Sesión 1: Git básico

Fundamentos de Git: arquitectura interna (working directory, staging area, repository), tipos de objetos y flujo de trabajo mínimo.

Objetivos

- Entender el modelo mental de Git y sus áreas.
- Configurar Git por primera vez y crear tu primer repositorio.
- Hacer commits atómicos y revisar historial con confianza.

Requisitos previos

- Conexión a internet y permisos para instalar software.
- Un editor de texto (VS Code, Vim, Nano, Notepad++, etc.).

Instalación de Git

- Linux

```
# Debian/Ubuntu
sudo apt update && sudo apt install -y git
# RHEL/CentOS/Fedora
sudo dnf install -y git || sudo yum install -y git
```

- Windows

```
winget install --id Git.Git -e
```

- macOS

```
brew install git
# o instala Xcode Command Line Tools
xcode-select --install
```

Configuración inicial

```
git config --global user.name "Tu Nombre"
git config --global user.email "tu.email@example.com"
# Rama por defecto y estrategia de pull
git config --global init.defaultBranch main
git config --global pull.rebase false
# Ver configuración
git config --list --show-origin
```

Conceptos clave

- Working directory: tus archivos en la carpeta del proyecto.
- Staging area (índice): cambios preparados para commit.
- Repository (.git): historial, objetos y referencias.
- Objetos: blob (contenido), tree (directorios), commit (snapshot + metadatos), tag.
- HEAD: puntero a la rama actual o a un commit.
- Ramas: punteros movibles a commits (baratas y rápidas).
- Remotos: repos en servidores (GitHub/GitLab/Bitbucket, etc.).

Flujo mínimo de trabajo

1) Inicializa repo y primer commit

```
mkdir mi-repo && cd mi-repo
git init
printf "# Mi Proyecto\n" > README.md
git add README.md
git commit -m "chore: inicializa repo con README"
```

1) Cambios iterativos

```
echo "Primera línea" >> notas.txt
git add -p
git commit -m "feat: agrega notas iniciales"
```

1) Inspecciona el historial

```
git log --oneline --graph --decorate --all
git show HEAD
git diff HEAD~1..HEAD
```

.gitignore rápido

```
cat > .gitignore << 'EOF'
# Dependencias
node_modules/
# Entornos
.env
# Editor
.vscode/
.DS_Store
EOF

git add .gitignore
git commit -m "chore: añade .gitignore"
```

Conectar un remoto (patrones por proveedor)

- GitHub: https://github.com/ORG/REPO.git o `git@github.com:ORG/REPO.git`
- GitLab: https://gitlab.com/ORG/REPO.git o `git@gitlab.com:ORG/REPO.git`
- Bitbucket: https://bitbucket.org/ORG/REPO.git o `git@bitbucket.org:ORG/REPO.git`

Ejemplo (HTTPS):

```
git branch -M main
git remote add origin https://github.com/ORG/REPO.git
git push -u origin main
```

SSH requiere claves (ver Sesión 4).

Buenas prácticas de commits

- Un cambio por commit; mensaje con prefijo semántico (feat, fix, chore, docs, refactor, test).
- Primera línea ≤ 72 caracteres; el cuerpo explica el porqué.

Ejercicio rápido

- Crea repo local, añade README y .gitignore.
- Genera 2–3 commits pequeños.
- Explora historial con `git log` y `git show`.