Git: les branches (exemple)

👍 Voici une **fiche d'exercice pédagogique complète** sur la manipulation des branches en Git, avec **consignes** et **solutions commentées**.

6TTR 5TTR 6TQ Découverte

Objectif

- Créer, explorer, fusionner et supprimer des branches.
- Comprendre comment Git isole les développements dans des branches différentes.

Partie 1 – Mise en place du projet

Consignes (élève)

1. Crée un dossier de travail et initialise un dépôt Git.
2. Ajoute un premier fichier avec un contenu simple.
3. Fais ton premier commit dans `main`.

Solution (enseignant)

```
mkdir projet-branches
cd projet-branches
git init
echo "Version initiale" > fichier.txt
git add .
git commit -m "Commit initial"
```

👉 L'historique contient maintenant un premier commit sur la branche `main`.

Partie 2 – Créer et travailler sur une branche

Consignes (élève)

1. Crée une branche nommée `feature-texte`.
2. Déplace-toi dessus.
3. Modifie le fichier pour y ajouter une ligne.
4. Enregistre ton travail avec un commit.

Solution (enseignant)

```
git checkout -b feature-texte
echo "Ajout depuis la branche feature-texte" >> fichier.txt
git commit -am "Ajout d'une ligne dans feature-texte"
```

👉 La branche `feature-texte` contient maintenant un commit qui n'existe pas encore dans `main`.

Partie 3 – Comparer les branches

Consignes (élève)

1. Reviens sur `main`.
2. Affiche le contenu du fichier.
3. Compare avec le contenu de `feature-texte`.

Solution (enseignant)

```
git checkout main
cat fichier.txt
# Résultat attendu : seulement "Version initiale"
```

Puis :

```
git checkout feature-texte
cat fichier.txt
# Résultat attendu : "Version initiale" + "Ajout depuis la branche feature-texte"
```

👉 Les branches isolent bien les changements.

Partie 4 – Fusionner une branche dans `main`

Consignes (élève)

1. Reviens sur `main`.
2. Fusionne la branche `feature-texte`.

3. Vérifie que les modifications apparaissent bien dans `main`.

Solution (enseignant)

```
git checkout main
git merge feature-texte
cat fichier.txt
# Résultat attendu :
# Version initiale
# Ajout depuis la branche feature-texte
```

👉 Un nouveau commit de merge est créé (sauf si fast-forward).

Partie 5 – Nettoyer les branches

Consignes (élève)

1. Supprime la branche `feature-texte` devenue inutile.
2. Vérifie la liste des branches restantes.

Solution (enseignant)

```
git branch -d feature-texte
git branch
# Résultat attendu : seule la branche main reste
```

👉 Bon réflexe : supprimer les branches intégrées pour garder un dépôt propre.

Résumé à retenir (élève)

- `git branch nom` → créer une branche.
- `git checkout nom` ou `git switch nom` → se déplacer.
- `git checkout -b nom` → créer + se déplacer en une seule commande.
- `git merge nom` → fusionner une branche dans celle où tu te trouves.
- `git branch -d nom` → supprimer une branche.

Variante pour aller plus loin (enseignant)

- Demander aux élèves de créer deux branches différentes (`feature-A` et `feature-B`) et de les fusionner l'une après l'autre.
- Faire volontairement deux modifications sur la **même ligne** dans deux branches différentes → provoquer un **conflit de merge** à résoudre.
