

# Fundamentos de programación

## Git & GitHub

### Intro al control de versiones

#### ¿Qué es el Control de Versiones?

Un sistema de control de versiones (VCS, por sus siglas en inglés) se encarga de mantener nuestro código de manera organizada y segura. Permite mantener un registro de todos los cambios realizados en el código, incluyendo quién los realizó y cuándo. Además, se pueden añadir mensajes al registrar un cambio para ayudar a entender las modificaciones realizadas. Estos cambios se llaman commits, y cada commit tiene un hash único que lo identifica, lo que permite referirse a cambios específicos al interactuar con el VCS.

#### Tipos de Sistemas de Control de Versiones

##### Sistemas Centralizados

Los sistemas centralizados, como Subversion, requieren de un servidor para gestionar los cambios y los usuarios. La desventaja es que si el servidor falla, no es posible gestionar ni compartir los cambios de manera sencilla.

##### Sistemas Distribuidos

Los sistemas distribuidos, como Git o Mercurial, permiten que cada usuario sea tanto servidor como cliente. Los usuarios pueden trabajar localmente sin conexión, añadiendo cambios y ramas, y luego sincronizar dichos cambios con servidores remotos. En caso de fallo del servidor, cualquier usuario puede sustituirlo temporalmente hasta que se restaure el sistema. Aunque son distribuidos, normalmente se utiliza un servidor central para tener una única fuente de conocimiento.

#### Versionado Semántico (Semantic Versioning)

El versionado semántico es una convención y un estándar que ayuda a transmitir el tipo de cambios introducidos en el sistema o aplicación.

- **Major (Versión Principal):** Representa la versión principal del software. Incrementar este número indica que el sistema puede dejar de ser compatible con versiones anteriores.
- **Minor (Versión Secundaria):** Se incrementa al añadir nuevas funcionalidades que no rompen la compatibilidad con versiones anteriores.
- **Patch (Corrección):** Se incrementa al añadir reparaciones o correcciones en el código.

#### Instalación de Git

Para instalar Git, puedes seguir las instrucciones disponibles en la página oficial de Git:

- **Windows:** Descarga y ejecuta el instalador.
- **Mac:** Sigue las instrucciones específicas para macOS.
- **Linux:** Utiliza el gestor de paquetes de tu distribución para instalar Git.

#### Uso de Git y GitHub

##### Configuración Inicial

###### 1. Configurar el nombre de usuario y correo electrónico:

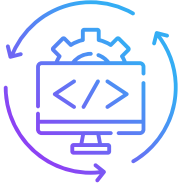
```
git config --global user.name "Tu Nombre"
git config --global user.email "tu.email@example.com"
```

##### Flujo de Trabajo Básico con Git

###### Clonar un repositorio existente:

```
git clone https://github.com/usuario/repositorio.git
```





## Intro al control de versiones

**Crear una nueva rama:**

```
git checkout -b nombre-de-la-rama
```

**Hacer cambios y añadirlos al área de preparación:**

```
git add .
```

**Hacer un commit con un mensaje descriptivo:**

```
git commit -m "Descripción del cambio"
```

**Enviar los cambios al repositorio remoto:**

```
git push origin nombre-de-la-rama
```

**Recursos Adicionales Recomendados**

- [Pro Git Book](#): Un libro gratuito y completo sobre Git.