## Moved to Github Repository - find latest copy there

## Setting up LaTeX

1. Install MikTeX:
   1. Find a MikTeX distributable [here](https://miktex.org/).
   2. You can install it just for the current user.
   3. Make sure you allow installation of packages on the fly (w or w/o approval)
2. Install [TeXnic Center](http://www.texniccenter.org/) (PC) or <insert program here> (Mac)
3. If this is your first time using LaTeX set up a new build profile:
   1. Open up TeXnic Center.
   2. Navigate to the “build” menu, and select “define output profile”
   3. Name the profile “GameTeX”
   4. Set the “source” path to: /Miktek/Miktek/bin/x64 (or whatever matches wherever you installed MikTeX on your computer.)
   5. In the next text box, “commands to compiler”, add the following: *-interaction=nonstopmode "%pm" -enable-write18*
   6. Append to the end of the text above (one space between the previous text and this):
   7. -include-directory="<path to LaTeX folder of game>"

E.g.: -include-directory="C:\Users\silve\Documents\GitHub\Neptune-Ball\LaTeX"

1. If you’ve used LaTeX for game writing before, to add additional games, just repeat step 3.g. It is recommended that rather than replacing the “include-directory” portion, that you simply append it to the end of the line of text. This ensures that you don’t lose the ability to compile previous games.
2. If you haven’t done so yet, set up GitHub on your computer, clone the repository to your computer, and adjust the “\*\_path.cls” file. All of these steps are described in the “[Setting up GitHub](#_fk0licm8ivdx)” section.

## Setting Up GitHub

By habit, we use GitHub for our version control. This makes collaborating with folks easier, and makes it possible to revert versions of documents in case something breaks.

1. If you don’t already have a github account, go to [github.com](https://github.com/) and sign up for one. It’s free. **Keep track of your username;** you’ll need it to get write access to the repository.
2. If a repository hasn’t been started yet, start one. Follow the steps described in the “[Setting up the Repository](#_cm4kh9xpcnbo)” section below.
   1. If someone else started the repository, provide your GitHub username for them to invite you as a collaborator. You must accept the invite to collaborate before can do anything.
3. **Recommended** (unless you are comfortable using Git with the command line): Download the [github desktop app](https://desktop.github.com/).
4. Clone the repository down to your desktop using either the web interface or the desktop app.
5. Copy the “<game name>\_path” file into the base folder. Open this file and change the path inside the quotation marks to match the path to the “LaTex” folder in the repository **on your own computer**. Make sure to switch backslashes (“\”) to forward slashes (“/”). Theoretically double backslashes (“\\”) also work, but we’ve had only mixed success with this method. Save and close the file.

## Using GitHub for Version Control

1. Before you start work, “fetch” and then “pull” from the repository to make sure you have the most up to date version of everything.
2. Make your changes. Save often.
3. At regular intervals, you should back your work up. Absolutely do so at the end of your working session, but I recommend doing it more frequently, especially if you’re making a lot of changes to many different files, or you know for sure that other people are working on things at the same time - this helps avoid conflicts.
   1. To back your work up, open the desktop app and select the relevant repository.
   2. Add a title and description to your changes. E.g.: “Added the King’s character sheet, assigned everyone their abilities.” The more descriptive you can be, the easier it will be to track back to when a particular change happened if we need to.
   3. “Commit” the change, then “push” it. You may get an error that you have to “fetch” and “pull” first. This means that someone else pushed something to the repository since the last time you pulled. Just do as instructed and “fetch” then “pull”. If there are no conflicts, you should then be able to “commit” and “push” your own changes.
4. If you get a “conflict” error:
   1. Use the interface to figure out which file(s) are in conflict). Open up the two files, figure out where the conflict occurred (usually because two people edited the same line of code/same paragraph and then both tried to “commit” and “push” their version. The second person will get the conflict error.) The conflict section should have a bunch of “>>>>” and “<<<<” around it.
   2. Use your judgement to arbitrate between the two versions; collaborate as necessary with whoever wrote the other version. (Often this is just something like which order the abilities are listed in; it’s a completely trivial fix and doesn’t require checking with your collaborators.) When you are done, save the file.
   3. Commit and push the change.
5. Unless you are very familiar with Git AND with GameTeX, please do NOT branch the project. You’ll probably end up creating more headaches for yourself and your fellow collaborators.

## Setting up the Repository

These are the details for setting up the repository. Only one person has to do this, and it only has to be done once at the beginning of each game.

1. Start a new project on GitHub.
   1. Make it public (private projects require a paid account as of 12Sep2019)
   2. Initiate with a read-me.
   3. Don’t worry about a git.ignore, we’ll add that in a later step.
   4. Pick the “MIT license”.
2. Clone the repository made in step 2 to your computer. You can save it wherever you like, but by default a “GitHub” folder will be created inside your “documents” folder. This is a reasonable default location.
3. Populate the repository:
   1. Open up the folder for the repository in your file folder on your own computer.
   2. Copy in the relevant folder structure (and contents) from another Game project. If you don’t already have one, clone this repository to your computer and copy the files from there <insert link to repository with just base folder structure>.
   3. Make sure the git.ignore file is included in this. If it isn’t, create a new “.txt” file in the base repository folder and copy in the text from this google doc <insert link>.
4. Modify the Game Path file:
   1. In the “LaTeX” folder inside your repository, find the file labeled “Game\_Path.cls”. Open that file. You may have to copy it by hand from another game.
   2. Update the path inside the quotation marks to match the path to the base folder in the repository. Make sure to switch backslashes (“\”) to forward slashes (“/”). Theoretically double backslashes (“\\”) also work, but we’ve had only mixed success with this method.
   3. Save and close the file.
   4. Rename the “Game\_Path” file to replace the word “Game” with the “Game Name” for your game (for example: “GL2020\_Path”, “GC2019\_Path”, “MermaidGame\_Path”). Whatever it is, pick something fairly short but unique; you’ll need to change it in a bunch of places.
5. Update the Game Name in all the relevant places:
   1. Open up the “Game.cls” file in the “LaTeX” folder. Search for “game name”. Replace the text in quotations (should be “game” if you copied files from the skeleton linked in 3.b) with whatever “Game Name” you picked in step 4.d.
   2. While you're here, if you know when the game is running, update the run date and take down by dates. Otherwise leave the placeholders.
   3. Save and close the file.
   4. Rename the file to “game name.” e.g.: “game.tex” -> “GL2020.cls”
   5. Open up each folder except the “handouts”, “latex”, and “production” folders. In each folder:
      1. Open the template file. Change the “game” tag at the top with the “game name”.
      2. Save and close the file
   6. In the production file, open **every** file and make the same change as described in step 5.d.
6. Open the “git.ignore” file. Add the file name for the “<game name>\_path.cls” file you prepared in step 4. You can replace the entry for “game\_path.cls”.
7. Commit your changes to the repository, then push them.
8. Invite collaborators to the repository using their username. Find the interface to add collaborators by navigating to the repository, then clicking on “settings”, then “collaborators”. They have to accept the invite before they can work on the project. Have them clone the repository to their desktop.
9. Email out a copy of the “game\_path” file. It will NOT be version controlled by Git. We specifically tell Git to ignore this file in step 6.