

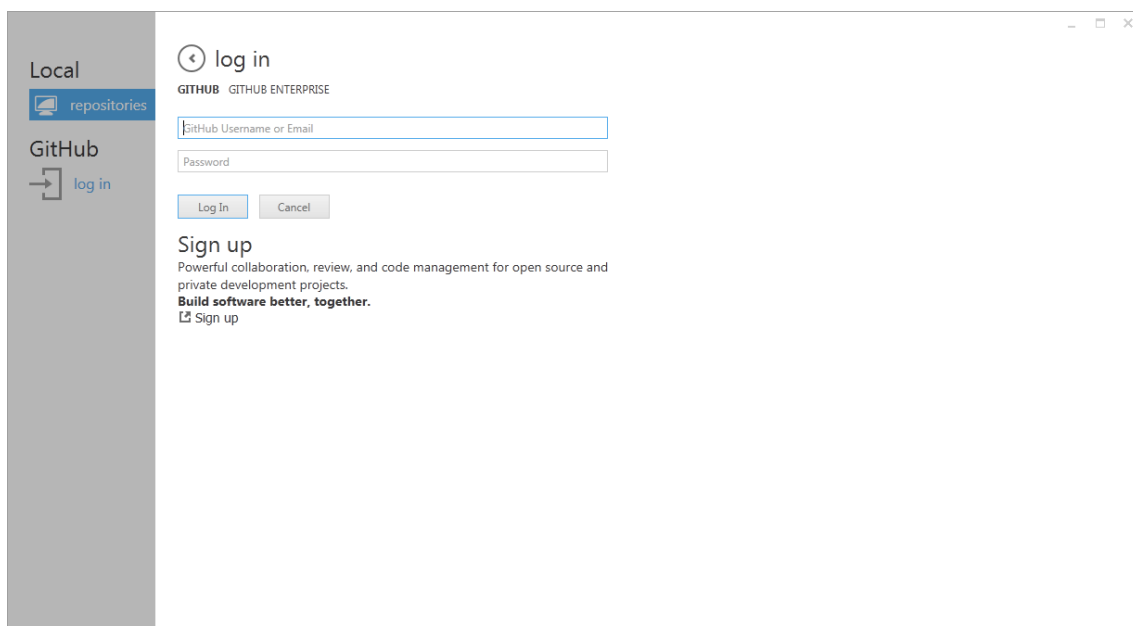
Manual de uso básico de GitHub con la herramienta de Windows y por el navegador web

GitHub posee una herramienta para Windows, la cual se puede descargar desde <http://windows.github.com/>

Para su uso hace falta tener una cuenta previamente creada en la página de GitHub <https://github.com/>

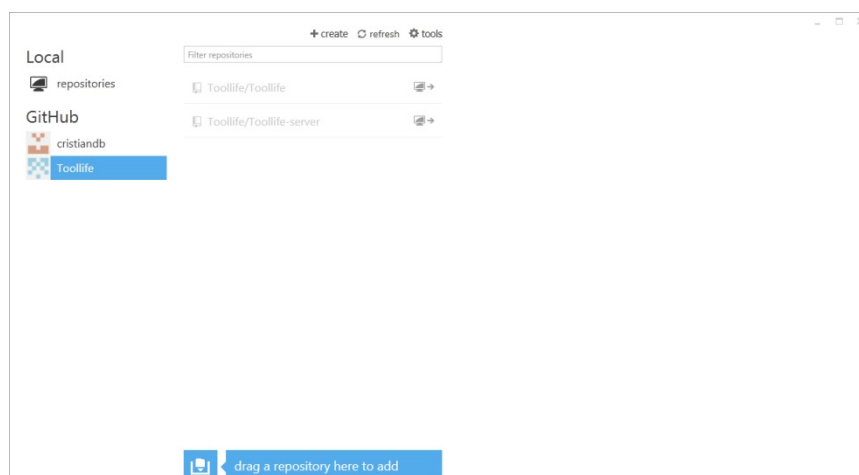
Log in

La primera vez que arrancas la aplicación se te pedirá que indiques tu cuenta, la cual será usada para gestionar tus repositorios.



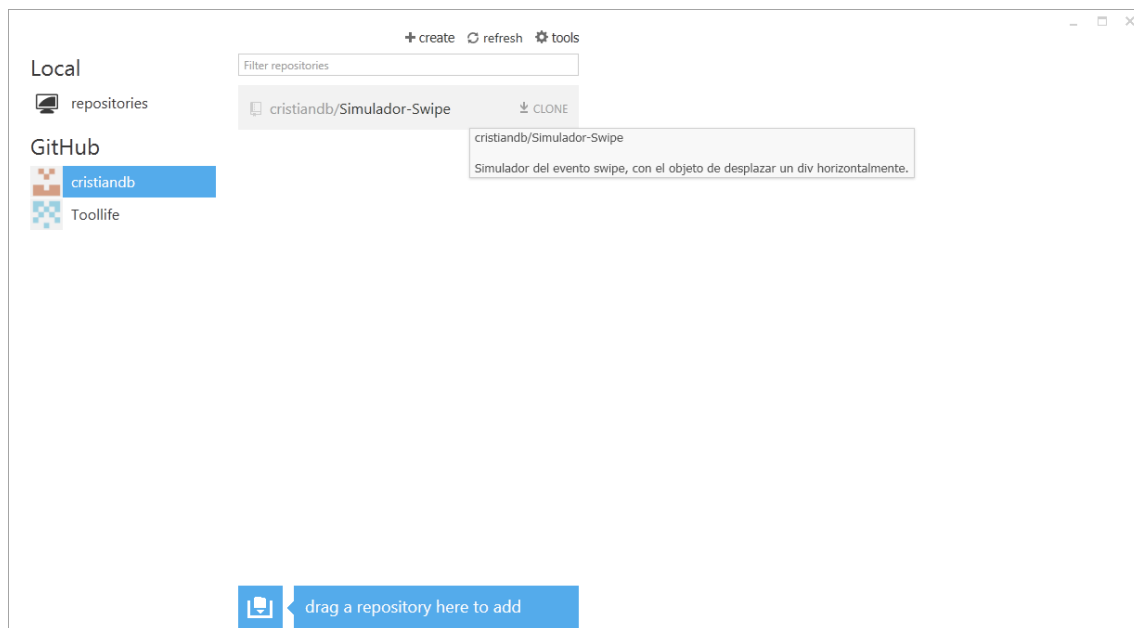
Repositorios

Una vez realizado el login, se nos mostrará la siguiente ventana:

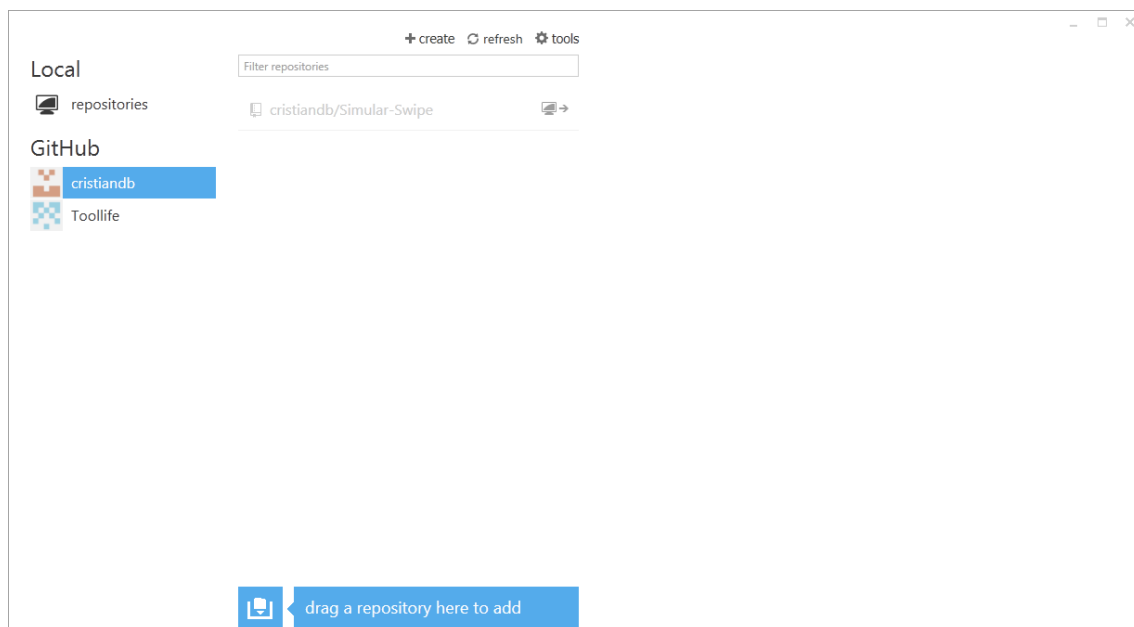
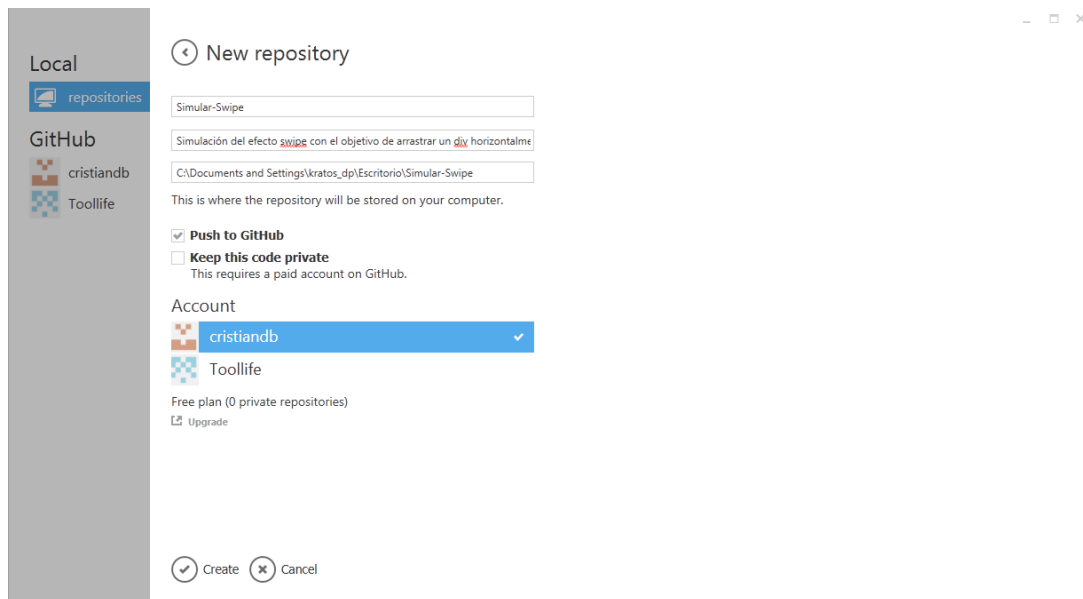


La sección “Local” te permite visualizar los repositorios que tienes almacenados en tu equipo (de forma local). La sección “GitHub” está formada por los grupos de desarrollo a los que perteneces y tu cuenta, si seleccionas alguno se te mostrará a la derecha los repositorios que hay en un determinado grupo/cuenta. Para crear un repositorio puedes arrastrar el directorio con el que estés trabajando dentro de la ventana o puedes pulsar el botón de créate.

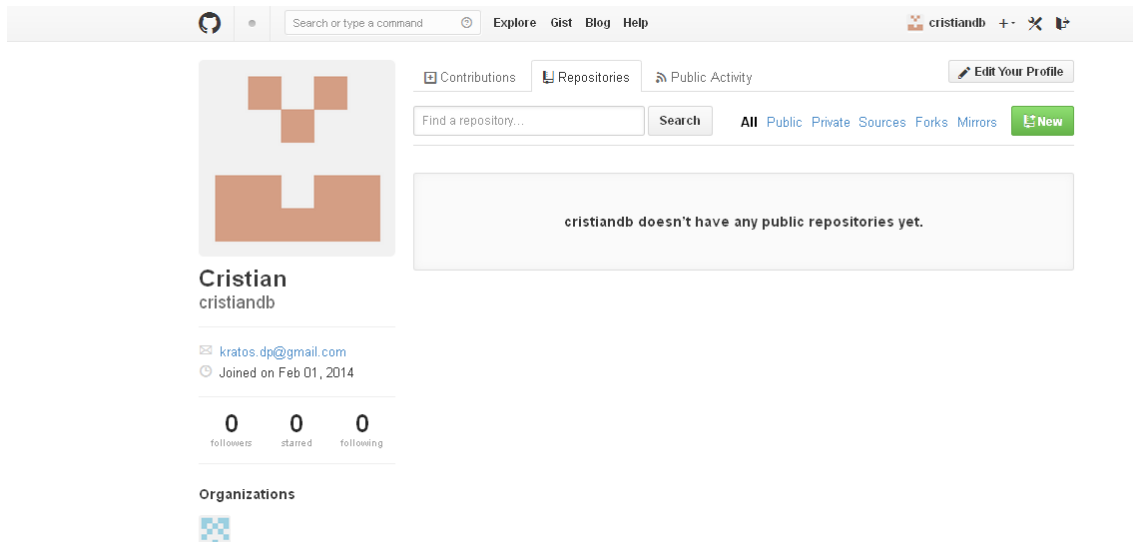
En el caso de que hayas creado un repositorio desde cualquier otro sitio, por ejemplo desde la página web, y no tienes una copia local, tan solo debes pulsar el botón CLONE, se te creará una copia local del repositorio.



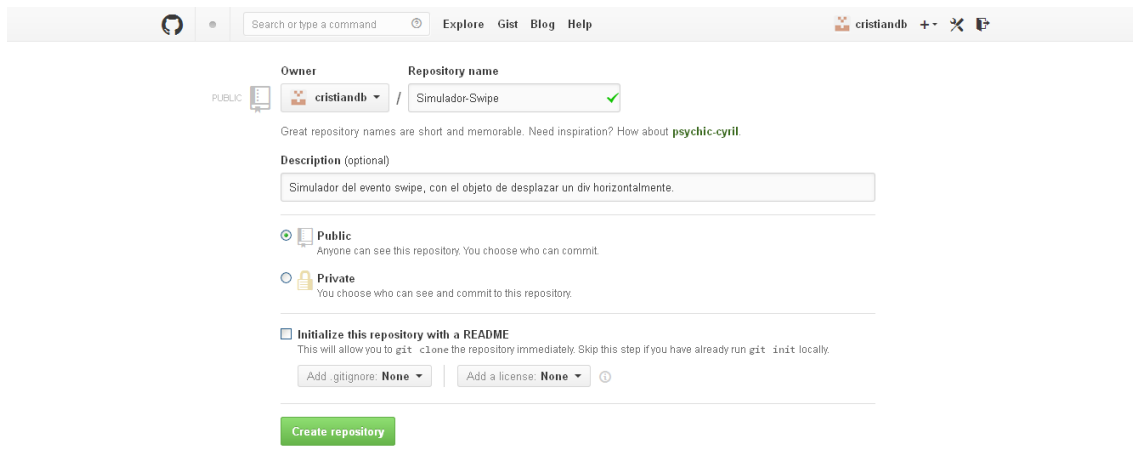
A continuación una captura de un repositorio que vamos a añadir a la cuenta cristiandb, en el momento en que es creado el repositorio de forma local, también lo hará en github, pero OJO!! No te sube los archivos del proyecto, tendrás que hacer un commit, el cual lo explicaremos a continuación.



Hemos visto como sería crear un repositorio local, ahora veremos cómo sería crear un repositorio desde GitHub. Ante todo debes realizar login en la página <https://github.com/>, una vez hecho podrás acceder a tu perfil y en la pestaña “Repositories” puedes crear un nuevo repositorio pulsando el botón “New”.



Se nos abrirá una página que nos indicará las propiedades del repositorio que queremos crear, para este caso indicaremos el nombre y usaremos las opciones por defecto.



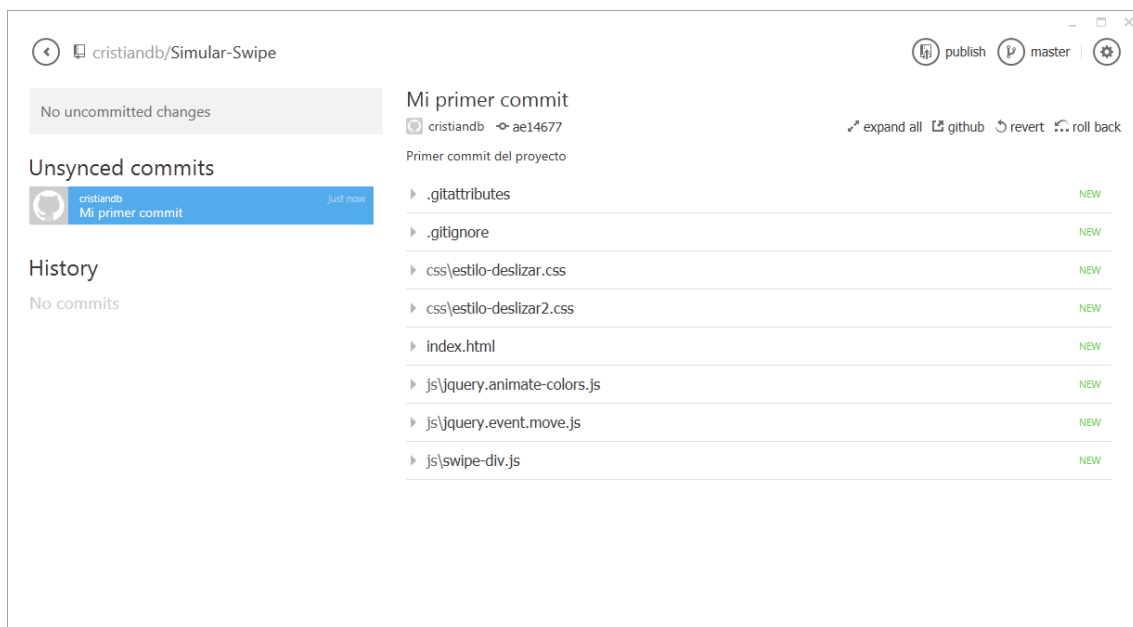
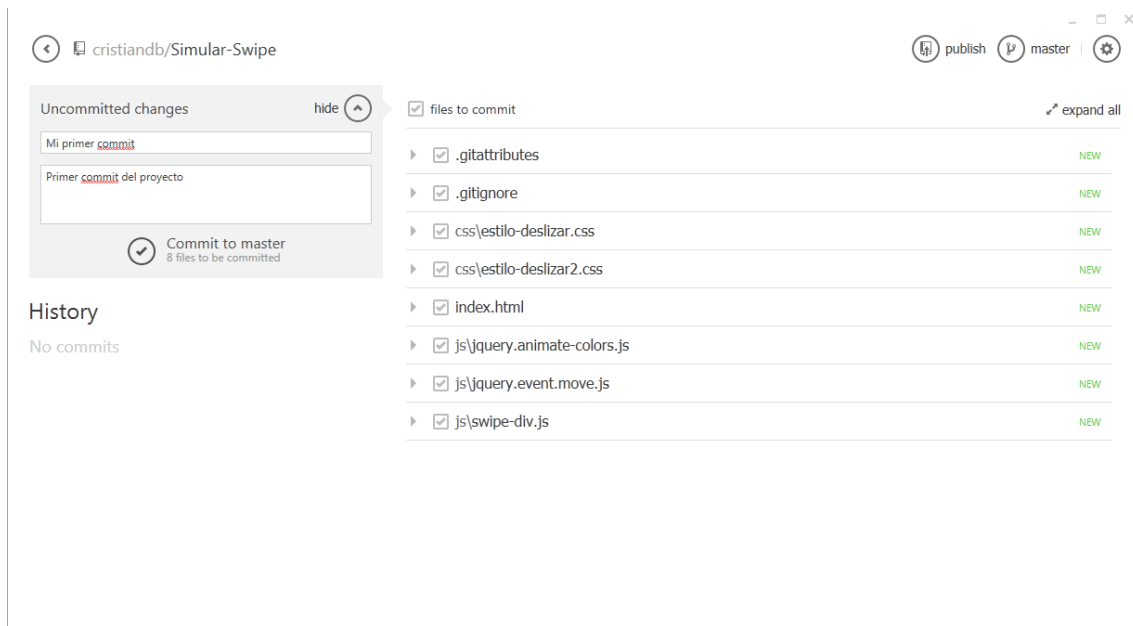
Creamos el repositorio y abrámoslo acabado.



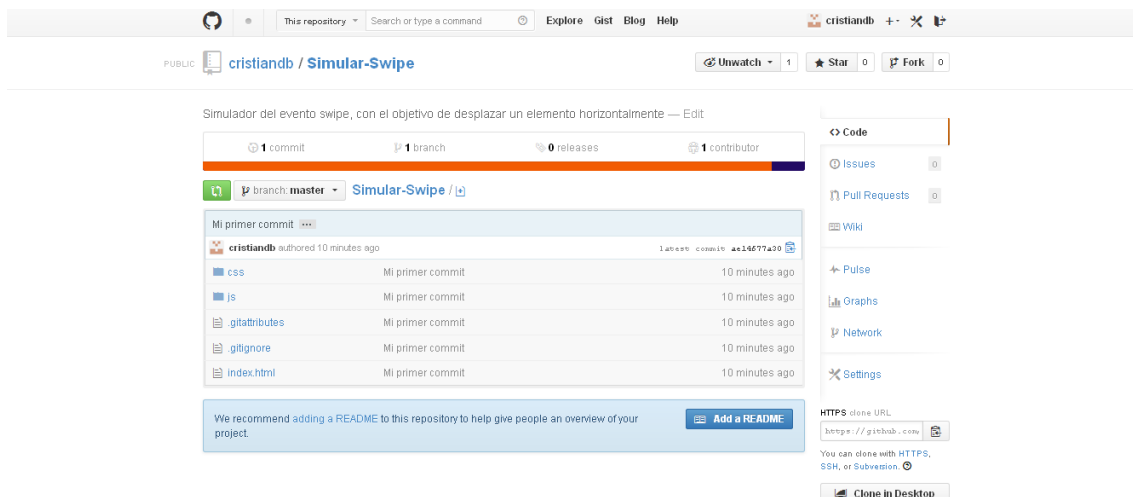
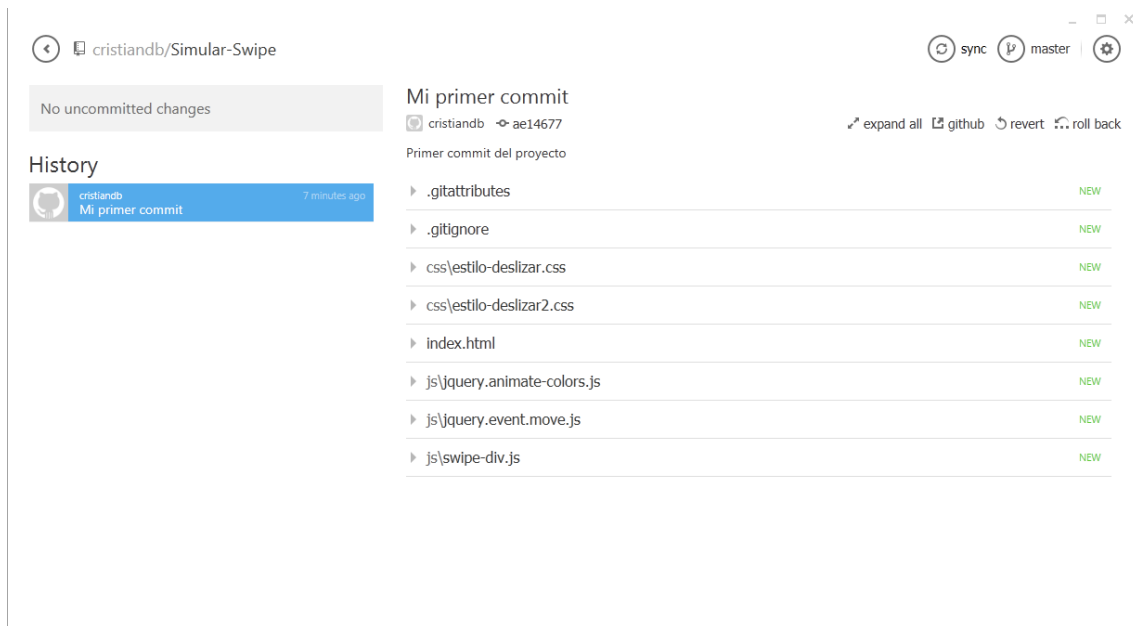
Commits

Para realizar commits se puede hacer desde la línea de comandos, desde una herramienta o añadiendo el archivo a mano desde la web, procedemos a explicar cómo hacer commits desde la herramienta de Windows. Si pulsamos en el icono del monitor, accedemos a una ventana que nos permite realizar la gestión del repositorio con GitHub. En nuestro caso, el repositorio que acabamos de crear en local contiene archivos, los cuales no están subidos en el servidor de GitHub, por lo tanto la herramienta nos muestra a la derecha los archivos que no hemos realizado un commit (en la sección “files to commit”). Cualquier archivo que modificamos en local se verá reflejado en esa lista.

Para realizar un commit, se debe indicar en el formulario de la izquierda (donde pone “Uncommitted changes”) una descripción sobre el commit que vas a realizar o un resumen, es obligatorio indicar uno de esos dos campos, y debemos indicar la rama en la que deseamos publicar el commit (más adelante explicaremos que es una rama/branch, en este caso publicamos en la rama por defecto, master) entonces se podrá pulsar el botón “Commit to master” y prepararemos el commit.



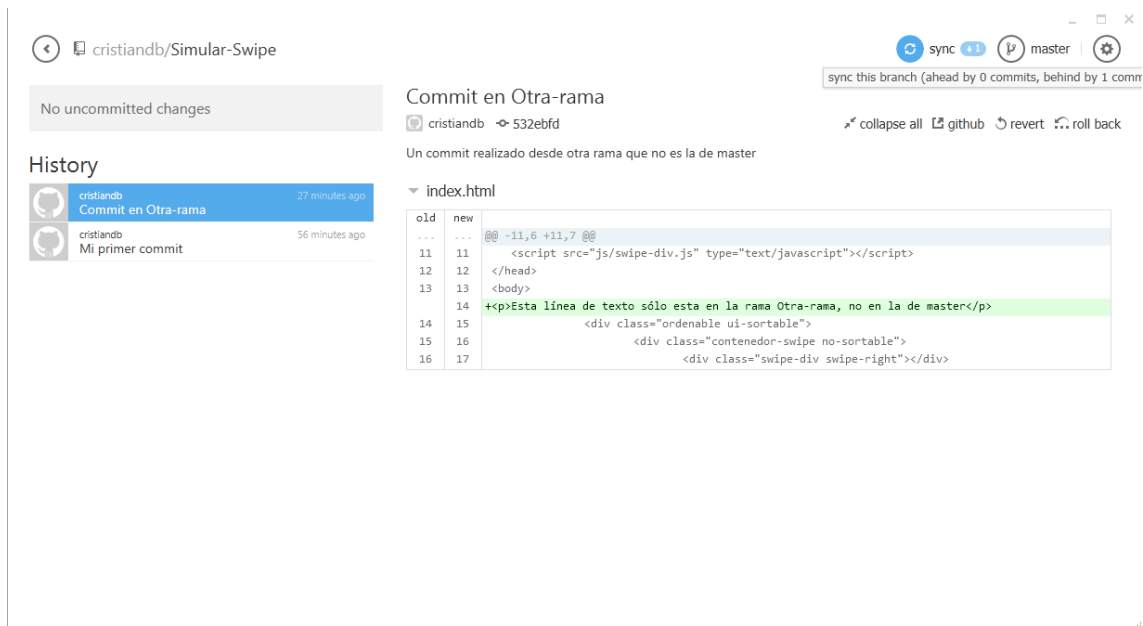
Todavía no hemos subido los cambios al repositorio de GitHub, lo hemos preparado. Podemos deshacerlo o revertir los cambios mediante los botones que están a la derecha, en este caso publicaremos el commit mediante el botón “publish”, una vez que finalice ya estaría los cambios propuestos en el commit en GitHub.



Los commit realizados en el repositorio aparecen en el historial (sección “History”), si pulsas en cualquiera de ellos podrás ver que archivos fueron modificados al hacer el commit, y lo más importante, podrás ver a la derecha que fue modificado en ese archivo.

Sincronizar el repositorio local con el repositorio de GitHub

Si hay algún cambio en el repositorio y no lo tienes en la copia local, te lo indicará el botón sync (debes seleccionar cada una de las ramas que tenga tu repositorio para ver si en alguna ha habido algún commit que tú no tengas en local). Tan solo debes pulsar el botón sync y automáticamente tendrás en local los cambios realizados sobre una determinada rama. También puedes pulsar el botón si deseas comprobar manualmente si tu copia local no corresponde con la que hay en el repositorio de GitHub.

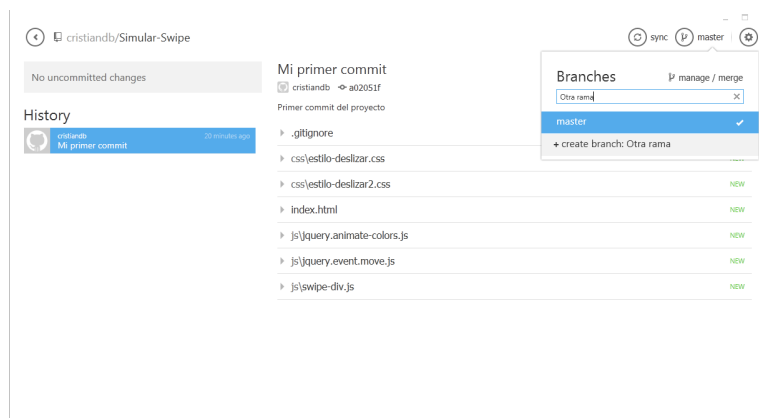


Ramas/branch

Cuando realizas un commit, debes indicar sobre que rama lo harás. Las ramas son, por así decirlo, “caminos alternativos de desarrollo” del proyecto. Es decir, cuando se desea realizar nuevas funcionalidades, pero no se desea afectar al proyecto (por ejemplo porque hay mucha gente trabajando en él y no quieres crear conflictos), creas una rama en la cual realizaras los commit, esos commit no se verán afectados en el resto de ramas del proyecto. De esta forma creas un “camino alternativo”, cuando desees que tu “camino” se vea reflejado en alguna de las ramas que hay en el proyecto, realizas un merge(mezcla) y pasas a unir las dos ramas.

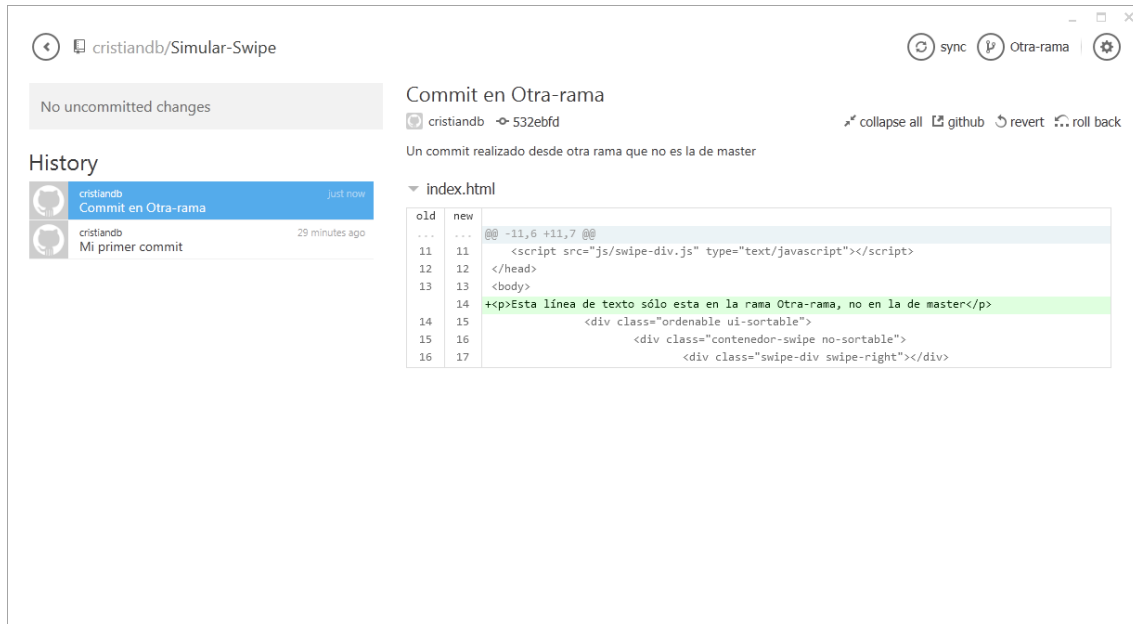
La filosofía de GitHub es trabajar sobre una rama maestra, si creas funcionalidades lo realizas en otras ramas y cuando estén realizadas realizas un merge sobre la rama maestra, de esa forma cada persona puede realizar funcionalidades de forma independiente y luego unificarlo con lo que haya realizado el resto de personas, de esta forma se pueden evitar conflictos si se han modificado los mismos archivos.

En la herramienta para crear o cambiar de ramas a la hora de realizar un commit se pulsa el siguiente botón:

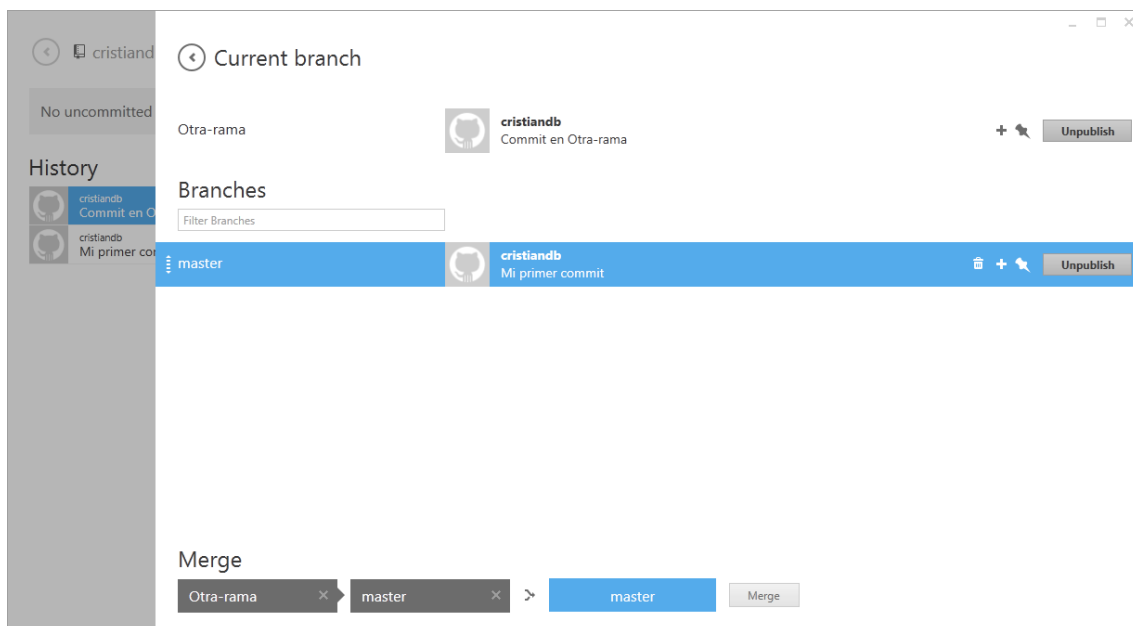


Para crear una rama tan solo habría que escribir el nombre de la rama y pulsar donde dice “create branch:”. Si deseas cambiar de rama tan solo debes seleccionar cual deseas, en este caso al solo haber una rama, está seleccionada la rama master.

A continuación mostramos un commit realizados con la rama “Otra-rama”.



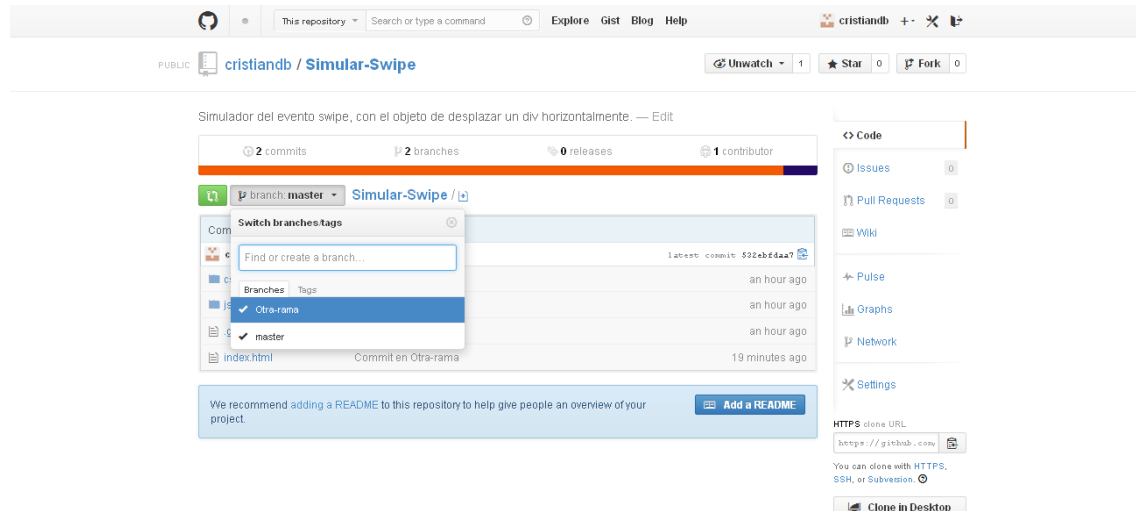
El cambio que hemos introducido tan solo está reflejado en la rama “Otra-rama”, vamos a realizar una mezcla para que la rama “master” tenga los cambios realizados. Para realizar una mezcla (merge) pulsamos en el botón “merge”, indicamos las dos ramas a mezclar arrastrándolas donde pone “drag branch here”, poniéndolo de la forma que la mezcla se vea reflejada en la rama que queramos y pulsamos el botón “merge”.



Pulsamos el botón “Current branch” y seleccionamos la rama en la que se han mezclado (en este caso “master”) y pulsamos el botón sync, de

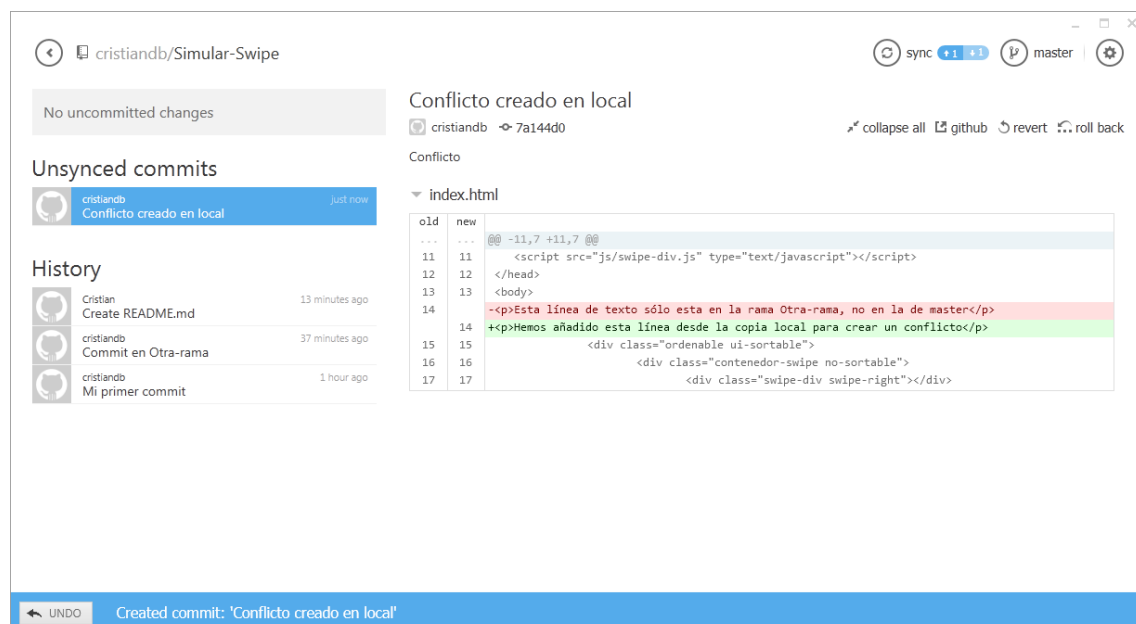
esta forma las dos ramas se habrán mezclado en la rama que hemos indicado como destino.

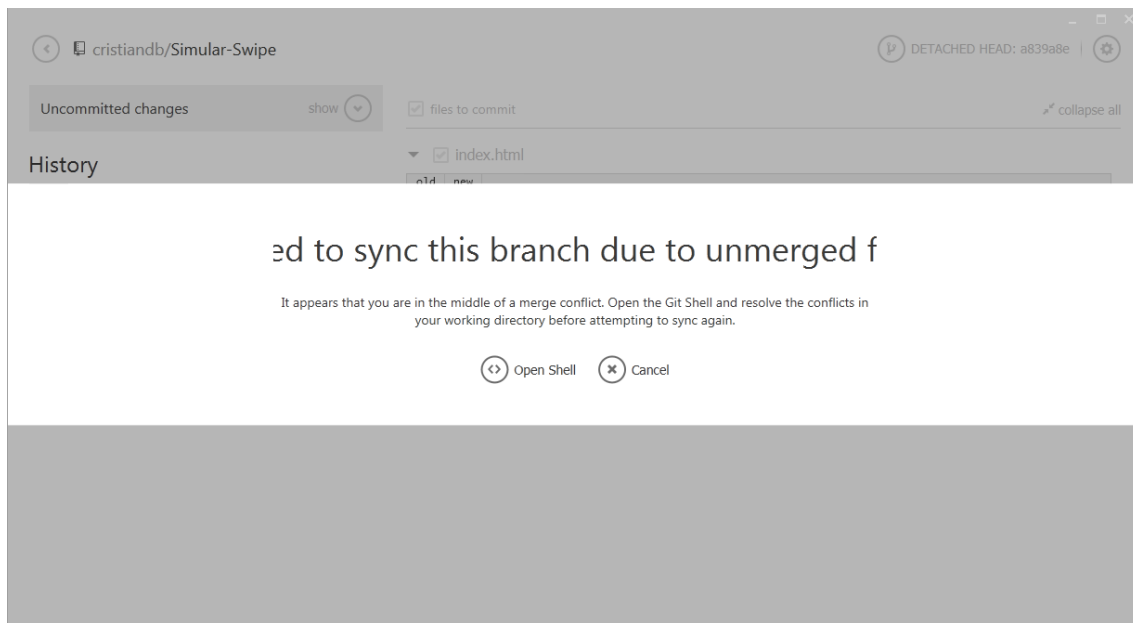
En la página de GitHub podemos ver en nuestros repositorios los archivos y modificaciones en cada rama de la siguiente forma:



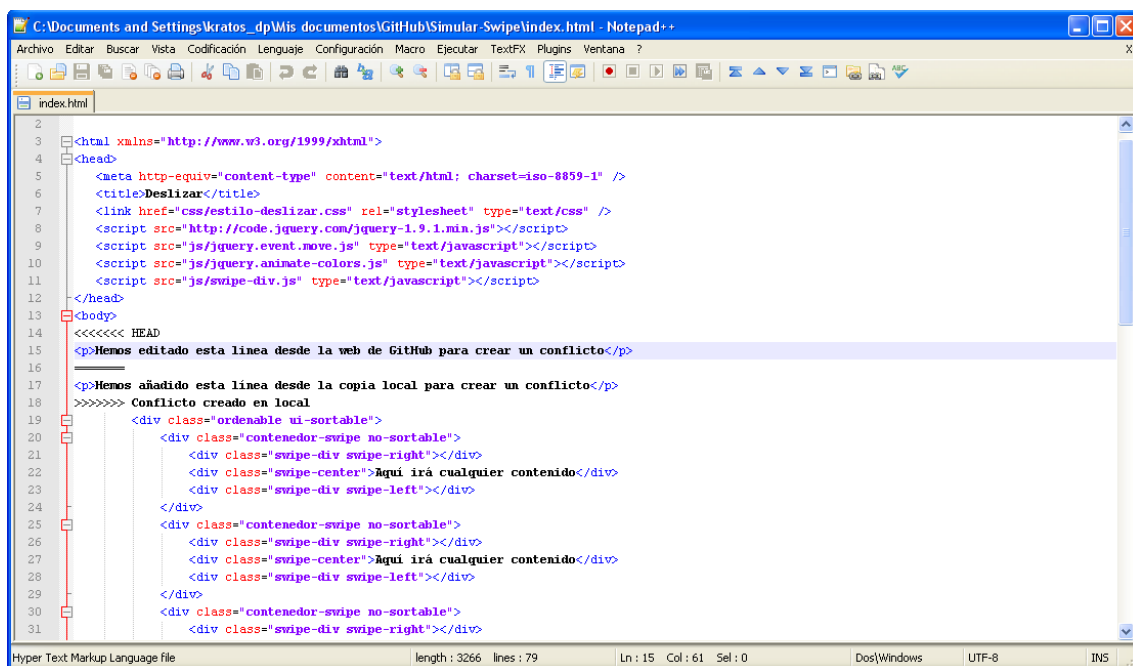
Conflictos

Un conflicto se produce cuando existe en local y en el repositorio de GitHub una versión modificada del mismo archivo, al intentar realizar el commit en local. Para poder ver un conflicto, hemos modificado el archivo `index.html` de la web de GitHub y hemos realizado un commit. Después se ha modificado el mismo archivo en la copia local y hemos intentado realizar un commit sin haber hecho un sync previamente (el cual nos informaba de que teníamos un commit pendiente en el repositorio de GitHub). Esto es lo que sucede:





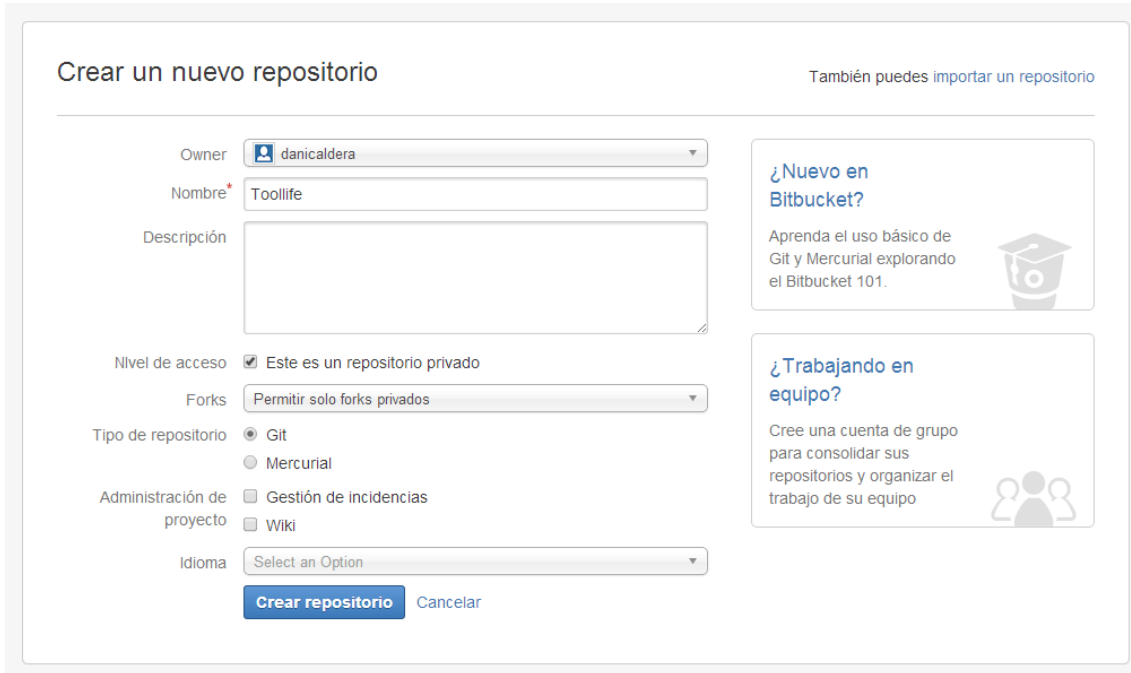
Y este es el aspecto actual del fichero modificado:



Para quedarnos con una versión definitiva del archivo tanto en la copia local como en el repositorio debemos modificar el archivo escogiendo el cambio con el que deseamos quedarnos y posteriormente realizar un commit, si se desea evitar este tipo de situaciones se es preferible crear ramas para evitarlo.

Manual de uso básico de Git con Git Bash, utilizando BitBucket como repositorio.

Creamos el repositorio en BitBucket:



The screenshot shows the 'Crear un nuevo repositorio' (Create a new repository) form in BitBucket. The form includes fields for Owner (danicaldera), Nombre (Toollife), and Descripción. It also has checkboxes for 'Este es un repositorio privado' (checked) and 'Permitir solo forks privados'. The repository type is set to 'Git'. There are also checkboxes for 'Gestión de incidencias' and 'Wiki'. A language dropdown is set to 'Select an Option'. On the right, there are two informational boxes: '¿Nuevo en Bitbucket?' with a graduation cap icon and '¿Trabajando en equipo?' with a group of people icon. At the bottom, there are 'Crear repositorio' and 'Cancelar' buttons.

Lanzamos la consola de git.

Se configura el usuario que vamos a utilizar en bitbucket:

```
git config --global user.name "Nombre Usuario"
```

```
git config --global user.email ejemplo@email.com
```

Nos situamos en el directorio de nuestro ordenador donde queremos crear el repositorio e iniciamos git utilizando el siguiente comando:

```
git init
```

Git nos permite establecer ficheros que se van a ignorar y por lo tanto no se tendrán en cuenta a la hora de hacer los commit y los push al servidor, para ello creamos un fichero en la raíz del repositorio llamado .gitignore

Añadir ficheros al índice local:

```
git add .
```

Realizar commit en la máquina local:

```
git commit -m "Mensaje"
```

Copiar repositorio a local (donde ruta puede ser una dirección web):

```
git clone ruta
```

Volver al último commit realizado:

```
git checkout -f
```

-f fuerza a que se sobrescriban los ficheros

Indicar el repositorio remoto en el cual vamos a trabajar:

```
git remote add origin https://dani@bitbucket.org/dani/toollife.git
```

Donde origin no es más que un nombre para hacer referencia al repositorio remoto, en caso de que origin ya esté asignado se puede usar cualquier otro nombre, o modificarlo con la siguiente línea:

```
git remote set-url origin https://dani@bitbucket.org/dani/toollife.git
```

Realizar los cambios en el servidor

```
git push -u origin master
```

Para crear una nueva rama:

```
git checkout -b NombredeRama
```

Para conocer en que rama estamos trabajando:

```
git branch
```

Saltar entre distintas ramas:

```
git checkout rama
```

Para mezclar las ramas nos situamos en las ramas en la que queremos que se integre el código nuevo y utilizamos el siguiente comando

```
git merge NombreRamaAMezclar
```

Borrar rama:

```
git branch -d Rama
```

Resumiendo, los pasos necesarios cada vez que queramos realizar un commit en el servidor local son los siguientes:

Añadir los cambios al índice local: `git add .`

Realizar el commit local: `git commit -m "Mensaje"`

Realizar la subida al servidor: `git push -u origin master`