcf2gds** tool in Rabix Composer in your project on CGC
   - **Create** the new tool
     - Click **Home**, then select **New Command Line Tool**
       - Choose **Platform**
       - App name: **vcf2gds**
       - CWL Version: **v1.0**
       - Pick the **destination** project where app will be created
       - Click **Create**
   - The tool editor opens
   - Specify the **Docker** Repository: *uwgac/topmed-master:latest*
   - Specify the **base command:** *Rscript /usr/local/analysis_pipeline/R/vcf2gds.R vcf2gds.config*
   - Specify the **inputs:**
     - Add an Input
     - Edit input **ID** to *vcf_file*
     - Set **Include in the command line** to *No*
     - Set **Required** to *Yes*
     - Add another Input
     - Edit input **ID** to *gds_file_name*
     - Edit input **Type** to *string*

    - Set **Include in the command line** to *No*
    - Set **Required** to *Yes*
- **Command line** preview should look like this:

```
Rscript /usr/local/analysis_pipeline/R/vcf2gds.R vcf2gds.config
```

- Specify the **output:**
    - Add an Output
    - Set **Glob** to **.gds*
- Specify the **FILE REQUIREMENTS**
    - Here we will create the config file for running this script
    - Click **Add** in FILE REQUIREMENTS section of the tool editor
    - Select **File**
    - Set File Name on the right side to *vcf2gds.config*
    - **Writing** content of the config file:
        - In the **File Content** window click on **</>** button
        - This will open JavaScript expression editor
        - Write the following JavaScript code in **Expression Editor**:

```
${
    config = "vcf_file \"" + inputs.vcf_file.path + "\"\n"
    config += "gds_file \"" + inputs.gds_file_name + "\"\n"
    return config
}
```

        - Click **Save**
- **Save** your tool by clicking the **Disk** icon in the upper right corner of the Composer
- Write some intuitive **revision note** and click Push
- Now your tool is in your project on **CGC**, let's run it!

2. **Run task** on vcf2gds app on CGC
    - Open CGC, click on Projects and navigate to the project you created in step 3 of Practice 1
    - First we need to upload vcf.gz file that will be converted to gds on CGC
        - Open **Files** tab of the project
        - Click **Add files** button
        - Navigate to FTP/HTTP tab
        - Copy this URL *https://github.com/UW-GAC/analysis_pipeline/raw/master/testdata/1KG_phase3_subset_chr1.vcf.gz* into the text field
        - Click Import
    - Click on **Apps**
    - Find **vcf2gds** app and click **Run**
    - Click on **Select file(s)** button and choose **Public Reference Files** tab
    - Search for *NA12878_GRCh38_exome_subset.vcf* file and select it
    - Click **Save selection** and **Copy**
    - You are now back on the **Task** page, make sure that Spot Instances are **On**
    - Click **Run** to run the task