<https://campus.lemoncode.net/>

https://app.slack.com/client/T04UHHCQFQW/dms

INDEX

Markdown 1

Install software and execute a js proyect in VisualStudio: 2

Bring the repository of other person to your GitHub 4

Created a new public empty repo on GitHub.com and push on it our local repository 5

Authenticate en GitHub 6

Commands 9

Merge GIT resolve conflicts 11

# Markdown

A screenshot of a computer

Description automatically generated

*We can see the result of use Markdown in VisualStudio on the Preview dice. We use special characters like : +, - o \* to give format to the text . If we want to writte this character in our text we have to scape the with \.*

*# Encabezado h1*

*## Encabezado h2*

*### Encabezado h3*

*#### Encabezado h4*

*<p>Soy un Párrafo</p>*

*\*\*\*o* **---** *para línea horizontal.*

*Soy un \*texto en cursiva\**

*Soy otro \_texto en cursiva\_*

*Soy un \*\*texto en negrita\*\*.*

*Una segunda forma de poner el \_\_texto en negrita\_\_.*

*Soy un \*\*\*texto en negrita y cursiva\*\*\*.*

*Soy otro \*\*\_texto en negrita y cursiva\_\*\*.*

*[texto enlace](URL donde vamos a navegar cuando pulsemos en el enlace)*

*[Visitar Lemoncode.](*[*https://lemoncode.net/*](https://lemoncode.net/)*)*

*![Texto alternativo imagen](URL de la imagen). O con:*

*<img src="./content/logo.png" alt="tituloimg" title="titulo img" />*

*> Soy una cita*

*> Soy una cita*

*>*

*> que ocupa varios párrafos*

*Listo of numbers:*

*1. Primer elemento*

*2. Segundo elemento*

*3. Tercer elemento*

*4. Cuarto element*

*For ball list we use: -, or \* or +*

*- Primer elemento*

*- Segundo elemento*

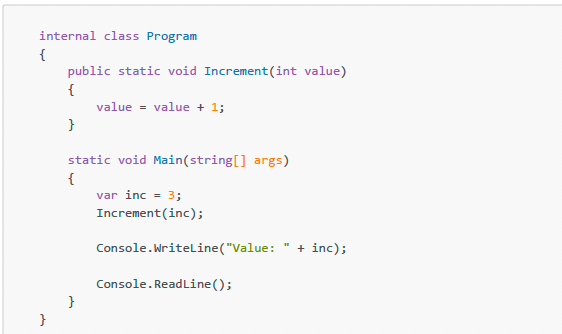
*- Tercer elemento*

*- Cuarto element*

*Introduce code:*

|  |
| --- |
| *```csharp*  *internal class Program*  *{*  *public static void Increment(int value)*  *{*  *value = value + 1;*  *}*  *00-introduccion-markdown.md 2/16/2023*  *7 / 9*  *static void Main(string[] args)*  *{*  *var inc = 3;*  *Increment(inc);*  *Console.WriteLine("Value: " + inc);*  *Console.ReadLine();*  *}*  *}*  *```* |

*I would look like :*

**

# Install software and execute a js proyect in VisualStudio:

|  |
| --- |
| brew install git  brew instala gitk  brew install git-gui  brew Install node  npm -v  10.2.4 |

We use **npm** that is part of **node** to manager code dependences in (NodeJS and JavaScript) software, we write this dependences in the file ”package.json”. The basic dependences are: rimraf (to can execute rm -rf) and Parcel (that optimize the code to work in different browsers).

A basic project that is inside a folder, contains the file “package.json” and the folder “src”. In the “src” folder are the files “index.html” and “index.js”.

|  |
| --- |
| {  "name": "01-basic",  "version": "1.0.0",  "description": "Let's start with a very basic sample, just add an html plus a simple console log (E5). This is what you can find in the getting started tutorial.",  "main": "index.js",  "scripts": {  "build": "rimraf dist && parcel ./src/index.html",  "build:prod": "rimraf dist && parcel build ./src/index.html",  "start": "rimraf dist && parcel ./src/index.html --open"  },  "keywords": [],  "author": "",  "license": "ISC",  "devDependencies": {  "parcel": "^2.8.3",  "rimraf": "^4.1.2"  }  } |

|  |
| --- |
| <html>  <body>  <h1>"Hello"</h1>  <script src="./index.js"></script>  </body>  </html> |

|  |
| --- |
| console.log("!Hola "); |

To execute it in chrome we write in a terminal:

|  |
| --- |
| npm install  BROWSER=Chrome npm start |

# Bring the repository of other person to your GitHub

A screenshot of a computer

Description automatically generated

ejem:

https://github.com/Lemoncode/git-training-2023

We click on “Fork”. When the repository appears in our gitHunb, we can copy it to our desktop with the command:

|  |
| --- |
| git clone “url de nuestro repo” |

We will need to authenticate to be able to clone the repo. We can use: account credentials, Token or a ssh Key.A screenshot of a computer

Description automatically generated

# Created a new public empty repo on GitHub.com and push on it our local repository

1-Go home

A screenshot of a phone

Description automatically generated

2-New

A screenshot of a phone

Description automatically generated

3-Create a empty repo

A screenshot of a computer

Description automatically generated

4-Using the console in your Mac o Linux, push your local files to git, ejem:

|  |
| --- |
| echo "# test" >> README.md  git init  git add README.md  git commit -m "first commit"  git branch -M main  git remote add origin https://github.com/LeonorMalaga/test.git  git push -u origin main |

####—Visual Studio Code ask for authorization, after this, The information appeared in the GitHub account####

\*If we don’t want to copy some folders to github, we have to write them in a file: “.gitignore”, then we can use the command “git add .” to add the rest of the file to the git database.

<https://git-scm.com/docs/gitignore>

# Authenticate en GitHub

## Delete credentials in the PC to can use token or SSH

|  |
| --- |
| $ git credential-osxkeychain erase  host=github.com  protocol=https  > [Press Return]  $ git credential-osxkeychain erase  host=github.com  protocol=ssh  > [Press Return] |

\*It works if doesn’t print anything.

## SSH key

Generate a SSH key

|  |
| --- |
| ssh-keygen -m PEM -t rsa -C keyForGitHub  Generating public/private rsa key pair. |

Enter file in (/Users/leonor/.ssh/id\_rsa) and copy the public key in

your git account->Settings/SSH

git clone git@github.com:LeonorMalaga/mirepositorio.git

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

git clone git@github.com: LeonorMalaga/git-training-2023.git

playing: I open the folder with VisualStudio. I modified a file and write:

|  |
| --- |
| git dad .  git commit -m “ ..”  git push |

, to save the changes in the cloud(GitHub).

A computer screen with white text and green text

Description automatically generated

## Token

->My profile/Settings/Developer settings/Personal Acces Token

A screenshot of a computer

Description automatically generated

ghp\_StsAEQtcS30HpgdtusBqKXFW45gpQY02O7oe

|  |
| --- |
| git init  git add .  git branch -M main  git remote add origin https://github.com/LeonorMalaga/test2.git  git push -u origin main |

# Git Commands (https://www.atlassian.com/es/git/tutorials/)

The files can be in our local repository in four different states, and we can move them from one state to other by commands.

A green and white squares

Description automatically generated

**git –help comman** 🡪 see the git instructions of used of the command.

**git log -10 --all --graph --oneline**—> to see a tree with the last 10 commits of all the branch, to exit press **q**.

**git branch ramaB** —> to create ramaB.

**git branch -d ramaB** —> delete ramaB.

**git checkout ramaB**-> move to ramaB.

**git checkout -b ramaB**—> to create branch b and move to it.

**git merge ramaB -m "important"—>** merge ramaB into the current branch tree with a comment “important”. (-s https://git-scm.com/docs/merge-strategies)

**git merge –-abort->** stop a merge, after a conflict before a commit.

**git branch -a** 🡪 see remote and local branchs.

**git branch –merge main** 🡪 see local branches merged with the main branch (delete candidates. **git branch –d “branchName”).** To delete a branch that is not merged and lost the work done in it writte: **git branch –D “branchName”.**

**git branch –move malnombre newnombre**-> change name of a branch.

**git push origin –delete malnombre**-> delete a branch in the remote repo.

**git reset**—> remove all the file from the staged state.

**git reset HEAD ./src/temporal.**txt -> remove from stage only the file ./src/temporal.txt (https://www.atlassian.com/es/git/tutorials/undoing-changes/git-reset)

**A computer screen shot of a program

Description automatically generated**

**git checkout --.** —> lost the last changes from the tracked files, we remove the file from the modified status.

**git clean -n --.** —> show files to delete (no tracked files) if we execute **git git clean --force**

**git commit -am “coment”** —> to pass from Modified to Stage Area without have to write “git add .”.

**git stach + git pop**—> to save without commit, the pass from Modified

#### git fech and git pull (bring information from remote to local)

**(https://www.atlassian.com/es/git/tutorials/syncing/git-fetch)**

The pull command is a fetch command + a merge. The fetch command downloads the remote code, but you have to merge it later after review it.

Eje:

|  |
| --- |
| git fetch origin  <!—bring the remote code ,ejem bring the branchs: origin/main, origin/develop, origin/some-feature--- for check the code you have to move to this branch-->  git log origin/main—> to see the commits  git checkout origin/main—> for testing and modify  <!—for merge-->  git checkout main  git merge origin/main |

git fetch - -prune -> delete all branches that are actually deleted in the remote repo.

# Merge GIT resolve conflicts

### CONFLICT (modify/delete):

*file2 deleted in mirama and modified in HEAD.*

To keep the file:

|  |
| --- |
| git add file2  git commit -m "recovering file 2" |

To delete the file:

|  |
| --- |
| git rm file2  git commit -m "recovering file 2" |

### CONFLICT (add/add):

Merge conflict in file2

Run the following commands in your terminal

git config merge.tool vimdiff

git config merge.conflictstyle diff3

git config mergetool.prompt false

This will set vimdiff as the default merge tool.

Run the following command in your terminal

git mergetool

You will see a vimdiff display in the following format:

╔═══════╦══════╦════════╗

║ ║ ║ ║

║ LOCAL ║ BASE ║ REMOTE ║

║ ║ ║ ║

╠═══════╩══════╩════════╣

║ ║

║ MERGED ║

║ ║

╚═══════════════════════╝

These 4 views are

LOCAL: this is the file from the current branch

BASE: the common ancestor, how this file looked before both changes

REMOTE: the file you are merging into your branch

MERGED: the merge result; this is what gets saved in the merge commit and used in the future

You can navigate among these views using ctrl+w

1. You can edit the MERGED view like this:
   * If you want to get changes from REMOTE
   * :diffg RE
   * If you want to get changes from BASE
   * :diffg BA
   * If you want to get changes from LOCAL
   * :diffg LO
2. Save, Exit, Commit, and Clean up

:wqa save and exit from vi

git commit -a -m "message"