

# TP Git – Branches et Merge

## Objectifs pédagogiques

- Comprendre l'intérêt des branches dans Git
- Créer et basculer entre des branches
- Fusionner des branches avec `merge`
- Résoudre les conflits simples

**Astuce :** Les branches permettent de travailler sur des fonctionnalités différentes sans impacter la branche principale (`main`).

## Contexte

Vous allez créer un dépôt local avec un fichier README, puis expérimenter les branches, modifications séparées et fusion.

## 1 Initialisation du dépôt

```
mkdir tp-branches
cd tp-branches
git init
echo "# TP Branches" > README.md
git add README.md
git commit -m "Initialisation du projet"
```

**Explication :** Vous avez créé un dépôt local avec un premier commit sur la branche `main`.

## 2 Créer et basculer sur une nouvelle branche

```
git branch nouvelle-fonctionnalite
git checkout nouvelle-fonctionnalite
```

**Explication :**

- `git branch nouvelle-fonctionnalite` : crée une nouvelle branche.
- `git checkout nouvelle-fonctionnalite` : bascule sur cette branche pour y travailler.

### 3 Modifier un fichier sur la branche

Ajoutez une ligne dans README.md :

```
echo "Fonctionnalit ajout e sur la branche" >> README.md
git add README.md
git commit -m "Ajout d'une fonctionnalit sur la branche"
```

**Explication :** Vos modifications sont maintenant spécifiques à cette branche et n'affectent pas main.

### 4 Revenir sur la branche principale

```
git checkout main
```

**Explication :** Vous êtes revenu sur la branche principale. Les changements effectués sur nouvelle-fonctionnalite ne sont pas encore visibles ici.

### 5 Fusionner la branche

```
git merge nouvelle-fonctionnalite
```

**Explication :** La commande `git merge` intègre les modifications de la branche nouvelle-fonctionnalite dans main.

### 6 Résolution des conflits (optionnel)

Si vous modifiez la même ligne sur les deux branches, un conflit apparaîtra. Git indiquera les fichiers concernés et il faudra choisir quelle version conserver.

```
# Exemple d' dition pour r soudre le conflit
nano README.md
git add README.md
git commit -m "R solution du conflit"
```

**Explication :** Les conflits sont fréquents lors du travail en équipe. Il est important de savoir les résoudre proprement.

## Conclusion

Vous avez appris à :

- Créer et basculer entre des branches
- Travailler indépendamment sur une branche
- Fusionner une branche dans la principale

— Gérer les conflits simples

**Conseil :** Habituez-vous à créer des branches pour chaque nouvelle fonctionnalité ou correction. Cela rend le projet plus sûr et plus facile à gérer.