



Gestión de proyectos con Git y GitHub: Modulo 3

1. <u>Introducción a Git y a sus comandos</u>	<u>2</u>
2. <u>El repositorio local y el directorio de trabajo: add, checkout, diff, init, log, mv, reset, rm, status y commit</u>	<u>7</u>
3. <u>Repositorios públicos en GitHub: new, push, import, fork y remote</u>	<u>26</u>

Juan Quemada, DIT - UPM



Git y GitHub

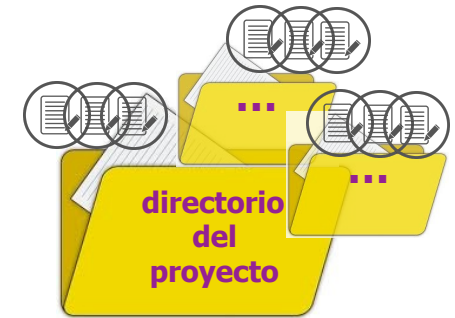
Introducción a Git y a sus comandos

Juan Quemada, DIT - UPM

Estructura del directorio de un proyecto

◆ Directorio de un proyecto:

- Contiene **todos** los **ficheros** y **subdirectorios** del proyecto
 - Estructura el contenido con convenios de nombres
- Algunos nombres muy habituales
 - **README.md** fichero resumen (formato GitHub markdown)
 - GitHub markdown: <https://github.com/github/markup/blob/master/README.md>
 - **LICENSE** fichero con licencia de distribución del proyecto
 - **public:** directorio donde almacenar **recursos Web**
 - **bin:** directorio donde almacenar **programas ejecutables**
 - **lib:** directorio con las **librerías** de software utilizadas
 - **test:** directorio con las pruebas de calidad del proyecto



◆ Herramientas de gestión del proyecto:

- Suelen utilizar **ficheros** y **subdirectorios** con nombres reservados, por ejemplo
 - **npm** (gestor de paquetes de Node.js)
 - Fichero **package.json**: guarda información del proyecto y los paquetes utilizados
 - Directorio **node_modules**: contiene las dependencias (paquetes) instaladas en el proyecto
 - **Git** (gestor de versiones)
 - Directorio **.git**: guarda el grafo de cambios (o de commits)
 - Fichero **.gitignore**: indica los ficheros a ignorar

•

Git



◆ Git es un gestor de **repositorios** de versiones software

- Desarrollado por Linus Torvalds en 2005 en código libre
 - Para soportar el desarrollo de Linux

◆ git es un comando de UNIX/Linux

- Documentación: <https://git-scm.com/documentation>
 - Tutorial: <https://www.atlassian.com/git/tutorials/setting-up-a-repository>
- Instalación de git o actualización a la última versión
 - Instrucciones de GitHub: <https://help.github.com/articles/set-up-git/>
 - Instrucciones git-scm: <https://git-scm.com/book/en/v2/Getting-Started-Installing-Git>
 - Tutorial Atlassian: <https://www.atlassian.com/git/tutorials/install-git>

\$ git --version # Indica la versión instalada. Si Git no está instalado, lo indicará.

\$ git --help # Equivale a git help y muestra lista de los comandos de git mas habituales.

```
venus:proy jq$
venus:proy jq$ 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>]
```

The most commonly used git commands are:

add	Add file contents to the index
bisect	Find by binary search the change that introduced a bug
branch	list, create, or delete branches

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



Git: manuales en línea y configuración

```
$ git init ... # git es un meta-comando, donde el primer parámetro (init) la operación solicitada  
               # Los parámetros add, bisect, branch, checkout,... invocan distintas operaciones.
```

```
$ git init --help # Equivale a git help init, ayuda del comando git init, igual para: add, bisect, ..
```

El comando **git config** configura git con las credenciales del desarrollador:

```
$ git config --global user.name "Pedro Ramirez"  
$ git config --global user.email pramirez@dit.upm.es
```

Consultar el valor de todas las opciones configuradas:

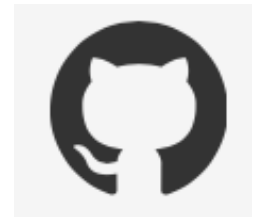
```
$ git config --list  
user.name=Pedro Ramirez  
user.email=pramirez@dit.upm.es  
color.ui=true  
.....
```

Consultar el valor de una opción:

```
$ git config user.name  
Pedro Ramirez  
$
```

El desarrollador debe **firmar** todos los **commits** que crea en la historia de un proyecto, porque Git es un software de colaboración. Por ello debe configurar sus credenciales antes de utilizar Git.

GitHub

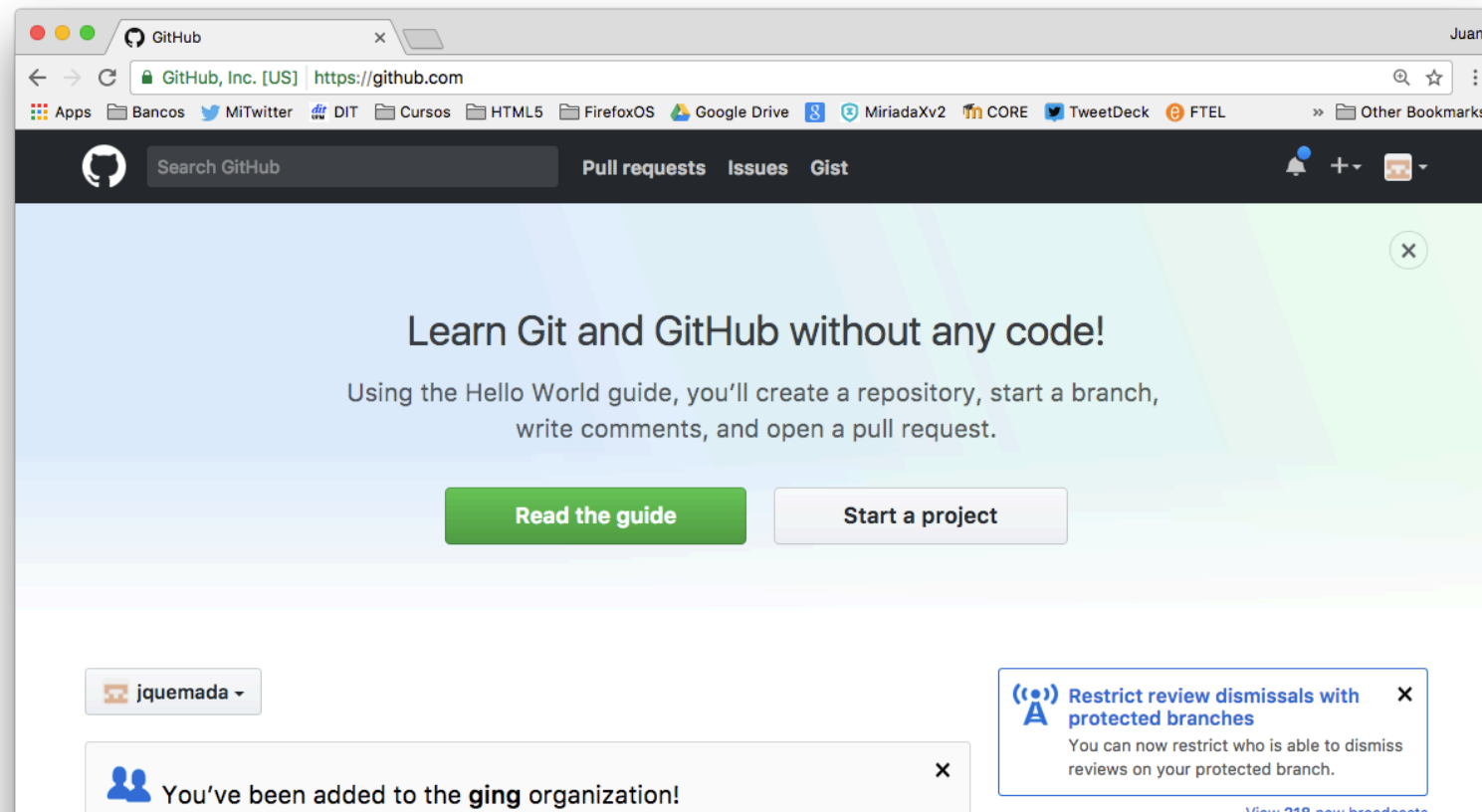


◆ Github

- Portal donde programadores comparten repositorios con proyectos Git
 - Nos da acceso a ellos a través del navegador Web y a través de Git

◆ Este curso requiere tener cuenta en GitHub: <https://github.com>

- Al crearla nos da instrucciones claras y precisas sobre uso de GitHub y Git



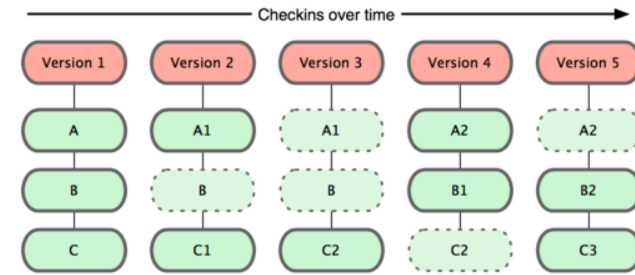


Git y GitHub

El repositorio local y el directorio de trabajo: add, checkout, diff, init, log, mv, reset, rm, status y commit

Juan Quemada, DIT - UPM

Proyecto Software



*S. Chacon, B. Straub: <https://git-scm.com/book/es/v1>

◆ Repositorio local

- Repositorio con un **directorio de trabajo asociado**, donde se trabaja en un proyecto y se guardan sus versiones (commits)

◆ Los proyectos software están activos durante largos periodos

- Durante su vida generan muchas versiones y variantes diferentes
 - Unas corrigen errores, otras añaden funcionalidad o adaptan nuevo hardware/software, etc.

◆ Commit o versión

- Instantánea del estado de los ficheros del proyecto, que puede restaurarse
 - Algunos commits se etiquetan con tags especiales de versión, p.e. v1, v1.3, ..

◆ Rama

- Secuencia de commits, ordenada por fechas, que soporta un desarrollo
 - Los nuevos commits se añaden al final de la rama activa (donde se desarrolla)
- La **rama principal** se denomina **master** y se crea con el primer commit

Directorio de trabajo y repositorio de commits



◆ Directorio de trabajo

- **Directorio** donde se crean las **versiones** del proyecto: código fuente, datos, ...
 - Se denomina también **área o espacio de trabajo** (workspace)
 - **árbol de trabajo** (working-tree) o **base de código** (codebase): contenido del direc. de trabajo (con sus subdir.)
 - También se denomina así a la información guardada en un commit.

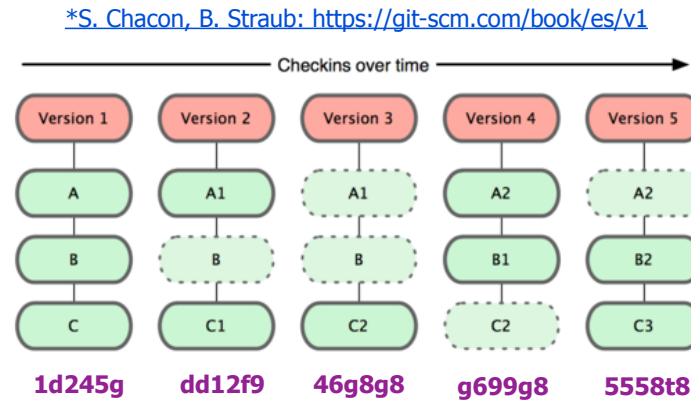
◆ El comando: **git init**

- Transforma un directorio en un **directorio de trabajo Git**
 - Añadiendo el **repositorio de commits** al directorio, lo que lo convierte en un directorio de trabajo Git
 - Lo añade en el **subdirectorio oculto .git** donde se guardarán los commits y sus relaciones

◆ Los comandos **git** son mas sencillos de invocar en el directorio de trabajo

- Por eso se suele entrar con "cd .." en el **directorio de trabajo Git** antes de ejecutar comandos Git

Nombre o identificador de commit



◆ git commit ... asigna un **identificador único** a cada nuevo commit

- El identificador actúa como nombre o referencia única del commit
 - Ningún otro commit en ningún otro repositorio poseerá el mismo identificador
 - Garantiza la integridad del commit: igualdad de identificadores implica igualdad de commits

◆ Nombre o identificador de commit

- Número hexadecimal de 40 dígitos generado como clave de hash SHA1
 - Ejemplo de identificador: **973751d2**1c4a71f13a2e729ccf77f3a960885682

◆ Se suele utilizar el **formato corto** (formato largo es incómodo)

- 7-8 dígitos iniciales (únicos en un proyecto): **973751d2**
 - Los comandos git permiten **identificadores cortos o largos**

Índice

◆ Índice (staging area, index)

- **Registro de cambios** del directorio de trabajo a incluir en el próximo commit
 - Los cambios no registrados en el índice no se incluyen al generar un nuevo commit
- Los ficheros **no modificados** del commit anterior permanecen en el siguiente commit

◆ git add ...

- registra en el índice los ficheros indicados
 - `git add .` registra en el índice todos los ficheros **nuevos** o **modificados**
 - `git add LICENSE README.md` registra los ficheros **LICENSE README.md** en el índice

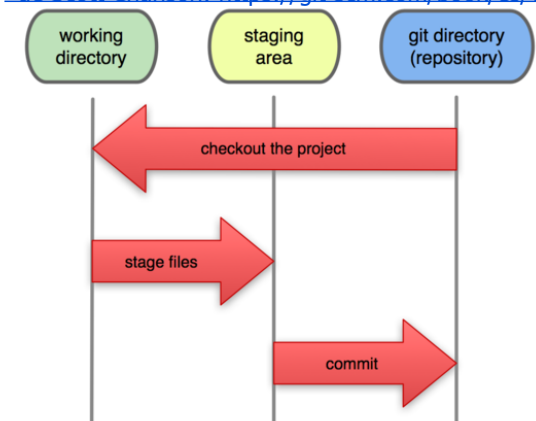
◆ git reset ...

- extrae ficheros del índice (deshace **git add ..**)
 - `git reset .` extrae del índice todos los ficheros
 - `git reset LICENSE` extrae el fichero LICENSE del índice

◆ git commit ...

- Genera un nuevo commit con lo registrado en el índice
 - `git commit -m "Descripción"` guarda nuevo commit con mensaje o título **"Descripción"**
 - `git commit` guarda nuevo commit y abre un editor para crear mensaje del commit
 - `git commit --amend -m "...."` modifica último commit con lo registrado en índice **!OJO cambia id de commit!**

*de Scott Chanson: <https://git-scm.com/book/es/v1>



Estado del directorio de trabajo: git status

♦ **git status** muestra el **estado** de los ficheros del **directorio de trabajo** respecto al **commit anterior**

♦ Estado de los ficheros

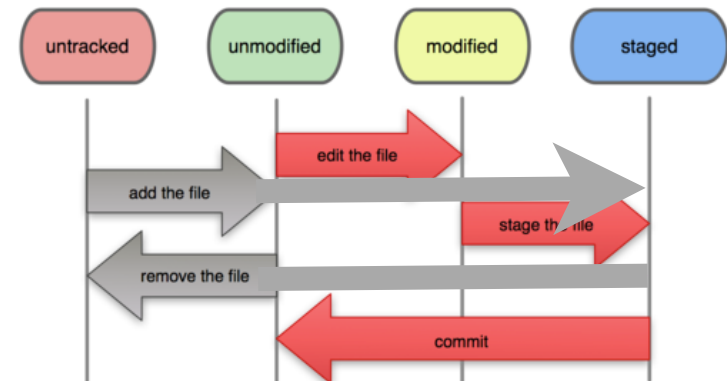
- **modified:** modificados respecto al commit anterior
- **untracked:** no existentes en el commit anterior
- **staged:** registrados para el próximo commit
-

♦ Estado de los ficheros

- **git status**
 - Muestra estado con detalle
- **git status -s**
 - Muestra estado en formato compacto
-

♦ Manual de referencia:

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



*de Scott Chanson: <https://git-scm.com/book/es/v1>

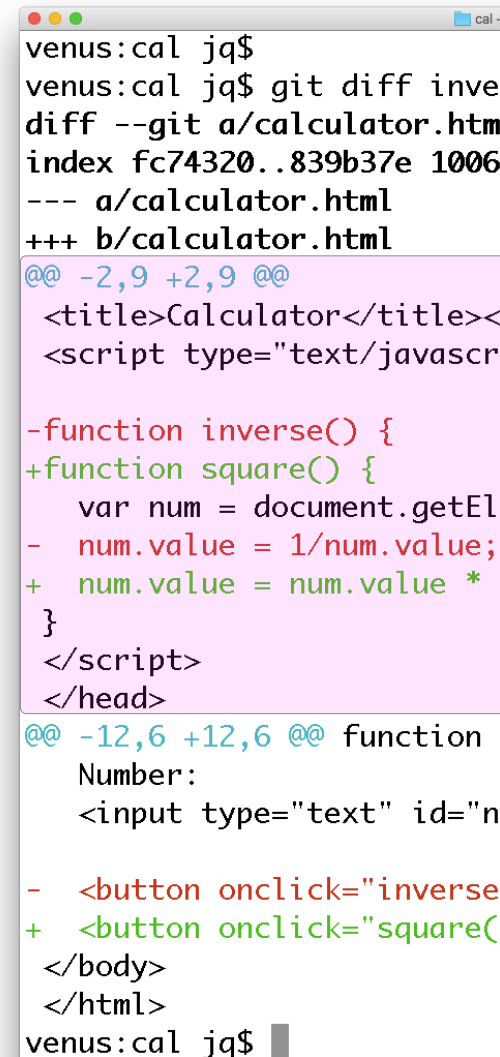
Diferencias con commit anterior: git diff

◆ Cada commit se genera por modificación de un commit anterior

- **git diff** muestra **diferencias** con el **commit anterior** o **entre dos commits cualquiera**
 - **Contexto:** se muestra en negro
 - **Líneas añadidas:** en verde y comienzan por +
 - **Líneas eliminadas:** en rojo y comienzan por -

◆ Algunos usos típicos del comando **git diff**:

- Mostrar cambios en ficheros **modified** respecto a commit anterior
 - **git diff** -> muestra cambios de los ficheros **modified**
 - **git diff file1, file2, ...** -> muestra cambios (si hubiere) de file1, file2, ...
- Mostrar cambios en ficheros **staged** respecto a commit anterior
 - **git diff --cache** -> muestra cambios de los ficheros **staged**
 - **git diff --cached file1, file2, ...** -> muestra cambios (si hubiere) de file1, file2, ...
- Mostrar cambios entre **2 commits**
 - **git diff 97d75 master** -> muestra cambios entre **97d75** y 1er commit de master
 - **git diff 97d75 master -- fich1** -> cambios en fich1 entre **97d75** y 1er com. master
- Algunas opciones de interés
 - **-b** -> Comparación no considera blancos, ni líneas vacías
 - **--name-status** -> Muestra solo nombres de ficheros y su estado (sin diferencias)
 - **--name-only** -> Muestra solo nombres de ficheros (sin diferencias)
 - **--stat** o **--numstat** -> Muestra estadísticas de cambios (sin diferencias)
 - **--unified=2000** -> Muestra 200 líneas de contexto (probablemente todo el fichero)
- Manual de referencia: <https://git-scm.com/docs/git-diff>



```
venus:cal jq$
venus:cal jq$ git diff inve
diff --git a/calculator.htm
index fc74320..839b37e 1006
--- a/calculator.html
+++ b/calculator.html
@@ -2,9 +2,9 @@
<title>Calculator</title><
<script type="text/javascr

-function inverse() {
+function square() {
    var num = document.getEl
-   num.value = 1/num.value;
+   num.value = num.value *
}
</script>
</head>
@@ -12,6 +12,6 @@ function
Number:
<input type="text" id="n

- <button onclick="inverse
+ <button onclick="square(
</body>
</html>
venus:cal jq$
```

Más comandos

◆ **git mv old_file new_file**

- Cambia el nombre de un fichero en el directorio de trabajo (y en el índice)
 - **git mv file1.js file2.js** cambia el nombre de file1.js a file2.js en el directorio de trabajo y en el índice

◆ **git rm file1, file2, ...**

- Borra los ficheros indicados del directorio de trabajo y registra lo borrado en el índice
 - **git rm file1.js file2.js** borra file1.js y file2.js del directorio de trabajo y del índice

◆ **git rm --cached file1, file2, ...**

- Borra los ficheros indicados del índice, pasan de staged a untracked
 - **git rm --cached file1.js file2.js** borra file1.js y file2.js solo del índice

◆ **git checkout file1, file2, ...**

- Elimina cambios de file1, file2, ... que pasan a **unmodified (Peligro! Cambios se pierden)**
 - **git checkout file1.js** elimina los cambios del fichero modified **file.js**

◆ **git checkout .**

- Elimina los cambios de todos los ficheros modified del directorio de trabajo que pasan a **unmodified (Peligro! Cambios se pierden)**
 - **git checkout .** elimina cambios en todos los ficheros modified del directorio de trabajo

stash

◆ **git stash [<name>]**

- Guarda en una pila las modificaciones del directorio de trabajo y el índice,
- deja restaurados el directorio de trabajo y el índice, por ejemplo
 - **git stash** guarda las modificaciones en la pila

◆ **git stash list**

- Lista el contenido de la pila de stashed p.e.
 - **git stash**

◆ **git stash apply [<name>] [<options>]**

- Aplica los cambios del último stash guardado, o los del stash llamado **name**,
- a los ficheros del area de trabajo,
- y no actualiza el índice, excepto si se usa la opción **—index**,
- y no elimina el stash aplicado de la pila. Por ejemplo:
 - **git stash apply**

◆ **git stash drop [<name>]**

- Elimina el último stash de la pila (o el indicado por **name**), por ejemplo
 - **git stash drop**

◆ **git stash pop [<name>]**

- Aplica el último stash (o el indicado por **name**) y lo elimina de la pila, por ejemplo
 - **git stash pop**

Crear directorio del S.O.

```
venus:proy jq$  
venus:proy jq$ mkdir cal  
venus:proy jq$ cd cal  
venus:cal jq$  
venus:cal jq$ git init  
Initialized empty Git repository in /Users/jq/proy/cal/.git/  
venus:cal jq$  
venus:cal jq$ git status -s  
venus:cal jq$  
venus:cal jq$ # Edit README.md & LICENSE  
venus:cal jq$  
venus:cal jq$ git status -s  
?? LICENSE  
?? README.md  
venus:cal jq$  
venus:cal jq$ git add .  
venus:cal jq$  
venus:cal jq$ git status -s  
A LICENSE  
A README.md  
venus:cal jq$  
venus:cal jq$ git commit -m "Readme & License"  
[master (root-commit) 1096247] Readme & License  
2 files changed, 23 insertions(+)  
create mode 100644 LICENSE  
create mode 100644 README.md  
venus:cal jq$  
venus:cal jq$ git log --oneline  
1096247 Readme & License  
venus:cal jq$
```

Crea el **directorio cal** y entra en él.



Transformar en directorio de trabajo Git

```
venus:proy jq$  
venus:proy jq$ mkdir cal  
venus:proy jq$ cd cal  
venus:cal jq$  
venus:cal jq$ git init  
Initialized empty Git repository in /Users/jq/proy/cal/.git/  
venus:cal jq$  
venus:cal jq$ git status -s  
venus:cal jq$  
venus:cal jq$ # Edit README.md & LICENSE  
venus:cal jq$  
venus:cal jq$ git status -s  
?? LICENSE  
?? README.md  
venus:cal jq$  
venus:cal jq$ git add .  
venus:cal jq$  
venus:cal jq$ git status -s  
A LICENSE  
A README.md  
venus:cal jq$  
venus:cal jq$ git commit -m "Readme & License"  
[master (root-commit) 1096247] Readme & License  
2 files changed, 23 insertions(+)  
create mode 100644 LICENSE  
create mode 100644 README.md  
venus:cal jq$  
venus:cal jq$ git log --oneline  
1096247 Readme & License  
venus:cal jq$
```

Transforma el directorio **cal** en un directorio de trabajo Git con su repositorio en **.git**

directorio de trabajo Git



Muestra directorio de trabajo limpio (sin ningún cambio).

Crear ficheros del commit

```
venus:proy jq$
venus:proy jq$ mkdir cal
venus:proy jq$ cd cal
venus:cal jq$
venus:cal jq$ git init
Initialized empty Git repository in /Users/jq/proy/cal/.git/
venus:cal jq$
venus:cal jq$ git status -s
venus:cal jq$
venus:cal jq$ # Edit README.md & LICENSE
venus:cal jq$
venus:cal jq$ git status -s
?? LICENSE
?? README.md
venus:cal jq$
venus:cal jq$ git add .
venus:cal jq$
venus:cal jq$ git status -s
A LICENSE
A README.md
venus:cal jq$
venus:cal jq$ git commit -m "Readme & License"
[master (root-commit) 1096247] Readme & License
 2 files changed, 23 insertions(+)
 create mode 100644 LICENSE
 create mode 100644 README.md
venus:cal jq$
venus:cal jq$ git log --oneline
1096247 Readme & License
venus:cal jq$
```

Llevar estos 2 ficheros al directorio de trabajo. Copiarlos o editarlos: vi, vim, sublime-text, Webstorm, Visual Studio Code, Atom,

directorio de trabajo Git



```
# cal
Educational Git project. Creates a simple calculator in
HTML and JavaScript in short steps.
```

MIT License

Copyright (c) 2016 Juan Quemada

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

Mostrar estado del directorio de trabajo

```
venus:proy jq$  
venus:proy jq$ mkdir cal  
venus:proy jq$ cd cal  
venus:cal jq$  
venus:cal jq$ git init  
Initialized empty Git repository in /Users/jq/proy/cal/.git/  
venus:cal jq$  
venus:cal jq$ git status -s  
venus:cal jq$  
venus:cal jq$ # Edit README.md & LICENSE  
venus:cal jq$  
venus:cal jq$ git status -s  
?? LICENSE  
?? README.md  
venus:cal jq$  
venus:cal jq$ git add .  
venus:cal jq$  
venus:cal jq$ git status -s  
A LICENSE  
A README.md  
venus:cal jq$  
venus:cal jq$ git commit -m "Readme & License"  
[master (root-commit) 1096247] Readme & License  
2 files changed, 23 insertions(+)  
create mode 100644 LICENSE  
create mode 100644 README.md  
venus:cal jq$  
venus:cal jq$ git log --oneline  
1096247 Readme & License  
venus:cal jq$
```

Muestra los 2 nuevos ficheros
todavía sin registrar en el índice.

directorio de
trabajo Git



Registrar nuevos ficheros en el índice

```
venus:proy jq$  
venus:proy jq$ mkdir cal  
venus:proy jq$ cd cal  
venus:cal jq$  
venus:cal jq$ git init  
Initialized empty Git repository in /Users/jq/proy/cal/.git/  
venus:cal jq$  
venus:cal jq$ git status -s  
venus:cal jq$  
venus:cal jq$ # Edit README.md & LICENSE  
venus:cal jq$  
venus:cal jq$ git status -s  
?? LICENSE  
?? README.md  
venus:cal jq$  
venus:cal jq$ git add .  
venus:cal jq$  
venus:cal jq$ git status -s  
A LICENSE  
A README.md  
venus:cal jq$  
venus:cal jq$ git commit -m "Readme & License"  
[master (root-commit) 1096247] Readme & License  
2 files changed, 23 insertions(+)  
create mode 100644 LICENSE  
create mode 100644 README.md  
venus:cal jq$  
venus:cal jq$ git log --oneline  
1096247 Readme & License  
venus:cal jq$
```

Registra ficheros en el índice

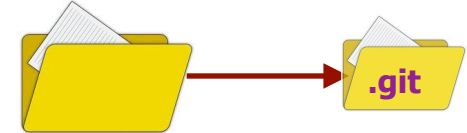
directorio de
trabajo Git



Mostrar estado del directorio de trabajo

```
venus:proy jq$  
venus:proy jq$ mkdir cal  
venus:proy jq$ cd cal  
venus:cal jq$  
venus:cal jq$ git init  
Initialized empty Git repository in /Users/jq/proy/cal/.git/  
venus:cal jq$  
venus:cal jq$ git status -s  
venus:cal jq$  
venus:cal jq$ # Edit README.md & LICENSE  
venus:cal jq$  
venus:cal jq$ git status -s  
?? LICENSE  
?? README.md  
venus:cal jq$  
venus:cal jq$ git add .  
venus:cal jq$  
venus:cal jq$ git status -s  
A LICENSE  
A README.md  
venus:cal jq$  
venus:cal jq$ git commit -m "Readme & License"  
[master (root-commit) 1096247] Readme & License  
2 files changed, 23 insertions(+)  
create mode 100644 LICENSE  
create mode 100644 README.md  
venus:cal jq$  
venus:cal jq$ git log --oneline  
1096247 Readme & License  
venus:cal jq$
```

directorio de
trabajo Git



Muestra los ficheros ya registrados. Estos se incluirán en el próximo commit.

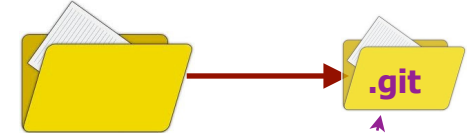
Crear nuevo commit y ver historia

```
venus:proy jq$  
venus:proy jq$ mkdir cal  
venus:proy jq$ cd cal  
venus:cal jq$  
venus:cal jq$ git init  
Initialized empty Git repository in /Users/jq/proy/cal/.git/  
venus:cal jq$  
venus:cal jq$ git status -s  
venus:cal jq$  
venus:cal jq$ # Edit README.md & LICENSE  
venus:cal jq$  
venus:cal jq$ git status -s  
?? LICENSE  
?? README.md  
venus:cal jq$  
venus:cal jq$ git add .  
venus:cal jq$  
venus:cal jq$ git status -s  
A LICENSE  
A README.md  
venus:cal jq$  
venus:cal jq$ git commit -m "Readme & License"  
[master (root-commit) 1096247] Readme & License  
2 files changed, 23 insertions(+)  
create mode 100644 LICENSE  
create mode 100644 README.md  
venus:cal jq$  
venus:cal jq$ git log --oneline  
1096247 Readme & License  
venus:cal jq$
```

Genera nuevo commit.

Muestra nuevo commit
en formato 1 línea

directorio de
trabajo Git



master

Readme & License

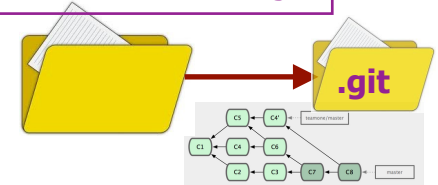
Inspeccionar historia y área de trabajo

```
venus:cal jq$  
venus:cal jq$  
venus:cal jq$ git log --oneline  
1096247 Readme & License  
venus:cal jq$  
venus:cal jq$ git status -s  
venus:cal jq$  
venus:cal jq$ # Edit calculator.html  
venus:cal jq$  
venus:cal jq$ git status -s  
?? calculator.html  
venus:cal jq$  
venus:cal jq$ git add calculator.html  
venus:cal jq$  
venus:cal jq$ git commit -m "x^2 button"  
[master b0e63ad] x^2 button  
1 file changed, 17 insertions(+)  
create mode 100644 calculator.html  
venus:cal jq$  
venus:cal jq$ git log --oneline  
b0e63ad x^2 button  
1096247 Readme & License  
venus:cal jq$
```

Muestra el primer commit ya generado.

Muestra el directorio de trabajo Git limpio.

directorio de trabajo



master

Readme & License

Añadir nuevo fichero

```
venus:cal jq$  
venus:cal jq$  
venus:cal jq$ git log --oneline  
1096247 Readme & License  
venus:cal jq$  
venus:cal jq$ git status -s  
venus:cal jq$  
venus:cal jq$ # Edit calculator.html  
venus:cal jq$  
venus:cal jq$ git status -s  
?? calculator.html  
venus:cal jq$  
venus:cal jq$ git add calculator.html  
venus:cal jq$  
venus:cal jq$ git commit -m "x^2 button"  
[master b0e63ad] x^2 button  
1 file changed, 17 insertions(+)  
create mode 100644 calculator.html  
venus:cal jq$  
venus:cal jq$ git log --oneline  
b0e63ad x^2 button  
1096247 Readme & License  
venus:cal jq$
```

Se añade el fichero **calculator.html** al directorio de trabajo con un editor, copiando (cp), ..

```
<!DOCTYPE html><html><head>  
<title>Calculator</title><meta charset="utf-8">  
<script type="text/javascript">  
  
function square() {  
    var num = document.getElementById("n1");  
    num.value = num.value * num.value;  
}  
</script>  
</head>  
<body>  
    Number:  
    <input type="text" id="n1"><p>  
  
    <button onclick="square()"> x<sup>2</sup> </button>  
</body>  
</html>
```

Number: 2

x^2

directorio de trabajo

Muestra el fichero **calculator.html** en el directorio de trabajo todavía sin registrar.

master

Readme & License

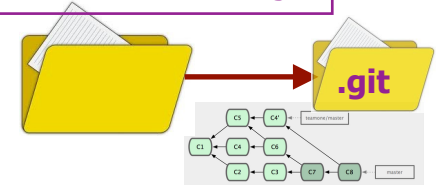
Añadir un nuevo commit a master



```
venus:cal jq$
venus:cal jq$
venus:cal jq$ git log --oneline
1096247 Readme & License
venus:cal jq$
venus:cal jq$ git status -s
venus:cal jq$ # Edit calculator.html
venus:cal jq$
venus:cal jq$ git status -s
?? calculator.html
venus:cal jq$
venus:cal jq$ git add calculator.html
venus:cal jq$
venus:cal jq$ git commit -m "x^2 button"
[master b0e63ad] x^2 button
1 file changed, 17 insertions(+)
create mode 100644 calculator.html
venus:cal jq$
venus:cal jq$ git log --oneline
b0e63ad x^2 button
1096247 Readme & License
venus:cal jq$
```

Registra fichero **cal_square.html** en el índice.

directorio de trabajo



Genera nuevo commit con mensaje "**x^2 button**"

Muestra los dos commits ya generados en la rama master (formato 1 línea).

master

x^2 button

Readme & License

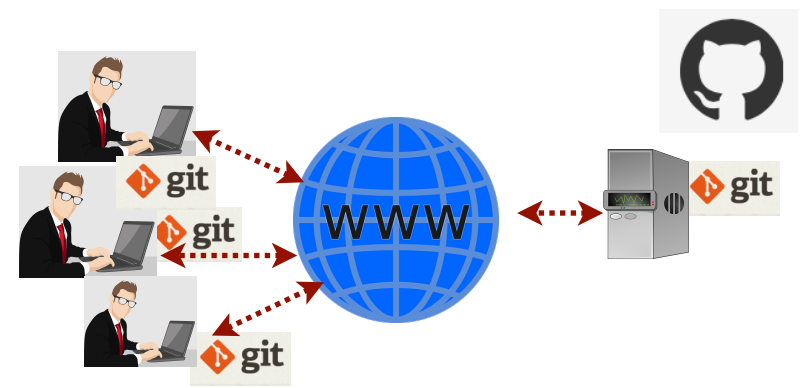


Git y GitHub

Repositorios públicos y remotos en GitHub:
new, push, import, fork y remote

Juan Quemada, DIT - UPM

Tipos de repositorio



◆ Repositorio local o de trabajo (**con** directorio trabajo)

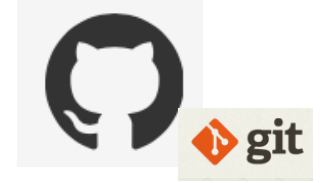
- Repositorio para **desarrollar** en el ordenador local



◆ Repositorio bare (**sin** directorio de trabajo)

- Se suelen alojar en servidores remotos en Internet
- Se utilizan para **compartir desarrollos**, **guardar backups**, etc.
- Se crean con: **git init --bare**





"remote"

◆ Un **repositorio remoto** se identifica por un **URL**, por ej.

- <https://github.com/jquemada/cal>
- https://github.com/jquemada/my_repo

◆ **"remote": nombre corto** dado en un repositorio local a uno **remoto**, por ej:

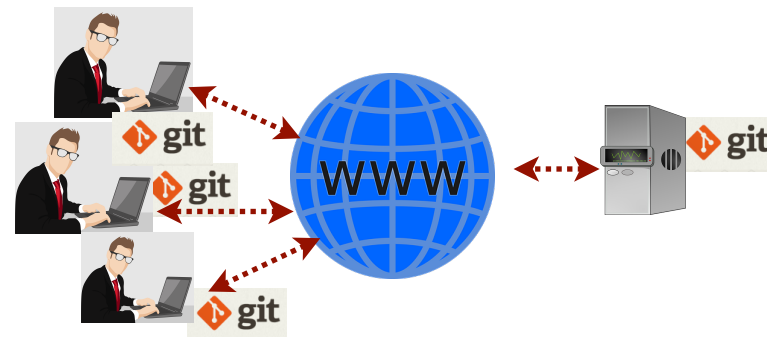
- **origin**: repositorio origen de clonación con "git clone"
 - La rama **master** de **origin** puede referenciarse como: **origin/master**
- **cal**: repositorio que contiene una calculadora guardada en GitHub
- **heroku**: repositorio para despliegue en heroku

◆ Un repositorio "remote" se define con el comando "git remote ...", por ej.

- **git remote [-v]**
 - Muestra los repositorios remotos definidos en un repositorio local (-v modo verboso)
- **git remote add cal <https://github.com/jquemada/cal>**
 - Define "remote" **cal** asociado al URL <https://github.com/jquemada/cal>
- **git remote remove cal**
 - Borra el "remote" **cal** del repositorio local (cal ya no podrá ser utilizado en comandos)

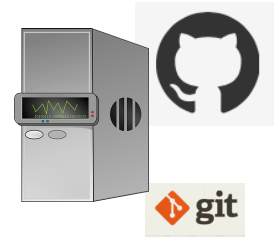


Sincronización de repositorios



- ◆ Git está pensado para trabajar en equipo y permite sincronizar repositorios locales y remotos con facilidad
- ◆ **git clone <https://github.com/jquemada/cal>**
 - Clona en local un repositorio remoto, que pasa a denominarse **origin**
- ◆ **git push**
 - **sube** a **origin** commits **nuevos** de master (si son compatibles)
- ◆ **git pull**
 - Trae desde **origin** commits nuevos de master (si son compatibles)
 - Los nuevos commits habrán sido generados por otros con acceso al repositorio
- ◆ Solo está permitido hacer **push** en **repositorios bare** (en locales no)

GitHub



◆ Github

- Servicio en la nube para albergar proyectos Git en repositorios remotos
- Permite acceder a los proyectos con Git o con un navegador
- Su lema es lema: "Social coding" porque tiene estructura de red social

◆ Repositorios **públicos son gratis**, los privados de pago

- Algunos proyectos libres en Github: Linux, Eclipse, jQuery, RoR, ...

◆ Este curso requiere tener cuenta en GitHub: <https://github.com>

- Al crearla nos da instrucciones claras y precisas sobre uso de GitHub y **Git**

◆ GitHub permite identificar sus repositorios con estos 3 tipos de **URL**:

- | | |
|---|--|
| • https://github.com/jquemada/cal | URL del rep. jquemada/cal en GitHub |
| • https://github.com/jquemada/cal.git | con extensión .git explícita (equivalente) |
| • git@github.com/jquemada/cal.git | URL Git (equivalente, poco utilizado) |

◆ Existen otros portales de repositorios, por ejemplo **Bitbucket**, ..

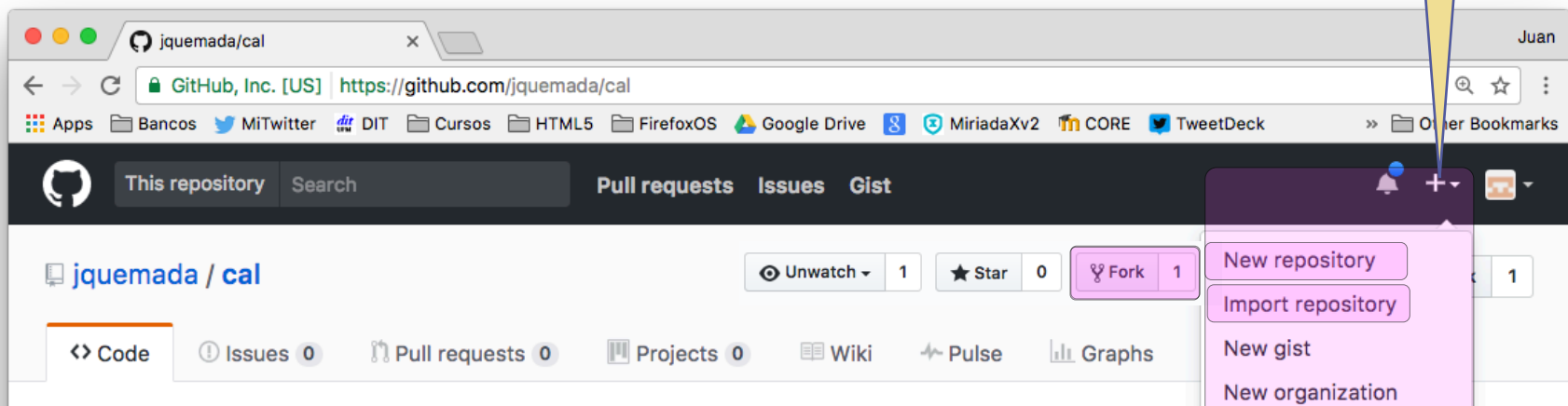
- Funcionalidad similar

Funciones principales de GitHub



- ◆ Las operaciones principales de un **usuario registrado** son
 - **Crear** repositorio (remoto) nuevo y vacío para albergar un proyecto
 - Utilizando el botón: **New repository**
 - **Copiar** un repositorio albergado en GitHub a otra cuenta (para contribuir)
 - Utilizando el botón: **Fork**
 - **Importar** un repositorio identificado por su URL a GitHub, incluso en otro formato
 - Utilizando el botón: **Import repository**
 - Se puede importar de otro servidor en Internet o de GitHub, incluso cambiando el formato
 - **Crear una organización** para albergar múltiples proyectos relacionados
 - Utilizando el botón: **New organisation**
 - Organización de asignatura CORE: <https://github.com/CORE-UPM>
 - Y otras operaciones de compartición, gestión y mantenimiento

Crear nuevos
objetos en
GitHub



Crear jquemada/cal vacío en GitHub

Crear nuevo repositorio vacío en GitHub

New repository
Import repository
New gist
New organization

Nombre del repositorio

Repositorio público sin .gitignore, LICENSE y README

Instrucciones de uso del repositorio.

Nuevo repositorio creado con URL:
<https://github.com/jquemada/cal>

Quick setup — if you've done this kind of thing before

Set up in Desktop or **HTTPS** **SSH** <https://github.com/jquemada/cal.git>

We recommend every repository include a [README](#), [LICENSE](#), and [.gitignore](#).

...or create a new repository on the command line

```
echo "# cal" >> README.md
git init
git add README.md
git commit -m "first commit"
git remote add origin https://github.com/jquemada/cal.git
git push -u origin master
```

...or push an existing repository from the command line

```
git remote add origin https://github.com/jquemada/cal.git
git push -u origin master
```

...or import code from another repository

You can initialize this repository with code from a Subversion, Mercurial, or TFS project.

[Import code](#)

Subir un repositorio local a Github con git push

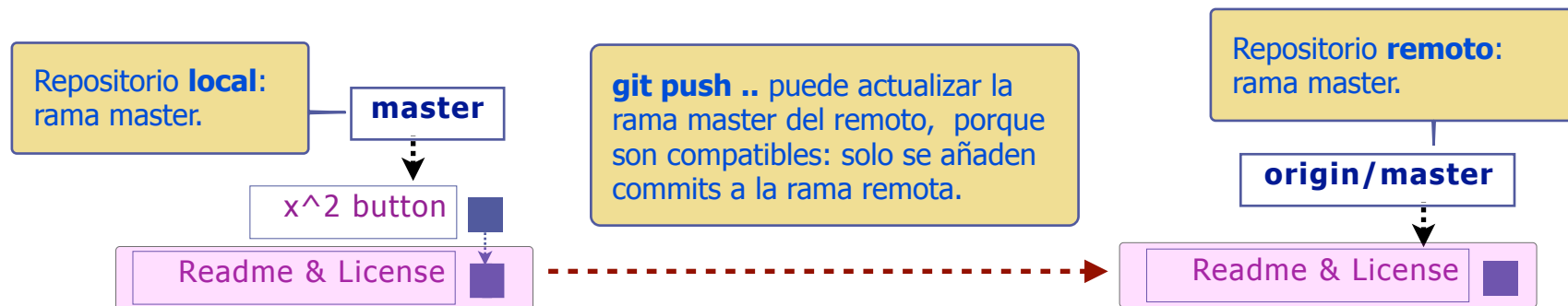
Actualizar un repositorio en GitHub con push

♦ git push ...

- Actualiza el repositorio remoto con los nuevos commits de una rama local
 - **git push**
 - actualiza commits nuevos de rama local **master** (por defecto) en repositorio **origin** (por defecto)
 - **git push origin master**
 - actualiza commits nuevos de rama local **master** en repositorio **origin** (similar al anterior)
 - **git push https://github.com/jquemada/cal_2com sqrt**
 - actualiza los nuevos commits de la rama local sqrt en el repositorio GitHub **jquemada/cal_2com**

♦ git push ... necesita 2 condiciones para finalizar con éxito

- Se debe tener credenciales de acceso al repositorio remoto
 - Por ejemplo, un repositorio en una cuenta u organización del usuario que lo actualiza
- La actualización de commits debe ser compatible con la rama actualizada en el remoto
 - Solo debe **añadir nuevos commits al final de la rama remota** o actualizar un **repositorio vacío**
 - **Peligroso!** La **opción -f** permite actualizar una rama incompatible, pero se pierden commits



Historia del repositorio local



```
venus:cal jq$  
venus:cal jq$ git log --online  
b0e63ad x^2 button  
1096247 Readme & License  
venus:cal jq$  
venus:cal jq$ git push https://github.com/jquemada/cal master  
Counting objects: 7, done.  
Delta compression using up to 4 threads.  
Compressing objects: 100% (7/7), done.  
Writing objects: 100% (7/7), 1.40 KiB | 0 bytes/s, done.  
Total 7 (delta 1), reused 0 (delta 0)  
remote: Resolving deltas: 100% (1/1), done.  
To https://github.com/jquemada/cal  
* [new branch]      master -> master  
venus:cal jq$
```

git log --online muestra los commits de la rama master del repositorio local **cal**.

master

x^2 button

Readme & License

(vacío)

El repositorio GitHub creado en <https://github.com/jquemada/cal> está vacío.

GitHub, Inc. [US] <https://github.com/jquemada/cal>

This repository Search Pull requests Issues Gist

jqemada / cal Unwatch 1 Star

<> Code Issues 0 Pull requests 0 Projects 0 Wiki Pulse Graphs Settings

Quick setup — if you've done this kind of thing before

Set up in Desktop or HTTPS SSH <https://github.com/jquemada/cal.git>

We recommend every repository include a [README](#), [LICENSE](#), and [.gitignore](#).

...or create a new repository on the command line

```
echo "# cal" >> README.md  
git init  
git add README.md  
git commit -m "first commit"  
git remote add origin https://github.com/jquemada/cal.git
```

Sincronizar rama master remota con la local



```
venus:cal jq$  
venus:cal jq$ git log --oneline  
b0e63ad x^2 button  
1096247 Readme & License  
venus:cal jq$  
venus:cal jq$ git push https://github.com/jquemada/cal master  
Counting objects: 7, done.  
Delta compression using up to 4 threads.  
Compressing objects: 100% (7/7), done.  
Writing objects: 100% (7/7), 1.40 KiB | 0 bytes/s, done.  
Total 7 (delta 1), reused 0 (delta 0)  
remote: Resolving deltas: 100% (1/1), done.  
To https://github.com/jquemada/cal  
* [new branch]      master -> master  
venus:cal jq$
```

Sube la rama **master** del repositorio **local** al repositorio **remoto** en GitHub con URL <https://github.com/jquemada/cal>, que está vacío. Ambos se sincronizan.

Es equivalente a:
`$ git remote add cal https://github.com/jquemada/cal`
`$ git push origin cal`

master

x^2 button

Readme & License

...../master

x^2 button

Readme & License

El repositorio en GitHub se ha sincronizado con el local.

El repositorio en GitHub

Repositorio en GitHub I

jquemada / cal

Es un repositorio público accesible con el URL <https://github.com/jquemada/cal> a cualquier persona través de Internet.

Unwatch 1

Star 0

Fork 0

Pulse

Graphs

Settings

No description, website, or topics provided.

New Add topics

Hay 2 commits (versiones)

2 commits

1 branch

0 releases

1 contributor

MIT

Branch: master

New pull request

El proyecto: último commit de la rama master con los 3 ficheros indicados.

New file

Upload files

Find file

Clone or download

jquemada x^2 button

Latest commit b0e63ad 5 days ago

LICENSE

Readme & License

5 days ago

README.md

Readme & License

5 days ago

calculator.html

x^2 button

5 days ago

README.md

cal

Educational Git project. Creates a simple calculator in HTML and JavaScript in short steps.

Fichero README.md se ve aquí. Es muy conveniente incluirlo en un fichero en GitHub describiendo el proyecto o repositorio.

Apps Bancos MiTwitter DIT

GitHub, Inc. [US] https://github.com/jquemada/cal

Google Drive

Repositorio en GitHub II

jquemada / cal

<> Code Issues 0 Pull requests 0 Projects 0

No description, website, or topics provided.

New Add topics

2 commits

Branch: master New pull request

jquemada x^2 button

LICENSE

README.md

calculator.html

README.md

cal

Educational Git project. Creates a simple calculator in HTML

© Juan

Último commit de la rama master con los 3 ficheros indicados.

This repository Search Pull requests Issues

jquemada / cal

<> Code Issues 0 Pull requests 0 Projects 0

Branch: master cal / calculator.html

jquemada x^2 button

1 contributor

18 lines (15 sloc) 350 Bytes

```
1 <!DOCTYPE html><html><head>
2 <title>Calculator</title><meta charset="utf-8">
3 <script type="text/javascript">
4
5 function square() {
6   var num = document.getElementById("n1");
7   num.value = num.value * num.value;
8 }
9 </script>
10 </head>
11 <body>
12   Number:
13   <input type="text" id="n1"><p>
14
15   <button onclick="square()"> x<sup>2</sup> </button>
16 </body>
17 </html>
```

Number: 2

x²

Repositorio en GitHub III

jqemada / cal

<> Code

Issues 0

Pull requests 0

Projects 0

Wiki

No description, website, or topics provided.

New Add topics

2 commits

1 branch

0 releases

Branch: master

New pull request

Create new file

Upload files

Find file

Clone or download

jqemada x^2 button

Latest commit b0e63ad 5 days ago

LICENSE

Readme & License

5 days ago

README.md

Readme & License

5 days ago

calculator.html

x^2 button

5 days ago

README.md

cal

Educational Git project. Creates a simple calculator in HTML and JavaScript in short steps.

Branch: master

Commits on Feb 12, 2017



x^2 button

jqemada committed 3 hours ago

Commits on Feb 9, 2017



Readme & License

jqemada committed 3 days ago

Number: 2

x²

Repositorio en GitHub III

jqemada / cal

Code

Issues 0

Pull requests 0

Projects 0

No description, website, or topics provided.

New Add topics

2 commits

1 branch

Branch: master

New pull request

jqemada x^2 button

LICENSE

README.md

calculator.html

README.md

cal

Educational Git project. Creates a simple calculator in

Verde: código añadido

No hay rojo, ni negro porque al ser un fichero nuevo solo se añade.

Branch: master

Commits on Feb 12, 2017



x^2 button

jqemada committed 3 hours ago

Commits on Feb 9, 2017



Readme & License

jqemada committed 3 days ago

Showing 1 changed file with 17 additions and 0 deletions.

17 calculator.html

```
... @@ -0,0 +1,17 @@
1 +<!DOCTYPE html><html><head>
2 +<title>Calculator</title><meta charset="utf-8">
3 +<script type="text/javascript">
4 +
5 +function square() {
6 +  var num = document.getElementById("n1");
7 +  num.value = num.value * num.value;
8 +}
9 +</script>
10 +</head>
11 +<body>
12 +  Number:
13 +  <input type="text" id="n1"><p>
14 +
15 +  <button onclick="square()"> x<sup>2</sup> </button>
16 +</body>
17 +</html>
```

0 comments on commit 0e9f90a

Crear repositorio jquemada/cal_2com

Crear jquemada/cal_2com

The screenshot shows the GitHub 'Import your project to GitHub' page. A callout box at the top left points to the 'Import repository' option in the GitHub sidebar. Another callout box points to the URL input field, which contains 'https://github.com/jquemada/cal'. A third callout box points to the 'Name' input field, which contains 'cal_2com'. A fourth callout box points to the 'Begin import' button at the bottom right.

Importar un repositorio a GitHub

URL de identificación del repositorio a importar.

Nombre del nuevo repositorio

Import your project to GitHub

Import all the files, including the revision history, from another version control system

Your old repository's clone URL

<https://github.com/jquemada/cal>

Learn more about the types of [supported VCS](#).

Your new repository details

Owner: [jqumada](#) / Name: [cal_2com](#) ✓

Your new repository will be **public**. In order to make this repository private you'll need to [upgrade your account](#).

Cancel Begin import

Nuevo repositorio creado con URL:
https://github.com/jquemada/cal_2com

The screenshot shows the GitHub repository page for 'jqumada / cal_2com'. The repository is public and has 2 commits and 1 branch. The repository description is 'Educational Git project. Creates a simple calculator in H...'. The repository files include 'LICENSE', 'README.md', and 'calculator.html'.

This repository Search Pull requests

jqumada / cal_2com

<> Code Issues 0 Pull requests 0 Projects 0

No description, website, or topics provided.

New Add topics

2 commits 1 branch

Branch: master New pull request

jqumada x^2 button

LICENSE Readme &

README.md Readme &

calculator.html x^2 button

README.md

cal

Educational Git project. Creates a simple calculator in H...

Crear repositorio CORE-UPM/cal con Fork

Fork: duplicar un proyecto de GitHub



The screenshot shows the GitHub web interface. At the top, a navigation bar includes 'This repository', 'Search', 'Pull requests', 'Issues', and 'Gist'. Below this, the repository 'jquemada / cal' is displayed. A modal window titled 'Where should we fork this repository?' is open, showing two options: '@ging' and '@CORE-UPM...'. The '@CORE-UPM...' option is selected. Below the modal, the repository 'CORE-UPM / cal' is shown, with a note 'forked from jquemada/cal'. On the left, the commit history is visible, showing two commits: 'x^2 button' and 'Readme & License'. On the right, the repository 'jquemada / cal' is shown again, with a note 'Copia de jquemada/cal creada en la organización CORE-UPM. El nuevo repositorio estará accesible en: https://github.com/CORE-UPM/cal'. A red dashed arrow points from the 'Fork' button on the right to the modal window. Another red dashed arrow points from the '2 commits' label to the commit history on the left. A third red dashed arrow points from the '2 commits' label to the '2 commits' label on the right. A fourth red dashed arrow points from the '2 commits' label to the '2 commits' label on the bottom right.

Repositorio cal del usuario jquemada en: <https://github.com/jquemada/cal>

Copia el repositorio a otra cuenta u organización del usuario en GitHub pulsando el **botón Fork**.

Where should we fork this repository?

Can't find what you're looking for?
You already have a fork of this repository:
[jquemada/cal_2com](#)

Copia de jquemada/cal creada en la organización CORE-UPM. El nuevo repositorio estará accesible en: <https://github.com/CORE-UPM/cal>

En el momento del **Fork** el repositorio **jquemada/cal** tiene estos **2 commits**. A partir de este momento cada repositorio evolucionara por separado a partir de estos 2 commits

Commits on Feb 12, 2017

- x^2 button
jquemada committed 3 hours ago

Commits on Feb 9, 2017

- Readme & License
jquemada committed 3 days ago

2 commits

1 branch

0 releases

1 contri



Final del tema