GIT: From Beginner To Fearless

GIT Branching Activity: A Single developer branching exercise

Brian Gorman, Author/Instructor/Trainer

Introduction

Branches are one of the key functions of working with GIT. While a single developer could work on a repository without ever branching, using branches is a critical way to make sure that it's very easy to protect your work and have the flexibility to make changes without worrying about messing up your repository.

In this activity, we're going to learn about using branches. By creating and using a branch, we have the opportunity to start working, save our changes with a commit, and easily switch back to the previous version if need be. Additionally, after we have our changes completed, we can then merge our changes into our main master branch.

This activity will NOT complete the circuit with a merge (we'll look at that in the next activity to finish this up). However, we'll take the time to learn about creating and working with branches, which will set us up for the next steps.

Let's get started!

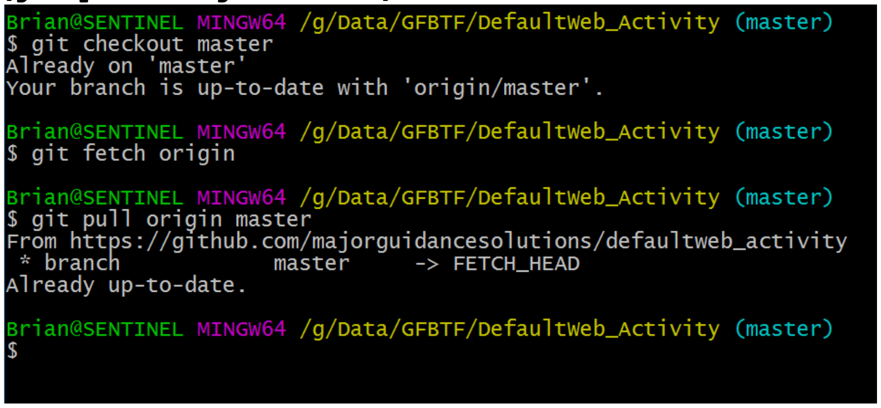
Notes

Step 1: Creating a new branch

a) Make sure you are in any repository, and that you are on master and up-to-date. For this activity, we'll be using our default web, but you can use any repository.

Always remember to run the following commands from the master branch before starting new work:

[**`git checkout master`**]
[**`git fetch origin`**]
[**`git pull origin master`**]

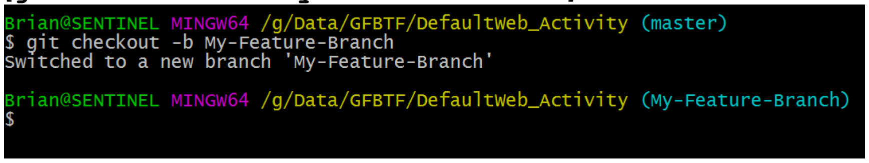


```
Brian@SENTINEL MINGW64 /g/Data/GFBTF/DefaultWeb_Activity (master)
$ git checkout master
Already on 'master'
Your branch is up-to-date with 'origin/master'.

Brian@SENTINEL MINGW64 /g/Data/GFBTF/DefaultWeb_Activity (master)
$ git fetch origin

Brian@SENTINEL MINGW64 /g/Data/GFBTF/DefaultWeb_Activity (master)
$ git pull origin master
From https://github.com/majorguidancesolutions/defaultweb_activity
 * branch            master     -> FETCH_HEAD
Already up-to-date.

Brian@SENTINEL MINGW64 /g/Data/GFBTF/DefaultWeb_Activity (master)
$
```

b) Create a new feature branch

You might think the command to create a new branch would include the word "branch" in it, however, that would be incorrect. Instead, if we want to create a new branch, we use the checkout command with a flag '-b'.

[**`git checkout -b My-Feature-Branch`**]



```
Brian@SENTINEL MINGW64 /g/Data/GFBTF/DefaultWeb_Activity (master)
$ git checkout -b My-Feature-Branch
Switched to a new branch 'My-Feature-Branch'

Brian@SENTINEL MINGW64 /g/Data/GFBTF/DefaultWeb_Activity (My-Feature-Branch)
$
```

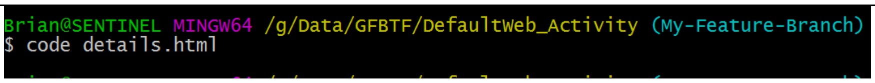
Note that not only did I create the new branch, but I also checked it out, and 'switched' to the new branch.

Step 2: Working on the branches

a) Work on your feature branch

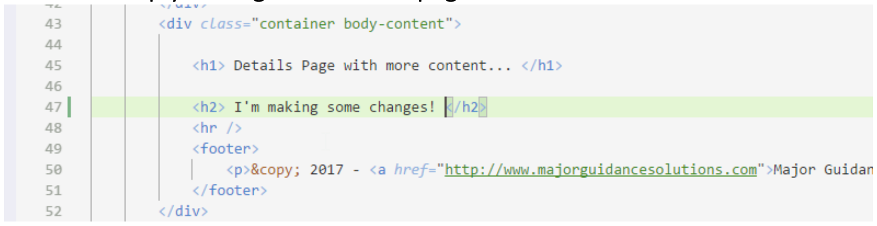
Create some changes on your branch using VSCode or VIM in the details.html file:

[**`code details.html`**]



```
Brian@SENTINEL MINGW64 /g/Data/GFBTF/DefaultWeb_Activity (My-Feature-Branch)
$ code details.html
```

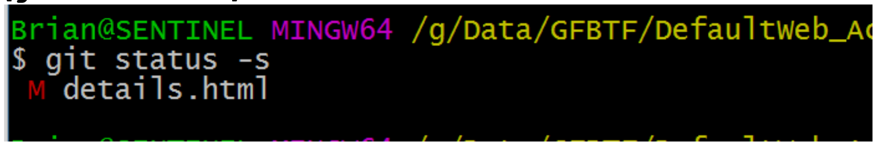
Here I'm simply adding an h2 to the page:



```
<div class="container body-content">

    <h1> Details Page with more content... </h1>

    <h2> I'm making some changes! </h2>
    <hr />
    <footer>
        <p>&copy; 2017 - <a href="http://www.majorguidancesolutions.com">Major Guidan
    </footer>
</div>
```

Save it and close your editor.

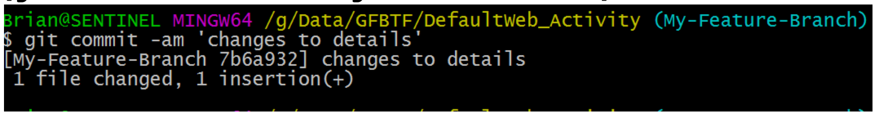
[**git status -s**]



```
Brian@SENTINEL MINGW64 /g/Data/GFBTF/DefaultWeb_Ac
$ git status -s
 M details.html
```

So I have changes. I want to add them. Switching branches with changes will keep the changes as we move to the new branch. This is important because if you forgot to move off master, now is the time to do it. You would check out the new branch and your changes would move with your checkout. Then commit on the branch:

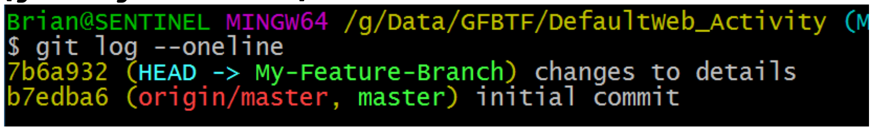
[**git commit -am 'changes to details'**]



```
Brian@SENTINEL MINGW64 /g/Data/GFBTF/DefaultWeb_Activity (My-Feature-Branch)
$ git commit -am 'changes to details'
[My-Feature-Branch 7b6a932] changes to details
 1 file changed, 1 insertion(+)
```

View our history:

[**git log --oneline**]

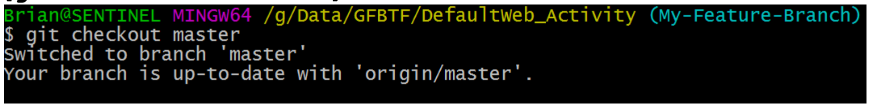


```
Brian@SENTINEL MINGW64 /g/Data/GFBTF/DefaultWeb_Activity (M
$ git log --oneline
7b6a932 (HEAD -> My-Feature-Branch) changes to details
b7edba6 (origin/master, master) initial commit
```

b) Switch branches

Next we're going to switch back to the master branch. By doing this, we'll see our change go away, as well as our commit:

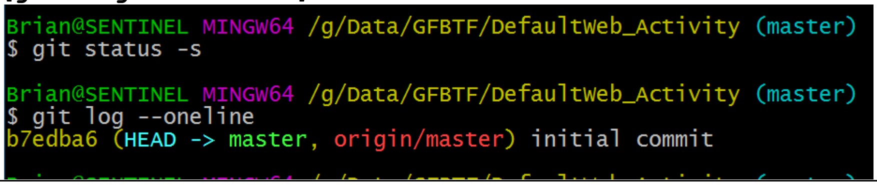
[**git checkout master**]



```
Brian@SENTINEL MINGW64 /g/Data/GFBTF/DefaultWeb_Activity (My-Feature-Branch)
$ git checkout master
Switched to branch 'master'
Your branch is up-to-date with 'origin/master'.
```

[**git status -s**]

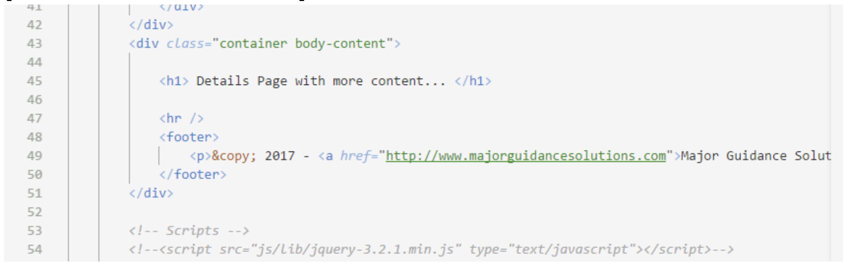
[**git log --oneline**]



```
Brian@SENTINEL MINGW64 /g/Data/GFBTF/DefaultWeb_Activity (master)
$ git status -s

Brian@SENTINEL MINGW64 /g/Data/GFBTF/DefaultWeb_Activity (master)
$ git log --oneline
b7edba6 (HEAD -> master, origin/master) initial commit
```

NOTE: the commit 7b6a932 is GONE!
This is EXPECTED. Our branch is "1 commit ahead of master" and that commit contains our change to details.

Open the file to see the change is missing:
[**code details.html**]



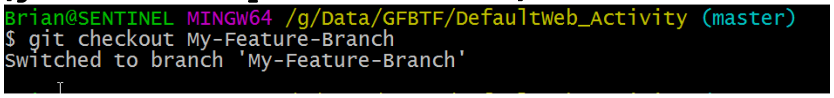
```
        </div>
        <div class="container body-content">

            <h1> Details Page with more content... </h1>

            <hr />
            <footer>
                <p>&copy; 2017 - <a href="http://www.majorguidancesolutions.com">Major Guidance Solut
            </footer>
        </div>

        <!-- Scripts -->
        <!--<script src="js/lib/jquery-3.2.1.min.js" type="text/javascript"></script>-->
```

[don't make any changes, and don't close it. Or close it and then re-open after switching branches].

We see that working on a branch allows us to keep our "master" branch protected in case some of the changes we are making go horribly wrong!

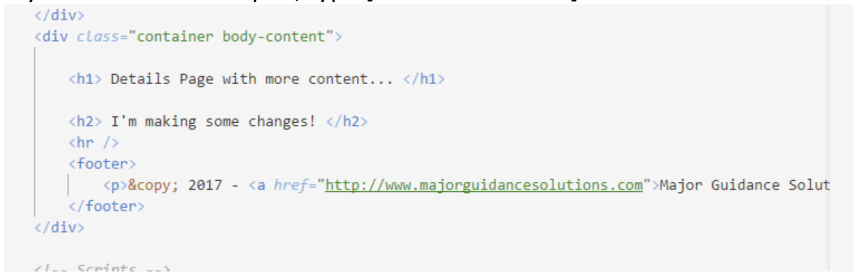
c) Switch back to feature branch

[**git checkout My-Feature-Branch**]



```
Brian@SENTINEL MINGW64 /g/Data/GFBTF/DefaultWeb_Activity (master)
$ git checkout My-Feature-Branch
Switched to branch 'My-Feature-Branch'
```

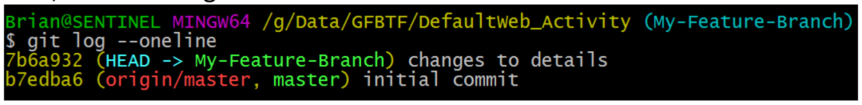
IF you left code open, look at it now!
If you didn't leave it open, type [code details.html]



```
</div>
<div class="container body-content">

    <h1> Details Page with more content... </h1>

    <h2> I'm making some changes! </h2>
    <hr />
    <footer>
        <p>&copy; 2017 - <a href="http://www.majorguidancesolutions.com">Major Guidance Solut
    </footer>
</div>
```

Our changes are back!

Also, check the log:

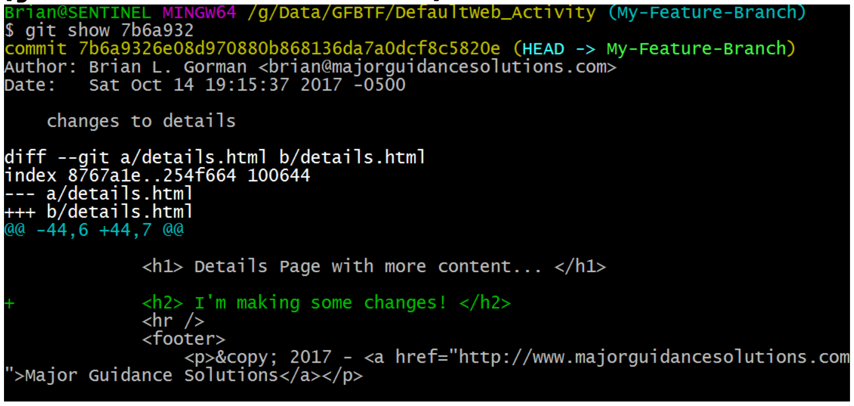


```
Brian@SENTINEL MINGW64 /g/Data/GFBTF/DefaultWeb_Activity (My-Feature-Branch)
$ git log --oneline
7b6a932 (HEAD -> My-Feature-Branch) changes to details
b7edba6 (origin/master, master) initial commit
```

Our commit is back, just like we expected.

Show the changes [note, you'll need to replace the commit id to what you see in your log report]:

[**`git show <some-commit-id>`**]



```
Brian@SENTINEL MINGW64 /g/Data/GFBTF/DefaultWeb_Activity (My-Feature-Branch)
$ git show 7b6a932
commit 7b6a9326e08d970880b868136da7a0dcf8c5820e (HEAD -> My-Feature-Branch)
Author: Brian L. Gorman <brian@majorguidancesolutions.com>
Date:   Sat Oct 14 19:15:37 2017 -0500

    changes to details

diff --git a/details.html b/details.html
index 8767a1e..254f664 100644
--- a/details.html
+++ b/details.html
@@ -44,6 +44,7 @@

             <h1> Details Page with more content... </h1>

+            <h2> I'm making some changes! </h2>
             <hr />
             <footer>
                 <p>&copy; 2017 - <a href="http://www.majorguidancesolutions.com
">Major Guidance Solutions</a></p>
```

Leave this branch intact. We're going to use this in the next activity (merging).

Step 3: Other [git branch] commands of note

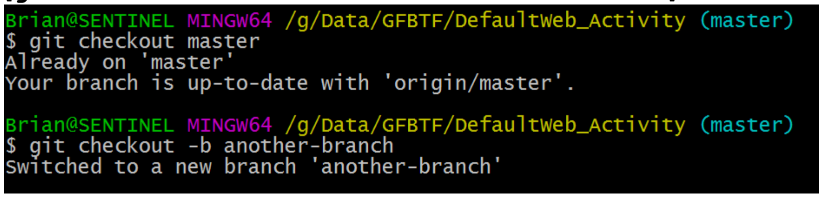
a) Create a branch locally to delete later.

Switch to master – this hurts nothing, no fear here:

[**`git checkout master`**]

Make sure you left the other branch alone. Switch to a new branch

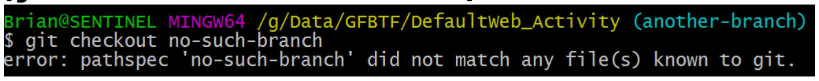
[**`git checkout -b <some-branch-name-here>`**]



```
Brian@SENTINEL MINGW64 /g/Data/GFBTF/DefaultWeb_Activity (master)
$ git checkout master
Already on 'master'
Your branch is up-to-date with 'origin/master'.

Brian@SENTINEL MINGW64 /g/Data/GFBTF/DefaultWeb_Activity (master)
$ git checkout -b another-branch
Switched to a new branch 'another-branch'
```

As an FYI – if you forget the –b, you'll get a 'pathspec' error:

[**`git checkout no-such-branch`**]

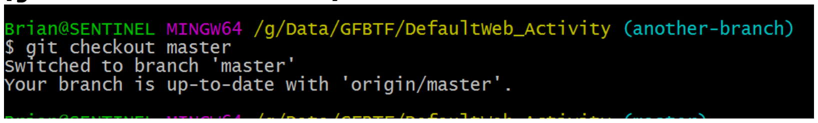


```
Brian@SENTINEL MINGW64 /g/Data/GFBTF/DefaultWeb_Activity (another-branch)
$ git checkout no-such-branch
error: pathspec 'no-such-branch' did not match any file(s) known to git.
```

b) Listing branches

Now switch back to master

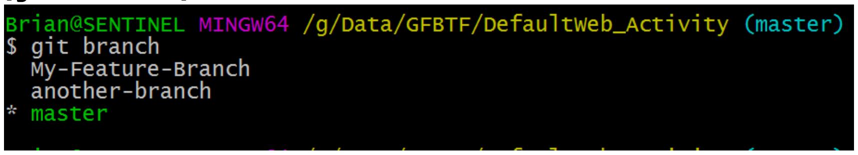
[**`git checkout master`**]



```
Brian@SENTINEL MINGW64 /g/Data/GFBTF/DefaultWeb_Activity (another-branch)
$ git checkout master
Switched to branch 'master'
Your branch is up-to-date with 'origin/master'.
```

List your branches

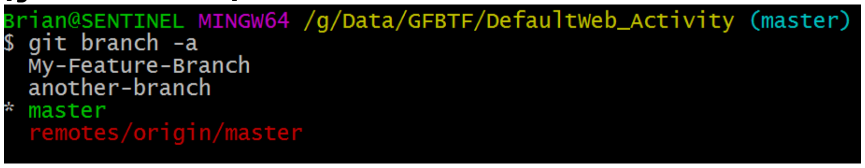
[**git branch**]



```
Brian@SENTINEL MINGW64 /g/Data/GFBTF/DefaultWeb_Activity (master)
$ git branch
  My-Feature-Branch
  another-branch
* master
```

List all the branches. Use the -a flag to list all branches, including branches at remote that we know about.

[**git branch -a**]

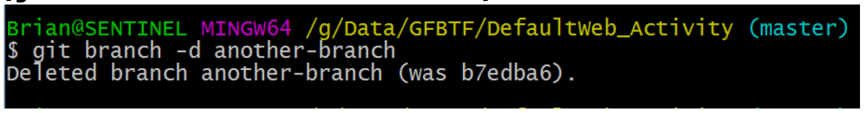


```
Brian@SENTINEL MINGW64 /g/Data/GFBTF/DefaultWeb_Activity (master)
$ git branch -a
  My-Feature-Branch
  another-branch
* master
  remotes/origin/master
```

c) Delete the local branch

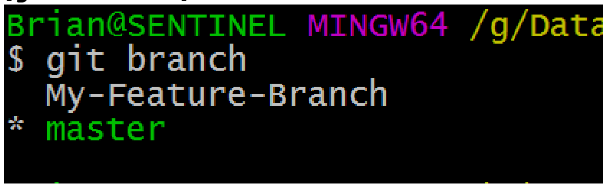
NOTE: you should NOT be on the branch you want to delete, and the branch to delete needs to not have committed changes at this point. Note: DO NOT DELETE the branch: My-Feature-Branch

[**git branch -d another-branch**]



```
Brian@SENTINEL MINGW64 /g/Data/GFBTF/DefaultWeb_Activity (master)
$ git branch -d another-branch
Deleted branch another-branch (was b7edba6).
```

[**git branch**]

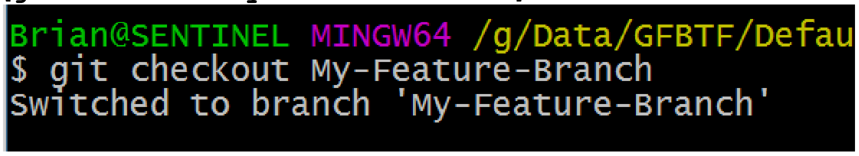


```
Brian@SENTINEL MINGW64 /g/Data
$ git branch
  My-Feature-Branch
* master
```

d) Force delete a local branch

Switch back to your feature branch that is one commit ahead of master

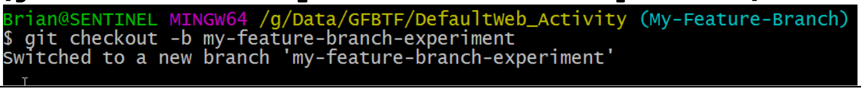
[**git checkout My-Feature-Branch**]



```
Brian@SENTINEL MINGW64 /g/Data/GFBTF/Defau
$ git checkout My-Feature-Branch
Switched to branch 'My-Feature-Branch'
```

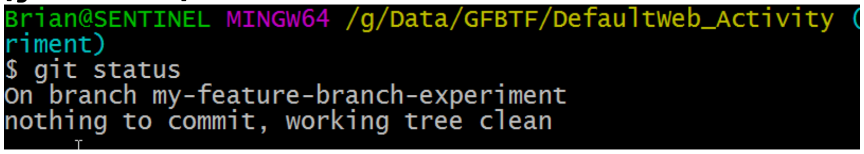
Now create a new branch from here [it will be like My-Feature-Branch, with a commit and changes – so anytime you need to experiment from any branch, you can always just branch off of the branch!]

[**git checkout -b my-feature-branch-experiment**]



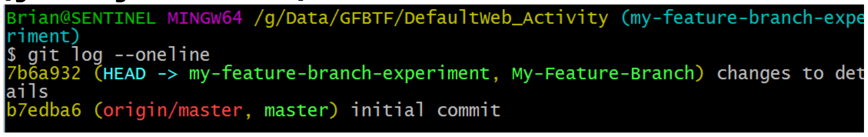
```
Brian@SENTINEL MINGW64 /g/Data/GFBTF/DefaultWeb_Activity (My-Feature-Branch)
$ git checkout -b my-feature-branch-experiment
Switched to a new branch 'my-feature-branch-experiment'
```

[git status]



```
Brian@SENTINEL MINGW64 /g/Data/GFBTF/DefaultWeb_Activity (
riment)
$ git status
On branch my-feature-branch-experiment
nothing to commit, working tree clean
```

[git log --oneline]

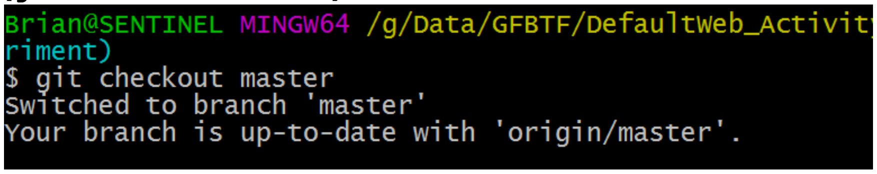


```
Brian@SENTINEL MINGW64 /g/Data/GFBTF/DefaultWeb_Activity (my-feature-branch-expe
riment)
$ git log --oneline
7b6a932 (HEAD -> my-feature-branch-experiment, My-Feature-Branch) changes to det
ails
b7edba6 (origin/master, master) initial commit
```

Note: the branch is on the same commit as the parent it was created from. If you wanted, you could do another commit and log to see the current feature branch move ahead another commit.

Switch back to master:

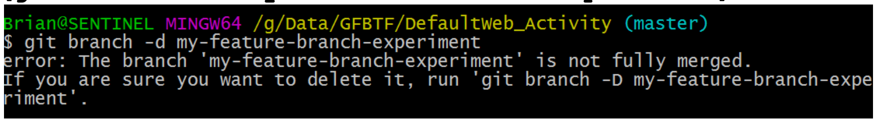
[git checkout master]



```
Brian@SENTINEL MINGW64 /g/Data/GFBTF/DefaultWeb_Activit
riment)
$ git checkout master
Switched to branch 'master'
Your branch is up-to-date with 'origin/master'.
```

And delete the experiment branch:

[git branch -d my-feature-branch-experiment]

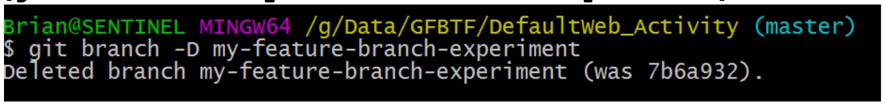


```
Brian@SENTINEL MINGW64 /g/Data/GFBTF/DefaultWeb_Activity (master)
$ git branch -d my-feature-branch-experiment
error: The branch 'my-feature-branch-experiment' is not fully merged.
If you are sure you want to delete it, run 'git branch -D my-feature-branch-expe
riment'.
```

What? Why did that happen?
Because the commit history shows that this branch is one ahead of master, just like the other branch would be, and GIT gives you a protective layer – an "are you sure you want to do this" type-of warning!

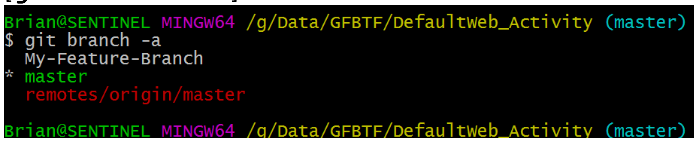
Now, we know we do want to do this, so we'll force the issue by changing the -d to -D. Yes, seriously, all it takes to force the issue is making a small d into a capital D.

[git branch –D my-feature-branch-experiment]



```
Brian@SENTINEL MINGW64 /g/Data/GFBTF/DefaultWeb_Activity (master)
$ git branch -D my-feature-branch-experiment
Deleted branch my-feature-branch-experiment (was 7b6a932).
```

[git branch -a]



```
Brian@SENTINEL MINGW64 /g/Data/GFBTF/DefaultWeb_Activity (master)
$ git branch -a
  My-Feature-Branch
* master
  remotes/origin/master

Brian@SENTINEL MINGW64 /g/Data/GFBTF/DefaultWeb_Activity (master)
```

This concludes our GIT: Branching activity

Closing Thoughts

In this activity, we have briefly seen how to work with branches. We noticed that branches are lightweight and easy to checkout, and our changes definitely live within the branches we commit to.

We got to see what it takes to create local branches, change some things, make a commit, and then switch branches to see the commit go away, followed by switching back to see it come back.

We also learned about deleting branches with no changes, and then we finished up by seeing how to delete a branch that had unmerged changes. During that last part, we also saw how the location branch we are in when creating a branch is critical. For this reason, most of the time you'll want to create branches from master. However, if you want to experiment, you can always create a branch from any location, and nest them infinitely, if you so desire.

Take a few minutes to make some notes about the various commands we've learned about in this activity, and practice using them.

Notes