PRUEBAS DE DESARROLLO DE CURSOS GIT Y GITHUB

VERSION CONTROL WTIH GIT COMPLETADO

Version Control with Git

RETOMAR LA CLASE 6

LESSON 1

What is Version Control?

Version control is an incredibly important part of a professional programmer's life. In this lesson, you'll learn about the benefits of version control and install the version control tool Git!

VIEW LESSON →

100% VISUALIZADO(S)   COMPRIMIR

Create A Git Repo   ✓ FINALIZADO

Version Control with Git   **RETOMAR LA CLASE 6**

Create A Git Repo   ✓ FINALIZADO

Review a Repo's History   ✓ FINALIZADO

Add Commits To A Repo   ✓ FINALIZADO

Tagging, Branching, and Merging   ✓ FINALIZADO

Undoing Changes   ✓ FINALIZADO

ALGUNAS PREGUNTAS

The HTML file has HTML and CSS changes on the Staging Index and an additional HTML change in the Working Directory. From what you've learned so far with how committing works, what do you think will get committed if a commit were made right now?

○ only the HTML file changes on the Staging Index

○ only the CSS file changes on the Staging Index

✓ the HTML and CSS file changes on the Staging Index

○ the HTML file changes in *both* the Working Directory *and* the Staging Index

Did you run `git init` yet? If not, do it now because you'll need it to answer this quiz!

After running `git init`, the text "Initialized empty Git repository in " followed by a path. should have appeared. The question is, has anything changed with your command prompt? If so, what?

✓ Yes - I now see the word "master".

○ No, it looks exactly the same.

Why don't you check out the documentation for `git clone` and pick the correct way to do it from the options below. The command should clone the blog project repo and store it in a directory named `blog-project`.

- ◯ git clone-new-dir https://github.com/udacity/course-git-blog-project blog-project

- ◯ git clone https://github.com/udacity/course-git-blog-project --out blog-project

- ◯ git clone https://github.com/udacity/course-git-blog-project --rename blog-project

- ✓ git clone https://github.com/udacity/course-git-blog-project blog-project

To answer this quiz, make sure you've `cd` ed into the `course-git-blog-project` project. If you've been following along in this lesson and haven't added any files to this project, then what does running `git status` display?

- ◯ Status: good

- ◯ On branch master
  Your branch is up-to-date with 'origin/master'.
  Initial commit
  nothing to commit (create/copy files and use "git add" to track)

- ◯ master branch
  Please commit some files

- ✓ On branch master
  Your branch is up-to-date with 'origin/master'.
  nothing to commit, working directory clean

To answer this quiz, make sure you've `cd` ed into the `new-git-project` project.

If you've been following along in this lesson and haven't added any files to this project, then what does running `git status` display?

○ Status: good

✓ On branch master
  Initial commit
  nothing to commit (create/copy files and use "git add" to track)

○ master branch
  Please commit some files

○ On branch master
  nothing to commit, working directory clean

After you've cloned the blog project repository, navigate to the project's directory using the command line. Once you're located inside the blog project, what is the very first thing you should do in a Git repository?

✓ run the `git status` command

○ open the project in a code editor

○ decide what new feature to work on

In the `index.html` file, take a look at the `<h1>Expedition</h1>` heading around line 15.

Based on what you can see here when was that heading added?

○ It was added on a Tuesday. Yeah, a Tuesday.

○ 3 weeks ago

✓ ¯\(ツ)/¯ I can't tell that by looking at the code.

Ok, so we're not quite sure *when* the heading was added. How about an easier question - *who* added this heading? Again, what can you tell from just looking at the code?

○ Richard did!

✓ No clue

QUIZ QUESTION

You're deep in the weeds of the `git log --oneline` command and want to get out of the `git log --oneline` output and return to the regular command prompt. What do you press on the keyboard to return to the regular command prompt?

○ the `esc` key

✓ the `q` key

○ the `ctrl` + `c` keys

○ the `ctrl` + `d` keys

`git log --stat` and `git log -p` are both really helpful commands. Wouldn't it be great if we could have both of their output at the same time? Hmmm...

What happens when `git log -p --stat` is run?

○ it displays only the patch information

○ it displays only the stats

○ it displays both with the patch info above the stats info

⊘ it displays both with the stats info above the patch info

What does the `-w` flag do to the patch information? For help, check this Git docs page.

○ it displays non-whitespace characters in blinking text

○ it displays non-whitespace changes in bold

⊘ it ignores whitespace changes

○ it shows a separate patch area with just new/removed content

Reviewing the guidelines on what makes a good commit message, is the following commit message good?

"Update the footer to copyright information"

- ✓ Yes
- ○ No

Is the following a good commit message?

"Add a

tag to the body"

- ○ Yes
- ✓ No

Is the following a good commit message?

"Add changes to app.js"

- ○ Yes
- ✓ No

By default, a Git tag will not appear in a log. What flag must be used to display the tag information in the output of `git log`?

- ○ --show-tags
- ○ --tags
- ○ --display-all
- ✓ --decorate

Which of the following will delete the tag `v-1`?

- ✓ git tag --delete v-1
- ☐ git remove v-1
- ✓ git tag -d v-1
- ☐ git delete v-1

Tag Older Commit?

Using the following `git log --oneline` information, what command would you run to give the commit with the message "style page header" a tag of `beta` ?

```
2a9e9f3 add breakpoint for large-sized screens
137a0bd add breakpoint for medium-sized screens
c5ee895 add space around page edge
b552fa5 style page header
f8c87c7 convert social links from text to images
```

git tag -a beta b552fa5

From what you know about both the `git branch` and `git tag` commands, what do you think the following command will do?

```
$ git branch alt-sidebar-loc 42a69f
```

○ will create a branch `alt` at the same commit as the `master` branch

○ will create the 3 branches `alt`, `sidebar`, `loc`

○ will move the master branch to the commit with SHA `42a69f`

✓ will create the `alt-sidebar-loc` branch and have it point to the commit with SHA `42a69f`

Which Branch Is Active?

Given the following output from `git branch`:

```
$ git branch
  barbara
* footer-fix
  master
  richard
  sidebar
  social-icons
```

Which branch is the active branch?

footer-fix

How about a quick quiz to see if you're following along with how merging works.

Let's say a repository has 4 branches in it:

- master
- allisons-mobile-footer-fix
- nav-updates
- jonathans-seo-changes

The changes on `master` and `allisons-mobile-footer-fix` need to be merged together. If HEAD points to `allisons-mobile-footer-fix`, which branch will move when the merge is performed?

○ master

✓ allisons-mobile-footer-fix

○ nav-updates

○ jonathans-seo-changes

CURSO DE GITHUB EN COURSERA-SEMANA 3 COMPLETADA A EXCEPCIÓN DE LA ACTIVIDAD CALIFICABLE QUE ES DE PAGA