# **GitHub and GitHub Desktop for Beginners**

|  |  |
| --- | --- |
| **Log in and initial setup** | |
| Create a login to GitHub at <https://github.com> |  |
| Select the free plan and complete setup. |  |
| You will be prompted to verify your email address. Check your emails and click the button to verify. |  |
| You will be prompted to create a new repository.  Give it a name, set it to PUBLIC to make it easier to share it with your teacher, initialise it with a README file (this is good practice), and click ‘Create repository’. |  |
| During your setup, you will see a link to learn GitHub.  This is optional – it is a good way to get familiar with the interface and with branches, but if you will be using GitHub mostly locally, you don’t need this so much. |  |
| Now you have your empty repo (repository) set up, you can make the link to a local program and your documents folders. |  |
| Click on ‘Clone or download’. |  |
| The next step depends on what program you are using to manage the GitHub files on your computer.  If you are working within another program, such as Atom, Visual Studio Code or Git Bash you will need the URL.  If you want to install and use GitHub Desktop you can use ‘Open in Desktop’ |  |
| **If using GitHub Desktop** | |
| If you are using the GitHub Desktop option, a few more steps.  Clicking on the ‘Open in Desktop’ link in your repo will take you to a download page. Download the appropriate version for your machine. |  |
| Click to run the install file… |  |
| Sign in using the GitHub credentials that you created for the GitHub website. |  |
| Pinning GitHub Desktop to your taskbar isn’t a bad idea, if you will be using it a lot. |  |
| You need to click on the Add button and clone your online repository to this computer. |  |
| You need to choose the repo you want to clone. You also need to choose where on your computer you want to save it. Choose a place that you can work from easily – eg your Documents or Google Drive File Stream.  Creating a new folder named after your project is a good way to keep things tidy. Clicking the blue ‘Clone’ button will set up a .git folder in your project folder, and add the README file (and any other files in your online GitHub repo) to it. |  |
| You should now be working in your new repo. Check under ‘Current repository’ at the top left of your GitHub Desktop window.  Your main screen should tell you that there are no changes yet – this means that your local version matches the online one. |  |
| Look in your documents to make sure that you can find your new folder and files. There is also a shortcut to them inside GitHubDesktop, but it’s important to know where they are saved, as you will often be working on them directly. You only really need to use GitHub Desktop when you want to make a Commit and push your files up to the online master repo. |  |
| **Adding and editing files** | |
| Now it’s time to see what happens when you make a change to the files in your local repository, and how to commit these changes and push them back up to your online repo.  Using your usual editing method, create or paste a new file into your new project folder. How you do this depends on what program and file type you are using – In my screenshot, to keep things simple I’ve just added a Word document. |  |
| You will see that GitHub Desktop recognises that your files have changed. |  |
| Once you have made the changes to your file that you wanted – made edits, added files, you will be ready to make your first ‘commit’.  You need to add a message and then click the ‘Commit it master’ button.  Sometimes it is tidier if you close the files you are working in first, but this isn’t always necessary. |  |
| Committing your file doesn’t upload your new file to the online origin repo, but it adds it as a commit to your local master version. It tells you that your local master is up to date by saying ‘No local changes’. If you choose to, now is a good time to click ‘Push origin’ and send your latest commit to the online version. |  |
| If you check your repo in online GitHub, your changes should now be there. You may need to refresh the page to see them. |  |
| From here on in, your workflow would be that you make changes to your files locally as you develop your outcome and after every significant change (or every few days), commit and push your files to update the online version. Use descriptive commit messages! | |
| If you need to roll back to or view a previous commit, you can click on ‘History’ in GitHub Desktop, or ‘commits’ in your online repo. |  |
| If you are working with a team, you can add them as ‘Collaborators’ online. This means you can work on the project at the same time. |  |
| You might want to track issues related to your project or have online discussions with others in your team. Issues is a good place to note things to fix or improve.  Issues lets you categorise items using labels, add issues to your project planning board, and comment on existing issues. |  |
| You can create your project planning or Kanban board within online GitHub. GitHub calls this a ‘Project’.  You can add or rename columns, add new tasks and drag to move them across the board. Adding styling with markdown lets you allocate tasks to people on your team, or add checklists and images. |  |
| **More tutorials:**   * Learn Git with an interactive sandbox at <https://learngitbranching.js.org/> * The Git and GitHub for Poets tutorial series at <https://www.youtube.com/playlist?list=PLRqwX-V7Uu6ZF9C0YMKuns9sLDzK6zoiV>. * An introductory tutorial lecture is at <https://youtu.be/xuB1Id2Wxak> - this is a good explanation of version control and Git, and goes on to teaching how to use it and basic commands. * An alternative tutorial is <https://youtu.be/1u2qu-EmIRc> - also very long but great information and demonstration. | |