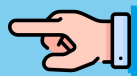
Terminal, Git, GitHub y GitHub Pages

Trabajando con Git y GitHub

{desafío} latam_

Crear un repositorio remoto en Github para controlar las versiones de un proyecto y publicar la página web utilizando Github Pages.

- Unidad 1: HTML y CSS
- Unidad 2: Bootstrap
- Unidad 3: JavaScript
- Unidad 4: Terminal, Git, GitHub y GitHub Pages



Te encuentras aquí

{desafío} latam_

¿Qué aprenderás en esta sesión?

- *Aplicar el procedimiento de habilitación de una página web con Github Pages, para que el contenido esté disponible públicamente.*

{desafío} latam_

/* Git branch */

Git branch

Conociendo las ramas del proyecto

- Una de las principales ventajas de git es la utilización de branch (ramas). Primero definiremos una rama simplemente como un camino donde está nuestro código, mientras que la definición técnica se refiere a un branch como un apuntador a donde van los commits que realizamos.

- Cada vez que inicializamos git, de forma automática se genera un branch main, que es donde se guardarán los nuevos commits.

Git branch

Conociendo las ramas del proyecto

- Al iniciar git en una carpeta, de forma automática se genera la rama main. Podremos conocer las ramas que tiene nuestro proyecto con el comando:

```
git branch
```

- Nos mostrará en consola todos los branch del proyecto. Como solo tenemos una rama, nos aparecerá **main**.

Git branch

¿Cuándo generar una nueva rama?

- Para crear una nueva branch utilizaremos el siguiente comando:

```
git branch nueva_branch
```

- Una vez creado, no cambiará de forma automática a la nueva rama, sino que nos quedaremos en la rama original. Para cambiarnos debemos hacerlo con el comando:

```
git checkout nueva_branch
```

Git branch

¿Cuándo generar una nueva rama?

- Si queremos crear una branch y situarnos enseguida en ella debemos ejecutar el siguiente comando:

```
git checkout -b otra_branch
```

- Veamos con el siguiente comando las ramas que hemos creado:

```
git branch
```

/* Git stash */

Git stash

- Existe una manera de reservar o “apartar” las modificaciones que estamos haciendo en nuestra rama actual para realizar una tarea emergente y asegurarnos que no perderemos los cambios que hemos realizado, para esto podemos optar por ocupar el comando:

```
git stash
```

Git stash

- Todos nuestros cambios han desaparecido, sin embargo, lo que sucedió es que fueron diferidos a un área en paralelo de nuestro historial de commits al cual podemos acceder con el siguiente comando:

```
git stash list
```

Git stash

- Para volver a unir el directorio actual con los cambios que estábamos trabajando y que fueron apartados del commit en el que nos encontrábamos.

```
git stash apply
```

- Este comando es muy útil para desarrollar sin temor a perder nuestras modificaciones, no obstante se recomienda usarlo en casos puntuales donde necesitemos de forma urgente solucionar problemas que fueron detectados mientras estamos trabajando en una nueva versión de nuestro proyecto.
-

/* Git Rebase */

Git rebase

- Cuando estamos trabajando en un equipo de desarrollo y varios usuarios manipulan el mismo repositorio en paralelo, es normal que se generen muchas ramas y muchos commits que dificultan la lectura del historial general, puesto que habrán modificaciones agrupadas en una rama A que serán unidas a una rama B a partir del último cambio generado.

- Para lograr un historial más limpio con menos ramas y menos commits distribuidos, git nos ofrece el siguiente comando:

```
git rebase otra_branch
```

/* GitHub Pages */

GitHub Pages

¿Qué es GitHub pages?

- GitHub pages, renderiza nuestro código de una rama específica en un dominio y espacio dentro de GitHub.

- Esto será de utilidad para poder mostrar nuestro trabajo de forma gratis, sin tener que recurrir a dominios o servidores externos.

GitHub Pages

Subiendo una página a GitHub pages

- Hay dos formas de realizar la subida a GitHub pages. La primera es manual a través de la consola:

- Crearemos la rama:

```
git branch gh-pages
```

- Recuerda que el nombre debe ser "gh-pages".

GitHub Pages

Subiendo una página a GitHub pages

- Ahora, pasaremos todos los commits de la rama main a la nueva rama, para luego subir los cambios al repositorio remoto.

```
git checkout gh-pages
```

```
git merge main
```

```
git push origin gh-pages
```

GitHub Pages

Subiendo una página a GitHub pages

- Finalmente, visitamos la página (reemplaza con tu nombre de usuario y nombre de tu repositorio donde corresponde).

```
https://TuNombreDeUsuario.GitHub.io/Nombre_de_tu_repositorio/
```

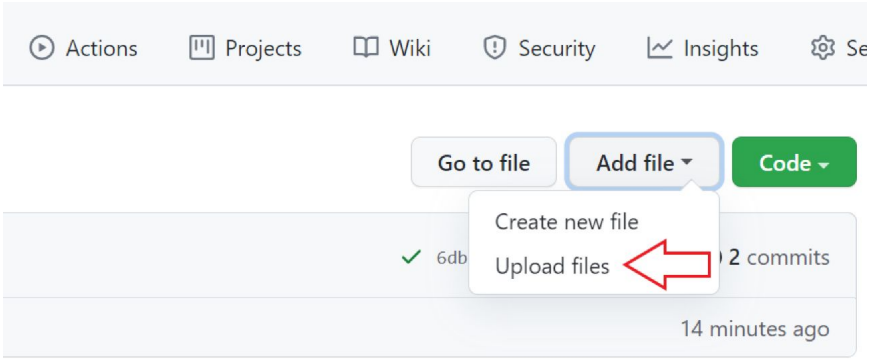
- Podremos ver la página almacenada en GitHub pages, que es completamente online. Se la puedes mostrar a quien quieras.

GitHub Pages

Subiendo una página a GitHub pages

- La segunda forma de hacer un despliegue de sitios web es a través de la interfaz gráfica de Github.

- Entrando al repositorio encontraremos un botón desplegable para agregar archivos a una rama y posteriormente arrastrar los mismos desde el explorador del sistema operativo, así como se muestra en las siguientes imágenes.



{desafío} latam_

Resumen

- En esta unidad hemos aprendido a trabajar con la terminal y sus comandos principales. Además, analizamos la importancia de versionar nuestro código y utilizar recursos tanto locales como remotos para gestionar proyectos.

- Estas herramientas son clave para trabajar desarrollando aplicaciones, pues nos permiten resguardar nuestro progreso, hacer pruebas de concepto sin impactar el código principal y lo que es más relevante aún, trabajar en equipos con responsabilidades diversas.

Próxima sesión...

- *Revisar material de estudio sincrónico que consiste en una **Guía de ejercicios** para practicar los conceptos aprendidos en todas las sesiones anteriores.*

{desafío} latam_

{desafío}
latam_

Academia de
talentos digitales