Librator development notes

# Environment

Python 3.7 + PyQt5 + Qt Designer

* For Mac, install Xcode command line tool first

xcode-select --install

* Install python 3.7 (use anaconda or homebrew)

brew install python3

* Install PyQt5 (use pip, this not work for PyCharm)

pip3 install pyqt5

* Install PyQt5 for PyCharm

File -> Other Settings -> References for New Projects -> Project Interpreter

Then install PyQt5 and PyQt5-sip

* Install Qt Designer

brew cask install qt-creator

* Install PySQLite

pip install pysqlite3

* Install pyinstaller

pip install pyinstaller

Use pyinstaller to generate .spec file:

pyinstaller -w MainLibrator.py

Edit .spec file to include data files and more options(Retina display support .etc, a template attached below, S. 1), then:

pyinstaller MainLibrator.spec

Note1: remember install all the required modules using pip. Modules installed in PyChram cannot be used by pyinstaller.

Note2: Don’t use -F option in the first step! Only use -w (must use -w) in the first step!

Note3: If you see ‘could not find qtwebengineprocess’ error, two possible reasons: 1) your PyQt5 version doesn’t compatible with your PyQtWebEngine version. For example, PyQt5==5.13.2 foesn't work with PyQtWebEngine 13.2, you need to downgrade PyQt5 to 5.13.0. 2) You might use -F option of pyinstaller. Try to repeat step 1 to generate a clean .spec file.

Note4: weblogo 3.7.1 has compatibility issue with Pyinstaller. You need to edit “\_\_init\_\_.py” file for weblogo/seq\_io/ . Note that PyCharm python libs may not the top priority for Pyinstaller, Pyinstaller usually search conda libs (/Users/leil/anaconda3/lib/python3.7/site-packages/)

If use customized icons, determine resources in a .qrc file

pyrcc5 -o images\_qr.py images.qrc

# IDE

PyCharm

<https://www.jetbrains.com/pycharm/>

cd /Users/PCW-MacBookProRet/Applications/Librator

pyuic5 MainLibrator\_UI.ui > MainLibrator\_UI.py

# Dependencies

***Clustal Omega***

Download from <http://www.clustal.org/omega/>

***Muscle***

Download from <https://www.drive5.com/muscle/>

Move those two executable files to $PATH (e.g. /usr/local/bin)

***Home brew(***<https://brew.sh/>***)***

The Missing Package Manager for macOS (or Linux)

/bin/bash -c "$(curl -fsSL https://raw.githubusercontent.com/Homebrew/install/master/install.sh)"

or

ruby -e "$(curl -fsSL https://raw.githubusercontent.com/Homebrew/install/master/install)" < /dev/null 2> /dev/null

***PyMOL*** (<https://pymol.org/2/>)

PyMOL for 3D structure visualization of mutated HA protein

Download from official website (license required after 30 days free trail)

or install from anaconda

conda install -c schrodinger pymol

or install from homebrew (recommended )

brew install brewsci/bio/pymol

or install from MacPorts

sudo port install pymol

or install from source

python setup.py --osx-frameworks install

***GhostScript*** (<https://www.ghostscript.com/>)

brew install ghostscript

***Pdf2svg*** (<https://github.com/dawbarton/pdf2svg>)

brew install pdf2svg

# Database

SQLite3 (local DB)

MySQL (remote DB)

CREATE TABLE Fragments (

Name TEXT NOT NULL,

Segment TEXT,

Fragment TEXT,

Subtype TEXT,

ID TEXT,

Template TEXT,

AAseq TEXT,

NTseq TEXT,

Instock TEXT,

PRIMARY KEY(Name(512)))

ENGINE=InnoDB;

# Functions

Functions here including both have and have not finished functions.

* Create database and import sequences from FASTA file PASS
* Load existing database PASS
* Epitope display under H1/H3 numbering PASS
* Multiple sequence alignment view PASS
* Sequence editing PASS
  + Mutation PASS
  + Merge sequence-base biased PASS
  + Merge sequence-cocktail PASS
* 3D visualization of HA protein structure PASS
* Gibson fragments design PASS
  + a database that stores all fragments we have PASS
  + design the optimized Gibson fragments plan PASS
  + generate Gibson fragments PASS
* Popup sequence/alignment viewer PASS
* Sequence fusion PASS
* All-in-one installation free package PASS

# Supplementary

**S.1** template of .spec file:

# -\*- mode: python ; coding: utf-8 -\*-

block\_cipher = None

added\_files = [

('/Users/leil/Documents/Projects/Librator/Resources/Data/H1\_AAVI.csv','Data'),

('/Users/leil/Documents/Projects/Librator/Resources/Data/H1\_PCT.csv','Data'),

('/Users/leil/Documents/Projects/Librator/Resources/Data/H3\_AAVI.csv','Data'),

('/Users/leil/Documents/Projects/Librator/Resources/Data/H3\_PCT.csv','Data'),

('/Users/leil/Documents/Projects/Librator/Resources/Data/NA\_AAVI.csv','Data'),

('/Users/leil/Documents/Projects/Librator/Resources/Data/NA\_PCT.csv','Data'),

('/Users/leil/Documents/Projects/Librator/Resources/PDB/3hto.pdb','PDB'),

('/Users/leil/Documents/Projects/Librator/Resources/PDB/4hmg.pdb','PDB'),

('/Users/leil/Documents/Projects/Librator/Resources/PDB/4jtv.pdb','PDB'),

('/Users/leil/Documents/Projects/Librator/Resources/PDB/3lzg.pdb','PDB'),

('/Users/leil/Documents/Projects/Librator/Resources/PDB/1ruz.pdb','PDB'),

('/Users/leil/Documents/Projects/Librator/Resources/PDB/1ru7.pdb','PDB'),

('/Users/leil/Documents/Projects/Librator/Resources/PDB/1ru7.pdb','PDB'),

('/Users/leil/Documents/Projects/Librator/Resources/Conf/db\_record.txt','Conf'),

('/Users/leil/Documents/Projects/Librator/Resources/Conf/db\_record.txt','Temp')

]

a = Analysis(['MainLibrator.py'],

pathex=['/Users/leil/Documents/Projects/Librator/Librator'],

binaries=[],

datas=added\_files,

hiddenimports=[],

hookspath=['hooks'],

runtime\_hooks=[],

excludes=[],

win\_no\_prefer\_redirects=False,

win\_private\_assemblies=False,

cipher=block\_cipher,

noarchive=False)

pyz = PYZ(a.pure, a.zipped\_data,

cipher=block\_cipher)

exe = EXE(pyz,

a.scripts,

[],

exclude\_binaries=True,

name='MainLibrator',

debug=False,

bootloader\_ignore\_signals=False,

strip=False,

upx=True,

console=False )

coll = COLLECT(exe,

a.binaries,

a.zipfiles,

a.datas,

strip=False,

upx=True,

upx\_exclude=[],

name='MainLibrator')

app = BUNDLE(coll,

name='MainLibrator.app',

icon='Flu.icns',

bundle\_identifier=None,

info\_plist={

'NSHumanReadableCopyright':"Copyright @ 2019, Wilson Lab, All Rights Reserved",

'NSHighResolutionCapable': 'True'

})

# User manual

User interface

1. Main Tab
2. Sequence Tab
3. Alignment Tab (RTF)
4. Alignment Tab (HTML)
5. Sequence DB Tab
6. Summary
7. Summary (HTML)
8. Fragment DB

Functions:

1. Path setting

Users can set paths for all required tools and databases.

Click Setting-> Preferences in menu or click  in tool bar.

1. Create new sequence database

Users can create new sequence database.

Click File-> New in menu or click  in tool bar.

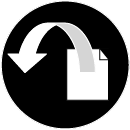
1. Open existing sequence database

Users can create new sequence database.

Click File-> Open in menu or click  in tool bar.

1. Import sequences

After load an existing sequence database, users can import sequence into current database. The input files should be in FASTA format.

Click Sequences-> Import in menu or click in tool bar or click “import sequences” button in Main tab.

1. Export sequences

After load an existing sequence database, users can import sequence into current database. Selected sequences will be exported to a FASTA file.

Click Sequences-> Export in menu or click  in tool bar.

1. Edit information of sequences

Users can edit information (sequence name, Role, Form, Subtype, NT sequence, reading frame, donor regions) of sequences.

For Sequence name, users can click sequence name text input to edit it.

For Role, Form, and Subtype, users can edit by the left panel. Users can select multiple sequences in active sequence panel to do bulk update.

For cording frame start and end, users can edit use the “Coding region” inputs.

For NT sequences, users can click “edit sequence” button to edit sequences.

Note: coding region and NT sequence editing is disabled for sequences with mutation information.

1. HA numbering

Users can access H1/H3 numbering of selected sequence.

Click Tools-> HA Numbering in menu or just click “Sequence” tab.

1. Multiple sequence alignment

Users can align multiple sequences together and check the results in a graphical viewer.

Two MSA viewers are available: RTF viewer (printable) and HTML viewer (interactive).

For RTF viewer:

Click Tools-> Multiple Alignment(RTF) in menu or just click in tool bar or click Alignment(RTF) tab.

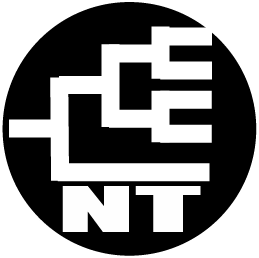
For HTML viewer:

Click Tools-> Multiple Alignment(HTML) in menu or just click  in tool bar or click Alignment(HTML) tab.

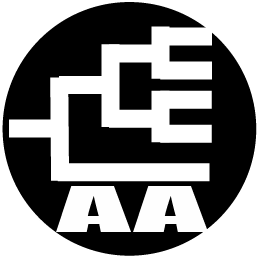
1. Generate phylogenetic tree (ML tree)

Users can generate phylogenetic tree (Maximum Likelihood tree) for selected sequences. Nucleotide and Amino Acid trees are available.

For Nucleotide tree:

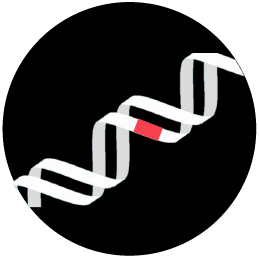
Click Tools-> Generate Maximum Likelihood Tree (nucleotide) in menu or just click in tool bar.

For HTML viewer:

Click Tools-> Generate Maximum Likelihood Tree (Amino Acid) in menu or just click  in tool bar.

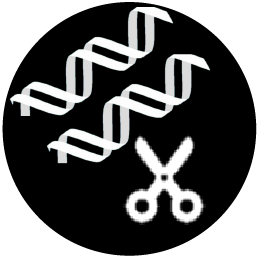
1. Mutate sequence

Users can generated new sequences with mutations based on any existing sequence.

Click Edit Sequence-> Mutation in menu or just click in tool bar.

1. Compare sequences and generate screening mutations

Users can compare sequences and generate consensus sequences or new sequences with screening mutations.

Click Edit Sequence-> Editing in menu or just click in tool bar.

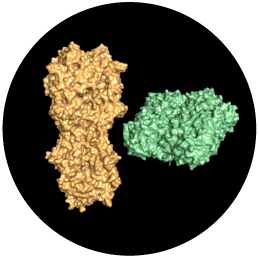
1. Epitope transplant across different subtypes

Users can transplant epitopes/regions from sequences of different subtypes.

Click Edit Sequence-> Fusion in menu or just click  in tool bar.

1. 3D visualization via PyMOL

Users can see 3D structure of selected sequence via PyMOL.

Click Visualization-> PyMOL in menu or just click  button on Main tab.

1. Generate Gibson Clone fragments

Users can generate Gibson Clone Fragments.

Click GibsonClone-> GibsonClone in menu or just click  button on Main tab.