



Git & GitHub

Introducción y uso de Git



Christian Prada Osuna
En colaboración con Yunemy
y Tech Talent South



Linux/Unix DevOps & Back-end Developer

- Administrador de Sistemas
 - IBM (Málaga)
 - CDMA (Granada)
 - Proyectos propios (Granada)

De aquí a...5 años? ¡Es hora de reciclarse!

- Back-end Developer: [Actualmente]
 - Tecnologías:
 - Python & Codenerix  Centrologic
 - Java Web JEE  Indra
 - iOS Native



Temario

1. Presentación
2. Creadores de Git
3. ¿Qué es Git?
4. Características
5. Propósitos de usar Git en Desarrollo
6. ¿Qué es GitHub?
7. Creamos nuestra cuenta en GitHub
8. Empezando a configurar Git
9. Protocolo SSH y Git (Claves públicas y privadas)
10. Creando nuestro primer proyecto con Git & GitHub
11. Comandos comunes de Git
12. Gestión de proyectos
13. GitKraken
14. Clonamos repositorio de GitHub
15. Uso de GitKraken
16. Resolución de incidencias



Creadores de Git



Torvalds, mundialmente conocido por crear y mantener el kernel de Linux en la actualidad.

Miembro de “The Linux Foundation” , fundación sin ánimo de lucro donde adoptan estándares de Software y Hardware para el crecimiento de Linux.



<https://github.com/torvalds>



Hamano, Ingeniero de Software Japonés, actualmente trabaja en Google y lleva el mantenimiento de Git desde su creación en 2005.

<https://github.com/gitster>



¿Qué es Git?

- Software de control de versiones diseñado inicialmente para dar soporte al desarrollo de Linux, utilizando un modelo abierto y descentralizado.
- Su uso se ha propagado tanto por el ámbito de desarrollo que es habitual solicitar perfiles en GitHub, GitLab o Bitbucket para los procesos de selección.



Características

- “Pública General de GNU(v2)” (software libre distribuido) .
- Soporta servidores HTTP, FTP, CVS y cifrado SSH.
- Rapidez en la gestión de ramas (Desarrollo no lineal) .
- Gestión distribuida (Copia local del historial de desarrollo) .
- Eficiencia en proyectos de gran dimensión.

Propósitos de usar Git en desarrollo

- Registro inicial y continuo de los cambios que realicemos en nuestro proyecto.
- Control del código fuente, de manera que podremos volver a un estado anterior del mismo, y seguir desde ese punto.
- Coordinación del trabajo que varias personas realizan sobre el mismo proyecto.
- Facilitar el desarrollo no solo en ámbito local sino también en remoto

¿Qué es GitHub?

- Plataforma de desarrollo de software colaborativo, se utiliza para alojar proyectos usando el sistema de control de versiones “Git” .
- El código se administra de forma:
 - Pública (Predeterminada por defecto)
 - Privada (De pago) (*GitLab permite repositorio privado por defecto*)
- ¿Para qué sirve?
 - GitHub almacena el repositorio de código y está orientado para el trabajo en equipo, Junto con “Git” , constituye una gran herramienta para el desarrollo de software.
 - Contribuir a la mejora del código de los demás, utilizando funcionalidades como: “**fork**” y “**pulls**”.

Creamos una cuenta en GitHub



Enlace a GitHub: <https://github.com/join?source=header-home>

Empezando a configurar Git (1/2)

- Abrimos la terminal y ejecutamos `git --version` para revisar si esta instalado.
- Si no, procedemos a descargar e instalarlo :
 - Windows: <https://git-scm.com/download/win>
 - Mac: <https://git-scm.com/download/mac>
 - Linux (abrir terminal):
 - Debian/Ubuntu: `sudo apt-get install -y git`
 - Fedora: `sudo dnf/yum install -y git`
 - ArchLinux: `sudo pacman -S git`
 - Comandos de ayuda:
 - `git help <comand>`
 - `git <comand> --help`
 - `man git-<comand>`

Empezando a configurar Git (2/2)

- Una vez descargado e instalado en nuestro sistema:
 - Pasos:
 - `git config --global user.name "USUARIO"`
 - `git config --global user.email "USUARIO @CORREO.es"`
 - Comprobamos la configuración con:
`git config --list`
 - Si queremos, podemos revisar el manual de configuración de git:
`git config --help`
 - Otros comandos habituales de configuración:
 - `git config --global core.editor vim`
(nano/vi/vim/emacs/...)
 - `git config --global merge.tool vimdiff`
(vimdiff/kdiff3/tkdiff/meld/emerge/...)

Protocolo SSH & Git (1/3)

- Pasos:

- Localizar si existe ya una clave SSH en nuestro equipo:

- Git bash on Windows / Linux or MacOS :

```
cat ~/.ssh/id_rsa.pub
```

- Si no existe generamos una clave nueva:

- Git bash on Windows / Linux or MacOS:

```
ssh-keygen -t rsa -C "USUARIO @github.com" -b 4096
```

Existe una alternativa para Windows usando el programa PuttyGen:

<https://the.earth.li/~sgtatham/putty/0.67/html/doc/Chapter8.html#pubkey-puttygen>

Protocolo SSH & Git (2/3)

- Se nos solicita un ruta para guardar la clave.
Nota: Si tuviésemos una clave de antes, añadiremos una ruta diferente, y tendremos que declararla dentro del fichero: `/.ssh/config`.
- Se nos solicitará escribir una contraseña (no es necesario pero si recomendable) .
Nota: podemos cambiar el par de claves SSH escribiendo: `ssh-keygen -p <keyname>`.
- A continuación copiamos el contenido de nuestra clave :
 - Windows: `cat ~/.ssh/id_rsa.pub | clip`
 - MacOS: `pbcopy <~/.ssh/id_rsa.pub`
 - Linux: `xclip -sel clip <~/.ssh/id_rsa.pub`
(En Linux es posible que necesitemos instalar: `xclip`)

Protocolo SSH & Git (3/3)

- El último paso es añadir a nuestro GitHub/GitLab nuestra clave SSH:
 - GitHub: *"Settings/SSH and GPG keys/New SSH Key/"*
Pegar en la sección y añadimos un título descriptivo.
 - GitLab: *"SSH Keys/Profile Settings/"*
Pegar en la sección y añadimos un título descriptivo.

¡Todo listo!

Empezando a configurar Git

Directorios comunes de un proyecto:

- Nombres de directorios y ficheros:
 - README.md (fichero resumen escrito en formato markdown)
<https://guides.github.com/features/mastering-markdown/>
 - LICENSE (licencia de distribución del proyecto)
 - public (directorio de recursos web)
 - bin (directorio de programas ejecutables)
 - lib (directorio de librerías de software)
 - test (directorio de pruebas de funcionamiento)
- Git (control de versiones)
 - .git (directorio de configuración de Git)
 - .gitignore (fichero donde incluiremos las rutas a omitir)
- npm (gestor de paquetes Node-js)
 - package.json (fichero con información del paquete y dependencias)
 - node_modules (contiene las dependencias instaladas del proyecto)

Creando nuestro primer proyecto con Git & GitHub (1/2)

- Pasos:
- Abrimos GitHub y creamos un nuevo repositorio llamado: `yunemy`
- Creamos una carpeta con nombre `yunemi` y nos movemos a ella:
 - `mkdir yunemy && cd yunemy`
 - A continuación iniciamos un repositorio vacío con:
 - `git init`
 - Creamos un archivo llamado `README.md` y dentro añadimos "Hello Yunemyers!!":
 - `echo "Hola Yunemyers" > README.md`
 - Añadimos el archivo de nuestro directorio con:
 - `git add README.md`
 - Creamos nuestro primer commit:
 - `git commit -m "[README] Creamos el archivo desde consola"`

Creando nuestro primer proyecto con Git & GitHub (2/2)

- Añadimos nuestro repositorio local a uno remoto de GitHub:
 - `git remote add origin git@github.com:USUARIO/yunemy.git`
- Verificamos la nueva URL:
 - `git remote -v`
- Hacemos `push` para subir nuestro repositorio a la rama `master`:
 - `git push -u origin master`
- Comprobamos en GitHub si se han subido los cambios.

¡Acabas de subir tu primer proyecto a GitHub!

Comandos comunes de Git (1/2)

- **Iniciar un proyecto**
 - `init` Crea un repositorio de Git vacío o reinicia el que ya existe
 - `clone` Clona un repositorio dentro de un nuevo directorio
- **Realizar cambios**
 - `add` Agrega contenido de carpetas al índice
 - `mv` Mueve o cambia el nombre a archivos, directorios o enlaces
 - `reset` Reinicia el HEAD actual a un estado específico
 - `rm` Borra archivos del árbol de trabajo y del índice
- **Historial y estado**
 - `bisect` Use la búsqueda binaria para encontrar el commit que introdujo el bug
 - `grep` Imprime las líneas que coinciden con el patron de busqueda
 - `log` Muestra los logs de los commits
 - `show` Muestra varios tipos de objetos
 - `status` Muestra el estado del árbol de trabajo

Comandos comunes de Git (2/2)

- Crece, marca y ajusta tu historial común
 - `branch` Lista, crea, o borra ramas
 - `checkout` Cambia ramas o restaura los archivos de tu árbol de trabajo
 - `commit` Graba los cambios en el repositorio
 - `diff` Muestra los cambios entre commits, commit y árbol de trabajo, etc
 - `merge` Junta dos o más historiales de desarrollo juntos
 - `rebase` Vuelve a aplicar commits en la punta de otra rama
 - `tag` Crea, lista, borra o verifica un tag de objeto firmado con GPG
- Colaborar
 - `fetch` Descarga objetos y referencias de otro repositorio
 - `pull` Realiza un fetch e integra con otro repositorio o rama local
 - `push` Actualiza referencias remotas junto con sus objetos asociados

Gestión de proyectos

- Vamos a utilizar:

- GitKraken:

- Windows / MacOS / Linux:

- <https://www.gitkraken.com/download>

- <https://support.gitkraken.com/how-to-install>



- Kdiff3:

- Windows / MacOS:

- <https://sourceforge.net/projects/kdiff3/files/kdiff3/>

- Linux (abrir en terminal):

- Debian/Ubuntu: `sudo apt-get install -y kdiff3`
 - Fedora: `sudo dnf/yum install -y kdiff3`
 - ArchLinux: `sudo pacman -S kdiff3`



Gestión de proyectos

- Otro Software conocido para la gestión de proyectos...

- SourceTree:

- Windows / MacOS:

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

- DiffMerge:

- Windows / MacOS / Linux:

- <https://sourcegear.com/diffmerge/downloads.php>

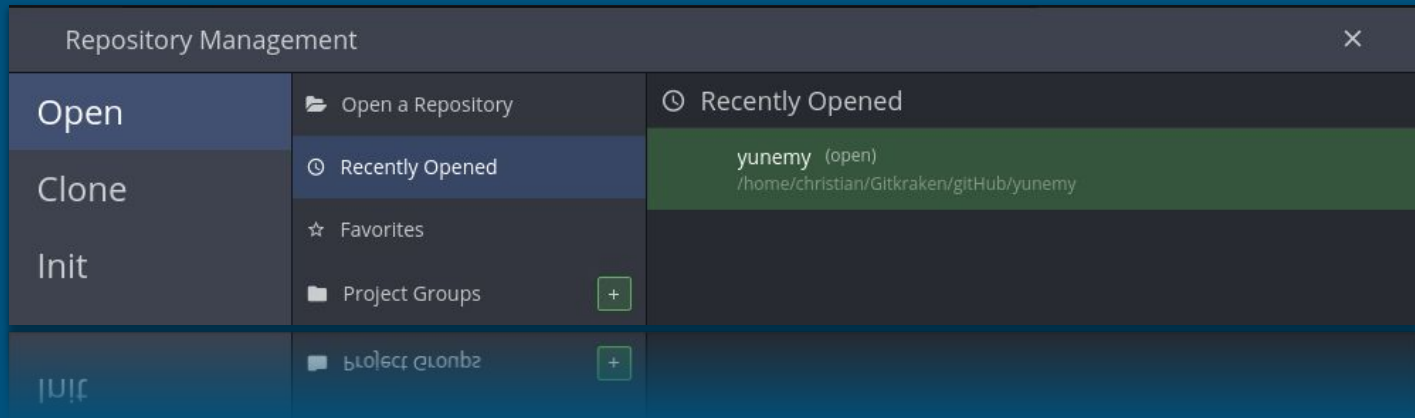
- Configuración linux:*

- https://sourcegear.com/diffmerge/webhelp/sec__git__linux.html

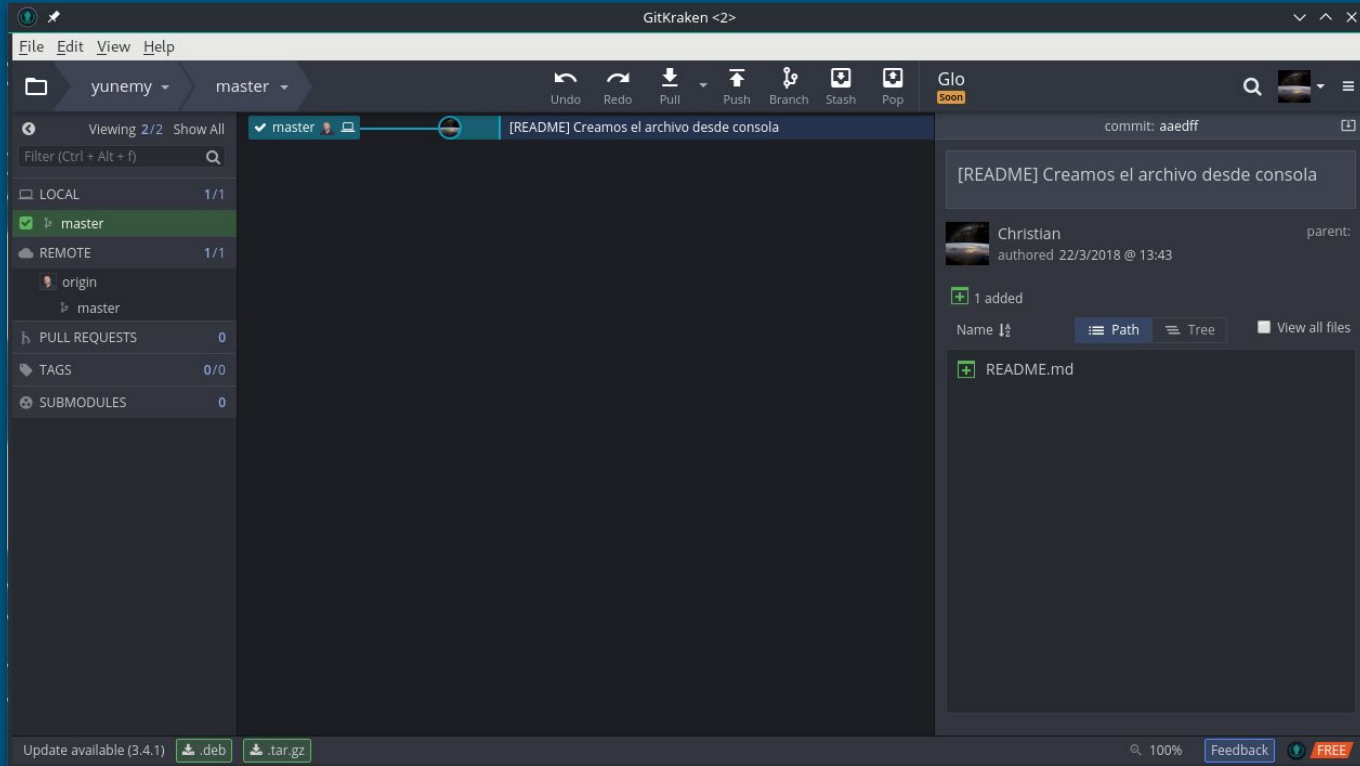


GitKraken(1/3)

- Ejecutamos GitKraken:
 - Abrimos nuestro repositorio:
 - File/Open Repo (Ctrl + O)
 - Open a Repository/ (Seleccionamos la ruta de nuestro repositorio local)

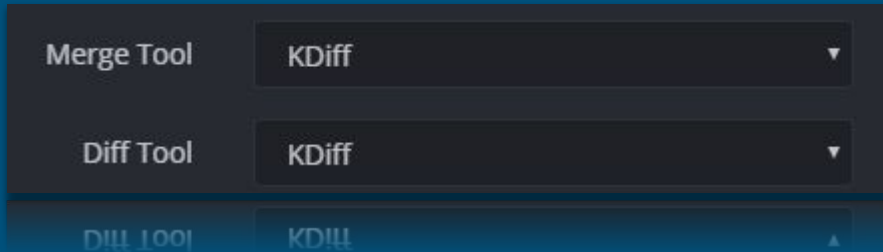


GitKraken(2/3)



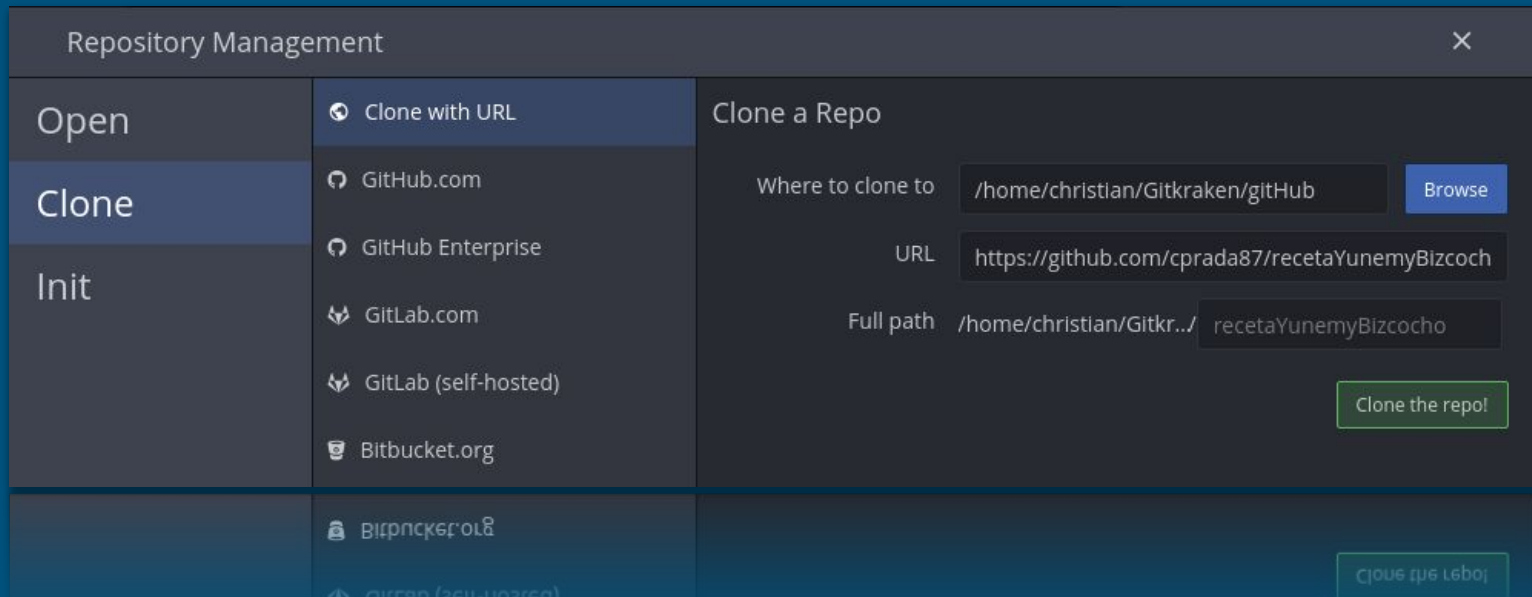
GitKraken(3/3)

- Configuración de kdiff3:
 - Seleccionamos como herramienta externa en:
 - Preferences/General/



Clonamos repositorio de GitHub (Gitkraken)

- Copiamos el siguiente URL: <https://github.com/cprada87/recetaYunemyBizcocho.git>
- Después copiamos el contenido de README.md y la carpeta en: yunemy



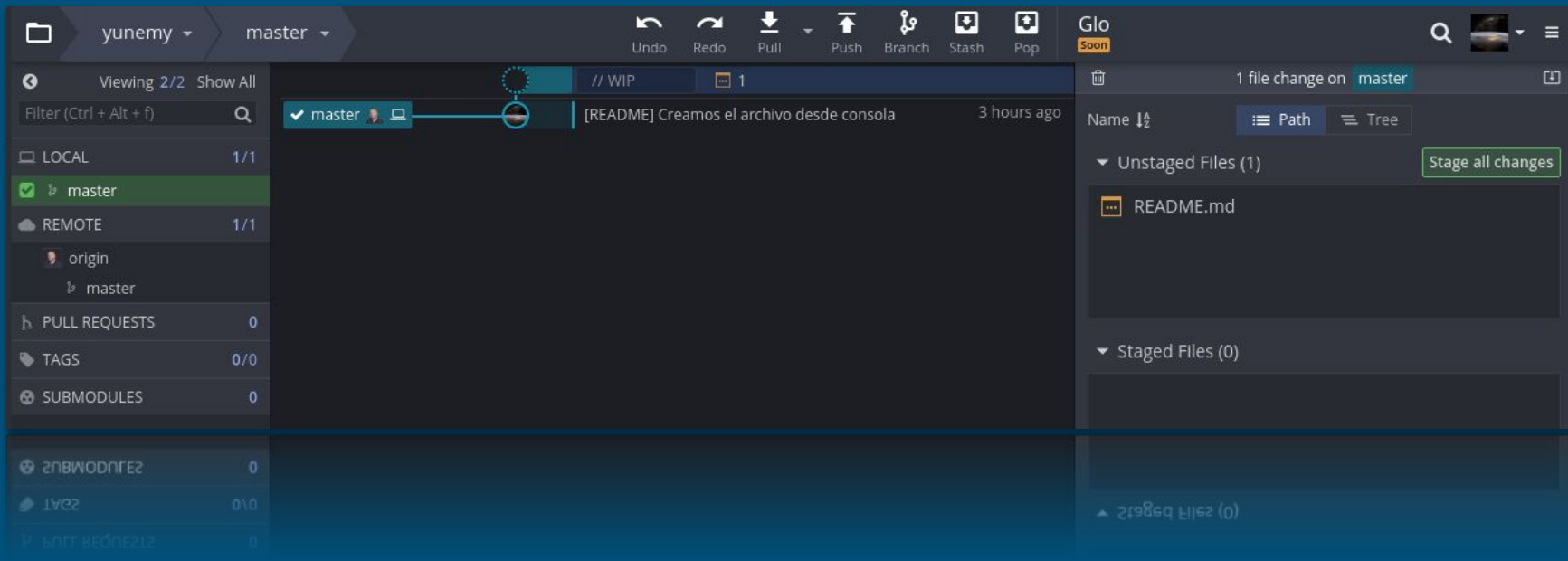
Clonamos repositorio de GitHub (Manual)

- Pasos:
- Volvemos al directorio principal:
 - `cd ../`
- Clonamos el repositorio:
 - `git clone https://github.com/cprada87/recetaYunemyBizcocho`
- Sobreescribimos el README.md de recetaYunemyBizcocho al del directorio yunemy
 - `cp -rf recetaYunemyBizcocho/* yunemy/`

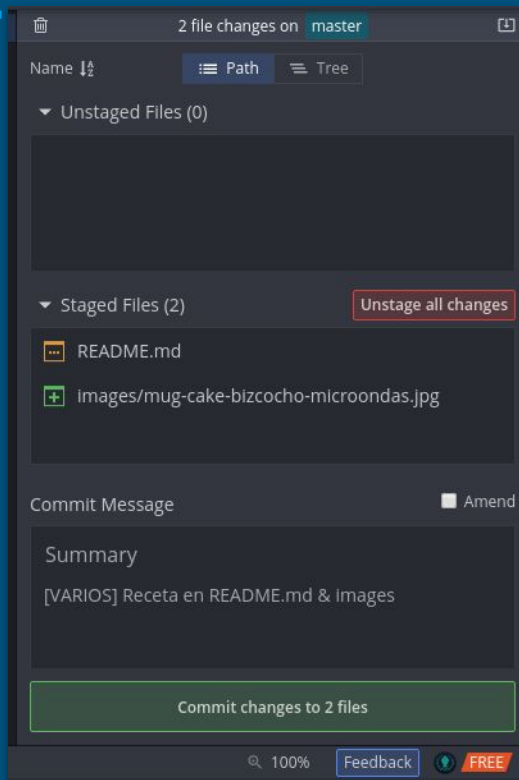
¡Seguimos!

Uso de GitKraken (1/2)

- Acto seguido GitKraken nos avisa de que un archivo a cambiado en master



Uso de GitKraken (2/2)



- Movemos los archivos a Staged Files
- Escribimos nuestro commit y le damos a: "Commit changes to 1 file"
- Hacemos click (arriba) en push :



- Podemos revisar nuestro repositorio en GitHub y comprobar los cambios en el archivo README.me

https://github.com/TU_USUARIO_AQUÍ/yunemi

Resolución de incidencias(1/14)

- Una de las herramientas más conocidas es Jira, incluye:
 - Tableros:
 - Scrum
 - Kanban
 - Herramientas de reportes
 - Gestión de incidencias según el tipo y también:
 - Gravedad:
 - Nivel low
 - Nivel normal
 - Nivel high

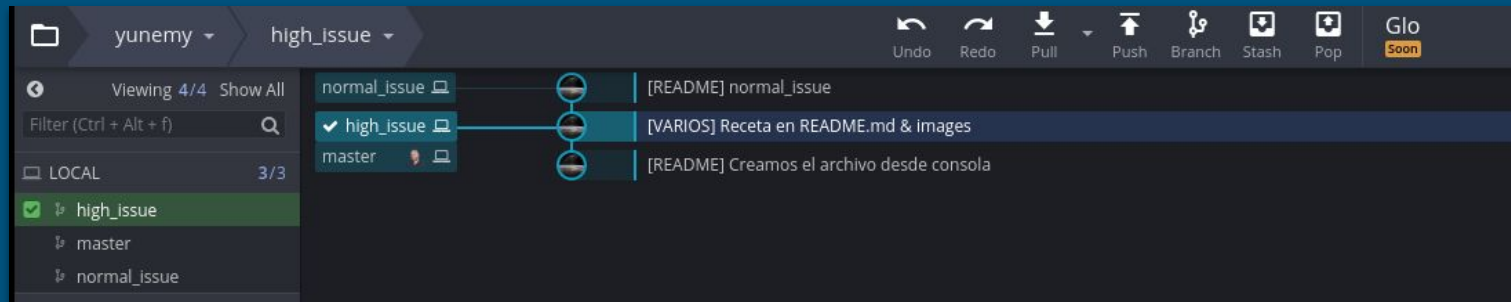
Resolución de incidencias(2/14)

- Por cada incidencia (`issue`) suele crearse una rama (`branch`) para resolverla.
 - A continuación simularemos un día habitual en nuestra Empresa:
 - El trabajo será mejorar una parte del código de una página (`README.md`) que en este caso es nuestra receta.
 - Crearemos una rama pulsando en `branch` y nombrandola: `normal_issue`
- Nos dicen que tenemos que modificar la línea número 15:

```
@@ -12,7 +12,7 @@
12 12 * 3 cucharadas de leche
13 13 * 3 cucharadas de aceite
14 14 * 1/4 de taza de trozos de chocolate (más o menos grandes)
-15 * 1 chorro pequeño de extracto de vainilla
+15 * 1 chorro pequeño de extracto de canela
16 16
17 17 ## Cómo hacer el bizcocho microondas o Mug cake en 3 minutos:
18 18
```

Resolución de incidencias(3/14)

- Resulta que nos llaman y nos dicen que dejemos lo que estemos haciendo y que la receta en vez de para una persona, hay que hacerla para dos...
 - Guardamos los cambios actuales, movemos el fichero a Staged Files, hacemos commit: `"[README] normal_issue"`
 - Volvemos a la rama anterior (segundo botón/Checkout origin/master)
([VARIOS] Receta en README.md & image)
 - Creamos una nueva rama, esta vez de carácter más urgente y la llamaremos: `high_issue`

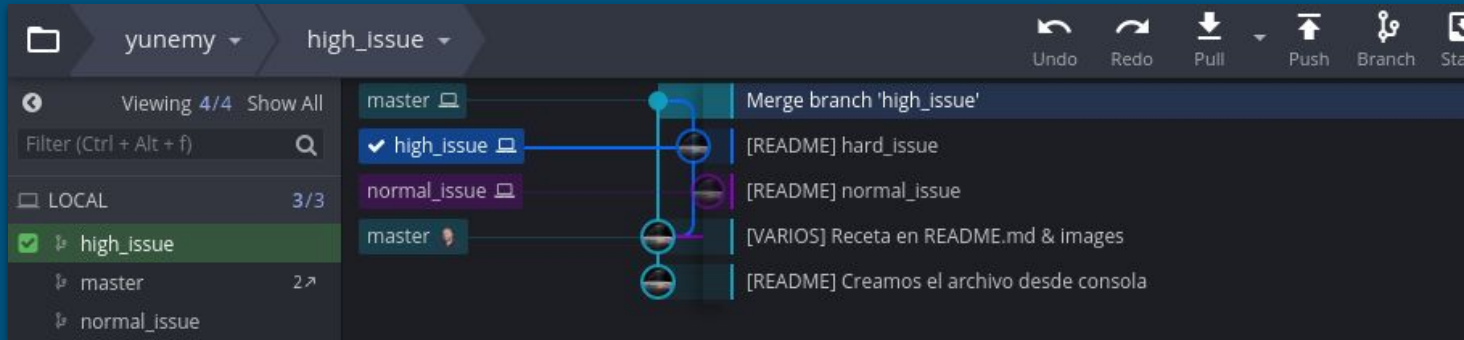


Resolución de incidencias(4/14)

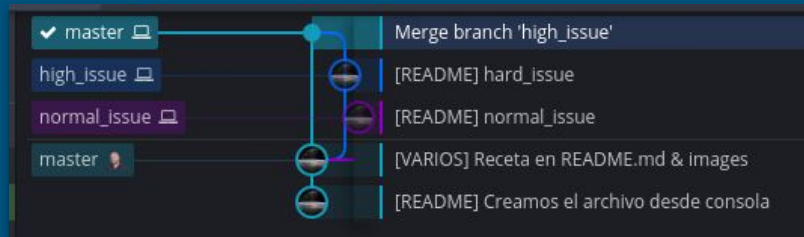
```
@@ -4,15 +4,15 @@
4      4
5      5  - Ingredientes:
6      6
-7      * 1 taza grande
-8      * 4 cucharadas de harina
-9      * 4 cucharadas de azúcar
-10     * 2 cucharadas de cacao
-11     * 1 huevo grande
-12     * 3 cucharadas de leche
-13     * 3 cucharadas de aceite
-14     * 1/4 de taza de trozos de chocolate (más o menos grandes)
-15     * 1 chorro pequeño de extracto de vainilla
+7      * 2 tazas grandes
+8      * 8 cucharadas de harina
+9      * 8 cucharadas de azúcar
+10     * 4 cucharadas de cacao
+11     * 2 huevos grandes
+12     * 6 cucharadas de leche
+13     * 6 cucharadas de aceite
+14     * 1/2 de taza de trozos de chocolate (más o menos grandes)
+15     * 2 chorros pequeños de extracto de vainilla
16     16
17     17  ## Como hacer el bizcocho microondas o Mug cake en 3 minutos:
18     18
```

- Modificamos los cambios en el fichero, movemos el fichero de nuevo a Staged Files, de nuevo commit:
[README] high_issue
- Como esta incidencia tiene prioridad, subimos primero los cambios de esta al repositorio remoto.
- Nos posicionamos en master, segundo botón/merge high_issue into master

Resolución de incidencias(5/14)



- Todavía estamos posicionados en `high_issue`, por lo que cambiamos de rama a `master`, segundo botón/checkout `master`



Resolución de incidencias(6/14)

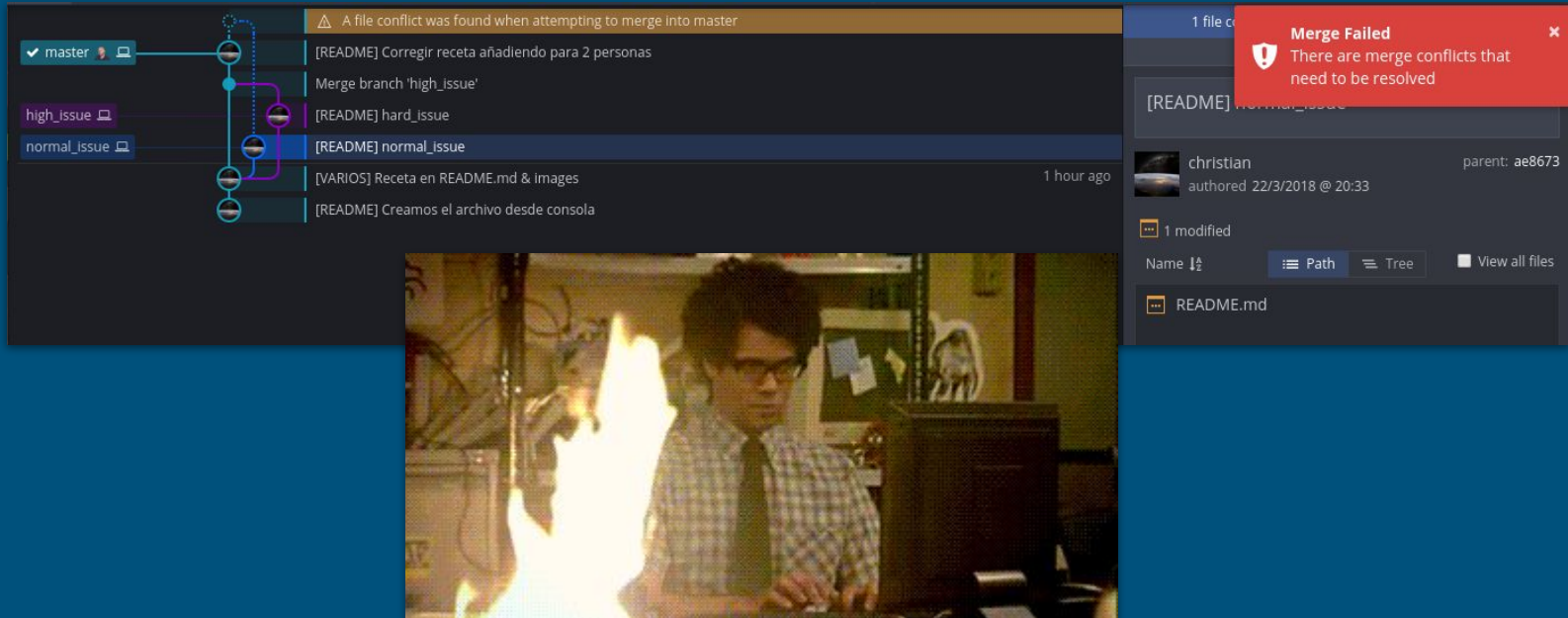
- Ahora que estamos posicionados en `master`, podemos hacer `push` al repositorio remoto y subir los cambios que nos han pedido.



- Hemos entregado la `high_issue` a tiempo...nos unimos la rama `normal_issue` a `master`
(`merge normal_issue into master`)

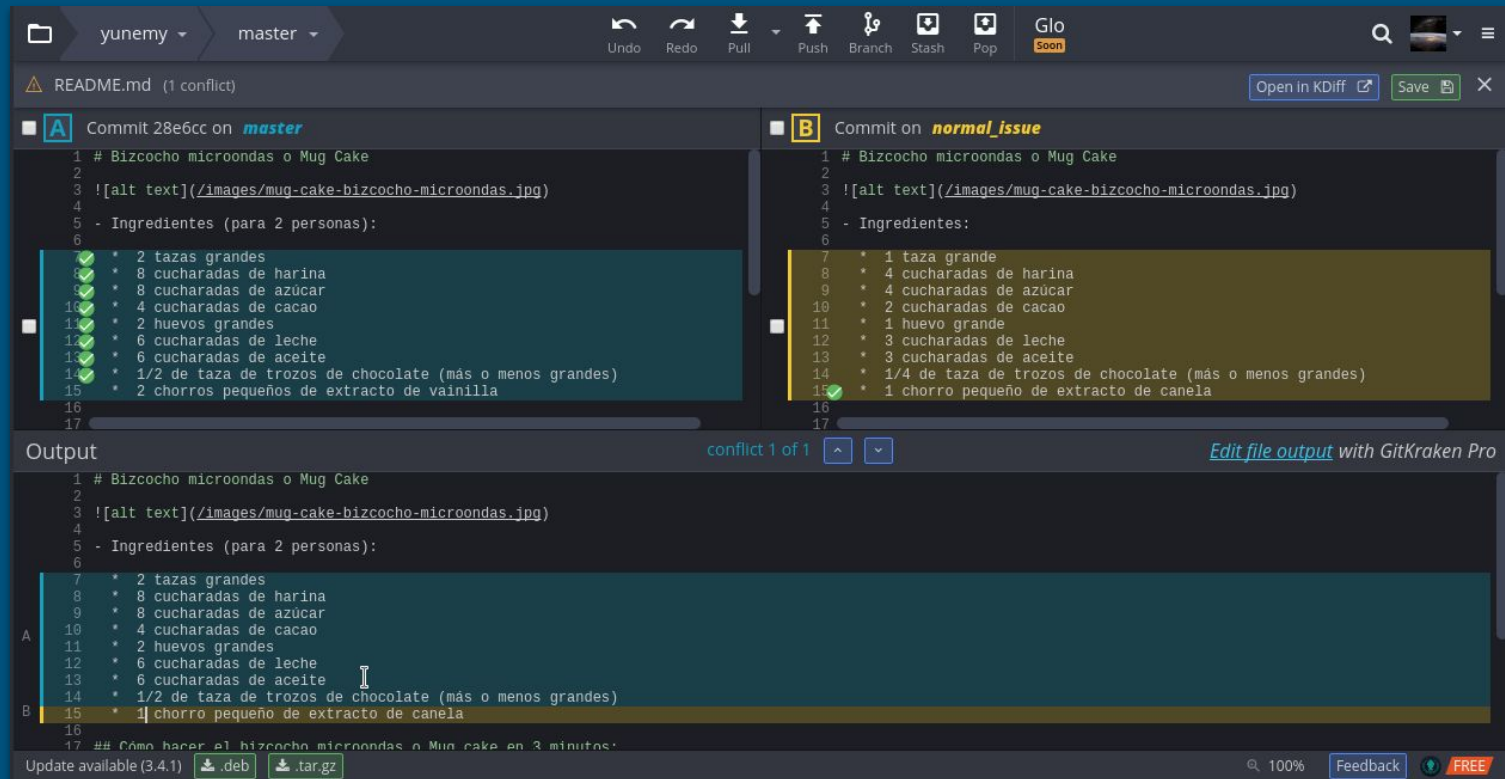
...

Resolución de incidencias(7/14)




The image shows a Git merge conflict resolution interface. On the left, a commit history tree shows a merge from 'high_issue' to 'master'. The commit messages include '[README] Corregir receta añadiendo para 2 personas', 'Merge branch 'high_issue'', '[README] hard_issue', '[README] normal_issue', '[VARIOS] Receta en README.md & Images', and '[README] Creamos el archivo desde consola'. A red banner at the top states: 'A file conflict was found when attempting to merge into master'. On the right, a 'Merge Failed' error message says: 'There are merge conflicts that need to be resolved'. Below the error, the commit details for 'christian' (parent: ae8673) are shown, including the file 'README.md' which was modified 1 hour ago. At the bottom, a meme of Moss from 'The IT Crowd' is shown, with a large fire effect behind him, symbolizing a critical system failure or a 'burning' problem.

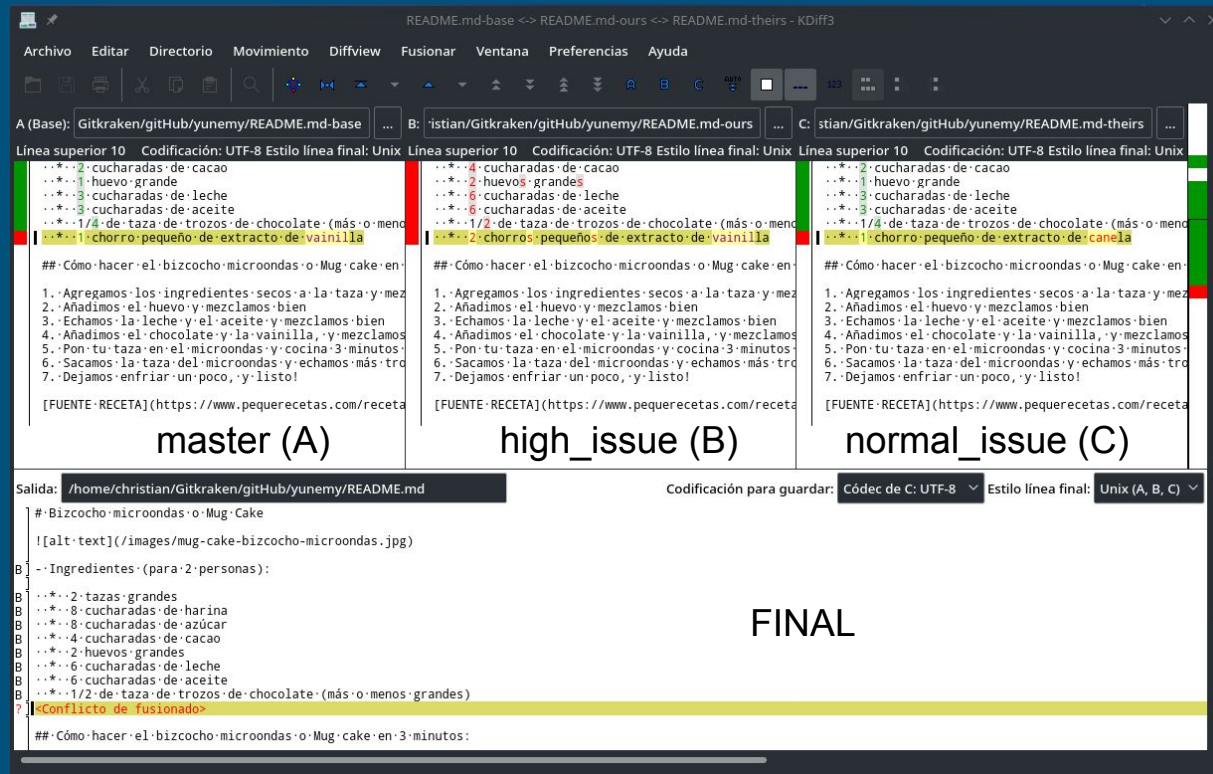
Resolución de incidencias(8/14)



Resolución de incidencias(9/14)

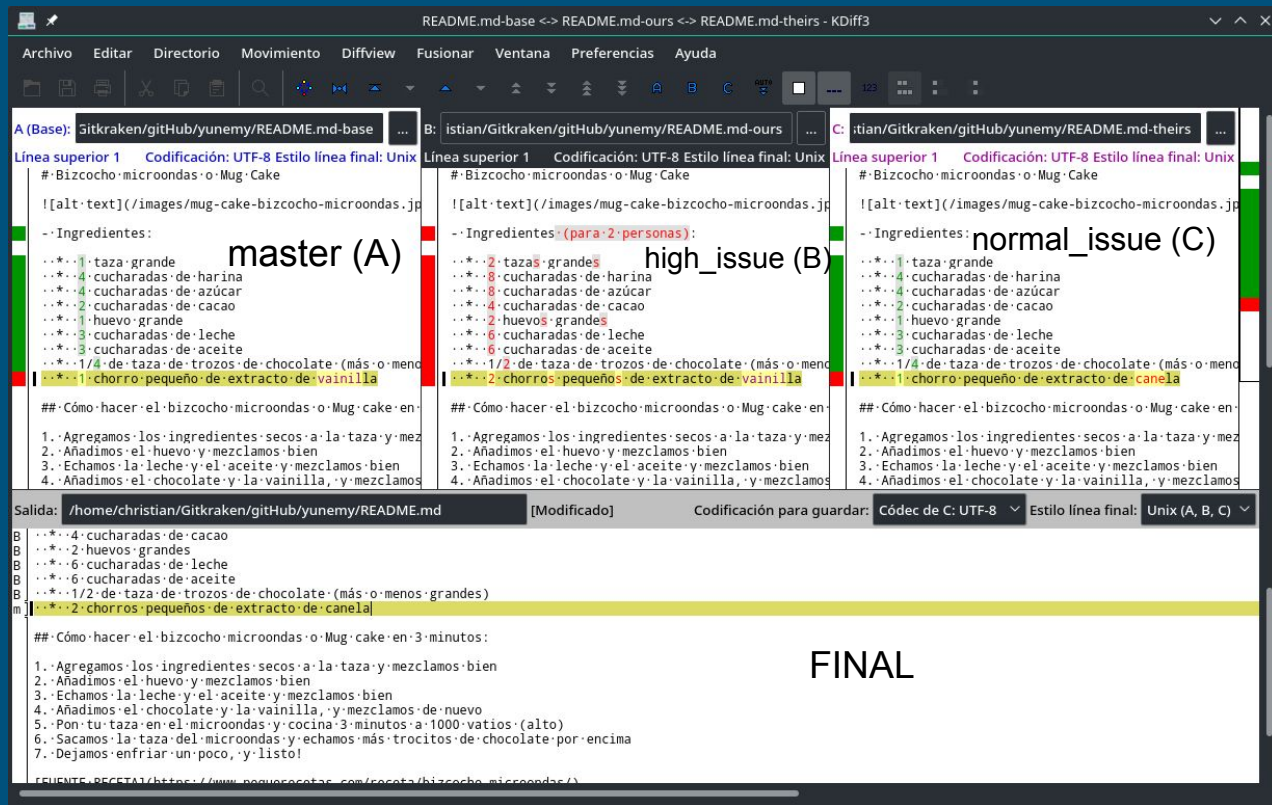
- Hacemos click en:
 y se nos
abrirá nuestro editor
gratuito:

■ KDiff3



Resolución de incidencias(10/14)

- Recordemos la prioridad de las incidencias, primero high_issue y despues normal_issue:



```
README.md-base <-> README.md-ours <-> README.md-theirs - KDiff3
Archivo  Editar  Directorio  Movimiento  Diffview  Fusionar  Ventana  Preferencias  Ayuda

A (Base): Gitkraken/gitHub/yunemy/README.md-base  B: istian/Gitkraken/gitHub/yunemy/README.md-ours  C: itian/Gitkraken/gitHub/yunemy/README.md-theirs
Línea superior 1  Codificación: UTF-8 Estilo línea final: Unix  Línea superior 1  Codificación: UTF-8 Estilo línea final: Unix  Línea superior 1  Codificación: UTF-8 Estilo línea final: Unix
# Bizcocho-microondas-o Mug-Cake
![alt:text](/images/mug-cake-bizcocho-microondas.jp
--Ingredientes:
...1-taza-grande
...4-cucharadas-de-harina
...4-cucharadas-de-azúcar
...2-cucharadas-de-cacao
...1-huevo-grande
...3-cucharadas-de-leche
...3-cucharadas-de-aceite
...1/4-de-taza-de-trozos-de-chocolate-(más-o-menos-grandes)
...1-chorro-pequeño-de-extracto-de-vainilla
##-Cómo-hacer-el-bizcocho-microondas-o-Mug-cake-en-3-minutos:
1.-Agregamos-los-ingredientes-secos-a-la-taza-y-mez
2.-Añadimos-el-huevo-y-mezclamos-bien
3.-Echamos-la-leche-y-el-aceite-y-mezclamos-bien
4.-Añadimos-el-chocolate-y-la-vainilla,-y-mezclamos

B
...4-cucharadas-de-cacao
B
...2-huevos-grandes
B
...6-cucharadas-de-leche
B
...6-cucharadas-de-aceite
B
...1/2-de-taza-de-trozos-de-chocolate-(más-o-menos-grandes)
m
...2-chorros-pequeños-de-extracto-de-canela

##-Cómo-hacer-el-bizcocho-microondas-o-Mug-cake-en-3-minutos:
1.-Agregamos-los-ingredientes-secos-a-la-taza-y-mezclamos-bien
2.-Añadimos-el-huevo-y-mezclamos-bien
3.-Echamos-la-leche-y-el-aceite-y-mezclamos-bien
4.-Añadimos-el-chocolate-y-la-vainilla,-y-mezclamos-de-nuevo
5.-Pon-tu-taza-en-el-microondas-y-cocina-3-minutos-a-1000-vatios-(alto)
6.-Sacamos-la-taza-del-microondas-y-echamos-más-trocitos-de-chocolate-por-encima
7.-Dejamos-enfriar-un-poco,-y-listo!

FUENTE: RECETA(http://www.papasroquetas.com/receta/bizcocho-microondas/)

Salida: /home/christian/Gitkraken/gitHub/yunemy/README.md [Modificado] Codificación para guardar: Códex de C: UTF-8 Estilo línea final: Unix (A, B, C)
FINAL
```


Resolución de incidencias(11/14)

yunemy master

Undo Redo Pull Push Branch Stash Pop Glo Soon

README.md (1 conflict) Open in KDiff Save X

A Commit 28e6cc on **master**

```
1 # Bizcocho microondas o Mug Cake
2
3 ![alt text](/images/mug-cake-bizcocho-microondas.jpg)
4
5 - Ingredientes (para 2 personas):
6
7 * 2 tazas grandes
8 * 8 cucharadas de harina
9 * 8 cucharadas de azúcar
10 * 4 cucharadas de cacao
11 * 2 huevos grandes
12 * 6 cucharadas de leche
13 * 6 cucharadas de aceite
14 * 1/2 de taza de trozos de chocolate (más o menos grandes)
15 * 2 chorros pequeños de extracto de vainilla
16
17
```

B Commit on **normal_issue**

```
1 # Bizcocho microondas o Mug Cake
2
3 ![alt text](/images/mug-cake-bizcocho-microondas.jpg)
4
5 - Ingredientes:
6
7 * 1 taza grande
8 * 4 cucharadas de harina
9 * 4 cucharadas de azúcar
10 * 2 cucharadas de cacao
11 * 1 huevo grande
12 * 3 cucharadas de leche
13 * 3 cucharadas de aceite
14 * 1/4 de taza de trozos de chocolate (más o menos grandes)
15 * 1 chorro pequeño de extracto de canela
16
17
```

Output conflict 1 of 1 Edit file output with GitKraken Pro

```
1 # Bizcocho microondas o Mug Cake
2
3 ![alt text](/images/mug-cake-bizcocho-microondas.jpg)
4
5 - Ingredientes (para 2 personas):
6
7 * 2 tazas grandes
8 * 8 cucharadas de harina
9 * 8 cucharadas de azúcar
10 * 4 cucharadas de cacao
11 * 2 huevos grandes
12 * 6 cucharadas de leche
13 * 6 cucharadas de aceite
14 * 1/2 de taza de trozos de chocolate (más o menos grandes)
15 * 1 chorro pequeño de extracto de canela
16
17 ## Cómo hacer el bizcocho microondas o Mug cake en 3 minutos:
```

Update available (3.4.1) deb tar.gz 100% Feedback FREE

Resolución de incidencias(12/14)

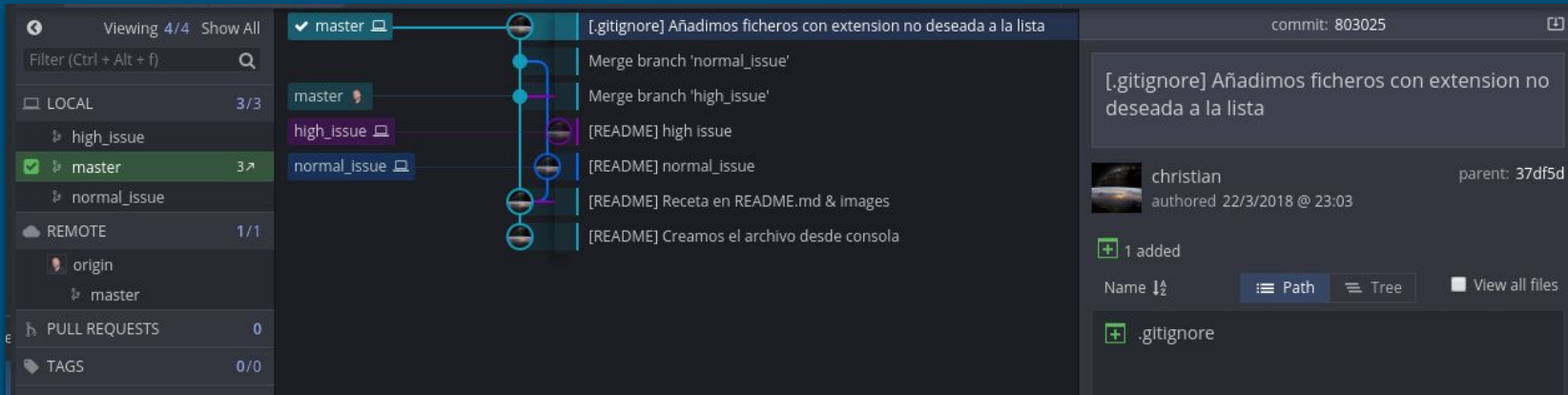
- Marcamos el conflicto como resuelto:

The screenshot displays the Git GUI interface during a merge conflict resolution. The top left pane shows a commit graph with branches: master (checked), high_issue, and normal_issue. The top right pane, titled 'Merge conflicts detected', shows the merge of 'normal_issue' into 'master' and lists 'Conflicted Files (1)' with 'README.md'. A 'Mark resolved' button is visible next to the file name. The bottom pane shows 'Staged Files (5)' including 'README.md-base', 'README.md-ours', and 'README.md-theirs'. A context menu is open over the 'README.md-ours' file, with the 'Ignore 3 files' option selected. The commit message at the bottom reads 'Merge branch 'normal_issue''.

- Ignoramos aquellos ficheros que no terminen en *.md
(se creará el fichero: .gitignore que posteriormente tendremos que commitear)

Resolución de incidencias(13/14)

- Hacemos commit de `.gitignore` y finalmente hacemos `push`



- Volvemos a estar en la rama `master`, y sin incidencias resueltas por el momento ;)

Resolución de incidencias(14/14)

- ¿Y si la liamos?...
 - `Reset master to this commit-->`
 - **Soft:** Guarda todos los cambios
 - **Mixed:** Mantiene una copia pero resetea el índice
 - **Hard:** Descarta todos los cambios
- Más info en:
 - <https://git-scm.com/book/es/v1/Fundamentos-de-Git-Deshaciendo-cosas>



¿Preguntas? :)

Fuentes consultadas

- https://es.wikipedia.org/wiki/Linus_Torvalds
- https://en.wikipedia.org/wiki/Junio_Hamano
- https://es.wikipedia.org/wiki/Fundaci%C3%B3n_Linux
- <https://git-scm.com>
- <https://github.com/join?source=header-home>
- <https://guides.github.com/>
- <https://guides.github.com/features/mastering-markdown/>
- <https://docs.gitlab.com/ce/ssh/README.html>
- <https://the.earth.li/~sgtatham/putty/0.67/html/doc/Chapter8.html#pubkey-puttygen>
- <https://www.sourcetreeapp.com/>
- <https://www.gitkraken.com/download>
- <https://support.gitkraken.com/how-to-install>
- <https://git-scm.com/book/es/v1/Fundamentos-de-Git-Deshaciendo-cosas>
- Curso de MiradaX: Gestión de proyectos Software con Git y GitHub

¡Muchas gracias por asistir!