GitHub for Mathematical Research

Luke Guatelli and Andrew Penland

June 13, 2018

1 Introduction to GitHub

1.1 Overview

Git is a version control system - a piece of software that can be used to track changes and other information about projects. *GitHub* is an online platform that uses Git. GitHub is widely used by software developers, and it contains a *lot* of open source software.

1.2 Why Use GitHub?

GitHub is a tool for project management. It allows all collaborators to keep track of goals (by using the Projects and Issues features), as well as enabling communication between collaborators. It promotes an efficient workflow where contributors can work on the project in small chunks.

GitHub is widely used in both open source and closed source software, but it can also be used on any project done on a computer. GitHub seems well-suited to current trends in mathematical research. The average number of coauthors on mathematical papers is increasing. There is an increase in large-scale, *crowdsourced* collaboration. An example is the PolyMath program [**?**], which has led to the solution of at least thirteen open problems.

In the end, each mathematician will have to decide on their own optimal work process, as well as whether the time invested in learning GitHub is worth the potential gains in efficiency and organization. We have found GitHub very useful.

1

2   How To Do Specific Things

2.1   Create An Account

The first thing we need is make an account. In any web browser type in the url https://github.com. Since it will be the first time visiting the site, this main page will prompt you to sign up with GitHub. We will use the following textboxes to choose a username, password and which email address we want to be associated with the account.
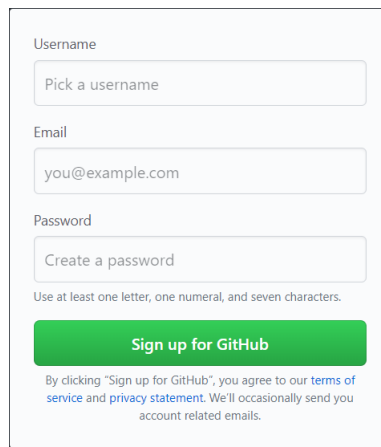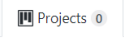

Figure 1: The Sign Up textboxes on the main page of GitHub.

Once these are filled in, select the green "Sign up for Github" to create the account. The next pages offers a choice between a free account or a paid one. The paid version allows for the creation of private repositories, while the free one only allows public repositories, but includes all other features of GitHub. It is worth noting that shortly after creation of an account, you will receive an email asking you to verify your email address. It is as simple as clicking the link in the email, taking you to your account page on GitHub.

2.2   Start a Project

2.2.1   Creating a Project Board

Next, we want to be able to create projects to work on. To do this from the main repository page click the [Projects 0] tab near the top. In the upper right hand corner of this tab, there will a green button like so, [Create a project].

The next page will prompt you to choose a name for the project, as well as giving the option for a brief description. The final option we need to choose before creating the project is the template style. The default no template option will allow you to create your own columns for this project. This basic kanban option will automatically include *To Do*, *In Progress*, and *Done* columns which will house issues that match those descriptions.
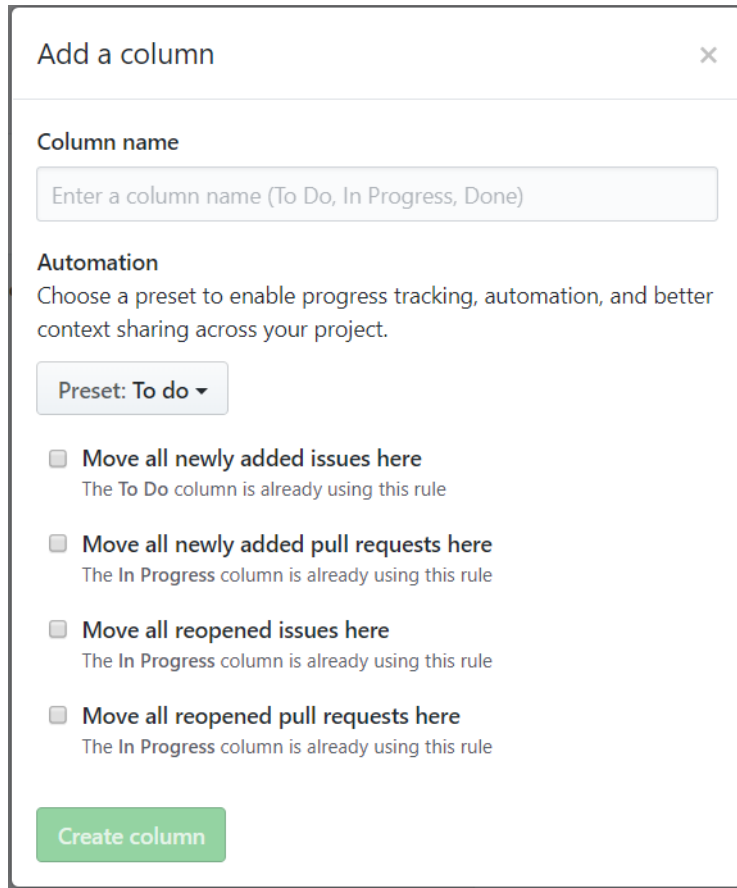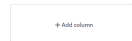

Figure 2: Creating a Project

2.2.2 Adding Columns to a Project Board

Once you have a project created you can add customized columns in addition to the ones from your chosen template. To do this we click on the which is to the right of any preexisting columns. First, we must give the new colmun and title. Then there we can choose between any of the preset options or create an entirely customized column. The preset columns have options such as "Move any new issues to this column" that make organization somewhat automated. The role of these columns is to the house

and organize issues related to certain projects. Later in the paper we will show how to add labels to issue in order associate it with a certain project. Once we have the column settings to our liking, we click "Create Column" at the bottom of the window to finish the process.

2.3 Add a File

2.3.1 From Your Computer

If you already have project files on your computer, you can upload them into the repository at any time by clicking the Upload files from the repository screen. This will take you to a page where you can add files either by dragging into the window or by clicking to open a dialog box. When you first open a

repository and do not have any files yet, you will be prompted to add files.

2.3.2   From Overleaf

One obvious disadvantage of using GitHub for mathematics is the lack of an ability to automatically compile .tex files into .pdf files. It is possible to integrate projects from Overleaf into GitHub, but it requires using the command line tool `Git`. A discussion of `Git` is beyond the scope of our current work, though we may include it in a future version. More information is available at the following links:

1. Collaborate Online and Offline with Git and Overleaf

2. How to connect an Overleaf project with a repo on GitHub, GitLab, or BitBucket

2.4   Edit a File

You can edit a file online by clicking on the file in the Repository's *Code* page,
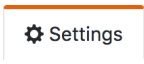
then clicking the icon on the right hand side. For organization's sake, we prefer to create a new branch before performing any substantial edits.

3   Collaborators

3.1   Invite a Collaborator

To invite someone to work on a repository with you,

1. Go to the repository.

2. Click the button.

3. From the "Settings" page, click on the button on the left side of the page.

4. Type the information into the search window

Search by username, full name or email address

You'll only be able to find a GitHub user by their email address if they've chosen to list it publicly. Otherwise, use their username instead.

Add collaborator

and click "Add a Collaborator".

3.1.1 Being Invited as a Collaborator

If you know that you have been invited to collaborate on a project, you can accept the invitation to collaborate by going to the repository at the link https://github.com/username/reponame/invitations, where username and reponame are replaced in the obvious way.

You will also receive an e-mail notifying you of the invitation to collaborate. You will **not** see anything in your GitHub notifications.
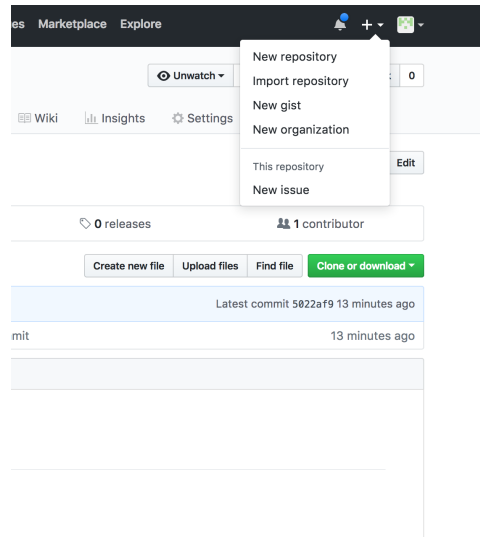
4 Issues

Issues are a way for team members to communicate and keep track of changes to the project. Here is GitHub's description of an issue: [2]
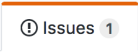
> Issues are used to track todos, bugs, feature requests, and more. As issues are created, they'll appear here in a searchable and filterable list. To get started, you should create an issue.

4.1 Creating An Issue

We'll follow their advice, creating our first issue. The first issue we will create will be a "todo", telling us to create an issue. (Thus, this issue resolves itself.) We have two options:

1. Under the in the upper left corner,we select the command "New Issue".
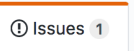
2. We click on the tab, then click the button.

Either way brings up a "New Issue" screen with fields for Title and Description. We fill in this information, then click the button.
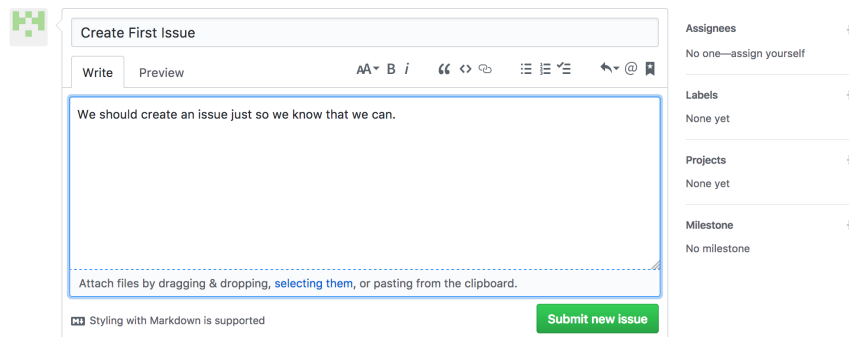

4.2 Checking For Issues

If we log in to GitHub, we see that there is a new Issue by checking the "Issues" tab. We see , telling us there is one new issue. We click on the tab to see what the issue is.

4.3 Closing An Issue

Once an issue is satisfactorily resolved, we want to close it. We can do this on the Issues screen by selecting the checkbox next to the issue, then going to the "Mark as" menu and selecting "Closed".
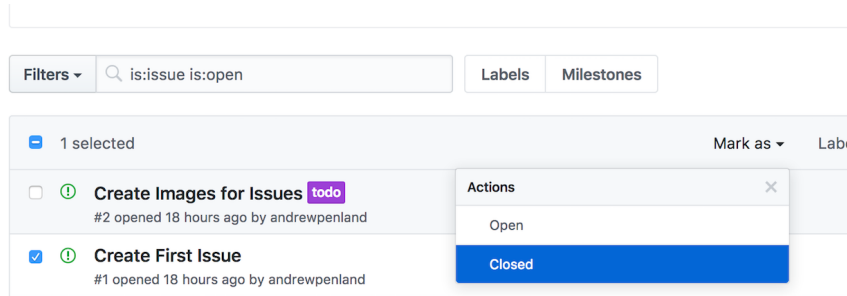
4.4 Labeling Issues

We can label issues so that we know what type they are. We can label an issue when we create it by using a dropdown menu called "Labels". GitHub has pre-existing labels for various categories. Simply type the name of the label you want to use or select it from the menu.

4.5 Creating Your Own Labels

To create our own label, we begin by typing the name we want to use for the label. GitHub will automatically bring up a suggestion to create a label with this name.

Figure 4.5 and Figure 4.5 show how to create a "todo" label, which is a common issue, but one that doesn't already have a premade GitHub label.

4.6 Assigning Issues

You can assign an issue to the attention of specific project members using the *Assignments* drop-down menu, which is to the right of the screen just above the *Labels* menu.
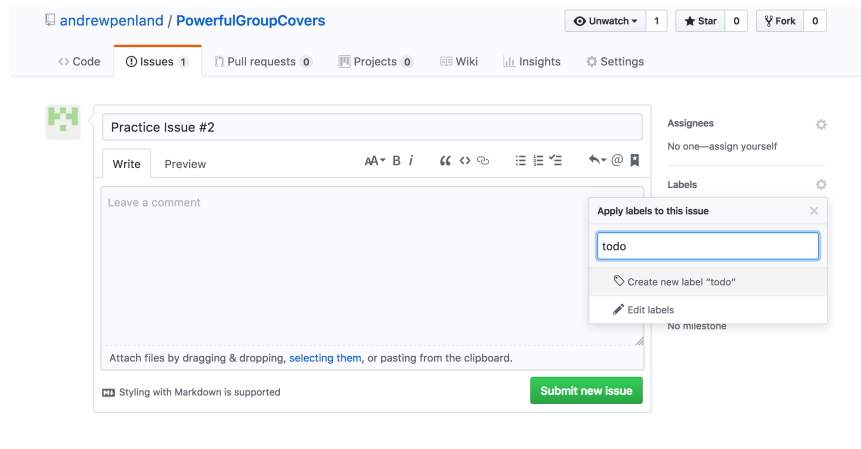
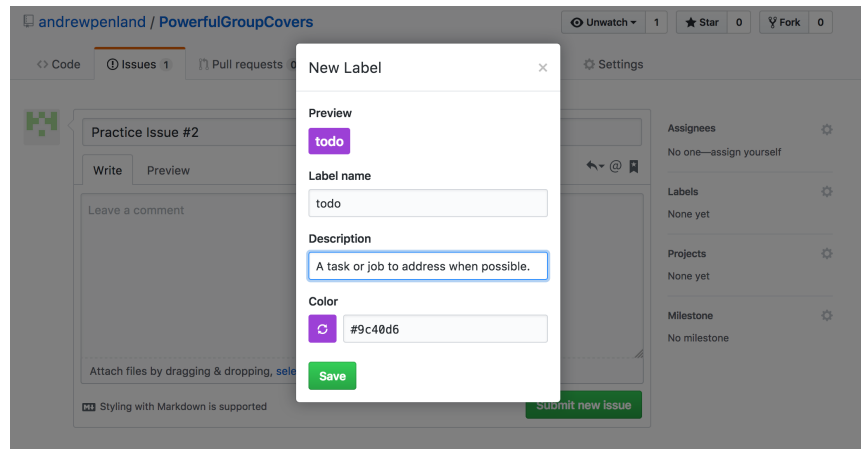Figure 3: Labeling An Issue using The Menu

Figure 4: Creating A Custom Label

5 Changes

5.1 Making Changes

5.1.1 Branches

One nice feature of GitHub is that it allows us to try things out in a different *branch*, so that we can edit the repository separately without changing the currently accepted version. Once we have made the desired changes in the branch, we will create a pull request to merge our branch with the master branch.

5.1.2 Creating a New Branch

We can create a new branch by selecting the [Branch: master ▾] button, then typing the name of the branch we want to create. In the resulting drop-down menu, select the *Create Branch* option.
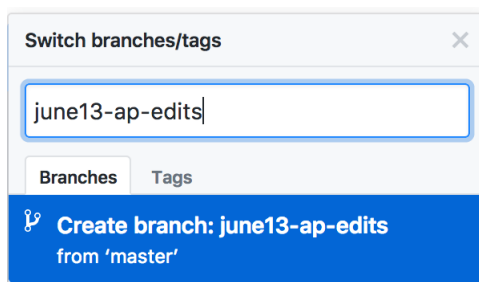
Figure 5: Creating a New Branch

5.1.3 Working Within the New Branch

After creating the new branch, we will make the desired changes. This usually involves adding and editing files, which can be done online as in subsections 2.3 and 2.4, or by downloading the repository to make changes on our own local machine.

If we want to work offline, we can use the [Clone or download ▾] button in the upper right hand part of the screen. We select *Download ZIP* from the resulting *Clone with HTTPS* box.

After downloading and unzipping the repository, we can make any desired edits on our local machine. We then re-upload the files from our computer to update the branch, as described in in subsection 2.3.
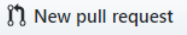
When we have made the desired changes, we will usually want to initiate a *pull request* so they become part of the master branch.

5.1.4   Pull Requests

Once we are done working on our new branch and would like to integrate the changes into the master branch, we need to generate a pull request. To do this we will first move over to the [Pull requests 0] tab. On this page we will find the [New pull request] button. Select this, and we will be prompted to choose which branches or forks we wish to merge using the drop down boxes shown below. We select the appropriate branches and click "Create New Pull Request".
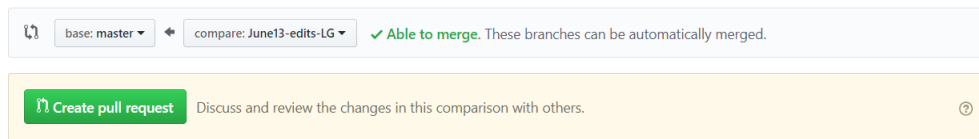

We can also arrive at this page from the list of branches. To the right of each active branch we see [New pull request] that will follow the same process for creating a pull request. Once we create the pull request, we may have conflicts in the code. If this is the case, the code will be displayed in an editable form. From here we simply pick the code from the discrepancies and the delete the markers delineating the two versions. Once all conflicts have been resolved, the branches will successfully merged.

5.2   Keeping Track of Changes

5.2.1   Changes To The Repository

If you want useful information about the project's history, look no further than the [Insights] tab. From the Insights page, you can see all Issues and Pull Requests associated to the project by clicking on [Pulse]. Other tools available from the Insights page include graphs that track data related to commits, visits, and code dependencies. Analysis of GitHub data is an emerging topic in computer science, with several tools intended for this purpose. (For just one example, see [1]).

5.2.2 Changes To An Individual File

To see changes in an individual file, open the file in GitHub, then use the *Blame* or *History* command. *Blame* will show the document along with a line-by-line annotation of which commit added each particular section. *History* gives a day-by-day overview of commits and their summaries.

6 Conclusion

We hope that you have found this brief tutorial useful. There are many other features that GitHub has; we have covered the ones we felt were most useful to a complete beginner. We welcome any feedback. Good luck with your research!

References

[1] Annalee Newitz. Data analysis of github contributions reveals unexpected gender bias. https://arstechnica.com/information-technology/2016/02/data-analysis-of-github-contributions-reveals-unexpected-gender-bias/.

[2] The Github Team. Issues. https://github.com/issues.