

# C++ Review DUNE

---

Una organización donde compartir notas acerca de C++ con PDFs escritos en  $\text{\LaTeX}$ .

22 de enero del 2022

 Pad de apuntes

 Liga del PDF

 Sesión grabada en `diode.zone`

\$ date ( Lima,  Bogotá,  Ciudad de México -1)

- Sat Jan 22 07:00:00 AM -05 2022.
- Sun Jan 23 07:00:00 AM -05 2022.

`git`  <https://man.archlinux.org/man/git.1>

`git-config`  <https://man.archlinux.org/man/git-config.1>

`git-init`  <https://man.archlinux.org/man/git-init.1.en>

`git-remote`  <https://man.archlinux.org/man/extra/git/git-remote.1.en>

`git-push`  <https://man.archlinux.org/man/git-push.1>

`git-stash`  <https://man.archlinux.org/man/git-stash.1>

`git-log`  <https://man.archlinux.org/man/git-log.1>

`git-show`  <https://man.archlinux.org/man/git-show.1>

`git-diff`  <https://man.archlinux.org/man/git-diff.1>

`git-gc`  <https://man.archlinux.org/man/extra/git/git-gc.1.en>

`git-mv`  <https://man.archlinux.org/man/extra/git/git-mv.1.en>

`git-rm`  <https://man.archlinux.org/man/extra/git/git-rm.1.en>

`git-add`  <https://man.archlinux.org/man/extra/git/git-add.1.en>

`git-grep`  <https://man.archlinux.org/man/extra/git/git-grep.1.en>

`git-branch`  <https://man.archlinux.org/man/git-branch.1>

`git-checkout`  <https://man.archlinux.org/man/git-checkout.1>

`git-rebase`  <https://man.archlinux.org/man/extra/git/git-rebase.1.en>

`git-lfs`  <https://man.archlinux.org/man/community/git-lfs/git-lfs.1.en>

`git-log`  <https://man.archlinux.org/man/extra/git/git-log.1.en>

`git-tag`  <https://man.archlinux.org/man/extra/git/git-tag.1.en>

`git-diff`  <https://man.archlinux.org/man/extra/git/git-diff.1.en>

1. Descargue el tarball del repositorio dune-pdelab.

```
$ curl -O https://gitlab.dune-project.org/pdelab/dune-pdelab/-/archive/master/dune-pdelab-master.tar.gz
```

2. Realice el tutorial de git <https://git-scm.com/docs/gittutorial>.

3. Instale asciinema y regístrese <https://asciinema.org>.

```
$ sudo pacman -S asciinema  
# suba las soluciones a, por ejemplo  
# https://asciinema.org/a/L7v4biUA6A7m3UNsE28UZy8WK
```

4. Rehacer los comandos de la sesión pasada.

# Ejemplos

```
git config
git config --list
git config --global user.name "Name"
git config --list
cd
whoami
mkdir -p Git
ls
pwd
# Primero ubicarse en el home del usuario /home/usuario
echo $USER
whoami
cd
cd /
pwd
ls
cd
pwd
mkdir -p Git/Hub
ls
ls Git/
cd Git/Hub/
ls
git init
git status
git config --global init.defaultBranch main
ls
git status
cd ..
ls
mkdir usuario
```

# Ejemplos

```
cd usuario
git init
git status
cd ..
ls
mkdir ramas
cd ramas/
pwd
cd
ls
cd Git/
ls
cd Hub/
git status
cd ..
ls
cd usuario
git status
cd ..
cd ramas/
git status
git init -b rama
git status
#origin/rama
#origin/master
#origin/main
sudo pacman -Sy
pacman -Ss github-cli
pacman -Si github-cli
git clone https://github.com/numpy/numpy.git
ls
```

# Ejemplos

```
cd numpy
git branch
git branch --list
git branch -a
pwd
cd
ls
pwd
git status
git branch -a
ls
mkdir src
touch README.md
git status
git add README.md
git add src
git status
git commit
ls -l
# r w x 4 2 1
# 6 4 4
ls -l src/
ls -lh src/
git status
git commit
git add .gitignore
git commit -m "Add .gitignore"
git status
git add src/vectores.cc
git commit -m "Create vectores.cc"
git status
```



```
git log
HASH_COMMIT=f5a1124eba1bf89e2baa65fc483b4b76146ae9be
git show $HASH_COMMIT^!
git status
git add src/vectores.cc
git commit -m "Change to float from integers"
git log
```

```
#include <iostream>

int main(int argc, char **argv)
{
    std::cout << "¡Hola mundo!" << std::endl;

    return 0;
}
```

```
#include <vector>
#include <iostream>

int main(int argc, char **argv)
{
    auto v = std::vector<float>{1.0, 2.0, 3.0, 4.0, 5.0, 6.0, 7.0};

    for (const auto &e : v)
    {
        std::cout << e << std::endl;
    }

    return 0;
}
```

```
#include <string>
#include <iostream>

int main(int argc, char **argv)
{

    std::string cadena = "hola cadena";
    std::cout << cadena << std::endl;

    return 0;
}
```