**Red text means it works as expected**

**Blue text means there are bugs or unexpected behaviour**

**Black text means the function was not tested (not counting BUGs)**

**Suggested Changes:**

1. For the unsaved changes prompt, consider adding a “Save and continue” option to the existing “Cancel” and “Discard” Options. Consider changing “Discard”  “Discard unsaved changes.”

**Shot prep Tool UI**

1. File Menu:
   1. New file
      1. Ensure that it will create a new h5 file with the correct file name
      2. Ensure that the GUI will open a warning dialog box if the user tries to create a file with the same filename as a file that already exists
      3. Ensure that the GUI will open a warning dialog if the user tries to create a new file while there are unsaved changes in the current file
      4. Verify that the GUI will raise a warning if there is a file lock on the h5 file

**BUG:** the GUI does not give a warning if there is a file lock on the h5 file. The python console logger does however.

Traceback (most recent call last):

File "C:/Users/Owner/GIT REPO/ENPH459Project/ShotPreparationTool/ShotPreparationToolUi.py", line 91, in actionNew

self.model = ShotPrepToolModel(self.fileName)

File "C:\Users\Owner\GIT REPO\ENPH459Project\ShotPreparationTool\ShotPrepToolModel.py", line 23, in \_\_init\_\_

os.remove(self.h5tempFileName)

WindowsError: [Error 32] The process cannot access the file because it is being used by another process: 'C:/Users/Owner/GIT REPO/ENPH459Project/ShotPreparationTool/test1.h5.tmp'

* + 1. Test that cancelling or discarding the operation works as expected

**BUG:** Attempting to create a new file before saving changes and choosing “cancel” does not cancel the action. The new file is still created and the .tmp file of the unsaved file remains open (in the directory, not in the actual tool), but its .h5 file cannot be opened. The .tmp files are not removed upon closing the program. The following python error is produced when attempting to open the original file:

Traceback (most recent call last):

File "C:/Users/Owner/GIT REPO/ENPH459Project/ShotPreparationTool/ShotPreparationToolUi.py", line 104, in actionOpen

self.model = ShotPrepToolModel(self.fileName)

File "C:\Users\Owner\GIT REPO\ENPH459Project\ShotPreparationTool\ShotPrepToolModel.py", line 23, in \_\_init\_\_

os.remove(self.h5tempFileName)

WindowsError: [Error 32] The process cannot access the file because it is being used

* 1. Open file
     1. Ensure that the prep tool will open an h5 file.
     2. Verify that a warning dialog will pop up if there are any unsaved changes.
     3. Verify that the GUI will raise a warning if the user attempts to load an invalid (corrupt or misnamed) h5 file. Verify that both options (Discard and Cancel) work as expected for the warning box.
     4. Verify that the GUI will raise a warning if there is a file lock on the h5 file

**BUG:** the GUI does not give a warning if there is a file lock on the h5 file when opening it. The python console logger does not produce an error either (unlike with creating a new file with the same file name as the file in use).

* + 1. Test that cancelling the operation works as expected

**BUG:** Attempting to create a new file before saving changes and choosing “cancel” does not cancel the action. The new file is still created and the .tmp file of the unsaved file remains open (in the directory, not in the actual tool), but its .h5 file cannot be opened.

* 1. Save file
     1. Ensure that the changes in the temporary file are saved when opening a new file. Verify with an external tool.
  2. Save as

**BUG:** ‘Save as’ does not function as expected. A warning message is produced to either ‘Cancel’ or ‘Discard’ unsaved changes.

* + 1. Ensure that the save as function will save the changes to the new file specified.
    2. Verify that the original file will not be changed.
    3. Ensure that save as will open a warning dialog box if the prep tool attempts to overwrite a file that already exists. Verify that both options (Discard and Cancel) work as expected for the warning box.
    4. Test that cancelling the saving operation works as expected
  1. Close
     1. Ensure that the file will be closed.
     2. Verify that a warning dialog will pop up if there are any unsaved changes
     3. Ensure that all temporary files are cleaned up on close
  2. Exit
     1. Verify that a warning dialog will pop up if there are any unsaved changes. Verify that both options (Discard and Cancel) work as expected for the warning box.

**BUG:** When using ‘X’ to exit, the ‘Cancel’ option works as expected, but ‘Discard’ exits and does not remove the .tmp file. When using ‘Exit’ to exit, the ‘Cancel’ option does not work and instead the program is exited without deleting the .tmp file. Choosing ‘Discard’ behaves as expected.

* + 1. Verify that the UI will exit.
    2. Verify for both the ‘X’ in the top right corner and the exit button in the file menu.
    3. Ensure that all temporary files are cleaned up on exit.

**BUG:** When using ‘X’ to exit, the .tmp file of the h5 file previously in use is not removed.

1. Device menu
   1. Verify that devices can be added to the h5 file
      1. Verify that the UI will show a meaningful warning if the user can’t add that device. For instance if the user attempts to add a device with the same name as one that already exists.

**BUG:** Attempting to create a device with the same name does nothing. The prompt to name a new device closes but no new device is added.

* 1. Verify that devices can be removed from the h5 file
  2. Verify that the tool rejects invalid Python variable names for device names. The user should be presented with a meaningful error message. Invalid names include the following:
     1. Names with illegal characters:
        1. Spaces
        2. '!@#$%^&\*()' etc.
     2. Names that begin with a digit
     3. Names that are reserved (i.e. Python keywords such as 'continue', 'class')
  3. Verify that the tool rejects duplicate device names. The user should be presented with a meaningful error message.

1. Row menu
   1. Verify that values can be added to the current device
      1. Verify that the UI will show a meaningful warning if the user can’t add that value. For example, if the user attempts to add a value with the same name as one that already exists.
      2. Verify that the tool rejects duplicate variable names within a device. The user should be presented with a meaningful error message.
      3. Verify that the tool will accept duplicate variable names across devices.
      4. Verify that the tool rejects invalid Python variable names as described in the device name test description. The user should be presented with a meaningful error message.

**BUG:** A warning is shown if the input for an attribute is invalid (ie. some random string), but the warning message simply notifies the user that the value is undefined and cannot be evaluated rather than it’s not a proper input.

* 1. Verify that attributes can be removed from the current device

1. File name test cases – For New File and Save As functions
   1. Test the prep tool with filenames with no explicit extension
   2. Test the prep tool with the wrong file extension.
2. Warning and dialog boxes.
   1. Ensure that the warning messages make sense.
   2. Verify that both the option buttons work as expected
3. Multiple prep tools open at the same time
   1. Ensure that the tool raises a warning dialog if the user tries to open an h5 file that’s already in use.

**BUG:** the GUI does not give a warning if there is a file lock on the h5 file when opening it. The python console logger does not produce an error either (unlike with creating a new file with the same file name as the file in use).

1. Test the buttons in various configurations and orders