Playing in the git-playground!

Start by making a fork of the git-playground repository using your GitHub account. We'll be using ASCII art instead of code to illustrate how things work. URL: https://github.com/NealHumphrey/git-playground

```
# Setup your repo:
> git clone <your-fork-of-git-playground>
> cd git-playground        #moves you into the folder
> git remote add upstream <url-of-original>  #adds remote called 'upstream'

# Use these between every step to see what's going on
> git status
> git branch
> git log --oneline --graph --decorate --all
```

Activity # 1 – Add an animal to the zoo, and make a pull request on GitHub!

Follow the standard workflow. Be sure to watch for common problems!

- Don't change branches if there are uncommitted changes
- Make sure your command line is *in* your project directory! (use 'cd my-folder' or 'cd ..' to navigate)
- Check status often and make sure it matches what you expect!
- Read the output of every command and look out for error messages

```
# Start your workflow
> git checkout dev          #always check for uncommitted changes before!
> git fetch upstream dev    #gets changes from 'upstream', if any
> git merge upstream/dev    #adds 'upstream/dev' to your local dev
> git branch                #lists all branches and says which you are on
> git branch <my-feature>  #Where you'll store your work
> git checkout <my-feature> #Move to the newly created branch

<<<Add an ASCII animal to the zoo.txt file, save the file>>>

#Commit your changes
> git status                #Do git status often, but esp. before a commit
> git add --all             #Stage the files - you decide what to commit!
> git status                #Make sure all the files are staged!
> git commit

# Tell your Github fork about your changes
> git push origin <my-feature>   #sends changes to the repo called 'origin'

<<<go to Github and make a pull request!>>>
```

Activity #2 - Merge two feature branches

Use all the same commands as above. But, after you commit your ASCII animal, make a second branch <my-second feature> that starts from the end of <my-cool-feature> branch. Then, you'll go back to <my-cool-branch> and merge in your second branch before pushing to Github.

```
# Merging branches
> git status                        # am I on the branch I want to modify?
> git checkout <my branch>          # if not, check it out. Remember when
                                    # merging, the branch that is checked out
                                    # gets the other changes brought in
> git merge <other-branch>          # <other-branch> will stay the same, but
                                    # <my-branch> will get updated!
```

Activity #3 - Working with friends

Try your hand at an advanced activity. All the commands you need are above, but you'll need to modify their arguments and when to use them to get this activity to happen.

1. Set up your repo to talk to a third repo - whoever is sitting next to you. Get the 'clone' path from their Github fork of git-playground. Use `git remote add…` to do this.
2. Modify the git fetch command to get all the changes from their repository - including the ASCII animal branch that they made in Activity #1.
3. Decide where you want to put their changes (e.g. a new branch you make off of dev called "combined-changes')
4. Check out this branch
5. Use `git merge <their-repo-name>\<their-branch-name>` to merge it in. Remember you came up with <their-repo-name> in step one, and they came up with <their-branch-name> in activity #1 !
6. Push your changes and make another pull request!!