

Control de versiones: Git

- Control de versiones.
- Git: Introducción
- Estado de un archivo
- Parche
- Flujo de trabajo
- Comandos útiles
- Configuración de Git



Control de versiones

- Controlar los cambios que se producen sobre un elemento.
- Guarda un historial de las modificaciones que se han producido.

- Versión: Estado específico de un elemento.
- Repositorio: Lugar donde están almacenados los elementos.
- Elementos: ficheros (de texto, binarios...)

Control de versiones

- Controlar los cambios que se producen sobre un elemento.
- Guarda un historial de las modificaciones que se han producido.

- Versión: Estado específico de un elemento.
- Repositorio: Lugar donde están almacenados los elementos.
- Elementos: ficheros (de texto, binarios...)



Control de versiones

- Controlar los cambios que se producen sobre un elemento.
- Guarda un historial de las modificaciones que se han producido.

- Versión: Estado específico de un elemento.
- Repositorio: Lugar donde están almacenados los elementos.
- Elementos: ficheros (de texto, binarios...)



Git: Introducción

- Software de control de versiones distribuida
- Diseñado por Linus Torvalds
- Eficiencia y confiabilidad en el mantenimiento de versiones de aplicaciones

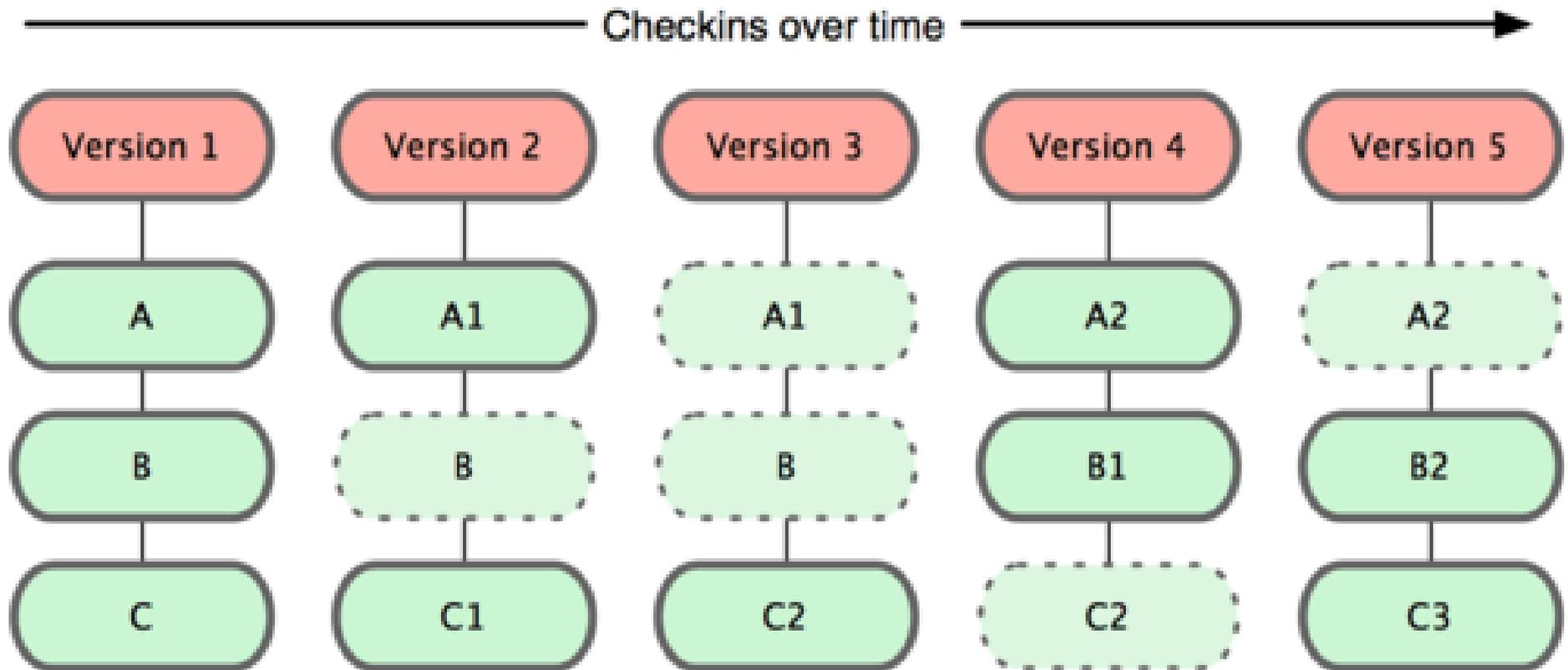


Git: Entendiendo Git

- Git modela sus datos como un conjunto de instantáneas de un mini sistema de archivos.
- Cada vez que confirmas un cambio, git hace una foto del aspecto de todos tus archivos en ese momento y guarda una referencia a esa instantánea.
- Se trabaja en local y luego se suben los cambios a los repositorios remotos.
- Integridad de los archivos. Checksum de los archivos.
- Imposible cambiar los contenidos de cualquier archivo o directorio sin que Git lo sepa.

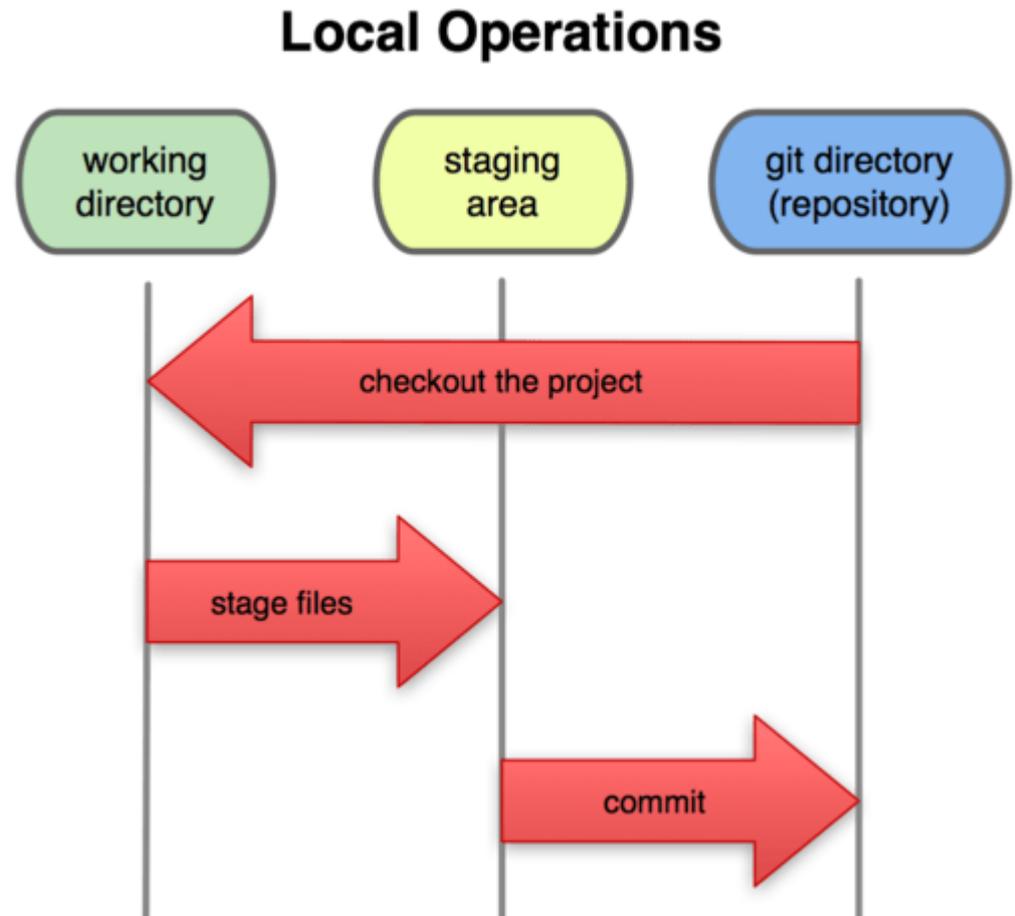


Git: Instantaneas



Git: Estados de un archivo/areas

- Modificado (modified):
Fichero con modificaciones.
- Preparado (staged):
Fichero con modificaciones marcado para ser commiteado.
- Confirmado (committed):
Fichero almacenado de manera segura en la base de datos local.



Git: parche

- Conjunto de modificaciones que se realizan sobre un archivo con el objetivo de corregir errores, agregarle funcionalidad, actualizarlo.



Git: Parche

```
From 9ab6adb389497e0d96881311157b02b8e6b40158 Mon Sep 17 00:00:00 2001
From: Alvaro Neira <alvaro@soleta.eu>
Date: Wed, 25 Feb 2015 12:20:39 +0100
Subject: [PATCH] =?UTF-8?q?Git:=20ejemplo1:=20A=C3=B1ado=20espacios=20entre=20?=
=?UTF-8?q?simbolos=20'=3D'=20y=20'+'?=
MIME-Version: 1.0
Content-Type: text/plain; charset=UTF-8
Content-Transfer-Encoding: 8bit
```

Siguiendo el código de estilo recomendado por el Kernel de linux, se añade un espacio entre los operadores aritméticos.

Signed-off-by: Alvaro Neira <alvaro@soleta.eu>

fichero1.txt | 4 ++--

1 file changed, 2 insertions(+), 2 deletions(-)

```
diff --git a/fichero1.txt b/fichero1.txt
```

```
index ff6ff4c..dc45346 100644
```

```
--- a/fichero1.txt
```

```
+++ b/fichero1.txt
```

```
@@ -1,5 +1,5 @@
```

Fichero de ejemplo para el curso de Programación en C moderno

```
-2+2=4
```

```
+2 + 2 = 4
```

```
-4-2=2
```

```
+4 - 2 = 2
```

```
--
```

1.7.10.4



Git: Flujo de trabajo

- Modificas una serie de archivos en tu directorio de trabajo.
- Preparas los archivos, añadiéndolos a tu área de preparación. (Marcas los fichero para que sean commiteados)
- Confirmas los cambios: toma los archivos del área de preparación y los almacena de manera permanente en tu directorio de Git.



Git: comandos básicos:

- `git status` => estado del repositorio: muestra los cambios preparados y no preparados
- `git diff` => Muestra las modificaciones en el area de trabajo: ver lo que has modificado pero aún no has preparado.
- `git diff --cached` => Muestra los archivos marcados como preparado y que irán en tu próxima confirmación
- `git add file` => añadir un fichero al "area de preparados"
- `git commit -s` => Confirmar los cambios: Hacer un commit con los cambios marcados. (-s añade la firma del autor)



Git: comandos básicos:

- `git log` => Historial de confirmaciones: con descripciones y asuntos
- `git log --oneline` => Historial de confirmaciones: asuntos.
- `git log --oneline -6` => Historial de confirmaciones de los últimos 6 commit: asuntos.
- `git show` => Información del último commit
- `git show [id-commit]` => Información del commit indicado



Git: Comandos ¿Y si me equivoco?

- `git commit --amend` => Añadir los cambios al último commit.
- `git checkout -- file` => Elimina los cambios realizados sobre ese fichero.
- `git reset --soft HEAD^` => Eliminar el último commit conservando las modificaciones en el area de trabado.
- `git reset --hard HEAD^` => Eliminar el último commit y las modificaciones realizadas en los ficheros involucrados



Git: Crear un parche

- `git format-patch HEAD^ => Formar un parche`
- `git format-patch --subject-prefix="nft PATCH" -o /tmp/ --to="netfilter-devel@vger.kernel.org" HEAD --cover-letter -n`
- `git format-patch`
 - `--subject-prefix="nft PATCH"`
 - `-o /tmp/`
 - `--to="netfilter-devel@vger.kernel.org"`
 - `HEAD` (HEAD~2)
 - `--cover-letter -n`



Git: Configuración

Fichero ~/.gitconfig

[user]

name = Alvaro Neira
email = anarey@gmail.com

name = Alvaro Neira
email = alvaro@soleta.eu

[core]

editor = vim

[color "diff"]

whitespace = red reverse
meta = blue black bold

[alias]

br = branch
st = status
ci = commit
co = checkout

[color]

ui = auto
diff = auto

[sendemail]

smtptencryption = tls
smtpserver = smtp.gmail.com
smtpuser = anarey@gmail.com
smtpserverport = 587

#[sendemail]

smtptencryption = tls
smtpserver = correo.soleta.eu
smtpuser = alvaro@soleta.eu
smtpserverport = 25



Git: config

- `logi = log --graph --pretty=format:'%h - %an, %ar (): %s'`
- `lg1 = log --color --graph --pretty=format:'%Cred%h%Creset-%C(yellow)%d%Creset %s %Cgreen(%cr) %C(bold blue)<%an>%Creset'--abbrev-commit`
- `lg = log --graph --pretty=format:'%Cred%h%Creset-%C(yellow)%d%Creset %s %Cgreen(%cr)%Creset' --abbrev-commit --date=relative`
- `last = log -1 HEAD`



Git: ejemplo uso de alias

```
ana@tux230 ~/devel/libmnl
(master) $ git logi -5
* 72aec11 - Ken-ichirou MATSUZAWA, 4 months ago (): doc: minor fix
* e374664 - Ken-ichirou MATSUZAWA, 5 months ago (): socket: creating a struct mnl_socket from a pre-existing socket
* 2c458b2 - Pablo Neira Ayuso, 5 months ago (): socket: calloc expects struct size as second parameter
* f94b2c6 - Pablo Neira Ayuso, 9 months ago (): include: cache copy of include/linux/socket.h
* 090a842 - Ken-ichirou MATSUZAWA, 1 year, 3 months ago (): examples: use mnl_socket_setsockopt

ana@tux230 ~/devel/libmnl
(master) $ git lg1 -5
* 72aec11- (HEAD, origin/master, origin/HEAD, master) doc: minor fix (4 months ago) <Ken-ichirou MATSUZAWA>--abbrev-c
* e374664- socket: creating a struct mnl_socket from a pre-existing socket (5 months ago) <Ken-ichirou MATSUZAWA>--ab
* 2c458b2- socket: calloc expects struct size as second parameter (5 months ago) <Pablo Neira Ayuso>--abbrev-commit
* f94b2c6- include: cache copy of include/linux/socket.h (9 months ago) <Pablo Neira Ayuso>--abbrev-commit
* 090a842- examples: use mnl_socket_setsockopt (1 year, 3 months ago) <Ken-ichirou MATSUZAWA>--abbrev-commit

ana@tux230 ~/devel/libmnl
(master) $ git lg -5
* 72aec11- (HEAD, origin/master, origin/HEAD, master) doc: minor fix (4 months ago)
* e374664- socket: creating a struct mnl_socket from a pre-existing socket (5 months ago)
* 2c458b2- socket: calloc expects struct size as second parameter (5 months ago)
* f94b2c6- include: cache copy of include/linux/socket.h (9 months ago)
* 090a842- examples: use mnl_socket_setsockopt (1 year, 3 months ago)

ana@tux230 ~/devel/libmnl
(master) $ git last
commit 72aec11703c7fda93af77cb6356f9692f18f9e9b
Author: Ken-ichirou MATSUZAWA <chamaken@gmail.com>
Date: Fri Oct 24 14:39:28 2014 +0900

    doc: minor fix

    mnl_attr_ok(): fix return value type
    mnl_attr_put_u8(): remove unused param - len
    mnl_attr_put_u8_check(): remove unused param - len
    mnl_nlmsg_ok(): fix return value type
    mnl_nlmsg_batch_stop(): not return batch size, but release it

Signed-off-by: Ken-ichirou MATSUZAWA <chamas@h4.dion.ne.jp>
Signed-off-by: Florian Westphal <fw@strlen.de>
```



Git: Comandos no tan básicos

- `git reset --hard @{upstream}`
- `git rebase -i bed27b6^`
- `git rebase --continue`

- `git pull --rebase`
- `git push`



Git: Flujo de trabajo

- `cd ejemplo1/` <= Directorio del repositorio
- `vi fichero1.txt` <= Modificamos el fichero
- `git add fichero1.txt`
- `git commit -s`

[master (root-commit) 3d07b07] Git: Ejemplo: hola mundo en git.

1 file changed, 5 insertions(+)

create mode 100644 fichero1.txt

- `git status`

On branch master

nothing to commit (working directory clean)



Git: Flujo de trabajo

- Modifico un archivo y hago un nuevo commit
- vi ejemplo1.txt
- git status

```
# On branch master
# Changes not staged for commit:
#   (use "git add <file>..." to update what will be committed)
#   (use "git checkout -- <file>..." to discard changes in working directory)
#
#   modified:   fichero1.txt
#
no changes added to commit (use "git add" and/or "git commit -a")
```

- git add fichero1.txt
- git commit -s
- git format-patch HEAD^

0001-Git-ejemplo1-A-ado-espacios-entre-simbolos-y.patch



Github

- Introducción
- Crear/configurar una cuenta de usuario
- Creación y uso de un repositorio.



Github

- Es un servicio de alojamiento de proyectos gestionado a través del sistema de control de versiones Git. 2010.
- Almacenamiento público (y privado con cuentas de pago)
- Comunidad de desarrollo alrededor de ella.
- Forma de almacenar vuestros proyectos personales y que estén siempre visible.
- Es tu CV demostrable. <https://github.com/>



Github: Crear/Configurar una cuenta.

- Accede a la pagina: <https://github.com/join>
- [En pc]:
 - `git config --global user.name "YOUR NAME"`
 - `git config --global user.email "YOUR EMAIL ADDRESS"`



GitHub: Nuevo repositorio

- [web] Creamos un nuevo repositorio.

Search GitHub

Explore Gist Blog Help

anarey

Owner anarey / **Repository name** AprendiendoC

Great repository names are short and memorable. Need inspiration? How about **petulant-ironman**.

Description (optional)
Repositorio para almacenar los ejercicios correspondientes al curso de C "Programación en C moderno"

Public
Anyone can see this repository. You choose who can commit.

Private
You choose who can see and commit to this repository.

Initialize this repository with a README
This will let you immediately clone the repository to your computer. Skip this step if you're importing an existing repository.

Add .gitignore: **C** | Add a license: **GNU LGPL v3.0**

Create repository

© 2015 GitHub, Inc. [Terms](#) [Privacy](#) [Security](#) [Contact](#) [Status](#) [API](#) [Training](#) [Shop](#) [Blog](#) [About](#)



Github: Nuevo Repositorio

anarey / **AprendiendoC**

Unwatch 1 Star 0 Fork 0

Repositorio para almacenar los ejercicios correspondientes al curso de C "Programación en C moderno" — Edit

1 commit 1 branch 0 releases 1 contributor

branch: master **AprendiendoC** / +

Initial commit		
anarey authored a minute ago		latest commit 7a74247bd2
.gitignore	Initial commit	a minute ago
LICENSE	Initial commit	a minute ago

Help people interested in this repository understand your project by adding a README! [Add a README](#)

- Code**
- [Issues](#) 0
- [Pull Requests](#) 0
- [Wiki](#)
- [Pulse](#)
- [Graphs](#)
- [Settings](#)

SSH clone URL
git@github.com:anarey,

You can clone with [HTTPS](#), [SSH](#), or [Subversion](#).

[Download ZIP](#)



Github: Clonando el repositorio

- `git clone https://github.com/anarey/AprendiendoC.git`

Cloning into 'AprendiendoC'...

remote: Counting objects: 4, done.

remote: Compressing objects: 100% (4/4), done.

remote: Total 4 (delta 0), reused 0 (delta 0), pack-reused 0

Unpacking objects: 100% (4/4), done.



Github: Primer commit en local

- `cd AprendiendoC`
- `touch aaa` => Creo un archivo vacio.
- `git add aaa`
- `git commit -s -m "Añado arhivo vacio aaa"`

```
[master 6bf8c8f] Añado arhivo vacio aaa
0 files changed
create mode 100644 aaa
```



Github: Información

git log

```
commit 6bf8c8f9d6b94e7c4e2be7498d4c137ca4b7cbf2
Author: Alvaro Neira <alvaroneay@gmail.com>
Date:   Wed Feb 25 13:40:36 2015 +0100
```

Añado arhivo vacio aaa

Signed-off-by: Alvaro Neira <alvaroneay@gmail.com>

```
commit 7a74247bd23ff104129b290e97cd163bf0cffc14
Author: Alvaro Neira <alvaroneay@gmail.com>
Date:   Wed Feb 25 13:21:11 2015 +0100
```

Initial commit

git log --oneline

```
6bf8c8f Añado arhivo vacio aaa
7a74247 Initial commit
```



Github: Subir los cambios a repositorio remoto

- git push

Username for 'https://github.com': alvneiayu

Password for 'https://anarey@github.com': *****

Counting objects: 4, done.

Delta compression using up to 4 threads.

Compressing objects: 100% (2/2), done.

Writing objects: 100% (3/3), 332 bytes, done.

Total 3 (delta 0), reused 0 (delta 0)

To https://github.com/anarey/AprendiendoC.git

7a74247..6bf8c8f master -> master



Github: Cambios en el repositorio remoto

The screenshot shows the GitHub interface for a repository named 'anarey / AprendiendoC'. At the top, there is a search bar and navigation links for 'Explore', 'Gist', 'Blog', and 'Help'. The repository name is displayed in blue, along with 'Unwatch' (1), 'Star' (0), and 'Fork' (0) buttons. Below this, a description reads: 'Repositorio para almacenar los ejercicios correspondientes al curso de C "Programación en C moderno" — Edit'. A summary bar shows '2 commits', '1 branch', '0 releases', and '1 contributor'. The main content area shows a commit by 'anarey' from 2 minutes ago with the latest commit hash '6bf8c8f9d6'. A table lists files: '.gitignore' (Initial commit, 22 minutes ago), 'LICENSE' (Initial commit, 22 minutes ago), and 'aaa' (Añadó archivo vacío aaa, 2 minutes ago). A green button 'Add a README' is visible. On the right, a sidebar contains links for 'Code', 'Issues' (0), 'Pull Requests' (0), 'Wiki', 'Pulse', 'Graphs', and 'Settings'. At the bottom of the sidebar, the 'HTTPS clone URL' is 'https://github.com/ani...', and a 'Download ZIP' button is present.

This repository

anarey / **AprendiendoC**

Repositorio para almacenar los ejercicios correspondientes al curso de C "Programación en C moderno" — Edit

2 commits 1 branch 0 releases 1 contributor

branch: master **AprendiendoC** / +

Añadó archivo vacío aaa

anarey authored 2 minutes ago latest commit 6bf8c8f9d6

.gitignore	Initial commit	22 minutes ago
LICENSE	Initial commit	22 minutes ago
aaa	Añadó archivo vacío aaa	2 minutes ago

Help people interested in this repository understand your project by adding a README! [Add a README](#)

Code

- Issues 0
- Pull Requests 0
- Wiki
- Pulse
- Graphs
- Settings

HTTPS clone URL

`https://github.com/ani...`

You can clone with [HTTPS](#), [SSH](#), or [Subversion](#).

[Download ZIP](#)



Vim: Índice

- Vim: Editor de texto.
- Vim: Configuración.
- Vim: Comandos más usados.



Vim: Editor de texto

- Editor de texto muy usado en entornos Linux (Como emacs)
- Es posible controlarlo sólo con un teclado.



Vim: Configuración

- Fichero: `/home/ana/.vimrc`

```
syntax on
```

```
set number
```

```
set pastetoggle=<F2>
```

```
highlight ExtraWhiteSpace ctermbg=green guibg=green
```

```
match ExtraWhiteSpace /\s\+$/
```

```
autocmd BufWinEnter * match ExtraWhiteSpace /\s\+$/
```

```
autocmd InsertEnter * match ExtraWhiteSpace /\s\+\%#\@<!$/
```

```
autocmd InsertLeave * match ExtraWhiteSpace /\s\+$/
```

```
autocmd BufWinLeave * call clearmatches()
```

```
:set colorcolumn=80
```

```
:set hlsearch
```

```
set background=dark
```



Vim: comandos más usados

- :wq
- i: insertar
- v: modo visual
- u deshacer el último cambio.
- d borrar una línea entera

- # copiar y pegar
- ctrl + v: seleccionamos una línea (y con el curso el total del bloque)
- ctrl + x: Cortar el texto seleccionado.
- ctrl + y: Copiar el texto seleccionado.
- Ctrl + p: Pegar el texto seleccionado.
- /palabra: Busca la palabra
- %s/a/A/cg: sustituir las 'a' por 'A' en todo el texto pidiendo confirmación.



Bibliografía

- <http://git-scm.com/>
- <https://www.kernel.org/doc/Documentation/CodingStyle>
- <https://training.github.com/kit/downloads/es/github-git-cheat-sheet.pdf>

