Contenido

1 Introducción 2
1.1 Ventajas de GIT 2
2 Instalaciones recomendadas: 3
2.1 Extensiones de VSCode 3
2.2 Tema que estoy usando en VSCode 3
3 Configuración 3
3.1 Primeros comandos 4
3.2 Configurando 4
3.2.1 Establecer nombre de usuario 4
3.2.2 Configurar la dirección de correo del usuario 4
3.2.3 Activar colores de la interfaz 4
3.2.4 Ver las configuraciones 4
4 Creando el primer repositorio 4
5 ¿Qué hace git por nosotros hasta el momento? 7

Práctica 02 – Primeros comandos

1 Introducción

Git es un software de control de versiones diseñado por Linus Torvalds, pensando en la eficiencia, la confiabilidad y compatibilidad del mantenimiento de versiones de aplicaciones cuando estas tienen un gran número de archivos de código fuente. Su propósito es llevar registro de los cambios en archivos de computadora incluyendo coordinar el trabajo que varias personas realizan sobre archivos compartidos en un repositorio de código.

1.1 Ventajas de GIT

1- Facilita el trabajo colaborativo: Distintos programadores pueden estar editando el mismo archivo, o versiones distintas del mismo archivo, y todos los cambios serán reflejados en el documento final.

2- Reduce considerablemente los tiempos de deploy (despliegue) de un proyecto, al subir solamente los cambios (no los archivos cambiados, sólo los cambios!), que en Git se conoce como "diff": las diferencias entre la versión local (la que estás trabajando) y la "master" que está en el servidor central.

3- Permite regresar a versiones anteriores de forma sencilla y muy rápida. En caso de haber realizado cambios negativos en un proyecto en producción, volver a la última versión estable es un simple comando, que retrocede a su estado previo todos los cambios realizados en la última modificación. Esto puede hacerse hacia cualquier versión del proyecto, sin importar la cantidad o calidad de los cambios posteriores.

4- Permite generar flujos de trabajo que facilitan el desarrollo y mantenimiento de proyectos de gran tamaño.

5- El ecosistema Git es increíble, y agrega un montón de herramientas a nuestra disposición para facilitarnos el trabajo, de forma robusta, rápida y profesional.

A través de los “hooks” de Git, los distintos servicios pueden detectar cambios en el historial de versiones y realizar acciones automáticas (como actualizar los archivos en el servidor o ejecutar una suite de tests y enviarnos su resultado), ¡dejándonos tiempo libre para cosas más productivas!

6- Las "branches" o ramas, permiten trabajar con una base de código paralela al proyecto en sí, donde podemos corregir bugs o desarrollar nuevas características para el producto sin afectar el "master", pero manteniendo todas las ventajas de usar un sistema de control de versiones. Una vez que estamos contentos con nuestro "branch", podemos combinarlo con el "master" o, en lenguaje Git, hacer un "merge".

7- Empezar a trabajar desde otro entorno es tan fácil como "clonar" el proyecto a tu nuevo entorno, trabajar sobre los archivos que se quieran, y subir los cambios al "master" o a una "branch".

8- Sistema de etiquetas, para etiquetar las distintas versiones del proyecto. Esto es un marcador a una versión específica del proyecto, sólo que en lugar de tener distintos backups de versiones anteriores, apuntamos a distintas versiones dentro de la misma base de código.

2 Instalaciones recomendadas:

1. VSCode - Visual Studio Code
2. Google Chrome
3. Git

2.1 Extensiones de VSCode

Activitus Bar

2.2 Tema que estoy usando en VSCode

- Tokio Night
- Iconos

3 Configuración

3.1 Primeros comandos

```
git --version
```

Nos muestra la versión de git

Muestra ayuda sobre los comandos

Muestra ayuda sobre un comando concreto. Ejemplo:

3.2 Configurando

3.2.1 Establecer nombre de usuario

3.2.2 Configurar la dirección de correo del usuario

Se usa para anotar que usuario realizó cada acción

3.2.3 Activar colores de la interfaz

3.2.4 Ver las configuraciones

4 Creando el primer repositorio

Un repositorio de Git es un almacenamiento virtual de nuestro proyecto. Permite guardar versiones del código a las que se puede acceder cuando se necesite.

Creamos una carpeta para contener nuestro repositorio y extraemos dentro algunos archivos por ejemplo el contenido del archivo comprimido “01-bases.zip” que contiene una página web.

Crea la carpeta C:\Git\Tu_nombre y copia en ella la carpeta 01-bases

Desde la línea de comandos nos situamos en la carpeta que será nuestro repositorio:

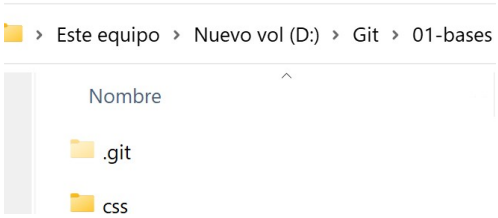
Añade una captura

En este caso la carpeta que será nuestro repositorio es 01-bases que es la que contiene la página web.

A continuación, iniciamos el repositorio con el comando:

Añade aquí una captura de pantalla

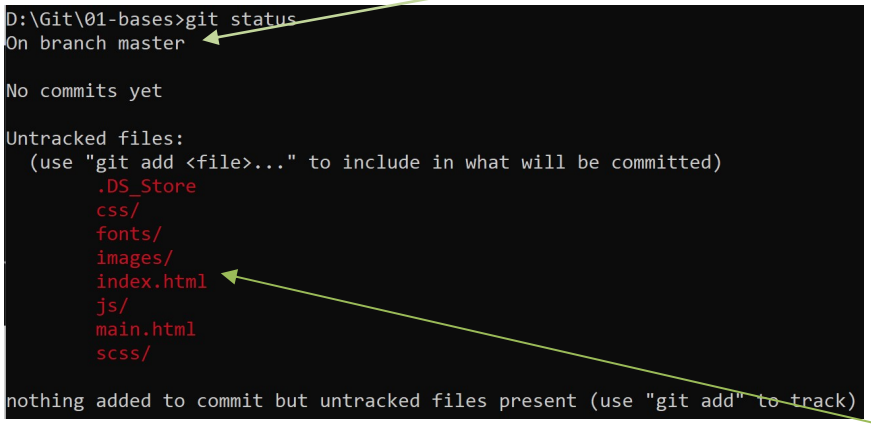
Dentro de la carpeta donde hemos creado el repositorio se crea la **carpeta oculta. git**:



Este directorio no se debe borrar porque es el directorio de trabajo de git.

Vamos a ver el estado de nuestro repositorio:

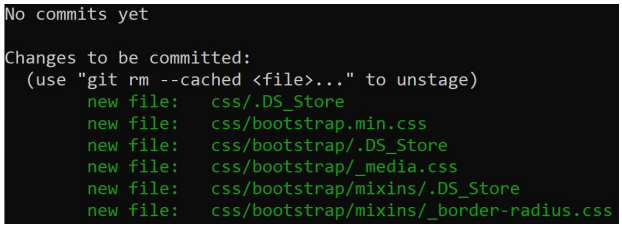
Vemos que estamos trabajando en la rama master que es la rama principal



Además, git nos indica que no estamos haciendo seguimiento de algunos archivos.

Vamos a indicar a git que queremos hacer seguimiento de un archivo:

Si vemos otra vez el estado con git status:



Vemos que ahora se nos muestran muchos archivos de los que hacemos seguimiento (los que están en color verde) ya que lo que hemos añadido en vez de un fichero suelto es un directorio y se han añadido todos los archivos dentro de él.

Si queremos añadir todo el contenido de la carpeta actual podemos usar el comando:

Añade aquí una captura de pantalla

El punto indica el directorio actual.

De la misma forma que podemos añadir seguimiento de archivos podemos eliminarlo con el comando:

Añade aquí una captura de pantalla

Cuando tengamos seleccionados los archivos de los que queremos hacer seguimiento procedemos a realizar una instantánea, es decir guardar el estado actual de los archivos. Para ello usamos el comando:

```
git commit –m “Primer commit”
```

Añade aquí una captura de pantalla

El texto entre comillas sirve para identificar la instantánea tomada.

Si usamos git status después de hacer el commit:

```
D:\Git\01-bases>git status
On branch master
nothing to commit, working tree clean
```

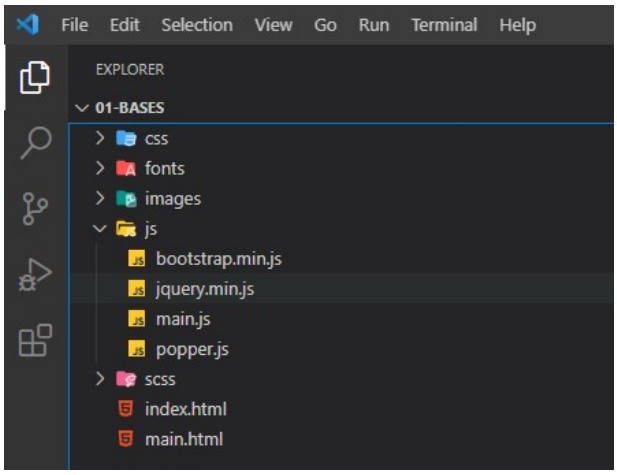
Nos indica que todo lo que habíamos configurado para hacer el seguimiento está guardado y no hay cambios que necesiten hacer commit ("nothing to commit").

Añade aquí una captura de pantalla

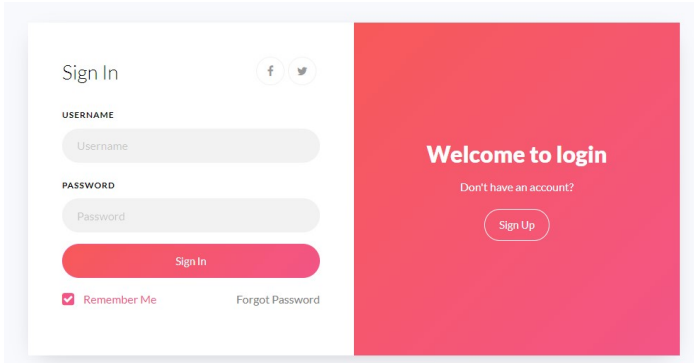
Reconstruye el proyecto al estado del último commit.

5 ¿Qué hace git por nosotros hasta el momento?

Arrastramos la carpeta 01-Bases al editor Visual Studio Code. Lo cual hace que se muestre toda la estructura de ficheros que contiene:



Si visualizamos el archivo index.html en el navegador vemos una página web:



Si modificamos el archivo index.html y borramos parte del contenido, puede llegar a no verse la página web, o verse de forma incorrecta como en la siguiente imagen:

Podemos arreglar las modificaciones de forma que la web se vuelva a ver correctamente:

Añade aquí una captura de pantalla