

---

title: Git FUN!damentals  
subtitle: Branching  
minutes:

---

## Branching

We've mentioned branching a few times now as the proper workflow. While you may think making changes on your local repository and tracking the changes is sufficient, many workflows will necessitate branching. Branching allows for multiple versions of a repository to exist together in your local files. If you are working on a new feature, you should first create a new branch and then work on the feature in that branch. This allows the master branch to continue serving as the working program, and additionally allows for other intermediate fixes to made to the master branch, all while you're working on your own new feature.

Let's make a new branch in our fruits repository:

```
$ git branch new_fruits  
$ git checkout new_fruits
```

We're going to use this branch to add new fruits, while someone else will take care of the fruits already there. If we type `git branch` now we can see the newly created branch. The checkout command changes the branch you are currently working on. We can switch between branches by typing:

```
$ git checkout master  
$ git checkout new_fruits
```

Each checkout will change the files your working on to the state of that branch. Let's add a couple fruits to `fruit_list.txt`:

```
$ nano fruit_list.txt
```

Adding:

```
banana  
kiwi  
peach  
orange  
grape
```

Then we add and commit the changes *to that branch* (it will automatically add and commit to the branch you are checked out on):

```
$ git add fruit_list.txt  
$ git commit -m "added some new fruits"
```

If the feature works, and we decide we like it, we can merge it into the master branch, after first checking out to master:

```
$ git checkout master
```

If we cat `fruit_list.txt`, we see that it does not include our new feature. To merge:

```
$ git merge new_fruits
```

Now the master branch has our new feature. We can delete the branch when work on the feature is over:

```
$ git branch -d new_fruits
```

When we type `git branch` we no longer see our branch.

## Vegetables

Everyone will now be working with the person next to them as if you were collaborating on a project together. As we've been working with fruits today, let's finish with vegetables. Partner 1 should create a new repository in GitHub – "vegetables", clone the repository to their computer, create a new branch, add a file titled `vegetables_list.txt`, which contains a list of vegetables, add the file, commit the file, merge it to the master branch, and push it to the remote repository. After this, you should see the file inside the repository on GitHub.

Partner 1 should then give access to the repository to Partner 2 by clicking on the repository, clicking the "Settings" tab, then "Collaborators", and adding Partner 2.

Partner 2 should then clone Partner 1's repository, create a new branch, modify `vegetables_.txt`, add the file, commit the file, merge to master, and push to the remote repository. The changes should now appear on the GitHub page.

To complete the process, Partner 1 should pull the changes from the remote to update their local repository.