Variant Explorer Tool
Version 1.0.0
September 2015

Content:

1 - Description

This tool is used on a Mac OS only. It is a variant explorer interface tool for the patients of the DDD project.
The variant explorer tool allows you to find variants spanning a genomic location if you have one of the following: a genomic location (chromosome: start - stop), a gene name or an HGVS term. It also provides other useful options for exploring variants.

2 - Module dependencies and installation

The interface explorer tool uses a python module "Pillow" for managing images. Pillow is not part of the python standard library therefore the first thing to do is to find out if the Pillow module is in your python library. In terminal type the following:

```
python
import PIL
```

If no error message appears, move on to part 3. Otherwise you will install this module through "PIP" because on some OS, "clang" is missing and needs to be installed if you were to install "Pillow" from source code. Place the "PIP" compressed folder on your Desktop, open a new terminal session and type the following:

```
cd Desktop
tar zxvf pip_7.1.0.tar.gz
cd pip_7.1.0
python setup.py install --user
/absolute/path/to/library/bin/pip install pillow --user
```

3 - Setting up the interface

Download the interface directory containing all necessary files and preferably place this directory on your Desktop. The first thing to do before launching the interface is to edit the 3 flat files that should

contain necessary login information. The names of these files start with a dot "." that makes them invisible.

Note: if you can not see files starting with a dot in the interface folder, try to edit the files from terminal or change the Finder defaults to be able to see the dot files by typing in terminal this line "defaults write com.apple.finder AppleShowAllFiles YES" and then re-launching Finder by holding the <ALT> key while pressing right click and choosing re-launch.

The 3 flat files:
Note: remove the square brackets when you edit the files with your information.

- The *.server_user* file:
  [server name]
  [your user name]
  [*your user name password] (this line is optional, do not fill in your password in case your ssh authentication is automatic)

- The *.ddd_prod_user* file:
  [find this data in the email sent about this interface]

- The *.igv_user* file:
  [user name for the IGV API]
  [your user name password]

Which servers to use?
Use one of the following servers: gen1, farm3-login, farm3-head1, farm3-head2, farm3-head3, farm3-head4, ddd-vm4.

4 - Launching the interface and troubleshooting

It is **NECESSARY** that you connect to the Sanger network through the 'network connect' on your Mac or by being on Sanger wired connection.

- Launching approach 1: double click on the "variant_explorer_tool.app" file.

OR

- Launching approach 2: in terminal type "python variant_explorer_tool/index.py".

Notes: If you launched the interface with approach 2, you will see some lines in terminal as the interface is running for example prompting you to enter a password (etc). **DO NOT** type your password or interact with these prompts because the interface will do this automatically for you.

In the case were the interface launches successfully from terminal but is not launching when you double click the "variant_explorer_tool.app" file do the following:
Applications -> Utilities -> activity monitor -> force quit variant_explorer_tool -> retry (double click the variant_explorer_tool.app file).

5 - Usage

Launch the interface to start and depending on what you are searching for you can:

A - Use the left sidebar to:

1) Find out the frequency of a variant in the cohort using a chromosome and start position:
Click 'variant frequency' in the sidebar to the left -> enter the chromosome and genomic location -> in a minute or so a popup will appear showing how much this position occurred in the cohort VCFs.

2) Compute the genomic coordinates of a gene name or an HGVS term:
Type a gene name in the 'gene' entry in the 'genomic location calculator' in the sidebar to the left -> press 'Find' -> the location this gene spans and the chromosome will be displayed near the 'chromosome', 'start', 'stop' labels on the sidebar.
Click the 'Refresh' button to clear out previous query details -> enter a complete HGVS term -> an Ensemble term will take longer to decipher than a Refseq term due to the steps designed to do that -> the location will be again displayed in the display fields.

B - Use the top menu to:

B.1) For a known child ID (person stable ID or decipher ID), find their variants in a genomic position using the genomic location (chromosome: start - stop) directly or through a gene name or an HGVS term.
The variants within that region of this child and their parents are displayed. If the 'IGV plot' was selected, the IGV plot of this trio will also be displayed.
In case the child has 1 parent or no parents at all (singleton), variants of the child or parent members found will be displayed.
Variants will be displayed in a tab called 'Variants'. The edit tools used to process these variants that are a local copy (note that the original VCFs are not altered in any way) include:

Remove
click on 1 or more variant lines to highlight it/them -> click remove: selected variants will be removed from your list.
Find
click find -> enter a chromosome and start position: if found, the variant will be highlighted in green in all the trio individual variant lists.
Info
click a variant in a list to highlight it -> click 'Info': another window will appear showing the contents of the variant line sorted alphabetically.
Filter out
choose an option from the list eg. 'variants with <DEL> ALT': the variants satisfying this characteristic will be removed from the trio individuals lists.
Export
choose the name of the file that will be saved containing the variants of this trio and their IDs.

The tools for the IGV plot include:
- (minus sign)

this minus sign will zoom out the image displayed.

+ (plus sign)

this plus sign will zoom in the displayed image.

Reload flanking bases

from the popup that appears, choose the number of the flanking bases on either side of the query start location: the IGV plot will be re-computed on the backend side, transferred to your local changeable directory and displayed on the interface tool.

Export

save the IGV image PNG regardless of the size you currently have displayed on your interface and choose the file name.

B.2) Find the variants in a genomic location for all the VCFs found in the cohort.

It is advised to use query parameters such as the MAX_AF cutoff, this will decrease the number of overall variants and make the data transfer from the backend side faster.

The more general the query parameters, the larger the number of variants, the longer it will take to transfer the data.

The edit tools for the cohort variants are the same as the trio variant edit tools above. The cohort option does not allow an IGV plot retrieval.

6 - Workflow

<u>Directly using genomic coordinates:</u>

Query parameters of the child are entered -> a script is created locally by a script builder to be executed on the backend side under the DDD profile ->

It first converts the decipher ID to person stable ID or keeps the person stable ID ->

Uses ddd_prod to find the IDs of the parents of this child ID (uses the following relations from ddd_prod: trio, families) ->

ddd_prod is used to find the latest uber VCF paths for the individuals of this trio (uses the dataset relation in ddd_prod) ->

the tabix VCF file is used to get the variants of the genomic location chosen in the query parameters ->

variants are filtered according to any CQ or MAX_AF parameters ->

the variants of this trio are saved to a file that is transferred to a local changeable directory in the interface folder.

IGV option -> a script is created locally that is executed on the backend side and that will communicate with the IGV API and create the IGV plot under the DDD profile -> IGV ".png" transferred to the local changeable directory.

<u>Finding variant frequency in cohort:</u>

The paths of the current VCFs in the cohort are retrieved from the ddd_prod database. The ‘dataset’ table is used and the paths to all the uber VCFs that are currently ‘current’ are extracted.

The occurrence of the query location is counted in these VCFs using multiprocessing to speed up the operation. Finding the variants in this case is done in batches being run in parallel.

Compute gene name genomic coordinates:
The gene name chromosome, start and stop are retrieved from the ddd_prod database table 'ddd_gene_detail'.
This data is transferred through a file from backend to frontend and displayed on the interface.

Compute HGVS genomic locations:
The complete HGVS term is entered -> in case of an Ensemble transcript: a script will find the chromosome of this term using the Ensemble REST API ->
the chromosome result is used to search the DDD VEP tabix VCF file -> when the term is found in a line in this VCF the location will be saved and transferred to the frontend to be displayed on the interface.
-> in case of a Refseq term: a python module is used to decipher the term using resources in a public database.

7 - Developer notes

- Local script builders will integrate the user parameters into scripts that will be executed on the backend side under the DDD profile.
- Output files from the script execution will be transferred to the frontend changeable directory in the interface directory mostly as JSON files.
- The standard error of the interface python script being executed locally is redirected to the red box at the bottom of the interface display with the title 'Error record'.
- When the interface is launched, if the server user file is correct a temporary backend directory and a frontend one will be directly created in the user's home directory on the server and locally in the interface folder (recent_runs directory).
- The directory name is: 'temp_interface_dump_%S_%M_%H_%d_%m_%y' (%S: second, %M: minute, %H: hour, %d: day, %m: month, %y: year).
- When the user quits the interface or exists successfully, the backend directory will be automatically deleted from their backend home directory, so will the frontend directory.
- The backend temporary directory is used to receive frontend scripts and store temporary outputs that are transferred to the frontend side.

- Interface tool directory structure:
  "documentation.pdf": this file you are reading.
  "local_scripts": directory containing script builder scripts and parsing_setups.py.
  "index.py": is the interface __main__ script.
  "result_widget_setups": is a module containing the definitions of widgets used in the result display of the interface.
  "widget_layouts": is a module containing definitions of widget layouts.
  "recent_runs": the local changeable directory that will contain the temporary frontend directory created every time the interface launches.

- Customized definitions are imported for use.
- System commands are used within the main script being executed and they include using script builders to build scripts ready to be executed on the server. Also "expect" scripts are

used only in case of users that do manual password entry during ssh to allow automatic password authentication.

- The “Pillow” module should be installed.
- The operations in these scripts assume that the INFO field elements in the VCF lines are separated by “;” and that the CQ attribute elements are separated by either “,” or “|” and that this applies to the MAX_AF as well.
- How does the MAX_AF cutoff work?
  - all elements of the MAX_AF should be less than the user-defined cutoff => True
  - if the VCF line does not have a MAX_AF => True
  - if there is any character element (not numerical) in the MAX_AF list of the VCF line => False

- How does the MAX_AF value work?
  - if the VCF line does not have a MAX_AF => False
  - if the elements of the MAX_AF list are all equal to the user-defined value => True

- The main elements of the interface are shown in Figure 1.

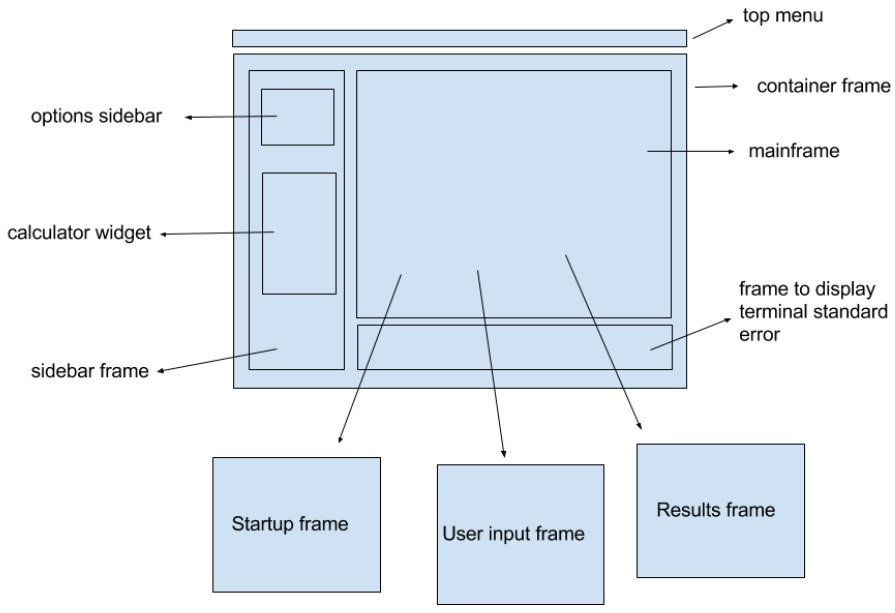

Figure 1: The general widgets of the interface.