

Agenda

sábado, 9 de septiembre de 2017 13:07



ÍNDICE

Duración: 16 horas

Objetivo: Al finalizar la acción formativa, los asistentes:

- Conocerán los elementos avanzados de la arquitectura de GIT
- Podrán aplicar determinadas técnicas a la gestión y organización de proyectos GIT

Requisitos:

Familiaridad con los repositorios de código fuente.
Idealmente, conocimientos básicos de GIT.

- Propósito y alcance
- Arquitectura
- Componentes fundamentales
- Operaciones CRUD
- Operaciones de organización de código
- Gestión de ramas y etiquetas
- Resolución de conflictos
- Integración con npm, Maven...
- Configuración de GIT
- Introducción a GIT en el lado servidor
- Programas cliente para operar con GIT

GIT AVANZADO

ESCUELA:
TECNOLOGÍA



 Indra Open University

- NOMBRE APELLIDO PROFESOR
Alejandro Cerezo

- VER PERFIL COMPLETO:



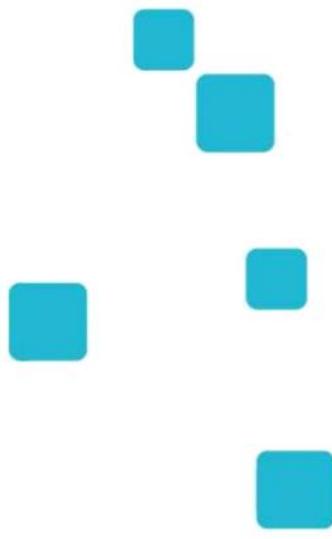
<https://www.linkedin.com/company/icono-training-consulting/>
<https://www.linkedin.com/in/alejandrocerezo/>

- CONTACTO



training@iconotc.com
alce65@hotmail.es

Título del documento | 3



Avda. de Bruselas 35
28108 Alcobendas,
Madrid España
T +34 91 480 50 00
F +34 91 480 50 80
www.indracompany.com



[Icono Training Consulting: iconotc.com](http://iconotc.com)

ACCIÓN FORMATIVA: GIT AVANZADO

Duración: 16 horas

Requisitos Asistentes: Familiaridad con los repositorios de código fuente. Idealmente, conocimientos básicos de GIT.

Objetivos:

Al finalizar la acción formativa, los asistentes:

- Conocerán los elementos avanzados de la arquitectura de GIT
- Podrán aplicar determinadas técnicas a la gestión y organización de proyectos GIT

Contenido:

- ❖ Propósito y alcance
- ❖ Arquitectura
- ❖ Componentes fundamentales
- ❖ Operaciones CRUD
- ❖ Operaciones de organización de código
- ❖ Gestión de ramas y etiquetas
- ❖ Resolución de conflictos
- ❖ Integración con Maven
- ❖ Configuración de GIT
- ❖ Introducción a GIT en el lado servidor
- ❖ Programas cliente para operar con GIT

Desarrollo de la Agenda

Lunes, 8 de enero de 2018 13:06

- Propósito y alcance
- Configuración de GIT
- Programas cliente para operar con GIT
- Arquitectura
- Componentes fundamentales
- Introducción a GIT en el lado servidor
- Integración con npm, Maven...
- Operaciones CRUD
- Operaciones de organización de código
- Gestión de ramas y etiquetas
- Resolución de conflictos
- Introducción: Propósito y alcance
 - Git: Control de versiones distribuido
 - Repositorios en la nube: GitHub, Bitbucket, GitLab
- Instalación y Configuración de GIT
 - Git. Terminales en Windows
 - Configuraciones
 - Clientes gráficos para operar con GIT
 - SourceTree
 - Cliente de GitHub
 - Integración en IDEs y Editores. Visual Studio Code
- Arquitectura y Componentes fundamentales
 - Áreas: Stashing - Working - Steaging - Repository
 - Creación de un repositorio, La carpeta .git
 - Punteros: HEAD
 - Commits
 - Readme.md y gitignore
- Introducción a GIT en el lado servidor
 - Repositorios bare
 - GitHub
 - GitLab
 - Bitbucket
- Integración con npm, Maven...
- Operaciones CRUD sobre los commits
 - C -> git add / git commit
 - R -> git status / git log
 - U -> git commit --amend
 - D -> git reset
- Operaciones de organización de código
 - Stashing
 - Reset
 - Revert
 - Repositorios remotos
- Gestión de ramas y etiquetas
 - Ramas
 - Merge
 - Resolución de conflictos
 - Tags
 - Rebase
- Workflow.
-

Introducción

lunes, 8 de enero de 2018 13:31

Introducción: Propósito y alcance

- Git: Control de versiones distribuido
- Repositorios en la nube:
 - GitHub - <https://github.com/>
 - Bitbucket <https://bitbucket.org/>
 - GitLab <https://about.gitlab.com/>

Sistemas de control de versiones (SCV)

martes, 25 de abril de 2017 0:01

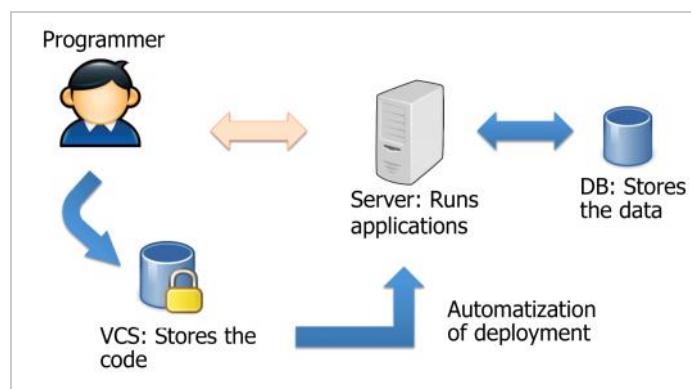
Los **sistemas de control de versiones** (SCV) son una herramienta esencial para manejar proyectos de software; lo que ha hecho de ellos herramientas de uso habitual en el desarrollo profesional de software desde hace décadas.

Proporcionan una serie de funcionalidades claves para el desarrollo de proyectos como es

- el control de cambios en el código,
- la reversibilidad de dichos cambios,
- la posibilidad de colaborar en el desarrollo del código.

Además, los SCV permiten tener en paralelo varias **versiones o ramas** del proyecto. Las ramas se utilizan para desarrollar funcionalidades aisladas de los cambios en otras partes del proyecto que posteriormente pueden integrarse en la rama principal.

Version Control System



Tipos de SVC

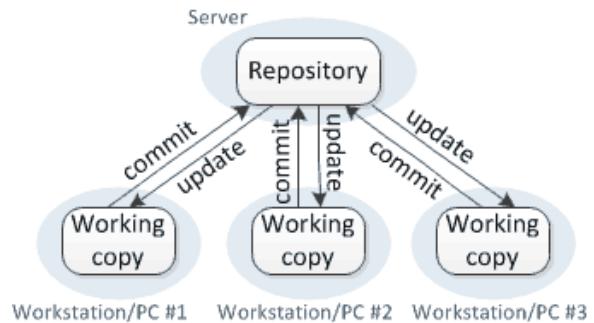
Lunes, 8 de enero de 2018 14:00

Los SCV se pueden clasificar en dos grandes familias:

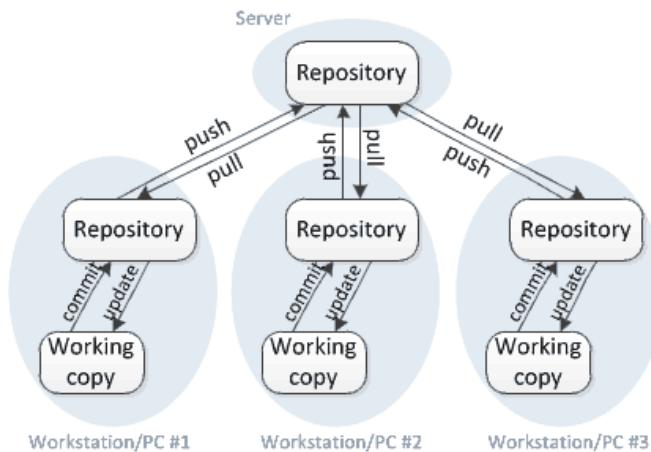
- SCV centralizados y
- SCV distribuidos.

Los SCV **centralizados** como CVS y Subversion (SVN) son sistemas cliente-servidor donde hay un repositorio canónico en el servidor que contiene toda la información de los cambios mientras que los clientes solo tienen copias de trabajo.

Centralized version control



Distributed version control



En sistemas **distribuidos** como Mercurial y Git no existe el concepto de repositorio canónico por lo que cada cliente ha de tener una copia completa del repositorio.

Desde 2010, la tendencia es utilizar cada vez más SCV distribuidos, en particular Git

SVC habituales

lunes, 8 de enero de 2018 14:02

- | | |
|---------------|---|
| Centralizados | <ul style="list-style-type: none">• RCS (el original, de Pardue University luego GNU)• CVS• Microsoft Visual SourceSafe / Team Foundation Version Control (TFVC)• Subversion (SVN) https://subversion.apache.org/ |
| Distribuidos | <ul style="list-style-type: none">• BitKeeper http://www.bitkeeper.org/• Git https://git-scm.com/• Mercurial (hg) https://www.mercurial-scm.org/• Bazaar (bzr) http://bazaar.canonical.com/en/ |

“CVS and SVN are remote backups that you use to save your changes.
Git is an editor that you use to write your code’s biography.”

— isaacs (@izs) May 14, 2010

Cronología

1972	<i>Source Code Control System (SCCS)</i>	<i>Closed source</i> , libre con Unix
1982	<i>Revision Control System (RCS)</i>	<i>Cross platform</i> . <i>Open Source</i> . Más características y más rápido Capaz de trabajar solamente con un fichero cada vez
1986-1990	<i>Concurrent Versions System (CVS)</i> :	<i>Open Source</i> . Usuarios concurrentes pueden trabajar simultáneamente con un mismo fichero
2000	<i>Apache Subversion</i> .	Instantáneas (<i>Snapshot</i>) del directorio, y no solo de un fichero.. <i>Commits</i> transaccionales .
2000	<i>BitKeeper SCM</i> .	<i>Closed source, proprietary</i> . Sistema distribuido (<i>Distributed version control</i>) "Community version" libre. Usado para el código fuente del <i>kernel</i> de Linux entre 2002-2005
2005, abril		La "community version" deja de ser de uso libre
2005	<i>Git</i>	Desarrollado por el equipo responsable del <i>kernel</i> de Linux, encabezado por Linus Torvalds. Intenta ser una mejora de <i>BitKeeper</i> .

Git. El SVC distribuido más utilizado

martes, 25 de abril de 2017 0:09

Git es un SCV distribuido diseñado para la gestión eficiente de flujos de trabajo distribuido no lineales. Git fue diseñado y desarrollado inicialmente por Linus Torvalds en 2005 para el desarrollo del kernel de Linux.

Git es un software de **control de versiones** diseñado por Linux Torvalds, pensando en la **eficiencia** y la **confiabilidad** del mantenimiento de versiones y aplicaciones cuando éstas tienen un gran número de archivos de **código fuente**.

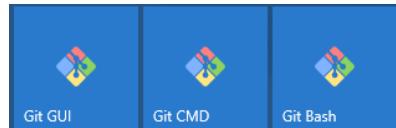
- Control de versiones
- Código fuente

automatizar la gestión de los cambios que se producen en archivos de todo tipo, tanto de código como de cualquier otro tipo, e.g. imágenes....

La licencia de Git es libre y hay distribuciones oficiales para los sistemas operativos

- Mac OS X,
- Windows,
- Linux y
- Solaris.

La distribución de Git incluye herramientas de **línea de comando** y de **escritorio**.



Además, hay disponibles herramientas proporcionadas por terceros que permiten una mayor integración con el escritorio o con entornos de desarrollo.

Orígenes

lunes, 8 de enero de 2018 14:15

Git fue diseñado y desarrollado inicialmente en 2005 con el objetivo concreto de dar soporte al desarrollo del kernel de Linux liderado por Linus Torvalds.

La experiencia del equipo en la gestión de la integración de las diferentes aportaciones en un proyecto distribuido de esta magnitud determinó los objetivos del proyecto

- Rapidez
- Simplicidad de uso
- Multiplicidad de versiones (Ramas)
- Distribuido:
capaz de trabajar sin conexiones
- Preparado para grandes proyectos

Estos objetivos se reflejaron en las siguientes decisiones de implementación:

Versiones no incrementales. Git almacena cada cambio como una instantánea de todos los archivos del proyecto. Para ser eficiente, si el archivo no ha sido modificado, sólo se almacena un enlace al archivo idéntico previamente almacenado. Los SCV anteriores a Git habitualmente almacenaban solo una versión base y las modificaciones hechas en cada cambio por archivo.

Trabajo fuera de línea. Por ser un sistema distribuido, cada repositorio de Git es un repositorio completo capaz de funcionar sin acceso a la red o al resto de los repositorios distribuidos gracias a que contiene una copia local de la historia completa del desarrollo del proyecto. Los cambios en la historia pueden copiarse de un repositorio a otro como nuevas ramas de desarrollo y se pueden copiar de la misma manera que una rama de desarrollo local.



- Cada operación se realiza en el repositorio local.
- No necesita conexión con un servidor.
- Se puede trabajar normalmente sin conexión (*offline*)
- Es más rápido que los repositorios centralizados

Autenticación criptográfica de la historia. El identificador de un cambio se computa utilizando un algoritmo criptográfico que utiliza como entrada el cambio y la historia completa de cambios. Esto permite que cualquier cambio de la información durante la transmisión o en sistema de archivos sea detectado por Git.

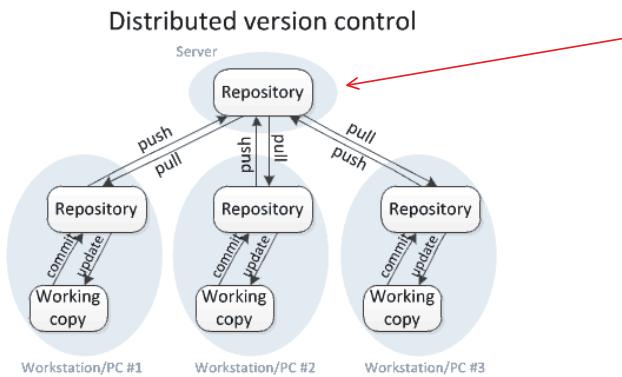


- Todo está identificado por secuencias *hash*.
- Es imposible cambiar el contenido de cualquier archivo o directorio sin que Git lo sepa.
- No se puede perder información ni dañar el archivo sin que Git pueda detectarlo.
- Ejemplo de secuencia de comprobación (*checksum*)

24b9da6552252987aa493b52f8696cd6d3b00373

Repositorios remotos

lunes, 8 de enero de 2018 15:57



Uno de los equipos actúa como repositorio de referencia, pero en realidad es solo una variación similar a cualquier otro

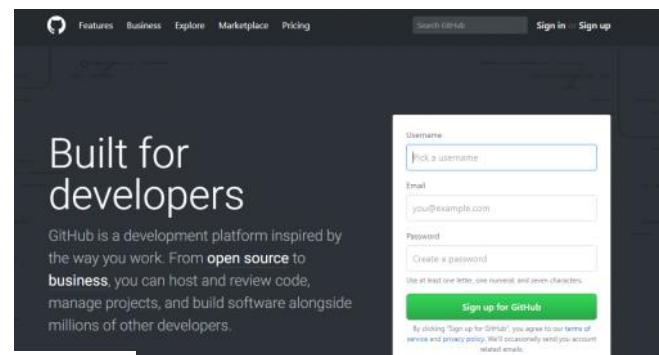
En la red de los usuarios

Existen versiones servidor de Git como

- *gitolite* <http://gitolite.com/gitolite/index.html>
 - *gitosis* <https://git-scm.com/book/es/v1/Git-en-un-servidor-Gitosis>
- o incluyendo una completa interfaz Web,
- *GitLab*, para Linux <https://about.gitlab.com/equipos>

En Internet

GitHub - <https://github.com/>



ATLASSIAN Bitbucket Features Integrations Enterprise Pricing

Log in Get started

For the code that delivers.

What will your code do?

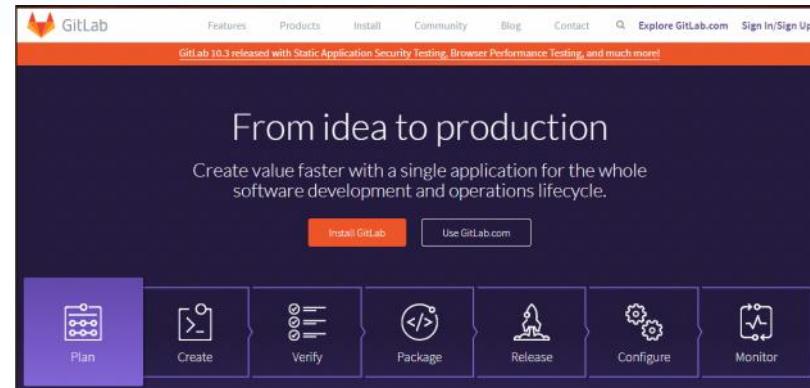
Get started for free

Or host it yourself with Bitbucket Server



Bitbucket <https://bitbucket.org/>

GitLab <https://about.gitlab.com/>



Información Extra

domingo, 29 de octubre de 2017 20:59

The screenshot shows a landing page for a Git guide. At the top, it says "git - la guía sencilla" and provides a link: <http://rogerdudler.github.io/git-guide/index.es.html>. Below the title, there's a large downward arrow pointing towards a summary section. The summary section contains several command snippets, such as "git add .", "git commit -m 'My message'", and "git log --oneline". To the right of the arrow, there's a sidebar with the text "want a simple but powerful git client for your mac?". The footer of the page includes a "download the cheat sheet now, it's free!" button and links to various languages.

Una referencia visual de Git

Otros Idiomas: [Deutsch](#) [English](#) [Français](#) [Italiano](#) [日本語](#) [한국어](#) [Polski](#) [Português](#) [Русский](#) [Slovenčina](#) [Tiếng Việt](#) [简体中文](#) [正體中文](#)

Si las imágenes no funcionan, puedes intentar la [versión zin-SVG](#) de esta página.

Esta página da una referencia breve y visual para los comandos más comunes en git. Una vez que conozcas un poco sobre la forma de trabajo de git, este sitio puede fortalecer tu entendimiento. Si estás interesado en cómo se creó este sitio, mira mi [repositorio de GitHub](#).

Contenidos

- [1. Uso Básico](#)
- [2. Convenciones](#)
- [3. Comandos en Detalle](#)
 - [a. Diff](#)
 - [b. Commit](#)
 - [c. Checkout](#)
 - [d. Committeando con un HEAD Detachado](#)
 - [e. Reset](#)
 - [f. Merge](#)
 - [g. Cherry Pick](#)
 - [h. Rebase](#)
- [4. Notas Técnicas](#)

<http://marklodato.github.io/visual-git-guide/index-en.html>

Otro resumen, sin un diseño tan cuidado

This screenshot shows a complex, multi-section reference page for Git commands. It's organized into several columns: "Resource", "Create Git", "Local Changes", "History", "Merge/Rebase", "Branching/Tagging", and "Useful Commands". Each column lists specific Git commands and their descriptions. For example, under "Resource", it has sections for "Online", "Download", and "Related". Under "Create Git", it covers "From existing directory", "From other repository", and "Merge/Rebase". Under "Local Changes", it includes "Changed in working directory", "Tracked file changes", and "Remove file". Under "History", it lists "Show all commits", "Short Format", and "Branches". The "Merge/Rebase" section covers "Merge branch into current", "Rebase", "Abort rebase", and "Merge tool to solve conflicts". The "Branching/Tagging" section includes "List branches", "Switch to branch", and "Delete branch". The "Useful Commands" section contains many common Git commands like "git status", "git add", "git commit", "git log", and "git merge".

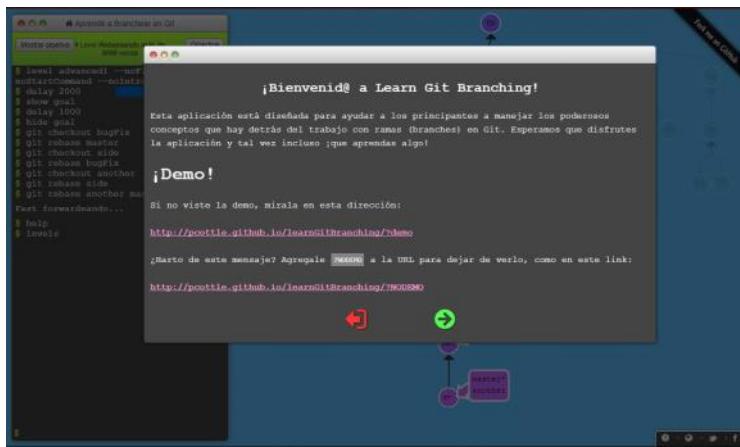
<http://overapi.com/git>

Referencia a los comandos de Git

This screenshot shows a "GIT CHEATSHEET" from NPD Software. It features a diagram illustrating the Git workspace structure across four main areas: "STASH", "WORKSPACE", "INDEX", "REPOSITORIO LOCAL", and "REPOSITORIO REMOTO". The "STASH" area contains commands like "stash", "stash pop", and "stash drop". The "WORKSPACE" area contains "status", "diff", "add", "rm", "mv", "commit", "checkout", "reset", and "clean". The "INDEX" area contains "add", "rm", "mv", "commit", "checkout", "reset", and "clean". The "REPOSITORIO LOCAL" area contains "clone", "pull", "push", "fetch", "merge", "rebase", "cherry-pick", and "revert". The "REPOSITORIO REMOTO" area contains "push", "pull", "fetch", "fetch --prune", "push --tags", and "push --all". A note at the bottom left states: "repositorio remoto: Version(es) del proyecto que están alojadas en Internet o una red, asegurando que todos los cambios estén disponibles para otros desarrolladores. Pueden ser 'origin'. Ramas típicas aquí son: master, shared-feature-x, release-y, etc.". A copyright notice at the bottom right reads: "© Andrew Petillo 2009 - All Rights Reserved".

<https://ndpssoftware.com/git-cheatsheet.html#loc=workspace>

Referencia interactiva con los comandados de Git superpuestos en la estructura del repositorio



<https://learngitbranching.js.org/?demo>

Animación del proceso de creación de ramas en Git

<https://try.github.io/levels/1/challenges/1>

Tutorial interactivo, en el que OCTOCAT nos enseña a manejar Git

Libros

lunes, 8 de enero de 2018 18:30

<https://git-scm.com/book/es/v2>

<https://git-scm.com/book/en/v2>

Original en inglés

Download Ebook



Book

The entire Pro Git book, written by Scott Chacon and Ben Straub and published by Apress, is available here. All content is licensed under the Creative Commons Attribution Non Commercial Share Alike 3.0 license. Print versions of the book are available on Amazon.com.



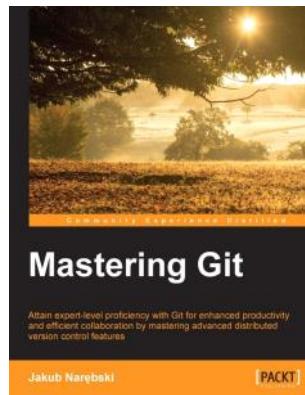
2nd Edition (2014)
Switch to 1st Edition

1. Inicio - Sobre el Control de Versiones

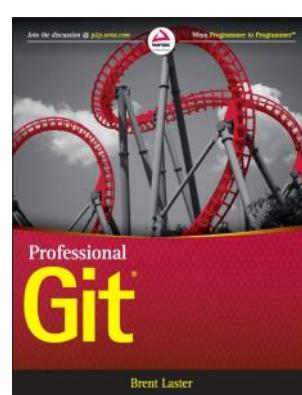
- 1.1 Acerca del Control de Versiones
- 1.2 Una breve historia de Git
- 1.3 Fundamentos de Git
- 1.4 La Línea de Comandos
- 1.5 Instalación de Git
- 1.6 Configurando Git por primera vez
- 1.7 ¿Cómo obtener ayuda?
- 1.8 Resumen



Git for Teams
Emma Jane Westby
O'Reilly, 2015



Mastering Git
Jakub Narębski
Packt Publishing, 2016



Professional Git
Brent Laster
John Wiley & Sons, 2016

Instalación y Configuración

lunes, 8 de enero de 2018 13:33

Instalación y Configuración de GIT

- Git. Terminales en Windows
 - Configuraciones
- Clientes gráficos para operar con GIT
 - SourceTree
 - Cliente de GitHub
- Integración en IDEs y Editores. Visual Studio Code

Instalación de Git

martes, 25 de abril de 2017 0:24

The screenshot shows the official Git website at git-scm.com. At the top, there's a search bar and a diagram illustrating the distributed nature of Git. Below the header, there are several links: 'About' (advantages over other systems), 'Documentation' (command reference, book, videos), 'Downloads' (GUI clients, binary releases for various platforms), and 'Community' (bug reporting, mailing lists, chat). A red box highlights the 'Downloads for Windows' link. Below these, there's a section for 'Pro Git' by Scott Chacon and Ben Straub, and a note about dead tree versions available on Amazon.

The screenshot shows the 'Git 2.6.3 Setup' wizard. It starts with the 'Welcome to the Git Setup Wizard' screen, which asks the user to close other applications before continuing. The next screen, 'Select Destination Location', shows the default installation path 'C:\Program Files\Git'. The third screen, 'Select Start Menu Folder', asks where to place the program shortcut, with the option 'Don't create a Start Menu folder' selected. A red arrow points from the 'Downloads for Windows' link on the Git website to this 'Select Start Menu Folder' screen.

The screenshot continues through the setup wizard. It shows the 'Adjusting your PATH environment' screen, which asks how to use Git from the command line. The 'Use Git from the Git Bash only' option is selected. The next screen, 'Configuring the terminal emulator to use with Git Bash', asks which terminal emulator to use. The 'Use Git/T (the default terminal of Msys2)' option is selected. Finally, the 'Completing the Git Setup Wizard' screen shows the summary of changes made, including enabling file system caching. A red arrow points from the 'Select Start Menu Folder' screen to this final completion screen.



Resultado



Comprobación

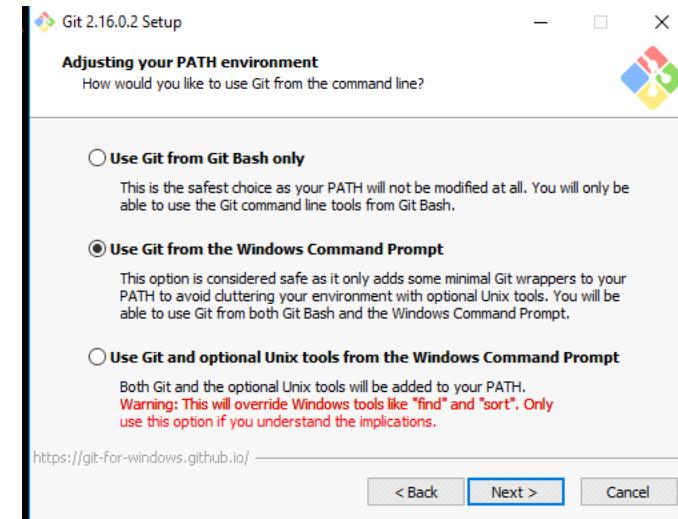
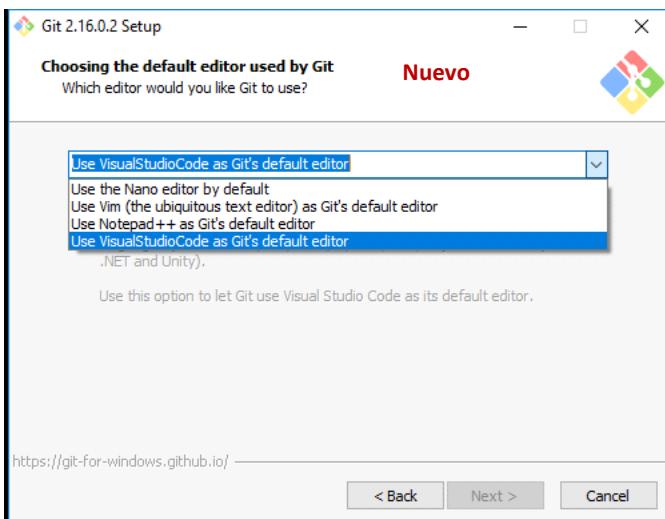
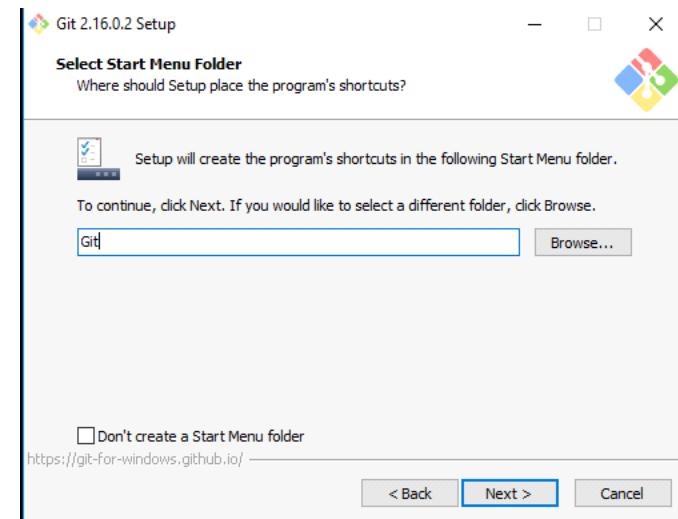
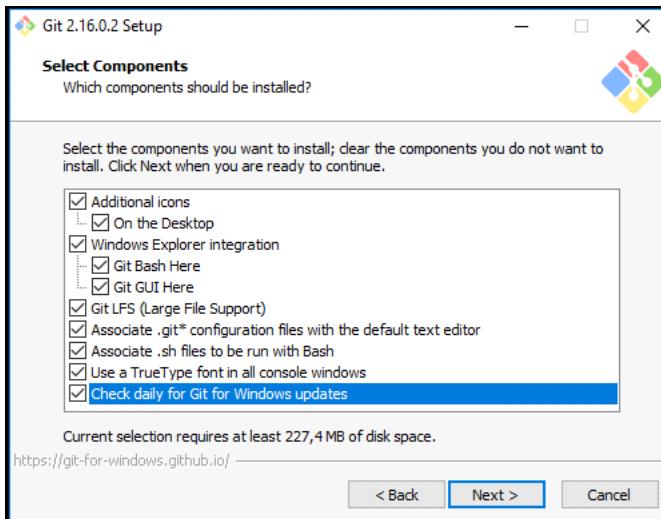
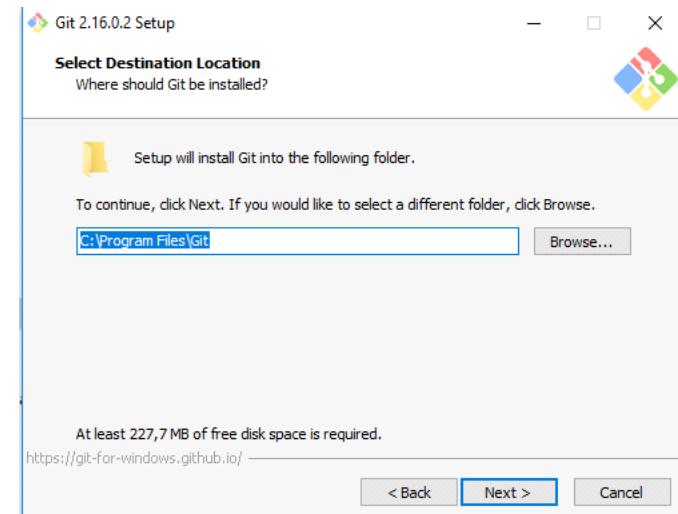
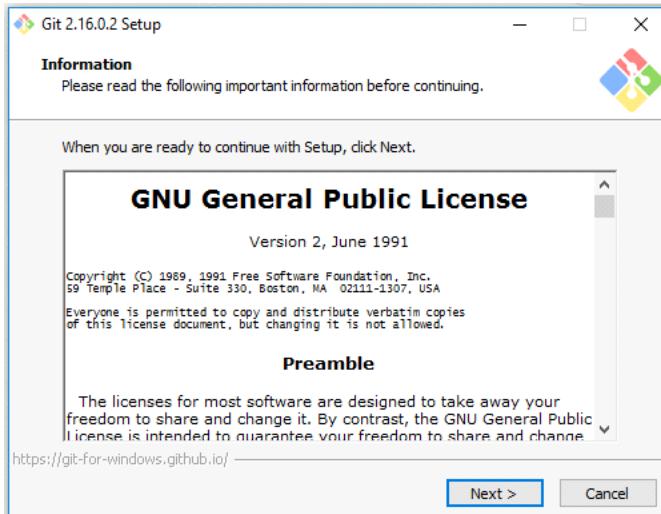
```

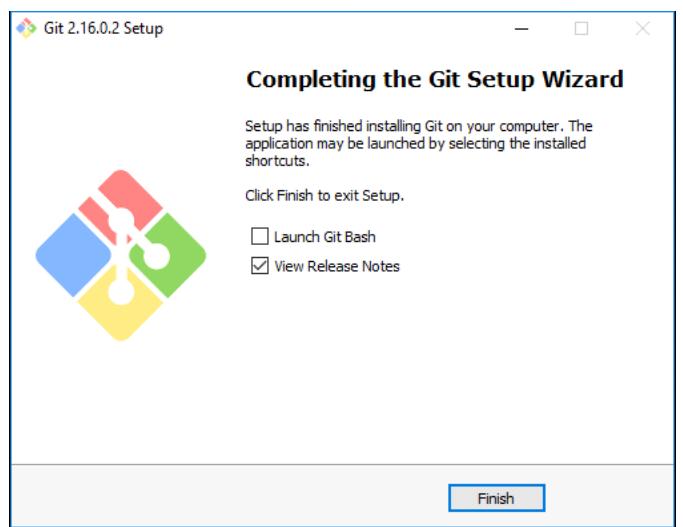
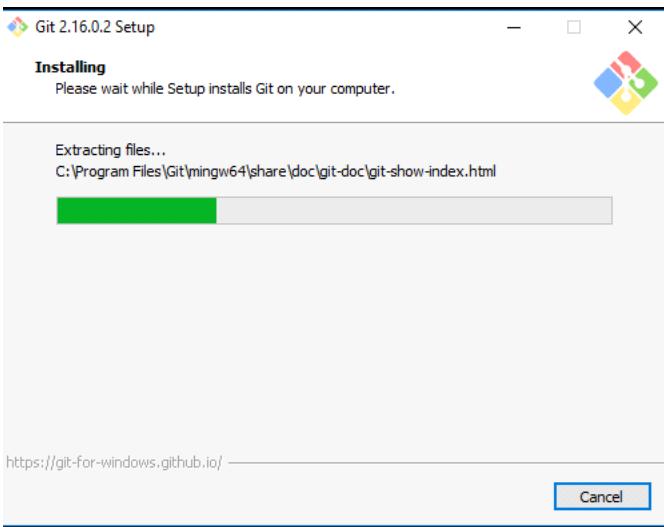
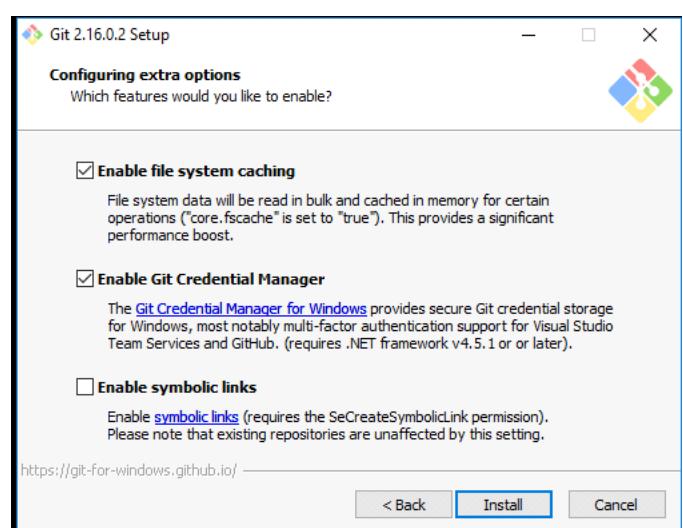
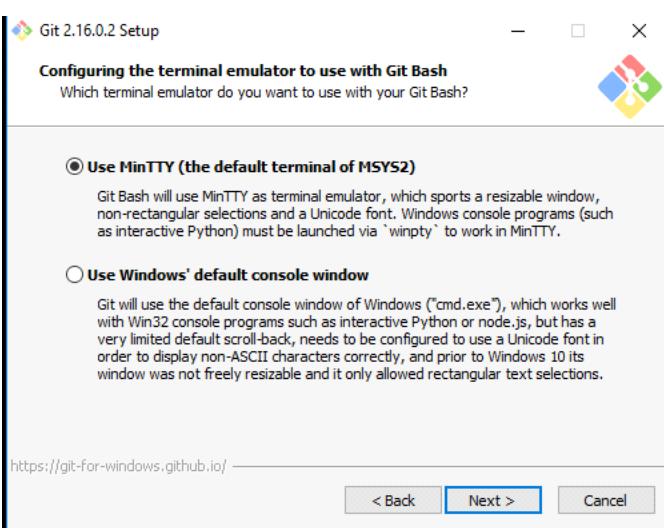
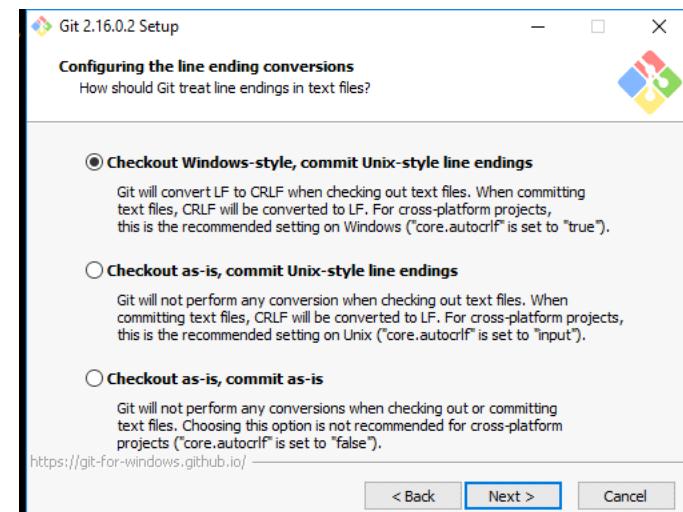
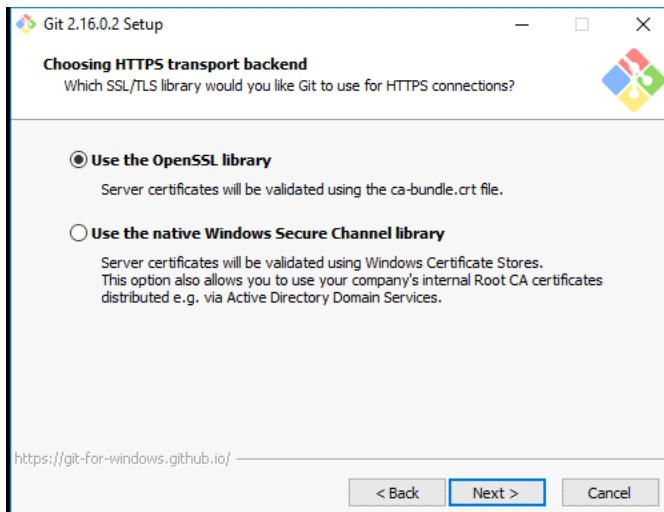
MINGW64:/d/Users/Alejandro
$ git --version
git version 2.12.1.windows.1
Alejandro@Cyborg8 MINGW64 ~
$ |

```

2.16.x

domingo, 21 de enero de 2018 22:08

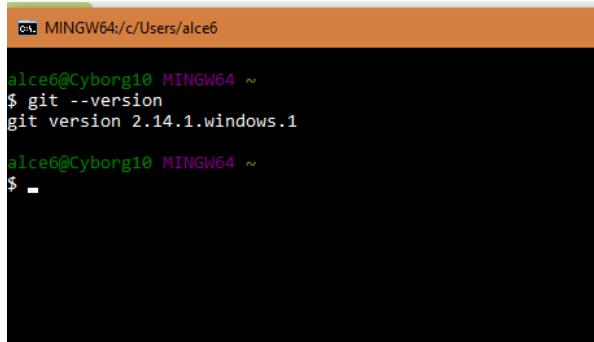




Terminales

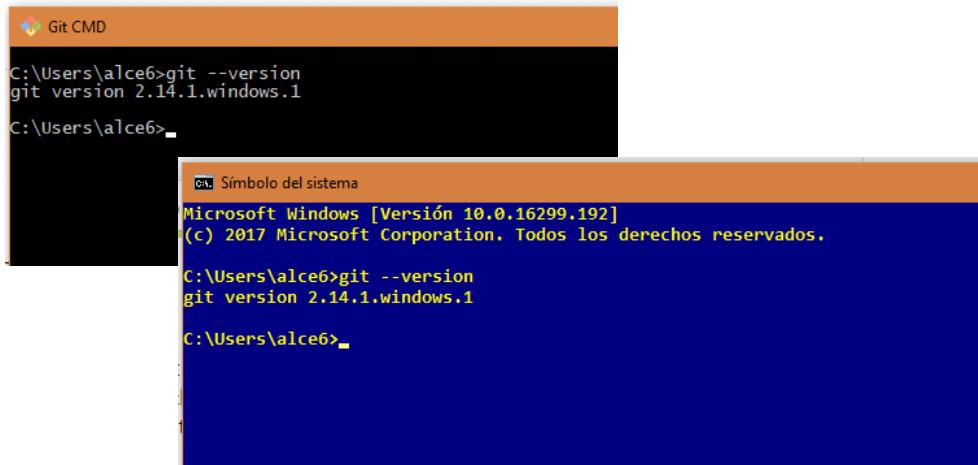
lunes, 8 de enero de 2018 14:40

Git Bash: shell Linux incluida en la instalación de Git



```
alce6@Cyborg10 MINGW64 ~
$ git --version
git version 2.14.1.windows.1
alce6@Cyborg10 MINGW64 ~
$ -
```

Símbolo del sistema (consola) de Windows



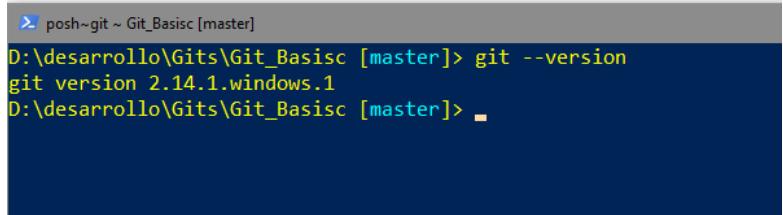
```
C:\Users\alce6>git --version
git version 2.14.1.windows.1
C:\Users\alce6> -
```



```
Microsoft Windows [Versión 10.0.16299.192]
(c) 2017 Microsoft Corporation. Todos los derechos reservados.

C:\Users\alce6>git --version
git version 2.14.1.windows.1
C:\Users\alce6> -
```

Power Shell configurado especialmente para Git



```
posh~git ~ Git_Basic [master]
D:\desarrollo\Gits\Git_Basic [master]> git --version
git version 2.14.1.windows.1
D:\desarrollo\Gits\Git_Basic [master]> -
```

Se utiliza el *plugin posh-git*

<https://github.com/dahlbyk/posh-git>

En la propia Web de GitHub se indica como instalarlo directamente o después de instalar *GitHub for Windows*, que ya incluye el plugin

<https://git-scm.com/book/sv/v2/Appendix-A%3A-Git-in-Other-Environments-Git-in-Powershell>

Para ello hay que crear / editar el fichero
%user%\Documents\WindowsPowerShell\Microsoft.PowerShell_profile.ps1

En él se incluirá

- . (Resolve-Path "\$env:LOCALAPPDATA\GitHub\shell.ps1")
- . \$env:github_posh_git\profile.example.ps1

Para la creación del script de PowerShell el proceso es el siguiente

<https://technet.microsoft.com/en-us/library/ff461033.aspx>

*At the Windows PowerShell prompt, type the following command, and then press ENTER:
Test-path \$profile*

*If the results of the previous command are false, Type the following command, and then press ENTER.
New-item -type file -force \$profile*

*If the results of the test are true, type the following command, and then press ENTER.
Notepad \$profile*

NOTA:

Windows PowerShell como **administrador** y ejecutamos el comando “**Get-ExecutionPolicy**” nos tendría que devolver “**Unrestricted**” o lo que es lo mismo “Restringido”. Para cambiar esta configuración basta con ejecutar “**Set-ExecutionPolicy Unrestricted**”.

<http://www.itprotoday.com/management-mobility/powershell-basics-console-configuration>

Configuración

domingo, 10 de septiembre de 2017 13:01

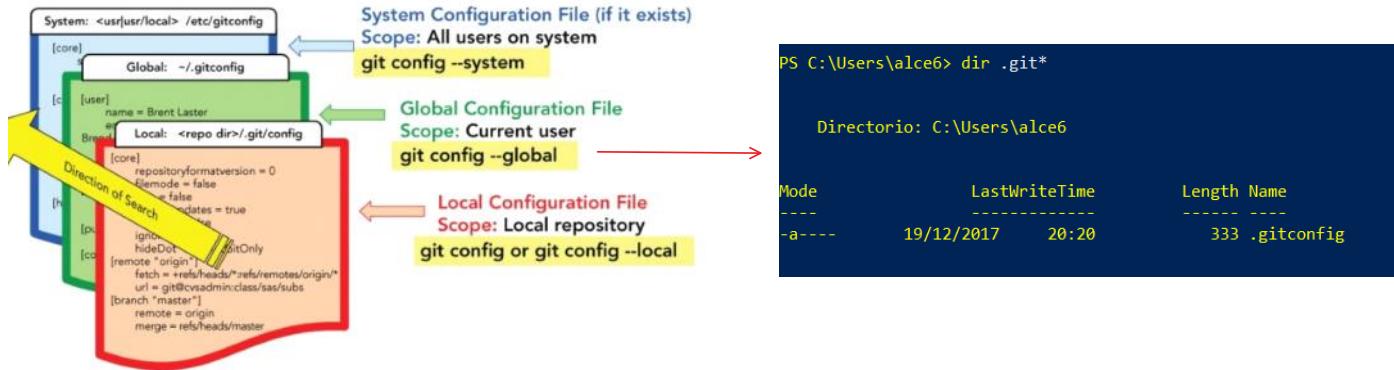
Git trae una herramienta llamada **git config** que te permite obtener y establecer variables de configuración, que controlan el aspecto y funcionamiento de Git.

Estas variables pueden almacenarse en tres sitios distintos, dependiendo de los parámetros del comando:

- **Archivo /etc/gitconfig**: Contiene valores para todos los usuarios del sistema y todos sus repositorios. Si pasas la opción **--system** a git config, lee y escribe específicamente en este archivo.
- **Archivo ~/.gitconfig**: Específico a tu usuario. Puedes hacer que Git lea y escriba específicamente en este archivo pasando la opción **--global**.
- **Archivo config "local"**, en el directorio de Git (es decir, *.git/config*) del repositorio que estés utilizando actualmente: Específico a ese repositorio. Cada nivel sobrescribe los valores del nivel anterior, por lo que los valores de *.git/config* tienen preferencia sobre los de */etc/gitconfig*

Como en cualquier entorno Linux, los modificadores utilizan
-- seguido de una palabra
- seguido de una letra, muchas veces como

Ámbito (Scope) de los ficheros de configuración en Git



Usuario

lunes, 8 de enero de 2018 14:27

Identidad del usuario

```
$ git config --global user.name "Pepe Perez"  
$ git config --global user.email pperez@example.com
```

Username

```
$ git config --global user.name "Your Name Here"  
# Sets the default name for git to use when you commit
```

Email

```
$ git config --global user.email "your_email@example.com"  
# Sets the default email for git to use when you commit
```

Editor por defecto

lunes, 8 de enero de 2018 14:31

Configuración del editor por defecto

git config --global core.editor "\"C:\Program Files (x86)\Microsoft VS Code\Code.exe\""

git config --global core.editor "code.exe --wait"

Propiedades configuradas

miércoles, 27 de diciembre de 2017 11:02

```
PS C:\Users\alce6> git config --list
core.symlinks=false
core.autocrlf=true
core.fscache=true
color.diff=auto
color.status=auto
color.branch=auto
color.interactive=true
help.format=html
diff.astextplain.textconv=astextplain
rebase.autosquash=true
filter.lfs.clean=git-lfs clean -- %f
filter.lfs.smudge=git-lfs smudge -- %f
filter.lfs.required=true
filter.lfs.process=git-lfs filter-process
http.sslcainfo=C:/Program Files/Git/mingw64/ssl/certs/ca-bundle.crt
http.sslbackend=openssl
diff.astextplain.textconv=astextplain
filter.lfs.clean=git-lfs clean -- %f
filter.lfs.smudge=git-lfs smudge -- %f
filter.lfs.required=true
filter.lfs.process=git-lfs filter-process
credential.helper=manager
filter.lfs.process=git-lfs filter-process
filter.lfs.required=true
filter.lfs.clean=git-lfs clean -- %f
filter.lfs.smudge=git-lfs smudge -- %f
filter.lfs.required=true
user.name=alce65
user.email=alce65@hotmail.es
core.excludesfile=D:\alce65\Documents\gitignore_global.txt
core.editor="C:\Program Files (x86)\Microsoft VS Code\Code.exe"
PS C:\Users\alce6>
```

Alias

martes, 9 de enero de 2018 0:36

Los alias fueron añadidos en Git 1.4.0

Evitan tener que teclear repetidas veces el mismo comando completo

Se pueden configurar

- en cualquiera de los ámbitos de configuración mencionados
- en cualquier caso,
 - o por comandos
 - o editando el fichero correspondiente

`git config --global alias.co commit`

- Editar `Programs/git/gitconfig`
- Editar `$HOME/<user>/.gitconfig` (**específico de usuario**)
- Editar `.git//config` (específico del repositorio)

```
[alias]  
co = commit
```

Ejemplo de uso

```
git config --global alias.hist git log --pretty=format:"%h %ad | %s%d [%an]" --graph --date=short
```

Configuración de proxies

lunes, 7 de agosto de 2017 21:38

Configuración git

```
git config --global http.proxy http://user:passw@proxy.empresa.es:8080
```

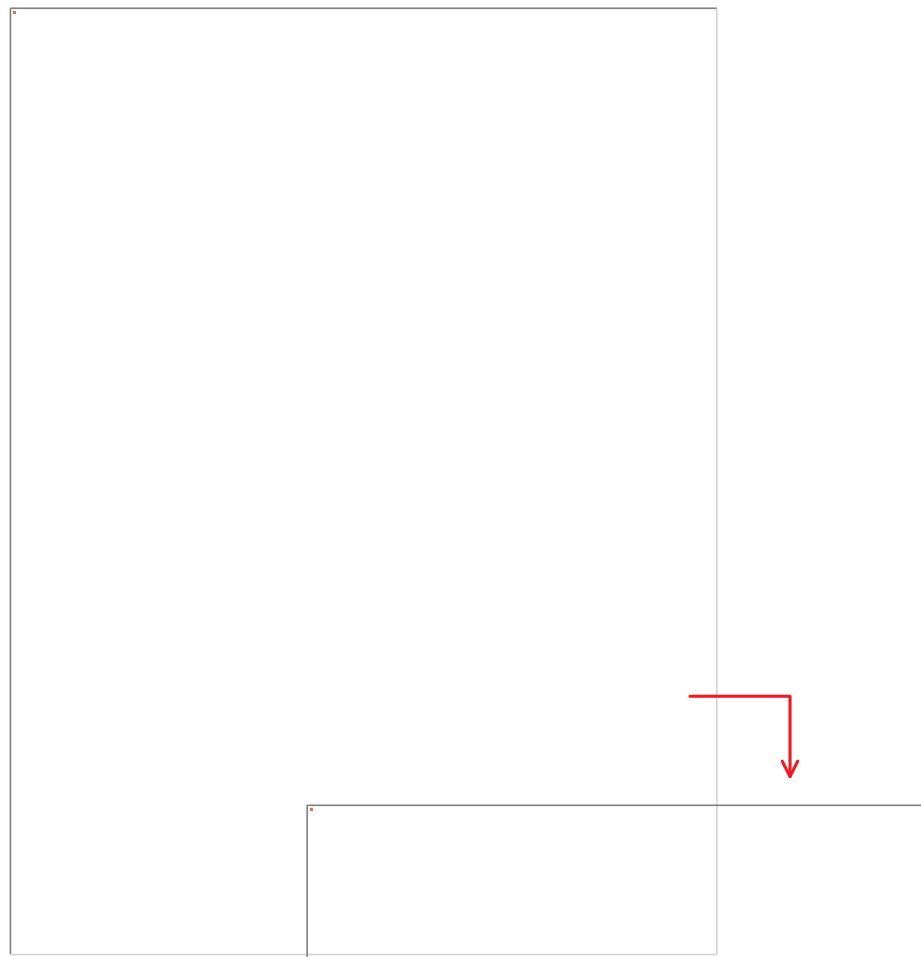
Otros aspectos de la configuración de conexiones

Configuración npm

```
$>npm config set proxy http://user:passw@proxy.emprea.es:8080
```

```
$>npm config set https-proxy http://user:passw@proxy.empresa.es:8080
```

Propiedades de internet





Permite usar localhost con los navegadores en combinación con IIS para probar el código desarrollado

Clientes gráficos

Lunes, 8 de enero de 2018 14:34

Cliente incluido en la instalación de Git

Otras opciones desde la propia Web de Git

<https://git-scm.com/download/gui/windows>

- SourceTree
- GitHub Desktop
- TortoiseGit
- Git Extensions
- GitKraken
- SmartGit
- Tower
- GitEye
- gitg
- ungit
- git-cola
- Cycligent Git Tool
- Aurees
- CodeReview
- gmaster
- GitAhead

SourceTree

Platforms: Mac, Windows
Price: Free
License: Proprietary

SmartGit

Platforms: Linux, Mac, Windows
Price: \$79/user / Free for non-commercial use
License: Proprietary

Git Extensions

Platforms: Linux, Mac, Windows
Price: Free
License: GNU GPL

ungit

Platforms: Linux, Mac, Windows
Price: Free
License: MIT

Aurees

GitHub Desktop

Platforms: Mac, Windows
Price: Free
License: MIT

Tower

Platforms: Mac, Windows
Price: \$79/user (Free 30 day trial)
License: Proprietary

GitKraken

Platforms: Linux, Mac, Windows
Price: Free for non-commercial use
License: Proprietary

git-cola

Platforms: Linux, Mac, Windows
Price: Free
License: GNU GPL

CodeReview

TortoiseGit

Platforms: Windows
Price: Free
License: GNU GPL

GitEye

Platforms: Linux, Mac, Windows
Price: Free
License: Proprietary

gitg

Platforms: Linux, Windows
Price: Free
License: GNU GPL

Cycligent Git Tool

Platforms: Linux, Mac, Windows
Price: Free
License: Proprietary

gmaster

Platforms: Windows

Price: Free
License: MIT

Aurees

Platforms: Windows
Price:
License: Proprietary

Price: Free
License: GNU GPL

CodeReview

Platforms: Linux, Mac, Windows
Price: Free
License: GNU GPL

License: Proprietary

gmaster

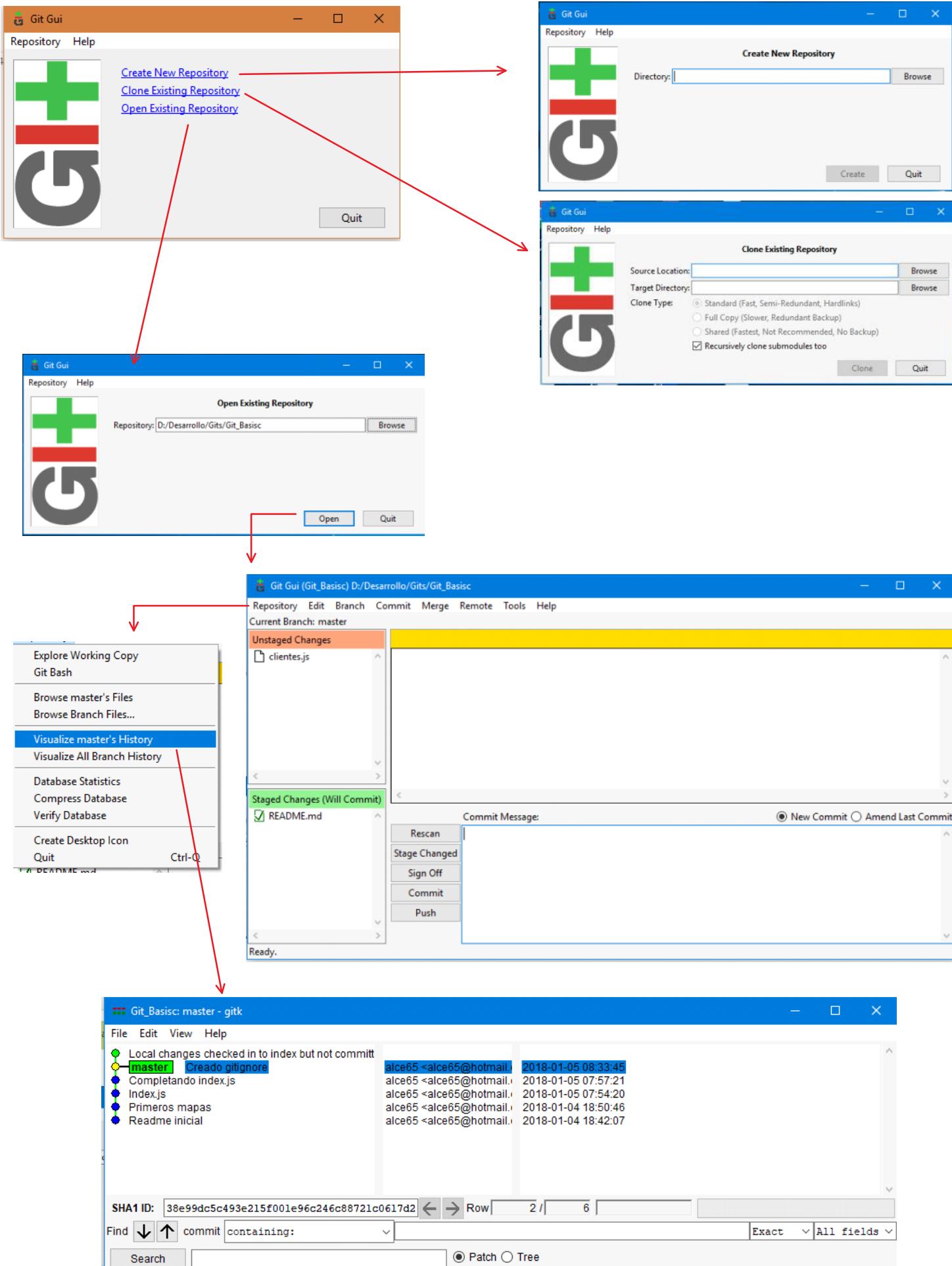
Platforms: Windows
Price: Beta / Free for non-commercial use
License: Proprietary

GitAhead

Platforms: Linux, Mac, Windows
Price: Free for non-commercial use
License: Proprietary

Git GUI

miércoles, 24 de enero de 2018 22:08



The screenshot shows the Gitk interface displaying the commit history for the 'master' branch of a repository named 'Git_Basic'. The commits are listed in chronological order from top to bottom:

- Local changes checked in to index but not committed**
- master Creado gitignore** (selected commit)
- Completando index.js
- Index.js
- Primeros mapas
- Readme inicial

Details for the selected commit ('Creado gitignore'):

- Author: alce65 <alce65@hotmail.es> 2018-01-05 08:33:45
- Committer: alce65 <alce65@hotmail.es> 2018-01-05 09:57:21
- Parent: [aabf62474ae960b89084b2bc6d4497165351938e](#) (Co)
- Branch: [master](#)
- Follows:
- Precedes:

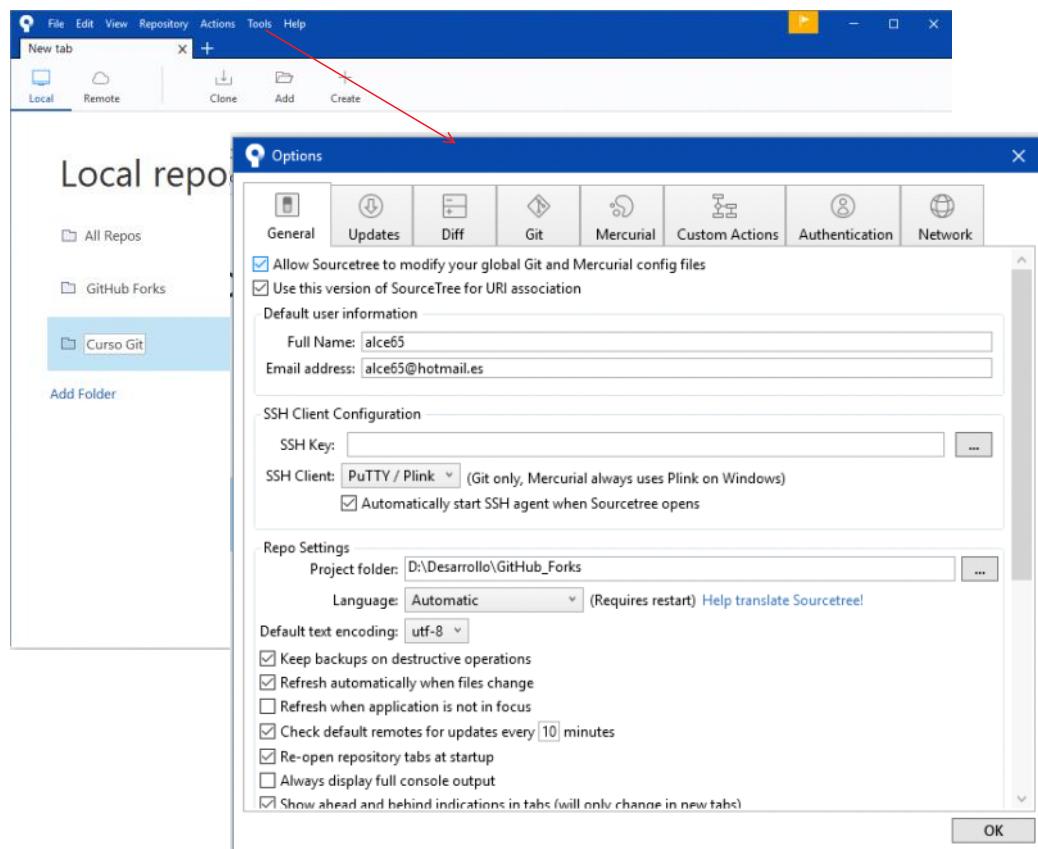
The commit message is "Creado gitignore". The diff output shows:

```
----- .gitignore -----  
new file mode 100644  
index 0000000..0e17662  
@@ @ @ -1 + @@  
<
```

SourceTree

Junes, 8 de enero de 2018 14:36

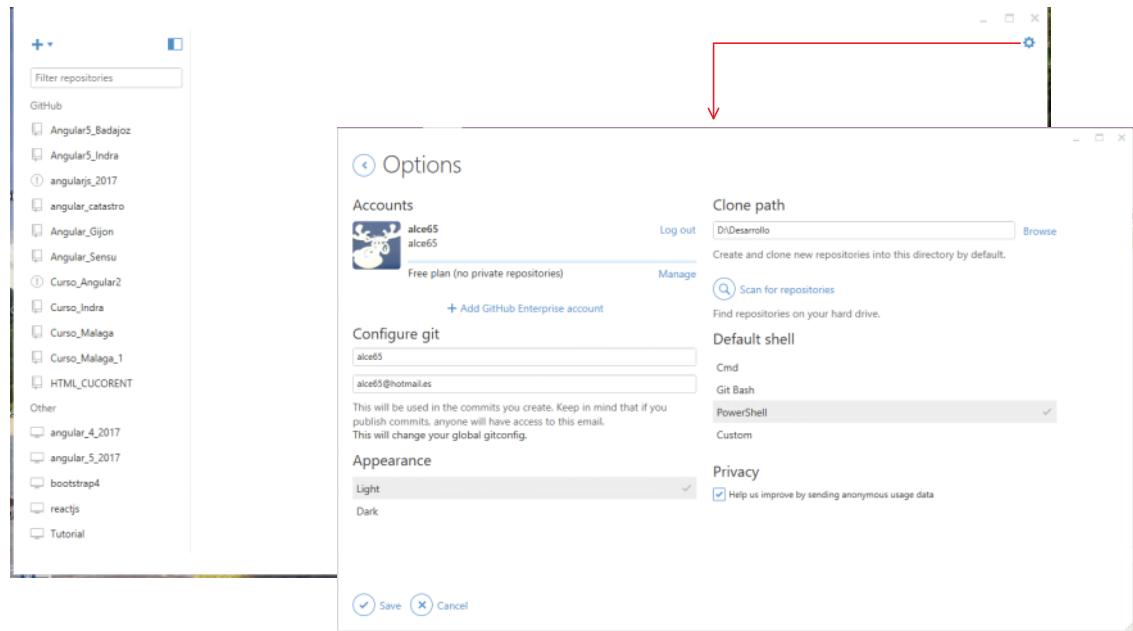
<https://www.sourcetreeapp.com/>



GitHub Desktop

lunes, 8 de enero de 2018 14:38

<https://desktop.github.com/>

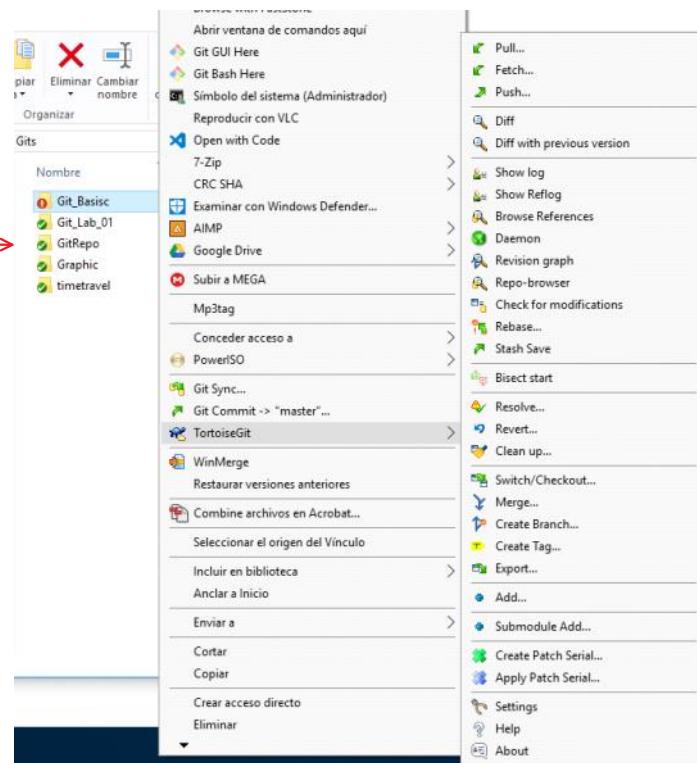


Tortoise

miércoles, 24 de enero de 2018 22:48



Iconos
representando el
estado de las
carpetas que
corresponden a
repositorios Git

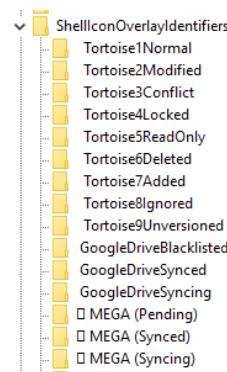


Ajustes en el registro de
Windows (10) para que
aparezcan los iconos

HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows
\CurrentVersion\Explorer\ShellIconOverlayIdentifiers\

Los espacios al principio de
los nombres garantizan que
estos sean los primeros al
ser ordenados
alfabéticamente

*values are not in the top 11
values, their icon overlays
will not be shown.*



<https://stackoverflow.com/questions/25156238/tortoisegit-not-showing-icon-overlays>

Entornos de desarrollo

lunes, 8 de enero de 2018 23:18

Editores de código



IDEs

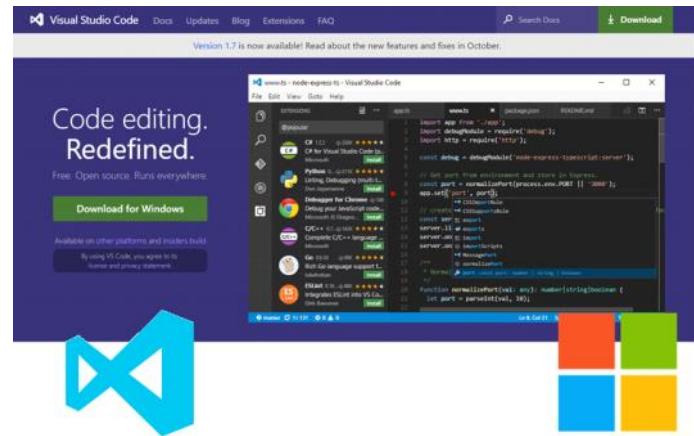
- Eclipse
- NetBeans
- Visual Studio



Editor de código: VSC

sábado, 20 de enero de 2018 12:10

<https://code.visualstudio.com/>



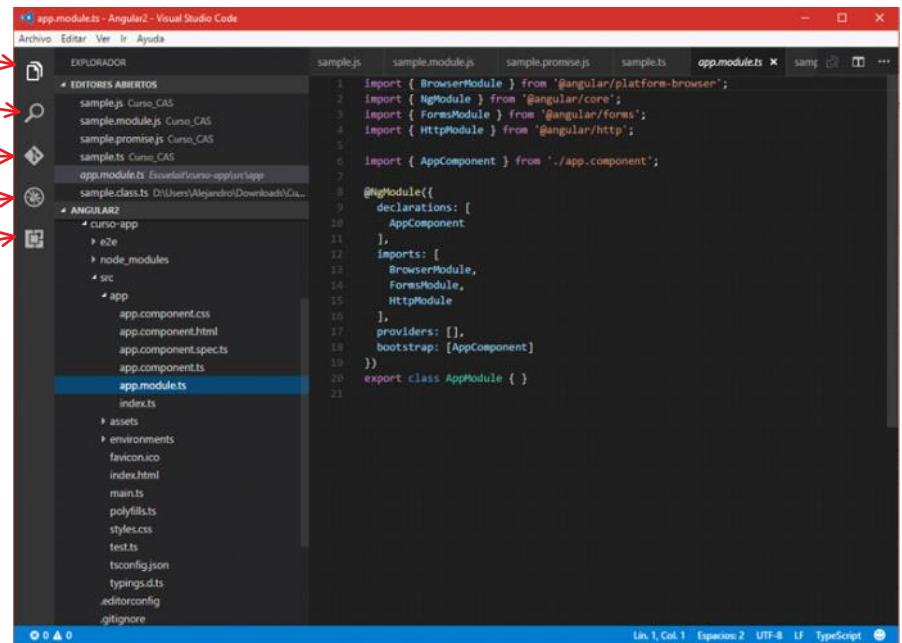
Carpetas y ficheros

Búsquedas en el proyecto

Git - GitHub integrado

Depurador

Extensiones



IDE: Eclipse

Junes, 8 de enero de 2018 23:44

The screenshot shows the Eclipse IDE interface with the following components visible:

- Package Explorer View:** Shows two Java files: `Klondike.java` and `Descarte.java`. `Klondike.java` contains a class `Klondike` and a method `mostrar()`. `Descarte.java` contains a class `Descarte` and methods `mostrar()`, `movea(Palo[] palos)`, and `movea(Columna[] columnas)`.
- Git Repositories View:** Shows a local repository named `java_desk [master t3 t2 - Merged]` with branches `Local` and `Remote Tracking`, and a remote branch `origin`.
- Outline View:** Shows the class hierarchy for `Klondike` and `Descarte`.
- Task List View:** Shows a list of tasks.
- Code Editor View:** Displays the Java code for `Klondike.java` and `Descarte.java`.

En resumen

sábado, 20 de enero de 2018 12:27

- Instalación de Git
- (Creación de cuenta en GitHub)
- Instalación de GitHub Desktop
- Configuración del usuario en Git
- Configuración de *proxies* en Git
- Configuración de PowerShell para Git
- Instalación de SourceTree
- Instalación de VSC
- (Instalación de Eclipse)

Arquitectura y Componentes

lunes, 8 de enero de 2018 13:36

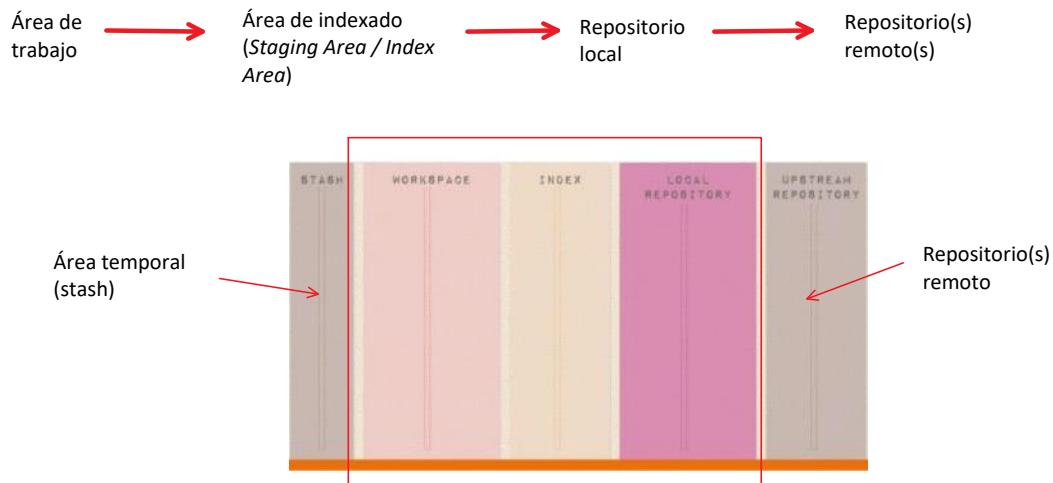
Arquitectura y Componentes fundamentales

- Áreas
 - Stashing
 - Working
 - Steaging
 - Repository
- Commits
- Punteros: HEAD
- Creación de un repositorio, La carpeta .git
 - Readme.md y gitignore

Estructura

jueves, 4 de enero de 2018 19:06

Git puede concebirse como un sistema de niveles en el que se promocionan los ficheros:



Área de indexado (Staging Area / Index Area)
zona intermedia donde

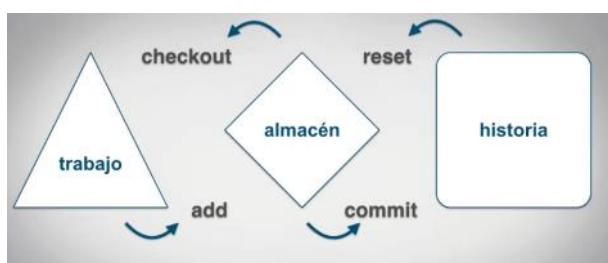
- se prepara el próximo commit
- se reparan o rehacen commits problemáticos (amend)

El repositorio creado tiene tres partes diferenciadas. El directorio es lo que se llama directorio o espacio de trabajo, y dentro de la carpeta .git que ha sido creada por el comando se encuentra el área de preparación, que actúa como una zona intermedia, y el historial de cambios.

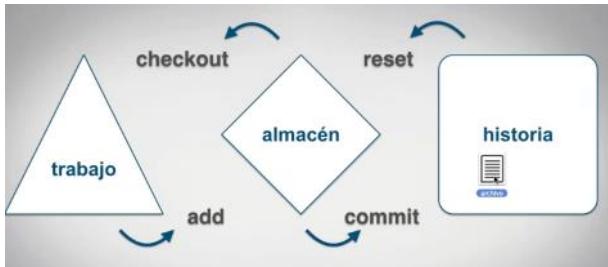
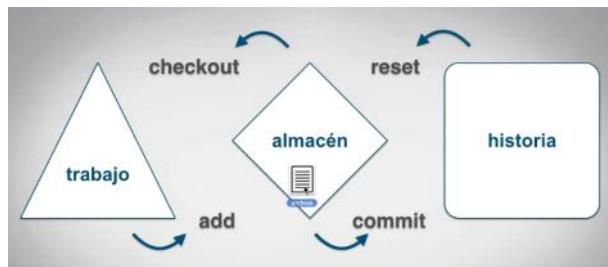
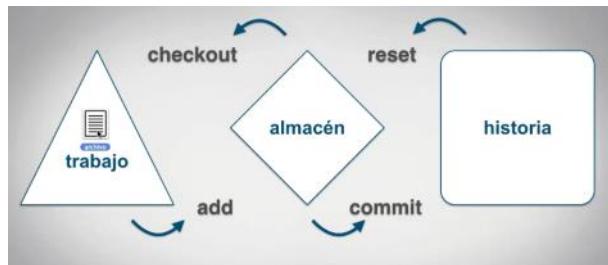
Etapas y flujo

jueves, 11 de mayo de 2017 0:08

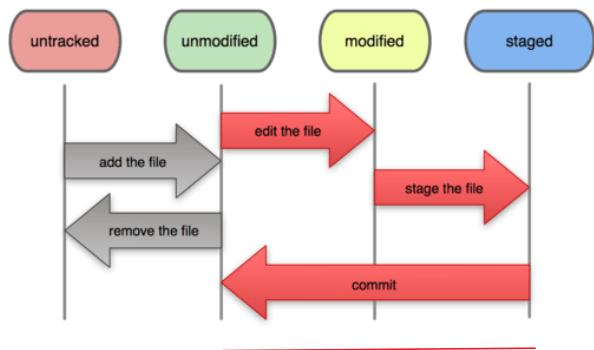
Estructura del repositorio



Ciclo de vida del fichero



File Status Lifecycle



Tracked

En su ciclo de vida (*Lifecycle*) los ficheros pasan por diversos estados

Cada fichero en el directorio de trabajo (*working directory* o *working area*) estará en uno de los dos estados posibles: rastreado (*tracked*) o sin seguimiento (*untracked*).

Tracked: ficheros que aparecen en la última instantánea (*snapshot*) del repositorio, pudiendo encontrarse modificados, preparados (*staged*) o no modificados y confirmados, .

Estados del archivo



Modificado

Preparado

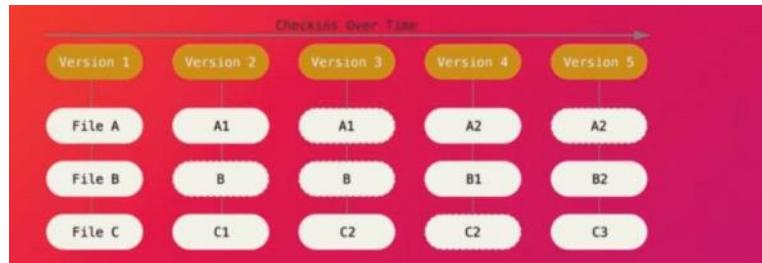
Confirmado

Untracked: cualquier otro fichero, del que el repositorio no tiene ninguna información.

Almacenamiento

sábado, 20 de enero de 2018 17:42

Git SVC: *Snapshot Storage Model*



OTROS SVC: *Delta Storage Model*

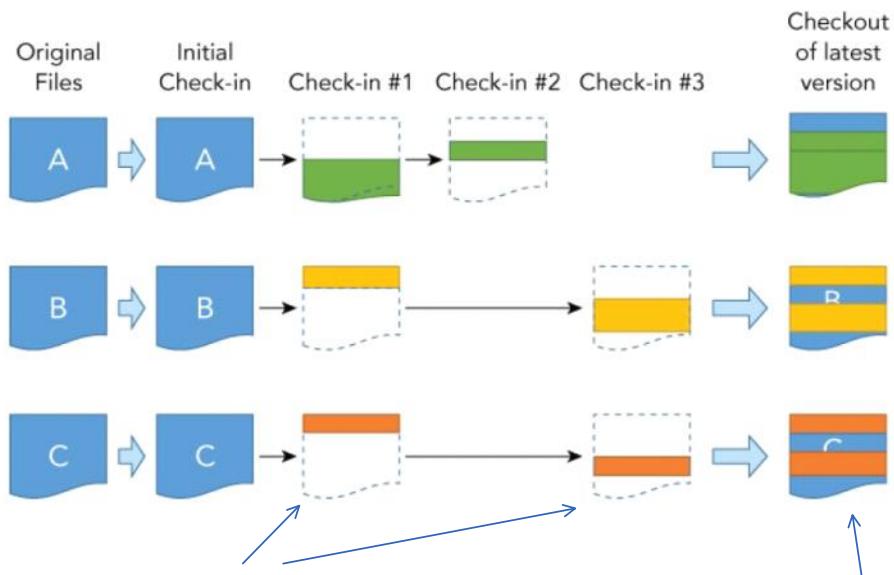
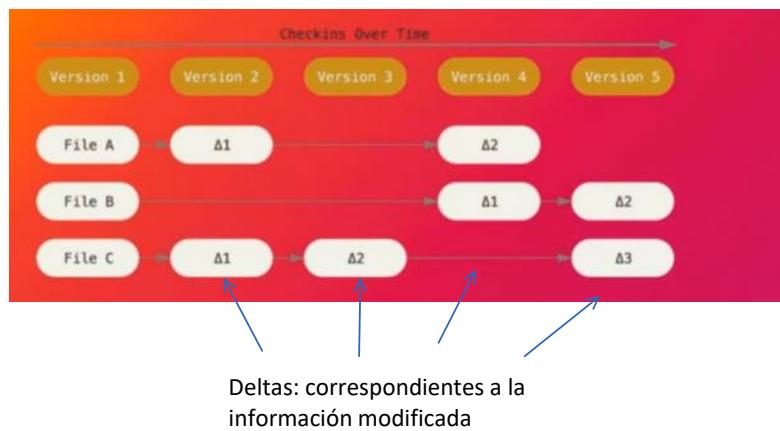


Otros sistemas SVC

sábado, 20 de enero de 2018 17:35

Delta Storage Model

Almacenamiento en base a ficheros



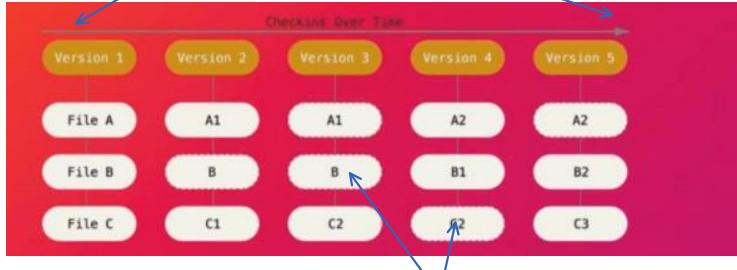
Professional Git
Brent Laster
John Wiley & Sons, 2016

GIT

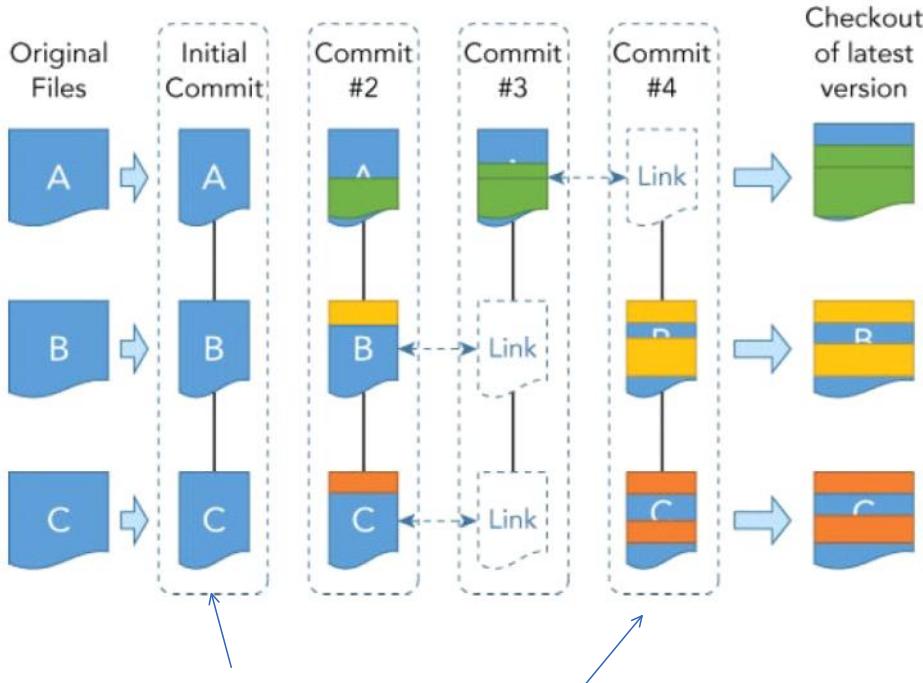
Git almacena *snapshots* (instantáneas) completas

Todos los procesos inicialmente son locales

OTROS SVC: *Delta Storage Model*



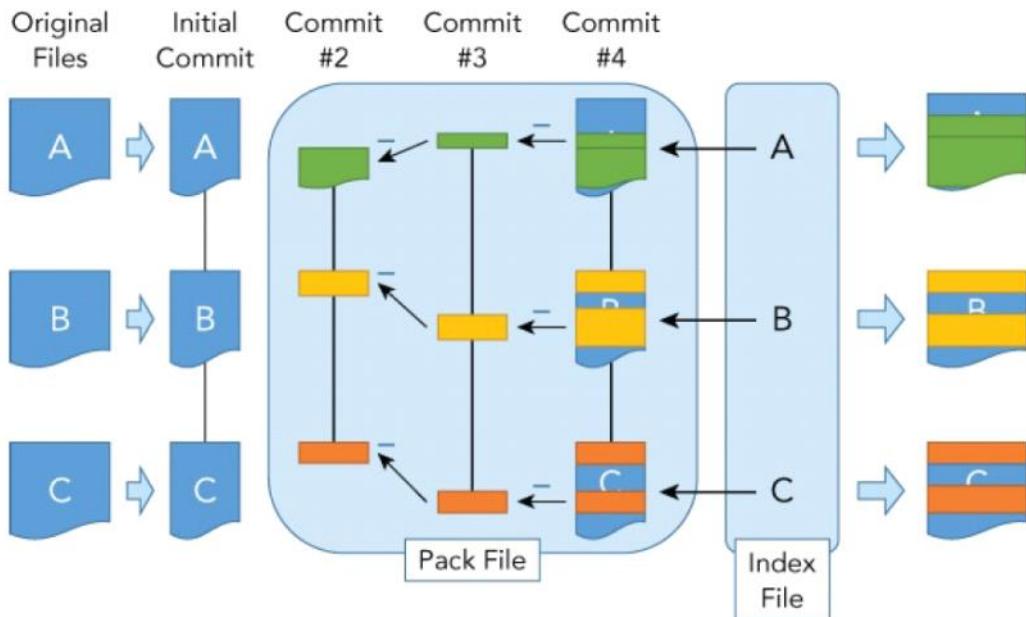
si no hay cambios se almacena un puntero al fichero sin cambios



Professional Git
Brent Laster
John Wiley & Sons, 2016

Empaque

Periódicamente, en ciertos momentos (*trigger points*), como cuando se ejecuta la funcionalidad de recolección de elementos no utilizados (*garbage collection*), Git busca contenido que sea muy similar entre revisiones y empaqueta esas revisiones juntas para formar un archivo de paquete comprimido (*compressed pack file*)



Desde su origen, este modelo está pensado para versionar el código, es decir ficheros de un tamaño bastante limitado. Posteriormente se han desarrollado extensiones que mejoran el tratamiento de ficheros muy grandes (imágenes, audio, video...) como son

- git-annex (<https://git-annex.branchable.com/and>)
- Git Large File Storage (Git LFS) (<https://git-lfs.github.com/>)

Git Large File Storage

Docs Downloads Source

An open source Git extension for versioning large files

Git Large File Storage (LFS) replaces large files such as audio samples, videos, datasets, and graphics with text pointers inside Git, while storing the file contents on a remote server like GitHub.com or GitHub Enterprise.

Download v2.3.4 (Windows)

Remote

Local

Large File Storage

file.psd

GitHub.com support now available. [Install the client to get started.](#)

Commits

miércoles, 27 de diciembre de 2017 20:00

Para registrar información sobre los cambios en el repositorio se utilizan **commits** → registros de la DB

Contenido de un *commit*:

- el *changeset* (i.e. los cambios) de cada uno de los ficheros afectados
- punteros a todos los ficheros que no han cambiado



Un *commit* es el único método de introducir cambios en un repositorio.

Los *commits* generalmente son generados de forma explícita por el usuario pero también pueden ser generados por Git, como sucede en el caso del *merge*.

Un *commit* se hace visible con el comando git show

Carácter transaccional de los *commits*

Cada *commit* en Git representa un conjunto de cambios (*change set*) único y → DB transaccionales

Se aplican todos los cambios incluidos en un *commit* o ninguno
(atomicidad)

Se puede estar seguro de que Git no deja un repositorio en un estado transitorio entre la instantánea de un *commit* y la del siguiente.

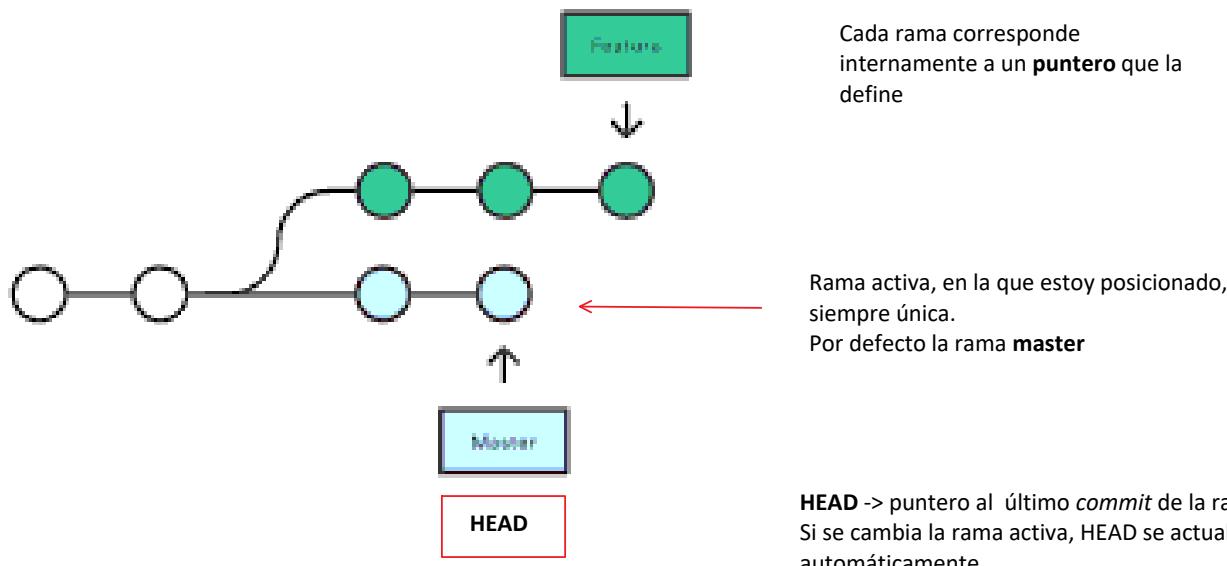
Cada *commit* tiene un *checksum* que garantiza la integridad de la información que contiene

Punteros: ramas y HEAD

lunes, 8 de enero de 2018 16:02

Los *commits* se organizan en secuencias temporales conocidas como ramas (*branch*)

Git internamente se basa en punteros



Cuando se añade un *commit*, HEAD cambia para seguir apuntando al último *commit* de la rama activa

Cuando se cambia a una rama diferente, HEAD cambia para seguir apuntando al último *commit* de la nueva rama.

Desde el S.O es posible comprobar como se produce este desplazamiento de punteros

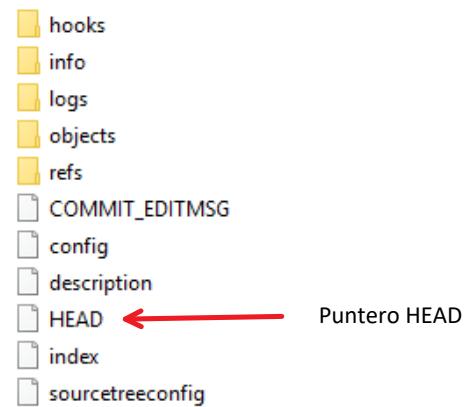
En *bash-shell*

```
$ cd .git  
$ cat HEAD -> ref: refs/heads/master  
$ git checkout -b newFeature  
$ cat HEAD -> ref: refs/heads/newFeature
```

Estructura de un repositorio (.git)

En Windows

```
cd .git  
type HEAD -> ref: refs/heads/master  
git checkout -b newFeature  
type HEAD -> ref: refs/heads/newFeature
```



Otras referencias

lunes, 22 de enero de 2018 0:57

Ciertas operaciones, como *merge* y *reset*, guardan la versión previa de HEAD como ORIG_HEAD justo antes de asignarle un nuevo valor a la primera.

El valor ORIG_HEAD puede usarse para revertir una operación recuperando el estado previo o para realizar las comparaciones necesarias.

Los nombres de las ramas, los de las ramas remotas o los tags son todos referencias.

En algunos casos se distinguen las referencias simbólicas (*symrefs*), como aquellas que apuntan indirectamente a los objetos en Git, pero en definitiva no dejan de ser referencias

Comandos

martes, 25 de abril de 2017 0:16

El diseño de Git se corresponde con la orientación a caja de herramientas, donde el comando git "contiene" un conjunto de subcomandos para las distintas tareas.

git <git-options> <command> <command-options> <operands>

Ejemplo git commit

Git está implementado como un conjunto de programas y scripts de *shell* que son fácilmente encadenables para formar nuevos comandos.

Además, Git cuenta con mecanismos para lanzar scripts de usuario cuando suceden ciertos eventos en el flujo de trabajo denominados puntos de enganche (*hooks*).

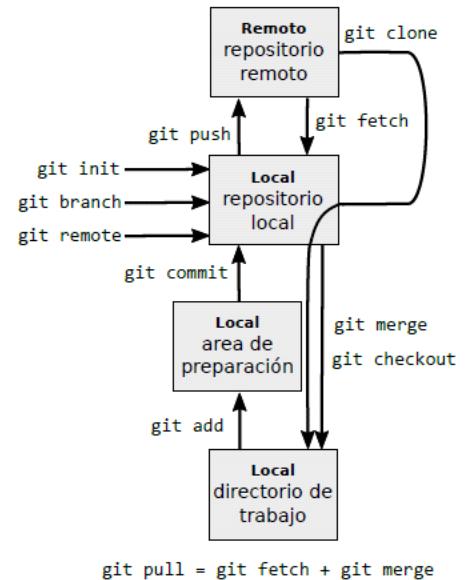
Resumen de comandos orientados al usuario
(*user friendly o Porcelain Commands*)

git init - crear un repositorio local en un directorio.
git clone - crear un repositorio local haciendo una copia de otro (local o remoto)
git add. - registra los ficheros del directorio de trabajo cuyos cambios se quieren
git commit - confirma los cambios de los ficheros registrados
git remote - configura los repositorios remotos

git branch - crear y destruir ramas
git checkout - permite cambiar de rama en el directorio de trabajo (Por defecto se trabaja en la rama denominada master)
git push - enviar los cambios a un repositorio remoto en la rama indicada
git pull - actualizar el repositorio con los cambios de un repositorio local

Este comando es esencialmente un encadenamiento de dos comandos:

git fetch - obtiene los cambios de una rama remota
git merge - fusiona si es posible estos cambios con una rama local.



Otros comandos

bisect
cherry
cherry-pick
config
diff
grep
help
log
mv
rebase
rerere
reset
revert
rm

show
status
submodule
subtree
tag
worktree

Mucho menos habituales son los comandos de bajo nivel (*Plumbing Commands*)

cat-file
commit-tree
count-objects
diff-index
for-each-ref
hash-object
ls-files
merge-base
read-tree
rev-list
rev-parse
show-ref
symbolic-ref
update-index
update-ref
verify-pack
write-tree

Ayudas

sábado, 20 de enero de 2018 19:03

```
~\Documents> git --help
usage: git [--version] [--help] [-C <path>] [-c name=value]
           [--exec-path[=<path>]] [--html-path] [--man-path] [--info-path]
           [-p | --paginate | --no-pager] [--no-replace-objects] [-bare]
           [--git-dir=<path>] [--work-tree=<path>] [--namespace=<name>]
           <command> [<args>]

These are common Git commands used in various situations:

start a working area (see also: git help tutorial)
  clone      Clone a repository into a new directory
  init       Create an empty Git repository or reinitialize an existing one

work on the current change (see also: git help everyday)
  add        Add file contents to the index
  mv         Move or rename a file, a directory, or a symlink
  reset     Reset current HEAD to the specified state
  rm         Remove files from the working tree and from the index

examine the history and state (see also: git help revisions)
  bisect    Use binary search to find the commit that introduced a bug
  grep      Print lines matching a pattern
  log       Show commit logs
  show      Show various types of objects
  status    Show the working tree status

grow, mark and tweak your common history
  branch   List, create, or delete branches
  checkout Switch branches or restore working tree files
  commit   Record changes to the repository
  diff     Show changes between commits, commit and working tree, etc
  merge   Join two or more development histories together
  rebase   Reapply commits on top of another base tip
  tag     Create, list, delete or verify a tag object signed with GPG

collaborate (see also: git help workflows)
  fetch   Download objects and refs from another repository
  pull    Fetch from and integrate with another repository or a local branch
  push    Update remote refs along with associated objects

'git help -a' and 'git help -g' list available subcommands and some
concept guides. See 'git help <command>' or 'git help <concept>'
to read about a specific subcommand or concept.
```

Función de autocompletado (<TAB>)

El doble guion --

Tiene una doble funcionalidad:

Separar opciones de una lista de argumentos
\$ git diff -w master origin -- tools/Makefile

Separar un nombre de archivo explícitamente identificado

Checkout the tag named "main.c"
\$ git checkout main.c

#Checkout the file named "main.c"
\$ git checkout -- main.c

Ejercicio. Primer repositorio

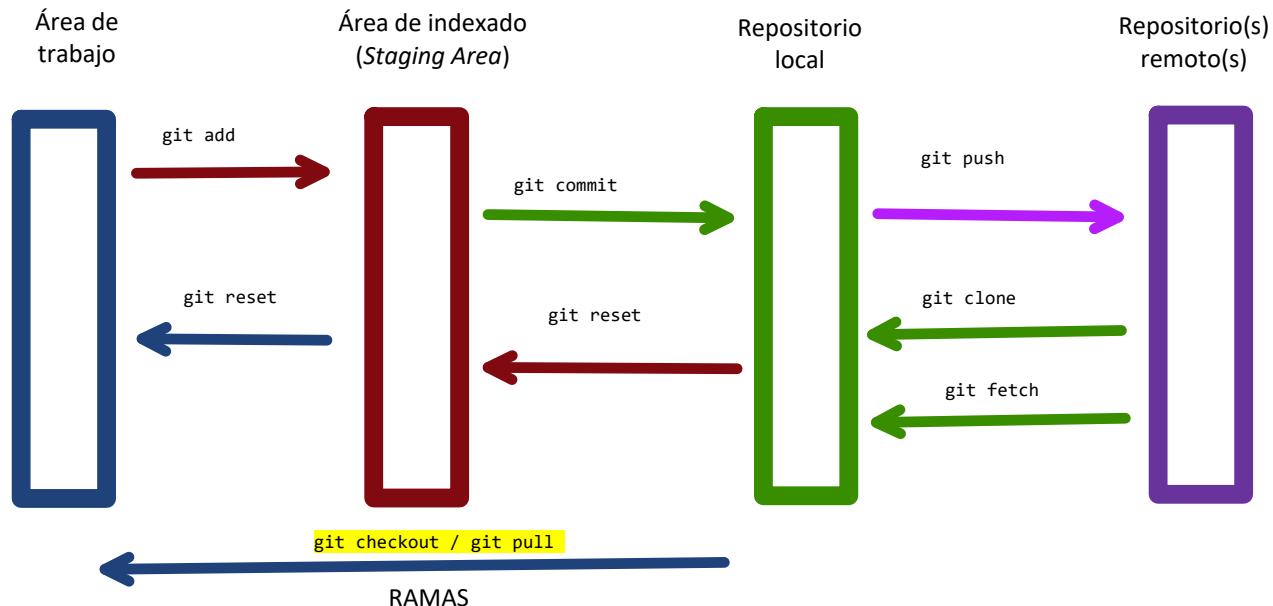
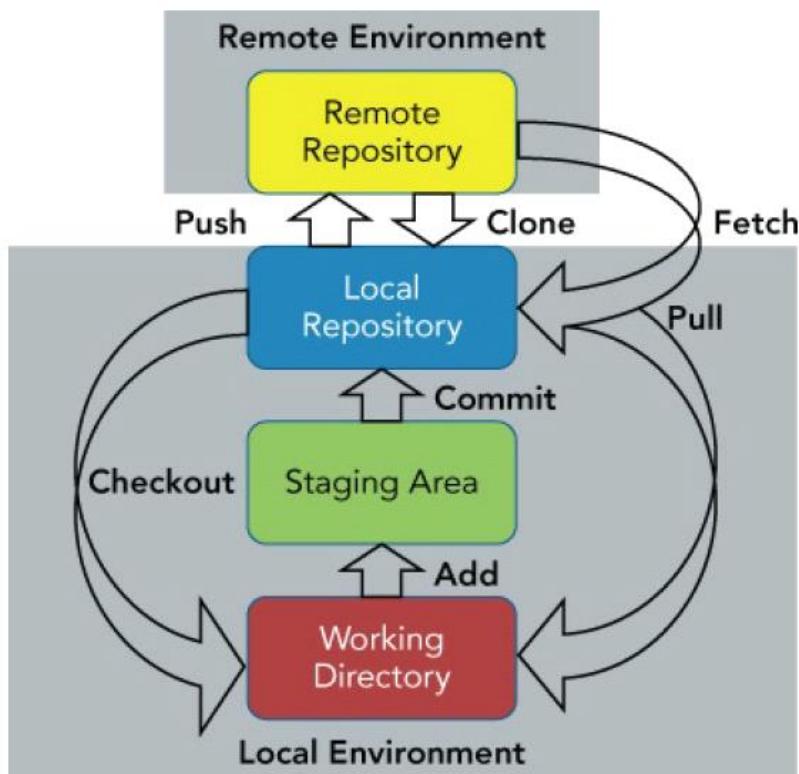
martes, 26 de diciembre de 2017 21:01

1. Se crea la carpeta raíz del proyecto, en la que se creará el repositorio
2. Se crea / inicializa el repositorio con el comando
`git init`

```
D:\desarrollo\Gits> cd .\Git_Basicc\  
D:\desarrollo\Gits\Git_Basicc> dir  
D:\desarrollo\Gits\Git_Basicc> git init  
Initialized empty Git repository in D:/desarrollo/Gits/Git_Basicc/.git/  
D:\desarrollo\Gits\Git_Basicc [master]>
```

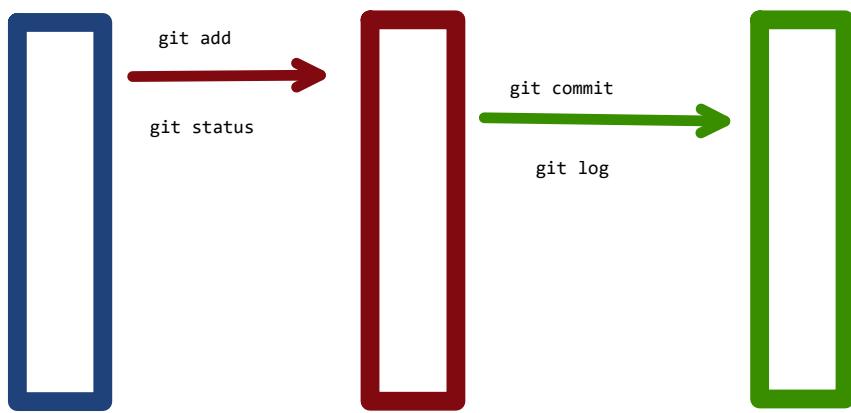
Flujos básicos

sábado, 20 de enero de 2018 18:22



Primeros comandos

Área de trabajo Área de indexado
(Staging Area)



Ejercicio: Primer commit

lunes, 8 de enero de 2018 20:48

```
D:\desarrollo\Gits\Git_Basic [master]> echo reaadme > README.md  
D:\desarrollo\Gits\Git_Basic [master +1 ~0 -0 !]> dir
```

```
Directorio: D:\desarrollo\Gits\Git_Basic  
  
Mode LastWriteTime Length Name  
---- ----- ----- ----  
-a--- 04/01/2018 18:31 20 README.md
```

Buena práctica: que este fichero del primer *commit* sea el [README.md](#)

```
D:\desarrollo\Gits\Git_Basic [master +1 ~0 -0 !]> git add README.md  
D:\desarrollo\Gits\Git_Basic [master +1 ~0 -0 ~]> git status  
On branch master
```

No commits yet

```
Changes to be committed:  
(use "git rm --cached <file>..." to unstage)  
  
new file: README.md
```

```
D:\desarrollo\Gits\Git_Basic [master +1 ~0 -0 ~]> git commit -m "Readme inicial"  
[master (root-commit) e4b3552] Readme inicial  
1 file changed, 0 insertions(+), 0 deletions(-)  
create mode 100644 README.md
```

```
D:\desarrollo\Gits\Git_Basic [master]> git log  
commit 833329b3f9bc80d44dc49931681d9f2f11906fd5 (HEAD -> master)  
Author: alce65 <alce65@hotmail.es>  
Date: Thu Jan 4 18:35:23 2018 +0100
```

```
Readme inicial  
D:\desarrollo\Gits\Git_Basic [master]> git log --oneline  
833329b (HEAD -> master) Readme inicial  
D:\desarrollo\Gits\Git_Basic [master]> ■
```

```
D:\desarrollo\Gits\Git_Basic [master]> git status  
On branch master  
nothing to commit, working tree clean  
D:\desarrollo\Gits\Git_Basic [master]> ■
```

Para todos los ficheros de la *workarea*:
git add .

git status

git commit

Hash: identificador único de cada *commit*
Sus 7 primeros dígitos es la versión
abreviada del identificador

git log --oneline
desencadena una versión compacta del
histórico

Elementos de un repositorio

domingo, 21 de enero de 2018 9:52

- carpeta git
- Buenas prácticas para el primer commit
 - Fichero Readme
 - Fichero gitignore

La carpeta .git

lunes, 8 de enero de 2018 21:45

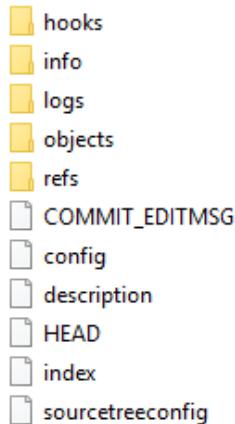
Comando : git init

```
D:\Desarrollo\Tools\GItRepo>git init
Initialized empty Git repository in D:/Desarrollo/Tools/GItRepo/.git/ ← Creación del
repository

D:\Desarrollo\Tools\GItRepo>dir /ad
El volumen de la unidad D es Data
El número de serie del volumen es: 10AD-0AF6

Directorio de D:\Desarrollo\Tools\GItRepo

28/10/2017 13:39 <DIR> .
28/10/2017 13:39 <DIR> ..
28/10/2017 13:39 <DIR> .git ← Carpeta oculta
    0 archivos          0 bytes
    3 dirs  197.347.762.176 bytes libres
```



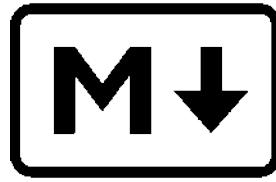
Fichero Readme

lunes, 8 de enero de 2018 20:45

Fichero de texto en formato *Markdown*

Contiene información descriptiva, instrucciones de uso, el historial de versiones

Se considera una buena práctica incluirlo en los repositorios, siendo prácticamente imprescindible en caso de repositorios compartidos, especialmente si se trata de proyectos de código libre / abierto o algún otro tipo de repositorio público.



Markdown

<https://es.wikipedia.org/wiki/Markdown>

The screenshot shows the ReText application interface. The title bar reads "Markdown syntax.md — ReText". The menu bar includes File, Edit, Help. The toolbar contains icons for New document, Open, Save, Preview, Undo, Redo, Find, Replace, Tags, and Symbols. A status bar at the bottom right shows "25 : 14". The main window has two panes. The left pane displays the raw Markdown code:

```
1 ## Some Markdown syntax examples
2
3 Welcome to ReText!
4
5 Here are some examples of Markdown syntax that you can use with
6 ReText.
7 To learn more, look at the [official documentation](http://
daringfireball.net/projects/markdown/syntax).
8
9 ### Basic formatting
10
11 **Bold text** | *Text in italics* | [Link](https://github.com/
retext-project/retext)
12
13 ### Bullet list
14
15 * Item one
16 * Item two
17
18 ### Ordered list
19
20 1. Item one
21 2. Item two
22
23 You can also use <u>some</u> HTML tags in your documents.
24
25 Happy editing!!
```

The right pane shows the rendered Markdown content:

Some Markdown syntax examples

Welcome to ReText!

Here are some examples of Markdown syntax that you can use with ReText.

To learn more, look at the [official documentation](#).

Basic formatting

Bold text | **Text in italics** | [Link](#)

Bullet list

- Item one
- Item two

Ordered list

1. Item one
2. Item two

You can also use some HTML tags in your documents.

Happy editing!

Markdown en GitLab

jueves, 4 de enero de 2018 20:35



Motor diferente al que utiliza GitHub

- ▷ Newlines: dos saltos de linea (mínimo)
- ▷ Italic: underscore
- ▷ URL link: automático (con http, https, ftp)
- ▷ Quote multiline: entre >>>
- ▷ Code: con `código` o con ``` para multilínea (también ``php)
- ▷ Inline diff: con [+ additions] o [- additions]
- ▷ Emoji: con :emoji: (ejemplo :zap:)
- ▷ @username para mencionar
- ▷ #123 para issue
- ▷ !123 para merge request
- ▷ \$123 para snippet
- ▷ ~123 para label (id)
- ▷ [README](doc/README) referencia
- ▷ Task lists...

Lo vemos: <https://docs.gitlab.com/ee/user/markdown.html>

Fichero *GitIgnore*

martes, 26 de diciembre de 2017 21:02

Un archivo *gitignore* especifica intencionalmente archivos sin seguimiento (untracked) que git debe ignorar.

Los archivos ya rastreados (*tracked*) por Git no se ven afectados.

Debe estar situado en la carpeta raíz.

Para forzar que se incluya un fichero descartado en *gitignore* se puede usar
git add -f <ficheros>

Patrones

#: Comentarios
!: Negación del siguiente patrón
algo/: Directorio con el nombre indicado (no ficheros con ese nombre)
*: cualquier valor

.html , !foo.html, /.js

Ejemplos de *gitignore* para diversos lenguajes

<https://github.com/github/gitignore>

Se crea *gitignore* y se incorpora al repositorio

```
D:\desarrollo\Gits\Git_Basic [master +1 ~1 -0 !]> echo gitignore > .gitignore
D:\desarrollo\Gits\Git_Basic [master +1 ~1 -0 !]> code .
D:\desarrollo\Gits\Git_Basic [master +1 ~1 -0 !]> git status
```

```
On branch master
Untracked files:
  (use "git add <file>..." to include in what will be committed)

    .gitignore

nothing added to commit but untracked files present (use "git add" to track)
```

```
D:\desarrollo\Gits\Git_Basic [master +1 ~0 -0 ~]> git add .
D:\desarrollo\Gits\Git_Basic [master +1 ~0 -0 ~]> git commit -m "Creado gitignore"
[master cc925b5] Creado gitignore
 1 file changed, 0 insertions(+), 0 deletions(-)
 create mode 100644 .gitignore
D:\desarrollo\Gits\Git_Basic [master]> git status
On branch master
nothing to commit, working tree clean
```

```
1 # ejemplo de gitignore
2 *.class
3 node_modules/
```

Puede ser necesario crearlo con el editor de código

Commit del propio *.gitignore*

Funcionamiento de *gitignore*

```
D:\desarrollo\Gits\Git_Basic [master]> echo clase > clase.class
D:\desarrollo\Gits\Git_Basic [master]> dir

    Directorio: D:\desarrollo\Gits\Git_Basic

Mode           LastWriteTime       Length Name
----          -           -
d----- 04/01/2018  18:46            mapas
-a---- 05/01/2018  9:16           107 .gitignore
-a---- 05/01/2018  9:19            16 clase.class
-a---- 05/01/2018  8:31           168 index.js
-a---- 04/01/2018  18:40            18 README.md

D:\desarrollo\Gits\Git_Basic [master]> git status
On branch master
nothing to commit, working tree clean
```

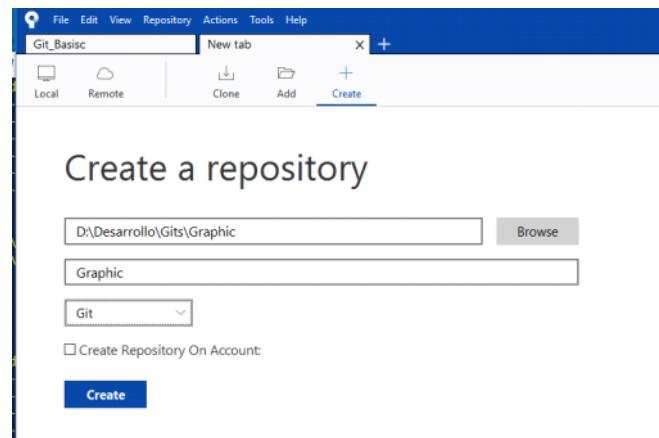
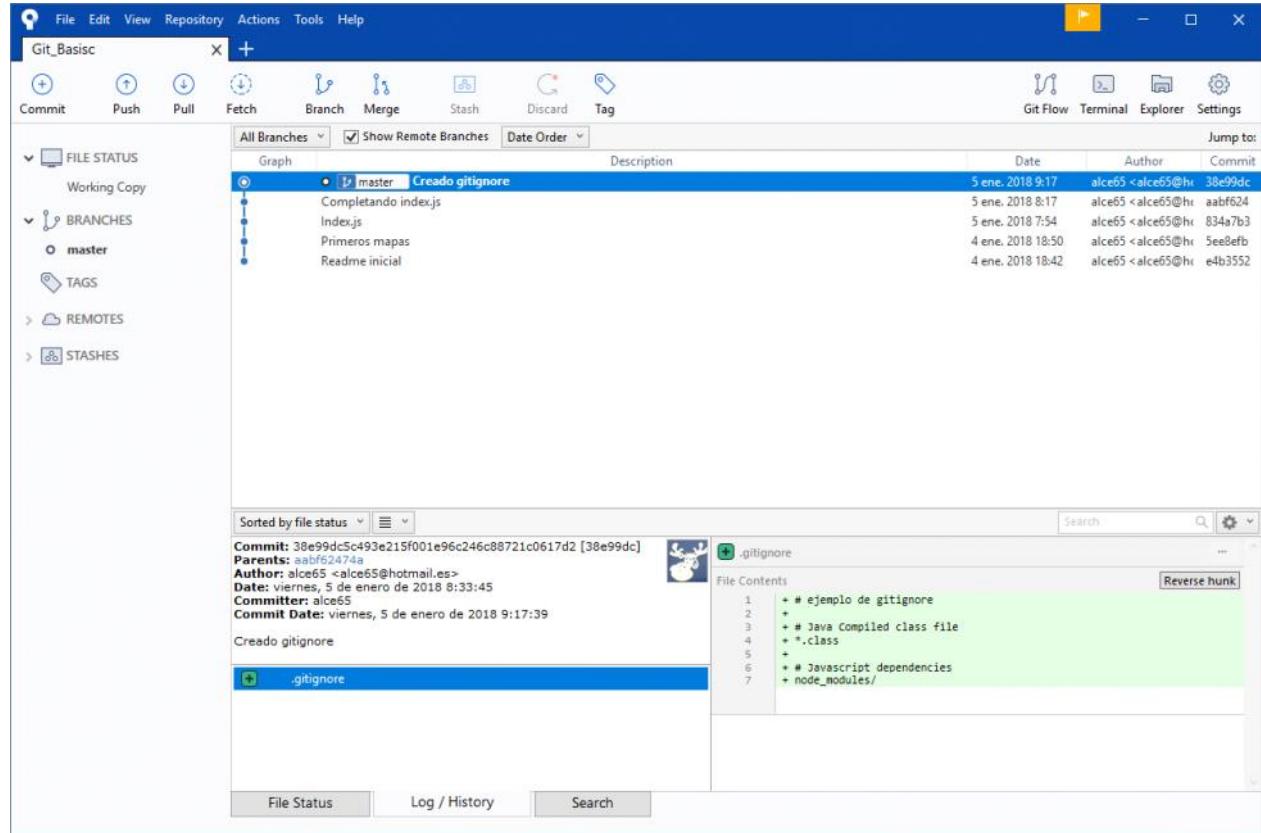
Se crea un fichero .class que no aparece como existente en la *working area* -> es completamente ignorado por el repositorio

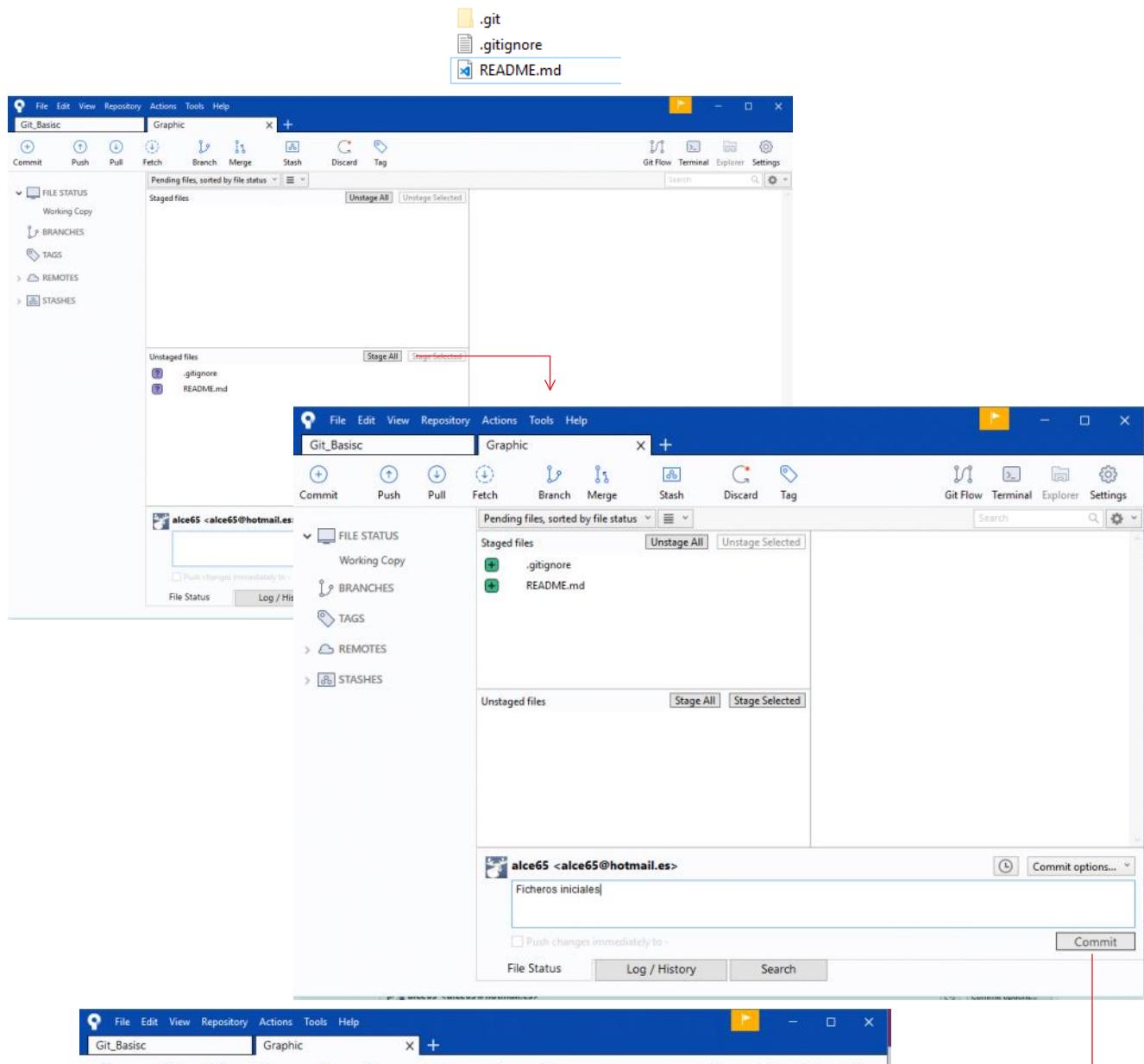
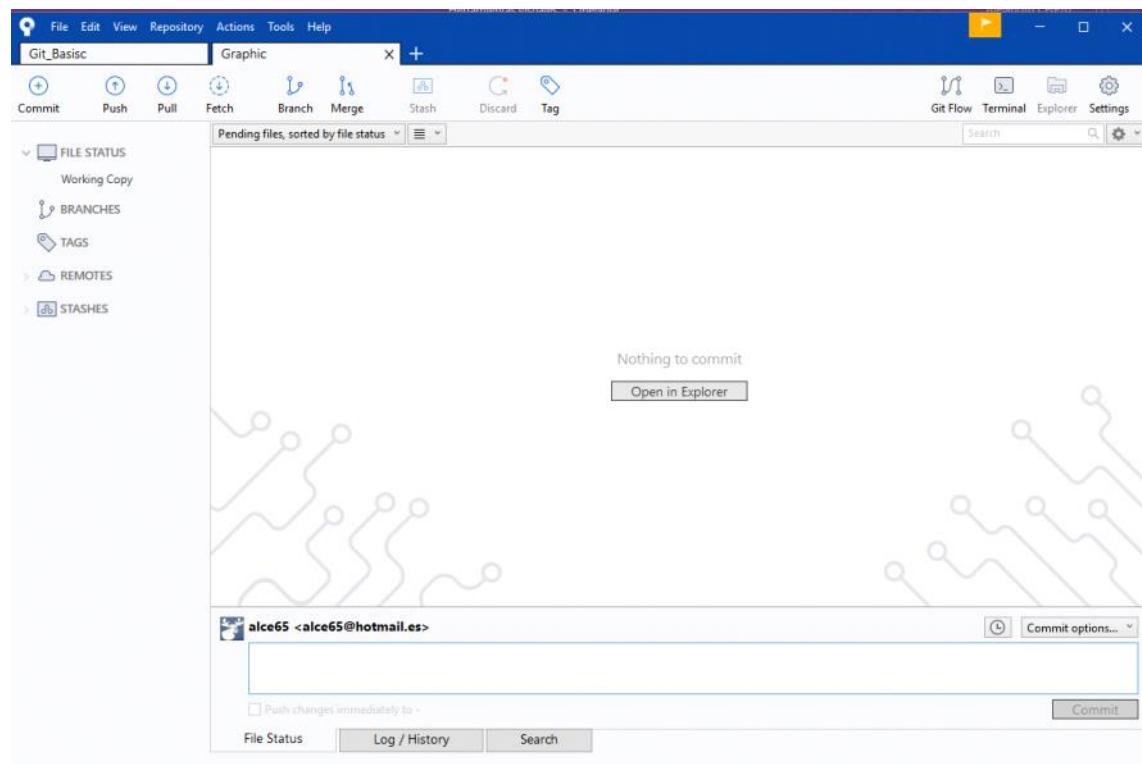
Herramientas visuales

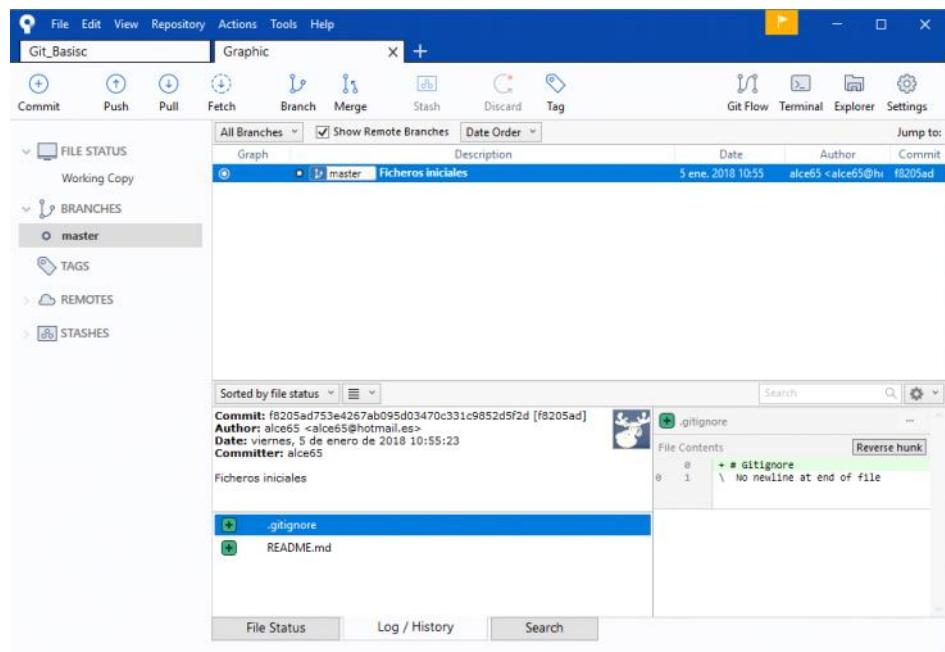
viernes, 5 de enero de 2018 10:41

Visualmente se pueden utilizar diversas herramientas, como *SourceTree*

<https://www.sourcetreeapp.com/>

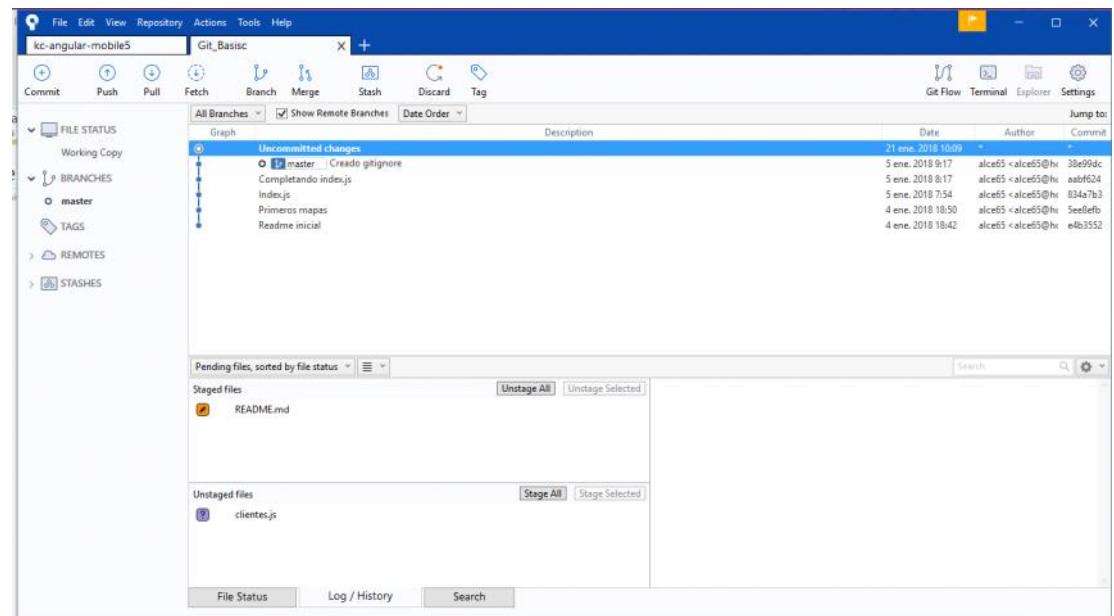






Ejercicio en SourceTree

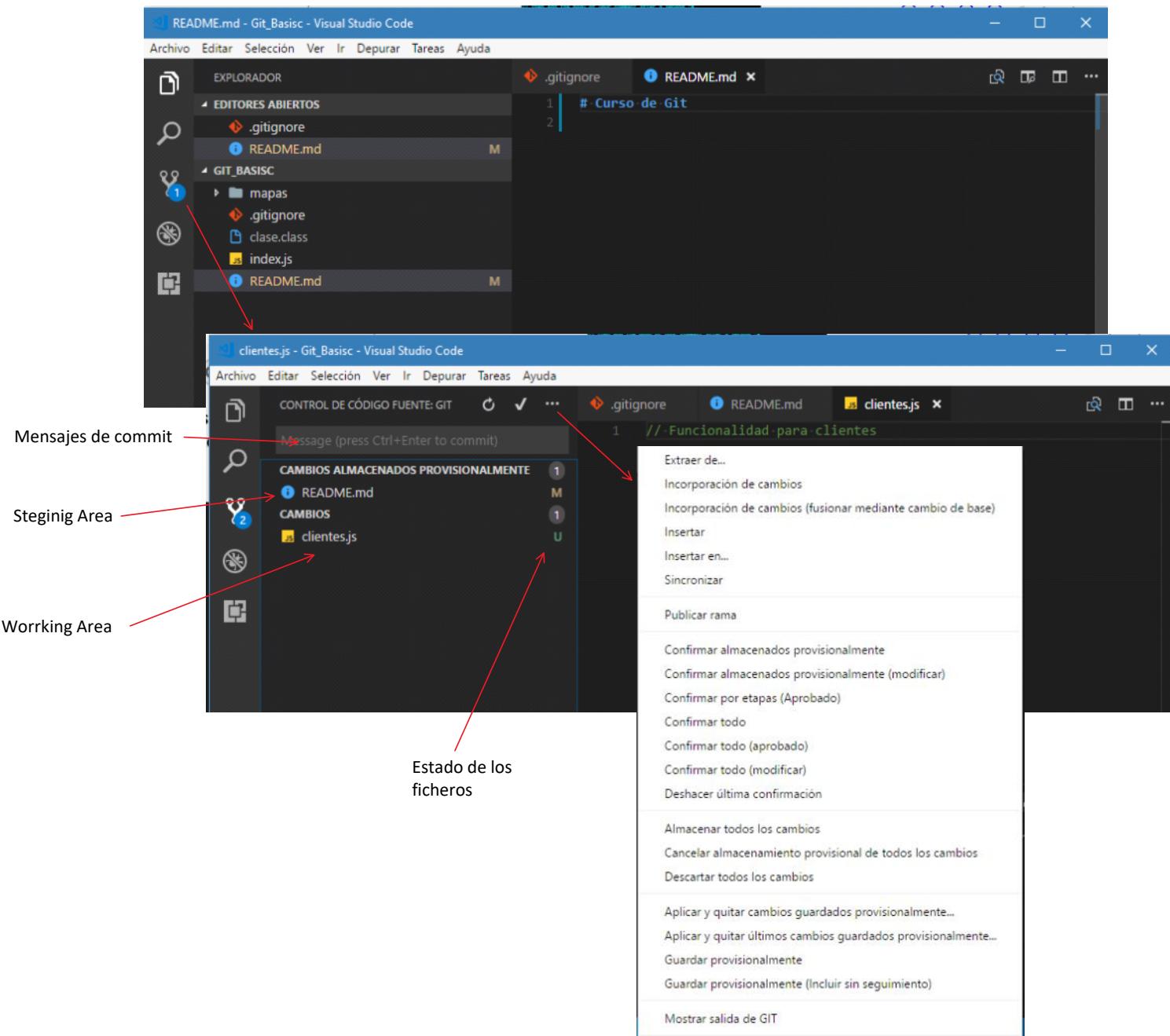
domingo, 21 de enero de 2018 9:54



Ejercicio en Visual Studio Code

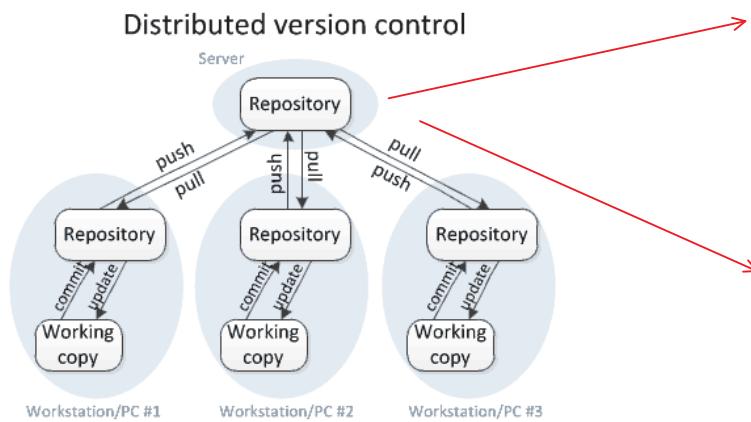
lunes, 8 de enero de 2018 23:31

Git Integrado



Introducción a GIT en el lado servidor

martes, 9 de enero de 2018 0:41



Servidores "propios"

- *gitolite* <http://gitolite.com/gitolite/index.html>
- *gitosis* <https://git-scm.com/book/es/v1/Git-en-un-servidor-Gitosis>
- *GitLab*, para Linux (interfaz Web,) <https://about.gitlab.com/equipos>

Hosting especializado de repositorios

- GitHub - <https://github.com/>
- Bitbucket <https://bitbucket.org/>
- GitLab <https://about.gitlab.com/>

• Repositorios *Bare*

miércoles, 3 de enero de 2018 13:37

Hay dos tipos diferentes de repositorios Git:

- *Bare* (desnudo, simple): no tiene directorio de trabajo. No se usa para el desarrollo directamente en el desarrollo de aplicaciones ya que en él no se pueden realizar directamente *commits*. De este tipo son los repositorios que se publican, en servidores estándar o especializados en el alojamiento (hosting) de repositorios
- Desarrollo: repositorio típico. Mantiene la rama en la que se trabaja, proporciona una copia extraída de dicha rama en el directorio de trabajo

Un repositorio *Bare* es crucial para el desarrollo colaborativo: otros desarrolladores clonian y recuperan de estos repositorios y los actualizan a partir de sus repositorios de desarrollo

Creación de un repositorio *Bare*

```
$ git init <repository> --bare
```

En el servidor donde se almacenan los repositorios *bare* son especialmente importantes algunos aspectos como

- la seguridad (por lo general basada en SSH)
- la gestión de las cuentas de usuarios

Aplicaciones

domingo, 21 de enero de 2018 12:21

gitolite

<http://gitolite.com/gitolite/index.html>

UNIX

The screenshot shows the official Gitolite website at <http://gitolite.com/gitolite/index.html>. The page title is "Hosting Git Repositories". The left sidebar contains links for "Hosting Git Repositories", "install/setup", "documentation", "TROUBLESHOOTING", "contact/support", "security issues", "mailing list(s)", "IRC", and "license". The main content area starts with a brief introduction: "Gitolite allows you to setup git hosting on a central server, with fine-grained access control and many more powerful features." Below this is a section titled "install/setup" with a note about source code availability and a link to a "quick install" page. A callout box provides a tip for users installing via package manager. The "documentation" section follows, with a note about self-explanatory sections and forward references. A sidebar on the right contains a message from April 2014 about a book on gitolite.

gitosis

<https://git-scm.com/book/es/v1/Git-en-un-servidor-Gitosis>

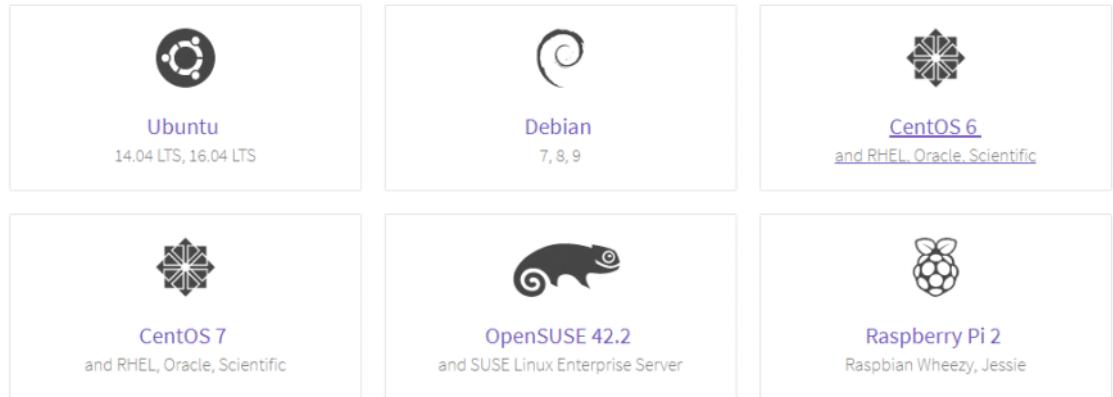
Es básicamente un conjunto de scripts que te ayudarán a gestionar el archivo 'authorized_keys', así como a implementar algunos controles de acceso simples.

instalación no demasiado simple
necesita de ciertas herramientas Python
se instala clonando el correspondiente repositorio Git

Versiones para diversas distribuciones Linux
Dotadas de un potente interfaz Web, como el del hosting de GitLab

<https://about.gitlab.com>

Omnibus package installation (recommended)



Github

miércoles, 27 de diciembre de 2017 11:12

GitHub



Ejemplo de hosting de repositorios Git
Puede actuar como remoto de cualquier repositorio Git

<https://github.com/>

Posibilidades del acceso anónimo

- Descarga / clonación de proyectos de software libre
- Perfil de los programadores (uso social / laboral)

Extensa documentación

<https://help.github.com/>

GitHub Help

Version ▾ Contact Support Return to GitHub

Sometimes you just need a little help.

How can we help?



Bootcamp

- › Set Up Git
- › Create A Repo
- › Fork A Repo
- › Be Social

Setup

- › Signing up for a new GitHub account
- › Verifying your email address
- › About commit email addresses
- › Setting your commit email address on GitHub
- › Setting your commit email address in Git
- › Blocking command line pushes that expose your personal email address
- › Setting your username in Git
- › Dealing with line endings
- › Supported browsers

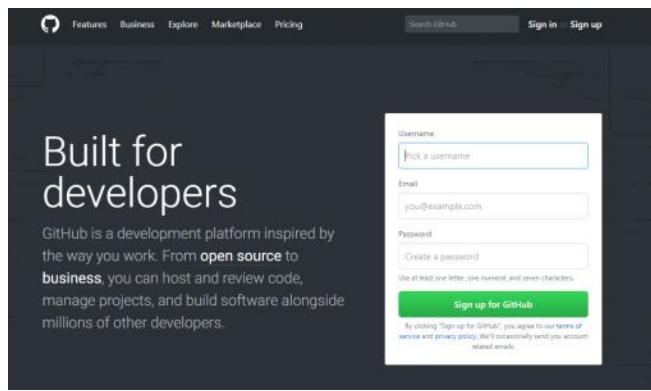
About GitHub

- › GitHub Glossary
- › Git and GitHub learning resources
- › Differences between user and organization accounts

Usuarios

domingo, 21 de enero de 2018 12:10

Alta de usuario



Perfil de usuario

GitHub Social

A screenshot of a GitHub user profile page for 'alce65'. The top navigation bar includes 'Search GitHub', 'Pull requests', 'Issues', 'Marketplace', 'Explore', and social icons. Below the header, there's a profile picture and the username 'alce65'. A red box highlights the 'Edit profile' button in the bio section, which contains the text 'Madrid, Spain' and an email address. Another red box highlights the 'Following' link in the top navigation bar, which shows the value '9'. The main content area displays 'Popular repositories' with cards for 'Curso_Malaga_2', 'Angular_IT', 'HTML_CSS', 'JS_Web', 'Curso_Malaga_1', and 'CursoJS'. At the bottom, a heatmap shows '388 contributions in the last year'.

Personal settings
Profile
Account
Emails
Notifications
Billing
SSH and GPG keys
Security
Blocked users
Repositories
Organizations
Saved replies
Applications
Developer settings
Organization settings
 Curso-Web

Public profile

Name

Public email

You can manage verified email addresses in your [email settings](#).

Bio

You can @mention other users and organizations to link to them.

URL

Company

You can @mention your company's GitHub organization to link it.

Location

Profile picture



[Upload new picture](#)

[Update profile](#)

Repositorios en GitHub

Lunes, 8 de enero de 2018 21:23

The screenshot shows the GitHub interface for creating a new repository. At the top, there are navigation links: Overview, **Repositories 33**, Stars 2, Followers 29, and Following 9. Below these are search fields for repositories, filters for Type: All and Language: All, and a green 'New' button. The main section is titled 'Create a new repository' with the sub-instruction: 'A repository contains all the files for your project, including the revision history.' A red arrow points from the first list item to the 'Repository name' input field, which is highlighted with a blue border. Another red arrow points from the second list item to the 'Create repository' button at the bottom.

Owner /

Repository name

Great repository names are short and memorable. Need inspiration? How about [bug-free-broccoli](#).

Description (optional)

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: **None** | Add a license: **None** | ⓘ

Create repository

- a. Solo se indica el nombre (y opcionalmente la descripción)

- b. Se Inicializa el repositorio con readme, gitignore y si procede información sobre la licencia

Enlace y clonado

Lunes, 8 de enero de 2018 21:11

Subir un repositorio a GitHub

Un **repositorio local**, después de ser creado puede ser **enlazado** con un **remoto vacío** (sin inicializar) para después por subirlo o "empujarlo" (*push*) siempre que sea necesario

```
git init  
...  
git remote add origin <url>  
...  
git push origin master
```

Las URL de los repositorios *bare* almacenados en GitHub terminan en .git
<https://github.com/alce65/GitRepo.git>

Añade una referencia al repositorio remoto identificada por un alias (*shorthand*), en este caso **origin**

Se puede definir cualquier número de remotos, cada uno con su alias o identificador propio

Podemos ver el resultado con el comando
git remote -v

Clonar un repositorio de GitHub

Un repositorio remoto, normalmente inicializado -> clonado a local

```
git clone <remote-url>
```

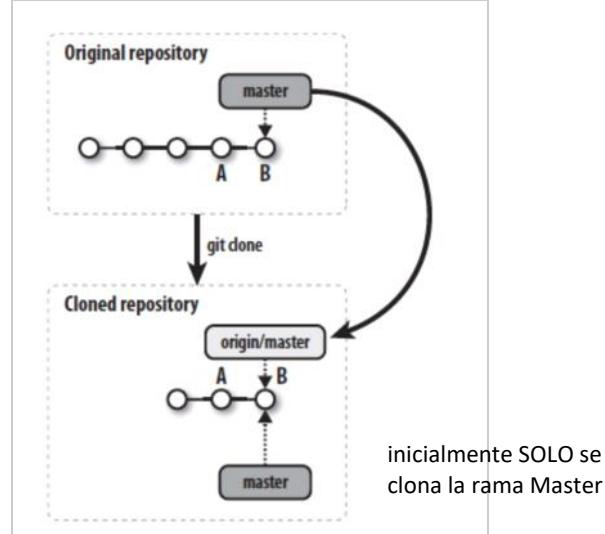
Crea el repositorio local con todo el contenido y le añade con el alias por defecto **origin** la url del remoto

El clonado se utiliza en 2 situaciones

- con nuestros propios repositorio
- con cualquier repositorio público

En este segundo caso, si no se dispone de permisos, no será posible subir nuevo contenido al remoto.

Por contra, para realizar esta operación ni siquiera es necesario tener sesión iniciada en GitHub ni tener una cuenta.



Un repositorio clonado queda vinculado al que lo origino:

- no reflejará automáticamente los cambios en el remoto
- podrá ser sincronizado mediante los comandos adecuados: *push - pull (fetch - merge)*

Ejercicios. Clonado

domingo, 21 de enero de 2018 20:07

Ejercicio.

- Creamos un remoto y lo enlazamos con el repositorio local que tenemos
- Creamos un repositorio en GitHub y lo clonamos en local

Comprobamos que el resultado es prácticamente el mismo

Desde Local a Remoto

- a. Creación del repositorio local
`git init`
- b. Primer *commit* para inicializarlo
`git add / git commit`
- c. Creación de remoto en GitLab vacío
- d. Conexión entre ambos
`git remote add origin <url>`
- e. Sincronización
`git push -u origin master`

Desde remoto a local

- a. Creación de un repositorio en GitHub añadiéndole un primer *commit* con los elementos iniciales
- b. Clonado del repositorio remoto creando así el repositorio local
`git clone <url>`

Forks

domingo, 21 de enero de 2018 12:09

Fork: clonado desde otra cuenta de GitHub a nuestro propio usuario de GitHub.

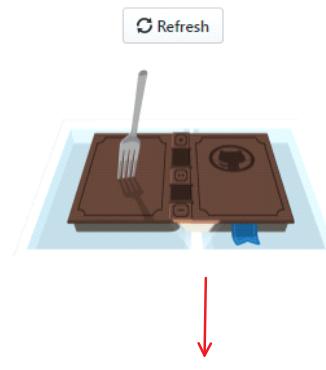
No añade nada en local

Puede hacerse desde cualquier repositorio público

This screenshot shows the GitHub repository page for 'angular/angular'. The top navigation bar includes 'This repository', 'Search', 'Pull requests', 'Issues', 'Marketplace', and 'Explore'. The main header displays the repository name 'angular / angular', a 'Watch' button (2,850), a 'Star' button (32,232), and a prominent 'Fork' button (7,957) which is highlighted with a red box. Below the header, there are tabs for 'Code', 'Issues 1,813', 'Pull requests 309', 'Projects 9', and 'Insights'. The repository description states 'One framework. Mobile & desktop. <https://angular.io>'. The repository stats show 9,352 commits, 30 branches, 196 releases, 570 contributors, and an MIT license. At the bottom, there are buttons for 'Branch: master', 'New pull request', 'Create new file', 'Upload files', 'Find file', and 'Clone or download'.

Forking miw-upm/IWVG

It should only take a few seconds.



This screenshot shows the GitHub repository page for 'alce65/IWVG', which was forked from 'miw-upm/IWVG'. The top navigation bar includes 'Unwatch' (1), 'Star' (0), and a 'Fork' button (58). The main header displays the repository name 'alce65 / IWVG' and the source repository 'forked from miw-upm/IWVG'. Below the header, there are tabs for 'Code', 'Pull requests 0', 'Projects 0', 'Wiki', 'Insights', and 'Settings'. A note says 'No description, website, or topics provided.' with an 'Edit' button. There is also an 'Add topics' link. The repository stats show 50 commits, 1 branch, 0 releases, and 1 contributor. At the bottom, there are buttons for 'Branch: master', 'New pull request', 'Create new file', 'Upload files', 'Find file', and 'Clone or download'. A note at the bottom of the page says 'This branch is even with miw-upm:master.' with links for 'Pull request' and 'Compare'. The commit history lists two entries: 'setillo interpreter' by 'aoo' (Latest commit b59dc11 on 8 Jul 2016) and 'interpreter' by 'doo' (2 years ago).

Actualizar remotos. Seguridad

domingo, 21 de enero de 2018 20:27

- Se puede clonar cualquier repositorio público
- Solo se pueden subir (*push*) actualizaciones a
 - los repositorios propios o
 - en los que se está autorizado como contribuidor

Para garantizar la autenticación, los accesos a los repositorios emplean uno de los siguientes protocolos seguros

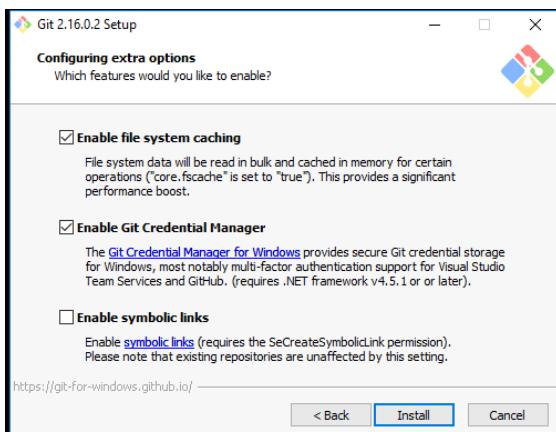
HTTPS:

- Habitual en entornos Windows
- Autenticación basada en usuario/clave

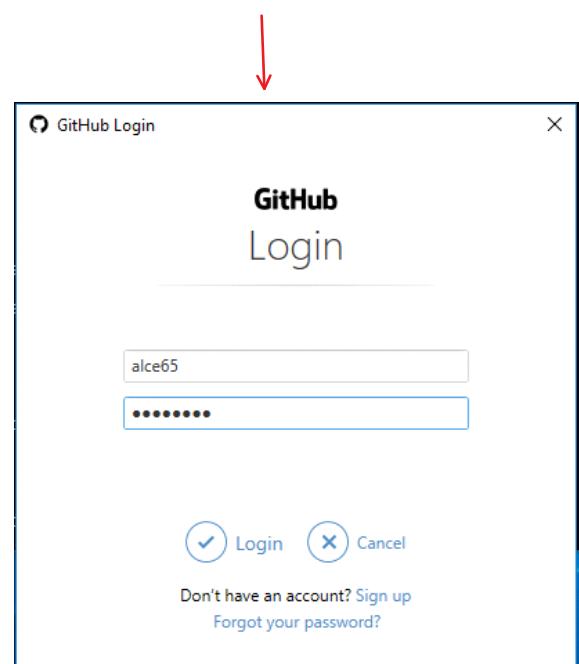
SSH (*Secure SHell*)

- Habitual en entornos UNIX/Linux
- Autenticación basada en diversas técnicas de cifrado, como la encriptación asimétrica y los hash

En la instalación



Al acceder por primera vez a una cuenta de GitHub para subir una actualización de cualquiera de sus repositorios

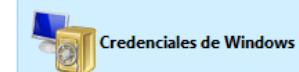


Panel de Control



Administrar credenciales

Vea y elimine su información de inicio de sesión guardada para sitios web, redes y aplicaciones conectadas.



[Copia de seguridad de credenciales](#) [Restaurar credenciales](#)

Panel de Control

Administrador de credenciales

Administrar credenciales

Vea y elimine su información de inicio de sesión guardada para sitios web, redes y aplicaciones conectadas.



Credenciales web



Credenciales de Windows

[Copia de seguridad de credenciales](#) [Restaurar credenciales](#)

Credenciales de Windows

[Agregar una credencial de Windows](#)

No hay credenciales de Windows.

Credenciales basadas en certificados

[Agregar una credencial basada en certificado](#)

No hay certificados.

Credenciales genéricas

[Agregar una credencial genérica](#)

MS.Outlook.15:alce65@outlook.com

Fecha de modificación: 21/11/2017

git:https://github.com

Fecha de modificación: Hoy

virtualapp/dilogical

Fecha de modificación: 21/11/2017

git:https://github.com

Fecha de modificación: Hoy

Dirección de red o Internet: git:https://github.com

Nombre de usuario: Personal Access Token

Contraseña:

Persistencia: Equipo local

[Editar](#) [Quitar](#)

Editar credencial genérica

Asegúrese de que el nombre de usuario y la contraseña que escriba se pueden usar para obtener acceso a la ubicación.

Dirección de red o Internet: git:https://github.com

Nombre de usuario:

Contraseña:

[Guardar](#)

[Cancelar](#)

Git Hub Social

domingo, 21 de enero de 2018 22:56

Actualizar remotos. Colaboraciones

domingo, 21 de enero de 2018 20:16

Se puede definir cualquier número de colaboradores autorizados a contribuir a un repositorio

The screenshot shows a GitHub repository page for 'alce65 / GitRepo'. At the top, there are buttons for 'Code', 'Issues 0', 'Pull requests 0', 'Projects 0', 'Wiki', 'Insights', and 'Settings'. To the right are buttons for 'Unwatch 1', 'Star 0', 'Fork 0', and 'Edit'. Below the header, the repository name is 'alce65 / GitRepo' and the description is 'Repositorio para prácticas con Git'. There is a 'Add topics' link. A red box highlights the '1 contributor' button in the top navigation bar. Below this, there are buttons for 'Branch: master ▾', 'New pull request', 'Create new file', 'Upload files', 'Find file', and a green 'Clone or download ▾' button. A file list shows 'index.html' with the commit message 'Creado index.html'. To the right, it says 'Latest commit 2fb236e on 28 Oct 2017 3 months ago'.

Información sobre las contribuciones

The screenshot shows the 'alce65 / GitRepo' repository's 'Insights' tab. On the left, a sidebar lists 'Pulse', 'Contributors', 'Community', 'Traffic', 'Commits', 'Code frequency', 'Dependency graph', 'Network', and 'Forks'. The main area displays a chart titled 'Oct 22, 2017 – Jan 21, 2018' showing 'Contributions to master, excluding merge commits'. A red arrow points from the 'Contributors' section of the sidebar to the 'Contributors' section of the Insights page. To the right, a 'Filter contributions' sidebar is open, showing options for 'Additions', 'Deletions', and 'Commits', with 'Commits' checked. Below the chart, a detailed view for a single commit is shown, with the commit message '1 commit 12 ++ 0 --' and a date range from 'Oct 29' to 'Jan 07'.

En la configuración, se pueden añadir contribuidores

The screenshot shows the GitHub repository settings page for 'alce65 / GitRepo'. The 'Collaborators' option in the sidebar is highlighted with a red box and a red arrow pointing down to the main content area. The main content area is titled 'Settings' and contains sections for 'Repository name' (GitRepo) and 'Features' (Wikis). Below this, the 'Collaborators' section is shown, which states 'This repository doesn't have any collaborators yet. Use the form below to add a collaborator.' It includes a search bar and an 'Add collaborator' button.

This repository doesn't have any collaborators yet. Use the form below to add a collaborator.

Search by username, full name or email address

You'll only be able to find a GitHub user by their email address if they've chosen to list it publicly. Otherwise, use their username instead.

Add collaborator

Ejercicio. Colaboración

domingo, 21 de enero de 2018 22:49

Grupos en GitHub

domingo, 7 de enero de 2018 20:02

Perfiles de personas

Equipos con determinados permisos

The screenshot shows the GitHub organization page for 'Escuelait'. At the top, there's a logo for 'escuelaIT' and a brief description: 'Esta es la escuela de la comunidad de DesarrolloWeb.com.'. Below the header, there are navigation links: 'Repositories' (highlighted with a red arrow), 'People 25', 'Teams 4' (also highlighted with a red arrow), and 'Settings'. There are also 'Filters' and a search bar ('Find a repository...'). A green button for '+ New repository' is visible. The main content area lists three repositories: 'Herramientas-Frontend-2015', 'HTML-CSS-2015', and 'Curso-angularjs-FS-2015'. To the right, there's a sidebar titled 'People' showing 25 members with their profile pictures. An 'Invite someone' button is at the bottom of the sidebar.

Herramientas-Frontend-2015
Repository del curso de Herramientas Frontend
Updated 16 days ago

HTML-CSS-2015
Repository con ejemplos y recursos del curso de HTML y CSS
<http://escuela.it/cursos/html-css/>
Updated on 2 Jul

Curso-angularjs-FS-2015
Updated on 23 Apr

People 25 >

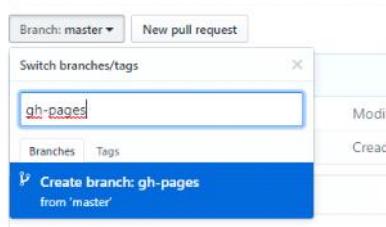
Profiles of 25 organization members.

Invite someone

Pages

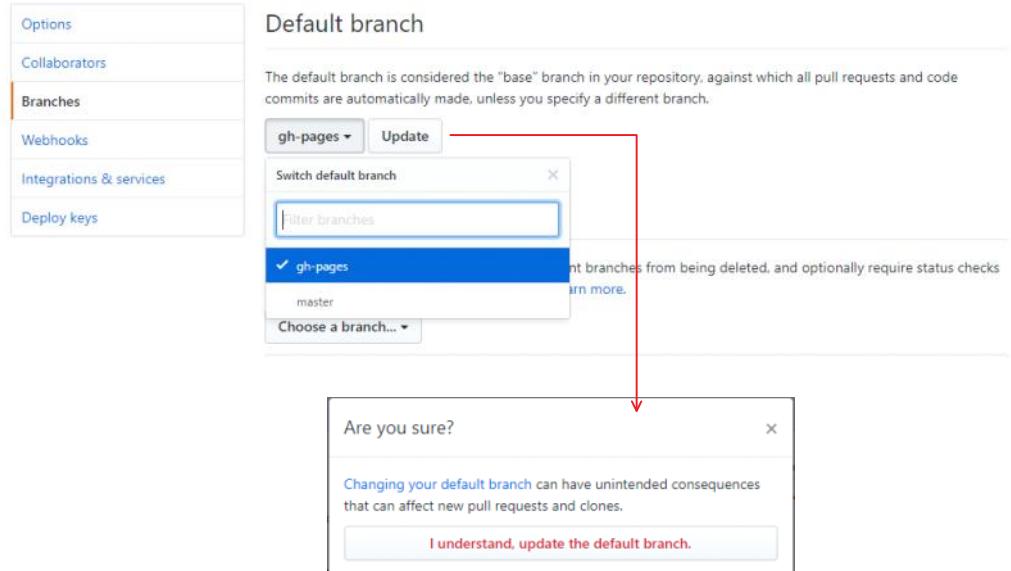
domingo, 21 de enero de 2018 22:52

En la página de nuestro repositorio vamos a dar clic en el botón Branch: master se nos desplegará un cuadro y dentro de la caja de texto escribimos gh-pages, de esta manera crearemos una nueva rama.



A continuación accederemos a la opción de settings y luego a branches.

Cambiaremos la rama por defecto "master" por "gh-pages", luego damos click en el botón "update" y aceptamos el mensaje que nos muestra.



The default branch is considered the "base" branch in your repository, against which all pull requests and code commits are automatically made, unless you specify a different branch.

Are you sure?

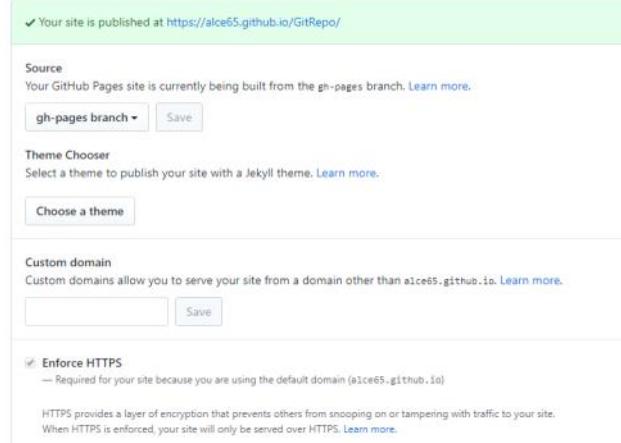
Changing your default branch can have unintended consequences that can affect new pull requests and clones.

I understand, update the default branch.

Regresamos a Options y bajamos hasta el cuadro llamado "GitHub Pages", ahí abrimos el enlace que se nos muestra.

GitHub Pages

GitHub Pages is designed to host your personal, organization, or project pages from a GitHub repository.



Your site is published at <https://alce65.github.io/GitRepo/>

Source
Your GitHub Pages site is currently being built from the gh-pages branch. Learn more.

gh-pages branch Save

Theme Chooser
Select a theme to publish your site with a Jekyll theme. Learn more.

Choose a theme

Custom domain
Custom domains allow you to serve your site from a domain other than alce65.github.io. Learn more.

Save

Enforce HTTPS
Required for your site because you are using the default domain (alce65.github.io)

HTTPS provides a layer of encryption that prevents others from snooping on or tampering with traffic to your site. When HTTPS is enforced, your site will only be served over HTTPS. Learn more.

<https://alce65.github.io/GitRepo/>



The screenshot shows a GitHub repository page for 'alce65 / GitRepo'. At the top, there's a navigation bar with links for 'Aplicaciones', 'Google', 'Tools', 'Redes Sociales', 'Referencia', 'Ocio', 'Tecnología', 'Proyectos', 'Work', 'Bookmarks', 'KeepCoding® Online', and 'Otros marcadores'. Below the navigation bar, the GitHub header includes 'This repository', 'Search', 'Pull requests', 'Issues', 'Marketplace', and 'Explore'.

The main content area displays the repository details:

- Repository name: alce65 / GitRepo
- Owner: alce65
- Commits: 4
- Branches: 1
- Releases: 0
- Contributors: 1

Below the summary, there are buttons for 'Branch: master', 'New pull request', 'Create new file', 'Upload files', 'Find file', and 'Clone or download'.

The file list shows the following entries:

- alce65 Modificado readme 2 (modified README.md)
- README.md (original README.md)
- index.html (original index.html)

A note indicates the latest commit was made 3 hours ago. The README.md file content is displayed below:

```
Repository prueba
Línea 1 Línea 2
```

Ejercicio. Git Pages

domingo, 21 de enero de 2018 23:31

GitLab

Lunes, 8 de enero de 2018 12:23

Qué es GitLab

- Git / Basado en la nube ->Control de código y revisión
- Integración continua (integra el antiguo GitLab CI)
- Deploy continuo
- Usuarios, Proyectos

Versiones de GitLab

- GitLab.com
- Servidores descargables
 - CE (free)
 - EE: Starter - Premium - Ultimate

- Proyectos y grupos. con plantillas, importaciones y... repositorios
- Grupos y subgrupos
- Actividad
- Milestones (hitos): metodologías ágiles
- Snippets ("gist")
- Chat Mattermost (chat tipo slack)
- IDE
- Ganttlab (diagramas)
- Nueva navegación (en 9.4 y otra en 10.0)
- Nuevo 10.0
 - Auto DevOps
 - Nueva navegación
 - Nueva forma de colaboración entre grupos
 - Resolver merge request antiguos de forma automática
 - Mejoras en subgrupos
 - API para la WIKI
 - Más "quick actions"

Preentación

GitLab Servers / GitLab.com

Grupos
Usuarios
Proyectos

Explorando proyectos

The screenshot shows the GitLab.com explore projects page. At the top, there are tabs for 'Projects', 'Groups', 'Activity', 'Milestones', and 'Snippets'. Below that, there are filters for 'Your projects', 'Starred projects', and 'Explore projects' (which is selected). There are also filters for 'Trending', 'Most stars', and 'All'. A search bar and a 'Last updated' dropdown are also present. The main area displays a list of trending projects:

- V Yale SDMP / Vesta**: A Ruby on Rails app to facilitate on-campus housing procedures. Developed for Yale's undergraduate housing process. Updated 2 days ago.
- T 김진우 / training-jwkim**: Creating and working with Matroska files. Updated about 7 hours ago.
- M Moritz Bunkus / MKVToolNix**: Creating and working with Matroska files. Updated 2 days ago.
- S eyeo / spec**: Functional specifications for all products at eyeo. Updated 2 days ago.
- F Kostyantyn Matlaiyev / Flights**: Updated a week ago.
- N NeoMutt Project / neomutt**: Mirror of the NeoMutt Project -- https://github.com/neomutt -- https://www.neomutt.org/. Updated a day ago.
- R Anthony Pesch / redream**: Work In Progress SEGA Dreamcast emulator. Updated a day ago.
- I python-devs / importlib_resources**: Design and implementation for a planned importlib.resources. Updated a week ago.

Ejemplo: el propio gitLab

The screenshot shows the GitLab.com explore projects page. The search bar contains 'gitlab-ce'. The 'Explore projects' tab is selected. The results show the 'GitLab.com / GitLab.com Support Tracker' and the 'GitLab.org / GitLab Community Edition' project, which is highlighted with a red box. The 'GitLab.org / GitLab Community Edition' project is described as an open source end-to-end software development platform with built-in version control, issue tracking, code review, CI/CD, and more. It has 117 stars and was updated 29 minutes ago.

Detalles
Actividad
Cycle Analytics

Repository

The screenshot shows the details page for the 'GitLab.org / GitLab Community Edition' repository. The sidebar on the left includes 'Overview', 'Details', 'Activity', 'Cycle Analytics', 'Repository', 'Issues' (9,646), 'Merge Requests' (490), 'CI / CD', 'Snippets', and 'Members'. The main content area shows the repository details for 'gitlab-ce'. It includes a star icon (3834), a fork icon (3,581), and a link to HTTPS://GITLAB.COM/GITLAB-ORG/G. The repository has 2.6 GB of files, 66,737 commits, 1,599 branches, and 661 tags. The 'Details' tab is selected, showing the last commit by Phil Hughes: 'Merge branch '38056-remove-unused-option' into 'master''. The commit was made 31 minutes ago. The repository has 117 stars and 3,834 forks. The 'Activity' tab shows a timeline of recent events, including a merge request from 'gitlab-ce' into 'master' and a merge branch '38056-remove-unused-option' into 'master'.

Pages

martes, 9 de enero de 2018 0:30

The screenshot shows the GitLab Pages interface. At the top, there's a navigation bar with a speech bubble icon and the text "GitLab pages". Below this, the main content area has a gradient background from orange to red. On the left side of the content area, there's a list of points:

- Igual que GitHub pages
- Puedes hacer una página estática: HTML, CSS y JS
- Puedes hacerla a mano o...
- Static pages generator: <https://gitlab.com/pages/>
- Puedes hacerlo también para tipo grupo o usuario
- PASOS:

1. Fork de uno de los repositorios
2. Quitar el fork relationship (settings → general → advanced)
3. Activar "shared runners" (settings → CI/CD)
4. Editar algún archivo → push → auto build
5. Edit → pages → i la URL !
6. (Opcional) rename a namespace.gitlab.io (namespace del group)
7. (Opcional) editar configuración → baseUrl

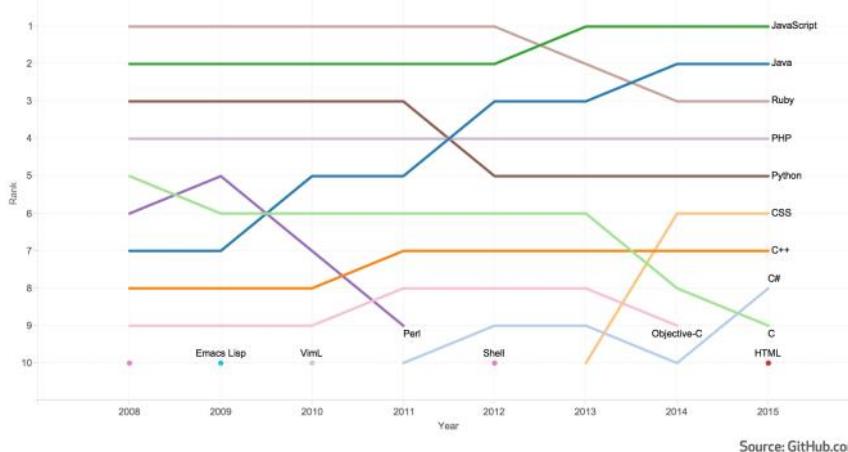
On the right side of the content area, there's a vertical sidebar with the text "Una rama para almacenar una página estática para el proyecto".

Clonado de proyectos

domingo, 21 de enero de 2018 10:19

Los repositorios de GitHub
son fundamentalmente
código

Rank of top languages on GitHub.com over time



El uso correcto de `gitignore` => los repositorios no deben contener nunca

- librerías de terceros, compiladas o no
- código compilado



Es habitual tener que "instalar" los proyectos después de clonarlos



Entran en juego herramientas específicas de cada lenguaje de programación

- `npm`
- `Maven`
- `gem`
- `Composer`

Un ejemplo con `npm`