**GIT**

**Contents**

* Git Intro
* Git get started
* Git new files
* Git staging environment
* Git commit
* Git Help
* Git branch

**Git Intro**

* Git is a popular version control system.
* It is used for

1.Track code changes

2.Tracking who made changes

3.coding collaboration

Working with Git

* Initialize Git on a folder, making it a **Repository**
* Git now creates a hidden folder to keep track of changes in that folder
* When a file is changed, added or deleted, it is considered **modified**
* You select the modified files you want to **Stage**
* The **Staged** files are **Committed**, which prompts Git to store a **permanent** snapshot of the files
* Git allows you to see the full history of every commit.
* You can revert back to any previous commit.
* Git does not store a separate copy of every file in every commit, but keeps track of changes made in each commit!

**Git Get Started**

Git install

Download git : [https://www.git-scm.com/](https://git-scm.com/)

Using Git with Command Line

git **- -version**

Configure Git

git config --global user.name “helloworld-test”

git congig –global user.email “helloworld.com”

Creating Git folder

git mkdir myproject

cd myproject

* **mkdir** makes a new directory
* **cd** changes the current working directory

Intialize Git

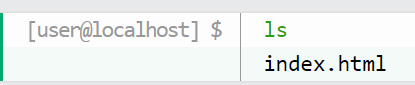
git **init**

* Initialized empty Git repository in /Users/user/myproject/.git/

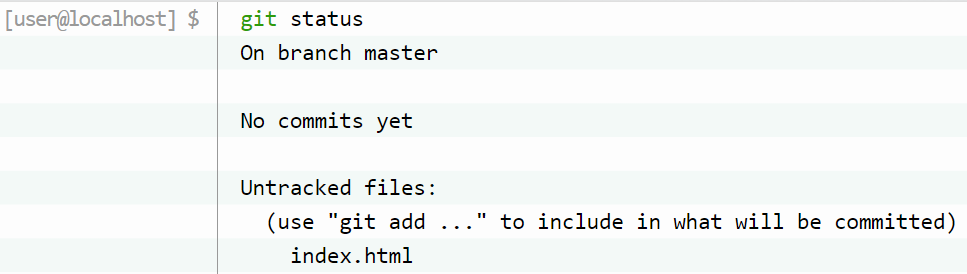
**Git New Files**

Git adding new files

* Created first local Git repo. But it is empty.So lets add some files
* index.html file created and save it in new folder
* Go back to the terminal and list the files in our current working directory and type



* **ls** will list the current directory



* **status** give git status

Git is aware of the file, but has not added it to our repository!

Files in your Git repository folder can be in one of 2 states:

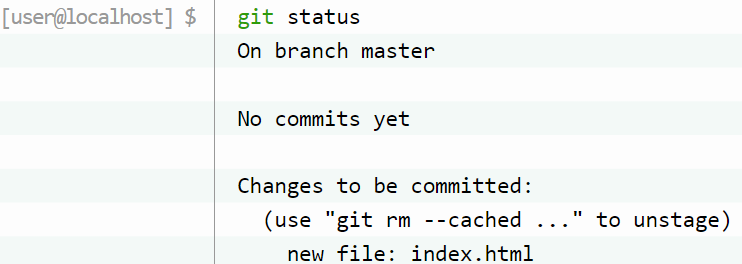
* **Tracked** - files that Git knows about and are added to the repository
* **Untracked** - files that are in your working directory, but not added to the repository

**Git Staging Environment**

* we are adding, editing and removing files. But whenever you do some changes ,you should add the files to a Staging Environment.
* **Staged** files are files that are ready to be **committed** to the repository you are working on
* we are done working with index.html. So we can add it to the Staging Environment



* File should be **Staged**. Let's check the status



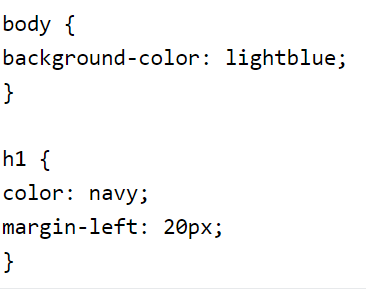
* Now file has been added to staging environment

Adding more than one file

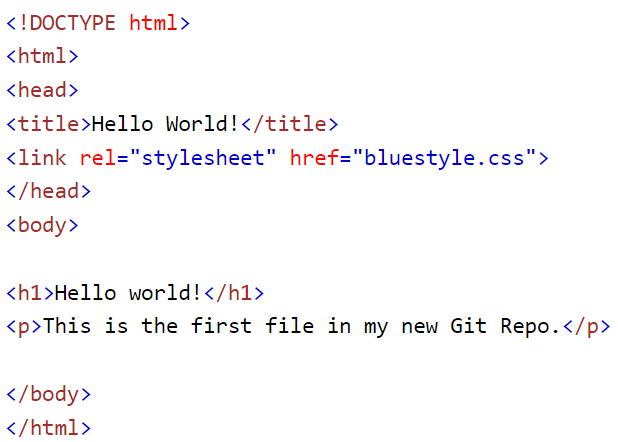
* Create a file **Readme.md**

# hello-world  
Hello World repository for Git tutorial

* Create **css** files



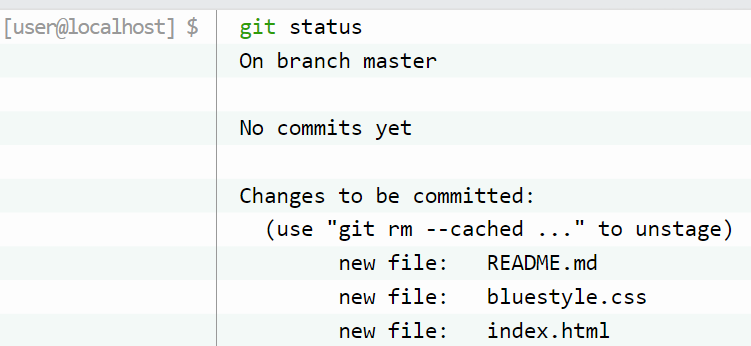
* And **update index.html** to include the stylesheet



* Now add all files in the current directory to the Staging Environment



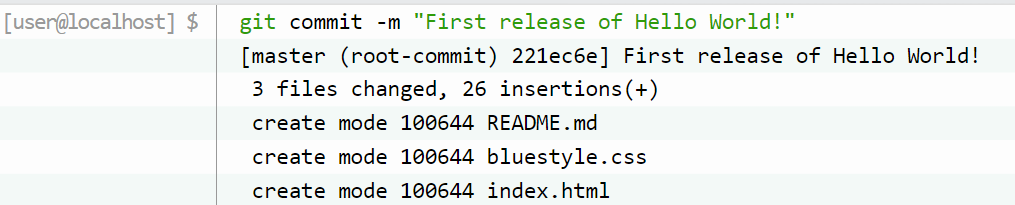
* Again check the status of the git



* Now all 3 files are added to the Staging Environment, and we are ready to do our first commit.

**Git Commit**

* We are moving from stage to commit for our repo.
* Git considers each commit change point or "**save point**",
* Commit **always** include a **message**.

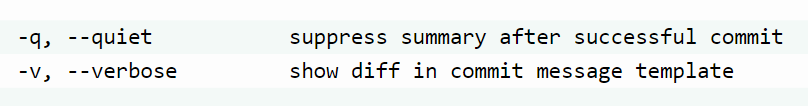
****

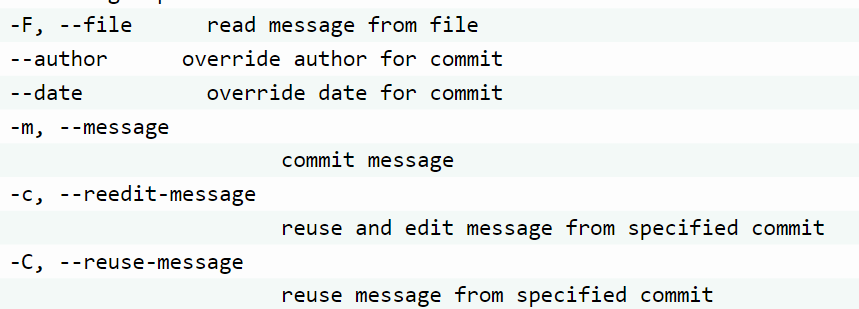
The **commit c**ommand performs a commit, and the **-m "message"** adds a message.

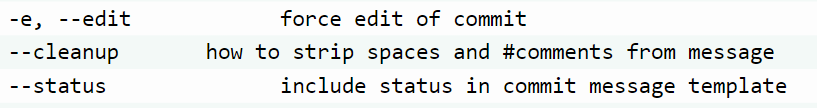
The Staging Environment has been committed to our repo, with the message:  
"First release of Hello World!"

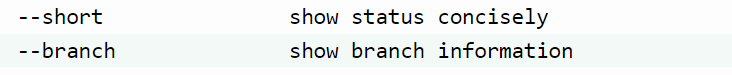
**Git Help**

* If you are having trouble remembering commands or options for commands, you can use **Git help.**
* **git *command* -help** -  See all the available options for the specific command
* **git help --all** -  See all possible commands
* You can also use **--help** instead of **-help** to open the relevant Git manual page.
* Some commands







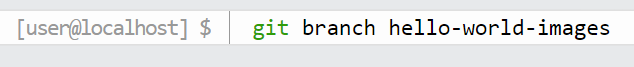


**Git Branch**

* In Git, a **branch** is a new/separate version of the main repository.

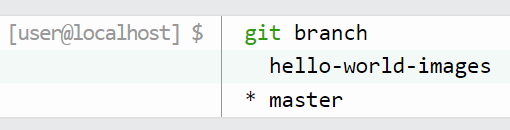
New Git branch

* Let add some new features to our **index.html** page.
* We are working in our local repository, and we do not want to disturb or possibly wreck the main project.
* So we create a new **branch**:

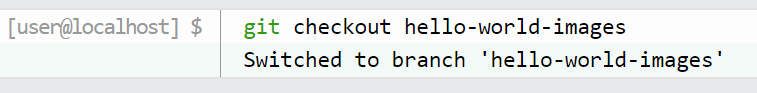
****

Now we created a new **branch**called "**hello-world-images**"

* Let's confirm that we have created a new **branch**:



* We can see the new branch with the name "hello-world-images", but the **\*** beside **master** specifies that we are currently on that **branch.**
* **checkout** is the command used to check out a **branch**. Moving us **from** the current **branch**, **to** the one specified at the end of the command:



* Now we have moved our current workspace from the master branch, to the new **branch**