Git: distributed version control system

Documentación:

- Web: https://git-scm.com/
- Documentación: https://git-scm.com/doc
- Pro Git Book: https://git-scm.com/book/en/v2

Control de versiones de nuestro proyecto

Instrucciones básicas

1. Descargamos e instalamos desde: https://git-scm.com/. Aceptamos las opciones por defecto.
2. Podemos usar Git Bash o la ventana de comandos. Nosotros lo haremos desde el propio Visual Studio Code.
3. Nos situamos en la carpeta del proyecto.
4. *git help* nos proporciona una primera ayuda.
5. Iniciamos el control de versiones en nuestro proyecto: *git init*
6. Añadimos los ficheros al estado intermedio *stage* mediante: *git add .*
7. Se pueden añadir archivos concretos: *git add index.html*
8. Para deshacer el paso al stage: *git reset*
9. Para ver el estado de nuestras acciones: *git status*
10. Creamos una versión con *git commit -m "Primera versión del proyecto"*
11. Si hacemos cambios en el proyecto, repetiremos estos comandos para crear nuevas versiones:
    - *git add .*
    - *git commit -m "Descripción de los cambios hechos"*
12. Para hacer ambas accions conjuntamente: *git commit -am "New commit description"*
13. Para ver el historial de versiones: *git log*
14. Para deshacer commits y volver a versiones anteriores:
    - *git reset --hard <idversioncommit>*
    - Si queremos mantener los cambios en el WD: *git reset <idversioncommit>*
    - *git reset HEAD^*
    - *git reset HEAD~3*
15. Modificar la descripción del commit:
    - *git commit --amend -m "Nueva descripción del último commit"*

Ramas

16. Crear una rama: *git branch <name>*
17. Saltar entre las ramas: *git checkout branchname*
18. Crear y moverse a la rama: *git checkout -b branchname*
19. Para fusionar las ramas: *git merge <branch-name>*

20. Renombrar una rama: *git branch -m [branchname] new-branchname*
21. Borrar una rama: *git branch -d <branchname>*

Viendo las diferencias

22. Mostrar los cambios de un fichero en el working directory: *git diff <file>*
23. Mostrar los cambios de un fichero en staging area: *git diff --staged <file>*
24. Mostrar los cambios de un fichero respecto a commit: *git diff <commit>*
25. Comparar dos ramas: *git diff <branch1>..<branch2>*

Repositorio remoto

1. Nos registramos en GitHub: https://github.com/
2. Creamos un repositorio para nuestro proyecto. Por ejemplo: https://github.com/markogalarza/demo-dsm-bootstrap-2023.git
3. Asignamos el repositorio remoto a nuestro proyecto: *git remote add <remotename> https://github.com/markogalarza/demo-dsm-bootstrap-2023.git*
4. Lo confirmamos haciendo *git remote --v*
5. Hacemos push de la rama al repositorio remoto:
    - *git push <remotename> <branch>*
6. Lo confirmamos en GitHub. Observad también las versiones.
7. También podemos controlar Git desde Visual Studio Code.
8. Es muy práctica la extensión Git Graph.

Colaboración

9. Podemos clonar en local repositorios públicos: *git clone https://github.com/markogalarza/demo-dsm-bootstrap-2023.git*
10. Añadimos colaboradores desde la pestaña *Settings* del repositorio de GitHub.
11. Así los miembros del grupo pueden realizar *push* de su código, ramas, commits…
12. Para actualizar nuestra copia con los avances del resto del equipo: *git pull*

Ejercicio

1. Instalar Git e iniciarlo en el proyecto de Bootstrap.
2. Crear el repositorio remoto y sincronizarlo con el proyecto local.
3. Crear y commitear los avances en una nueva rama.
4. Fusionarla con la rama master.
5. Instalar la extensión Git Graph y comprobar el histórico en VS Code.