Git-Squashing commits

Git uses the VIM editor by default. It can be changed to your preferred editor by setting the core.editor property. For example, if you want to use Notepad++ on Windows run:

```bash
    git config --global core.editor "d:\programs\notepadpp\notepadpp.exe" -multiInst -notabbar -nosession -noPlugin
```

You need to provide an absolute path and optional parameters for your editor in quotes.

**Performing a squash with Git – example**

There are a couple of commits made to the RESTWS-702 branch of the project module (openmrs-module-webservices.rest). We can see these commits by running the following command and then the Enter button while in the terminal or git-bash window while in the root directory of the project,

```bash
    git log
```

After quitting the VIM screen, type in the following to squash commits (in this case the last 2 commits),

```bash
    git rebase -i HEAD~2
```

The VIM editor will open showing the the last two and most recent commits in reverse order.

Type in ‘q’ in the space next to the ‘:‘ character to quit the VIM screen.

After quitting the VIM screen, type in the following to squash commits (in this case the last 2 commits),

```bash
    git rebase -i HEAD~2
```

The VIM editor will open showing the the last two and most recent commits in reverse order.
Don’t try to edit text directly because all will get messed up, instead press the 'I' key on the keyboard to enter **Insert** mode, edit the second line, and replace **pick** with **f**
Press the Esc key to enter command mode, then Shift + : and finally type in x.

Vim closes and Git performs the squashing and when finished, displays a status message:

Type the command git log to confirm the number of commits due to your changes is one.
Finally, push changes to your fork and check the pull request to ensure only one commit is left.