GitEditor User Manual

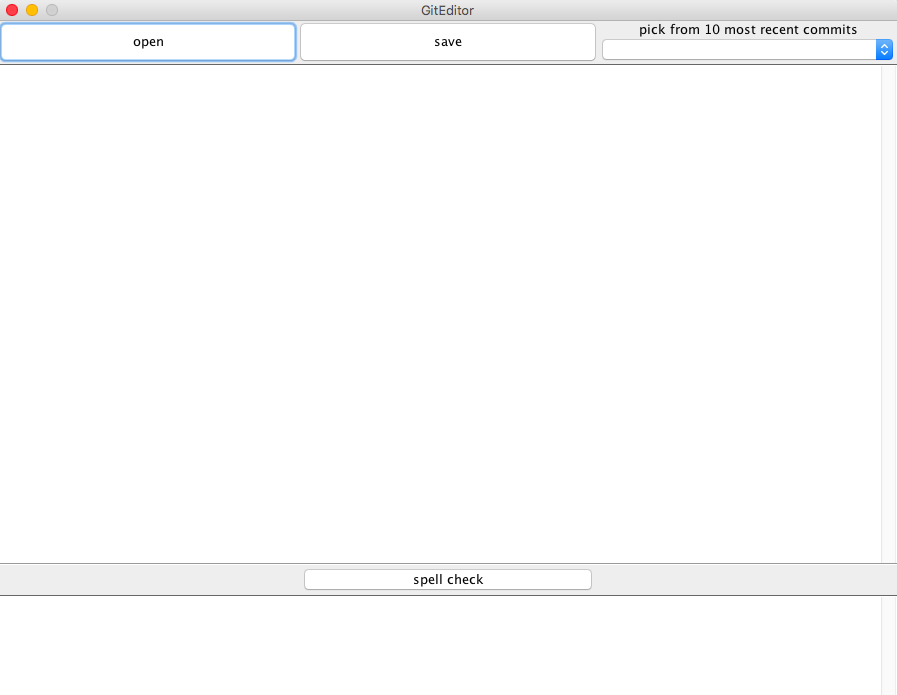
# What is GitEditor

GitEditor is a plain-text editor that uses a git-like backend to retrieve previous versions of the working file that also incorporates spell checking. That is, instead of just saving the current state of a file as most current text processors do, GitEditor implements a save-as and lets you provide a comment for easier retrieval. However, GitEditor hides the git-like features so users do not need to interact with the terminal or know specific git commands; instead, all of this is driven by the gui.

# The Anatomy of the GitEditor Window

3

2



6

5

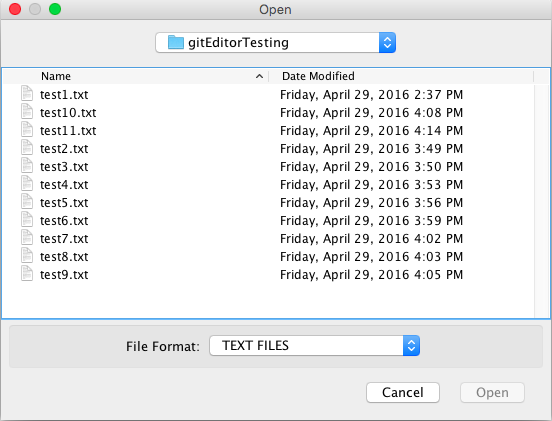
4

1

1. The **Open** button opens previously saved text file. GitEditor works best with plain-text text file.
2. The **Save** button functions as both the commit button and the traditional method of saving a file.
3. The drop-down menu to retrieve a previous commit the most 10 recent commits. With each subsequent save/commit, the drop down list will get populated with each commit message and time stamp.
4. This is the main text area for writing your document.
5. The **Spell Check** button will operate on the text that is currently in the main text area (4).
6. The results of the spell checking button will appear in this text area.

# How to Open a File

Clicking on the Open button will bring launch a dialog box that will let you browse to the directory and fie you would like to open.

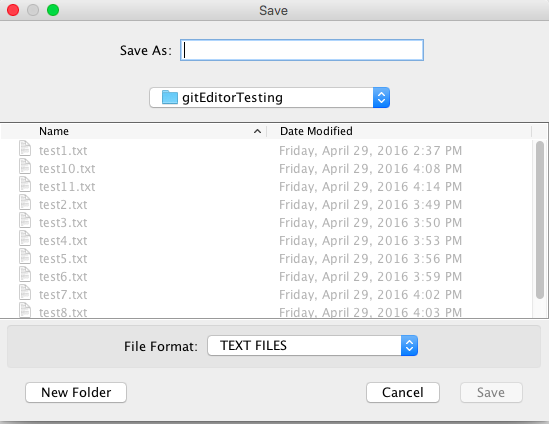


After selecting the document to open, the text in that document will appear in the main text area (4).

If the opened file already has previous commits to retrieve, the drop-down menu to retrieve will be automatically populated. Also, a new commit was be automatically generated and labeled “Reopening.”

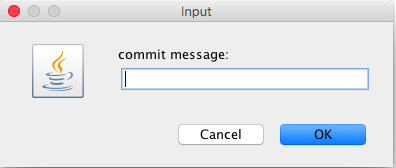
# How to Save a File

If the working file has not be saved yet and the **Save** button is pressed, you will be prompted to save the file first.



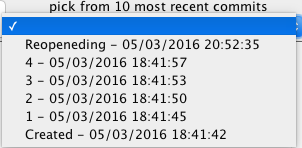
The initial save will also create the first commit which is labeled “Created” and will have the time stamp for that save.

If the file has already been saved, a prompt will appear asking you to provide a commit message. This will only appear if there is a change between the current commit and the previous one. If there is no difference, nothing will appear.



# How to Retrieve a Previous Commit

By selecting a previous commit from ../../../../../Desktop/Screen%20Shot%202016-05-03%20at%208.52., you pick in order (newest to oldest) which commit to retrieve which will populate the main text area with that commit’s data. Each commit will have the commit message and the commit time listed, as shown below

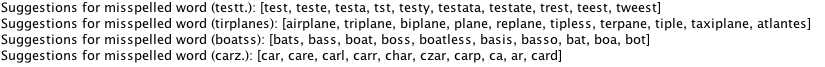


It is also important to note that a user needs to save, if changes have been made since the last save/commit, before retrieving a previous commit.

# How to Spell Check

You can run the spell checker on the current text that is in the main text pane by clicking the **spell check** button (5). The spell check pane will be populated with the results. If no errors are found, ../../../../../Desktop/Screen%20Shot%202016-05-03%20at%208.42. will appear in the spell check pane. If error are around, it will appear as this:

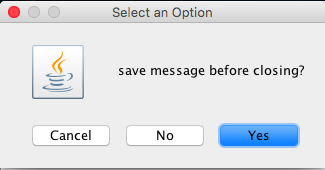
With this same text, “This is a testt. I like and tirplanes and boatss and carz,” the spell check pane will show:



You can then return to the main text area and update your text and click on the spell check button to ensure there are no misspellings.

# How to Exit GitEditor

You exit GitEditor by closing the window the way you would normally do for your operating system. However, if GitEditor notices that you have made changes since your last save/commit, it will prompt you to make sure you really want to exit.



If you select **Cancel**, it aborts the closing and keeps you in the window. If you select **No**, the program will exit and the window will close. If you select **Yes**, the program will put you in the path to save the file and provide a commit message.

It is also important to note that the window will take a fraction of a second to close as it needs to gracefully close out other background operations and components. If there is an exception thrown when you open the git editor, it is probably because somebody messed up the hidden folder “.gitDatabase” in the root directory where the src and bin folder located. Just delete the “.gitDabatase” folder and restart the program.