

Programa DevOps



Working Time #1

≡ Desafío 1 - Git

Es hora de que pongas en práctica todo lo aprendido. 🤖

Este apartado tiene el objetivo de ayudarte a seguir potenciando tus habilidades, por lo que a continuación encontrarás diferentes **desafíos** que podrás resolver de forma independiente y a tu ritmo.

En cada ejercicio conseguirás las resoluciones para que valides tus respuestas y puedas monitorear tu progreso. 😊

¡Manos a la obra!

1. Introducción:

Vamos a trabajar sobre comandos Git Básicos, estos comandos te permitirán conocer cómo manejar el código fuente, esto te permitirá manejar tanto tu propio código, como puede ser laC o Pipelines como también brindar ayuda a los desarrolladores con sus problemas al subir código a un repositorio.

2. ¿Dónde se lleva a cabo?

Línea de Comandos de GIT (git cmd)

Puede también, si su SO es Linux, utilizar git bash pero no voy a mostrar todos los comandos para Linux, deberá traducir los mismos de CMD a Bash en algunos casos.


3. Tiempo de dedicación

Entre 30 minutos y 1 hora.

4. Recursos

Instale la versión de Git para su Sistema Operativo.

<https://git-scm.com/downloads>

( Nota: Las versiones de git para Windows, Linux y macOS pueden variar sutilmente dependiendo del sistema operativo, esto es debido a que git CMD utiliza sintaxis de DOS mientras que git Bash utiliza la de Unix. Asimismo, puede haber variantes en la versión macOS debido al mismo punto.)

5. Desafío

Ejercicio 1

Configurar Git definiendo el nombre del usuario, el correo electrónico y activar el coloreado de la salida. Mostrar la configuración final.

SOLUCIÓN:

Abrir git cmd y posicionarse en el directorio donde desee trabajar

```
> git config --global user.name "TuNombreCopleto"
> git config --global user.email "tudireccion@decorreo"
> git config --global color.ui auto
> git config --list
```

Ejercicio 2

Crear un repositorio nuevo con el nombre libro y mostrar su contenido.

SOLUCIÓN:

Windows

```
> md libro
> cd libro
> git init
> dir /a
```

Linux

```
> mkdir libro  
> cd libro  
> git init  
> ls -la
```

Ejercicio 3

1. Comprobar el estado del repositorio.
2. Crear un fichero indice.txt con el siguiente contenido:

Capítulo 1: Introducción a Git
Capítulo 2: Flujo de trabajo básico
Capítulo 3: Repositorios remotos
3. Comprobar de nuevo el estado del repositorio.
4. Añadir el fichero a la zona de intercambio temporal.
5. Volver a comprobar una vez más el estado del repositorio.

SOLUCIÓN:

Windows

```
> git status  
> notepad.exe indice.txt  
Capítulo 1: Introducción a Git  
Capítulo 2: Flujo de trabajo básico  
Capítulo 3: Repositorios remotos  
(Guardar los cambios al salir)  
  
> git status  
> git add indice.txt  
> git status
```

Linux: Reemplazar notepad por cat y usar Ctrl+D para salir salvando cambios

Ejercicio 4

Realizar un commit de los últimos cambios con el mensaje "Añadido índice del libro." y ver el estado del repositorio.

SOLUCIÓN:

```
> git commit -m "Añadido índice del libro."  
> git status
```

Ejercicio 5

1. Cambiar el fichero indice.txt para que contenga lo siguiente:

Capítulo 1: Introducción a Git
Capítulo 2: Flujo de trabajo básico
Capítulo 3: Gestión de ramas
Capítulo 4: Repositorios remotos

2. Mostrar los cambios con respecto a la última versión guardada en el repositorio.

3. Hacer un commit de los cambios con el mensaje "Añadido capítulo 3 sobre gestión de ramas".

SOLUCIÓN:

```
> notepad.exe indice.txt  
    Capítulo 1: Introducción a Git  
    Capítulo 2: Flujo de trabajo básico  
    Capítulo 3: Gestión de ramas  
    Capítulo 4: Repositorios remotos  
(Guardar los cambios al salir)  
> git diff  
> git add indice.txt  
> git commit -m "Añadido capítulo 3 sobre gestión de ramas"
```

Ejercicio 6

1. Mostrar los cambios de la última versión del repositorio con respecto a la anterior.

2. Cambiar el mensaje del último commit por "Añadido capítulo 3 sobre gestión de ramas al índice."

3. Volver a mostrar los últimos cambios del repositorio.

SOLUCIÓN:

```
> git show  
> git commit --amend -m "Añadido capítulo 3 sobre gestión de ramas al índice."  
git show
```

Ejercicio 7

1. Mostrar el historial de cambios del repositorio.
2. Crear la carpeta capitulos y crear dentro de ella el fichero capitulo1.txt con el siguiente texto:

Git es un sistema de control de versiones ideado por Linus Torvalds.

3. Añadir los cambios a la zona de intercambio temporal.
4. Hacer un commit de los cambios con el mensaje "Añadido capítulo 1."
5. Volver a mostrar el historial de cambios del repositorio.

SOLUCIÓN:

```
> git log  
> md capitulos  
> notepad.exe capitulos/capitulo1.txt  
Git es un sistema de control de versiones ideado por Linus Torvalds.  
(Guardar los cambios al salir)  
> git add .  
> git commit -m "Añadido capítulo 1."  
> git log
```

Ejercicio 8

1. Crear el fichero capitulo2.txt en la carpeta capitulos con el siguiente texto:

El flujo de trabajo básico con Git consiste en: 1- Hacer cambios en el repositorio. 2- Añadir los cambios a la zona de intercambio temporal. 3- Hacer un commit de los cambios.

2. Añadir los cambios a la zona de intercambio temporal.
3. Hacer un commit de los cambios con el mensaje "Añadido capítulo 2."
4. Mostrar las diferencias entre la última versión y dos versiones anteriores.

SOLUCIÓN:

```
> notepad.exe capitulos/capitulo2.txt  
El flujo de trabajo básico con Git consiste en:  
1- Hacer cambios en el repositorio.  
2- Añadir los cambios a la zona de intercambio temporal.  
3- Hacer un commit de los cambios.
```

(Guardar los cambios al salir)

```
> git add .
> git commit -m "Añadido capítulo 2."
> git diff HEAD~2..HEAD
```

Ejercicio 9

1. Crear el fichero capitulo3.txt en la carpeta capitulos con el siguiente texto:

Git permite la creación de ramas lo que permite tener distintas versiones del mismo proyecto y trabajar de manera simultánea en ellas.

2. Añadir los cambios a la zona de intercambio temporal.

3. Hacer un commit de los cambios con el mensaje "Añadido capítulo 3."

4. Mostrar las diferencias entre la primera y la última versión del repositorio.


SOLUCIÓN:

```
> notepad.exe capitulos/capitulo3.txt
```

Git permite la creación de ramas lo que permite tener distintas versiones del mismo proyecto y trabajar de manera simultánea en ellas.

(Guardar los cambios al salir)

```
> git add .
> git commit -m "Añadido capítulo 3."
> git log
```

 Nota: Si luego de ejecutar git log la pantalla no responde, presionar la tecla Q para volver a git cmd. Cada commit tiene su propio hash, que puede verse como un código en el log. Ej: bd7f788903cf1b1adbde028d474068cebb6ac870

Reemplace en el siguiente comando por su hash lo que está entre <>

```
> git diff <código hash de la primera versión>..HEAD Esto le indicará las diferencias entre ambas versiones
```

Ejercicio 10


1. Añadir al final del fichero indice.txt la siguiente línea:

Capítulo 5: Conceptos avanzados

2. Añadir los cambios a la zona de intercambio temporal.
3. Hacer un commit de los cambios con el mensaje "Añadido capítulo 5 al índice."
4. Mostrar quién ha hecho cambios sobre el fichero indice.txt.

SOLUCIÓN:

```
> Notepad.exe indice.txt
(Agregue al final del archivo)
"Capítulo 5: Conceptos avanzados"
(Guardar los cambios al salir)
> git add .
> git commit -m "Añadido capítulo 5 al índice."
> git annotate indice.txt
```

 Nota: El comando Annotate permite ver los cambios de un archivo en particular, de todos modos, recomendamos usar alguna herramienta para ver diferencias entre archivos en lugar de este comando, como pueden ser winmerge o visual studio. Que visualmente le darán los cambios de manera más sencilla. De todos modos no lo haremos en este momento.

