Cas d'usage

Pendant le développement, plusieurs commits intermédiaires peuvent être créés pour conserver les étapes . Cela peut multiplier la création de commits qui ne sont pas forcément nécessaires.

A la fin, on souhaite garder un commit unique qui rassemblera tous les résultats de tous les commits intermédiaires sans en conserver l'historique.

Git nous permet de le faire de 2 manières

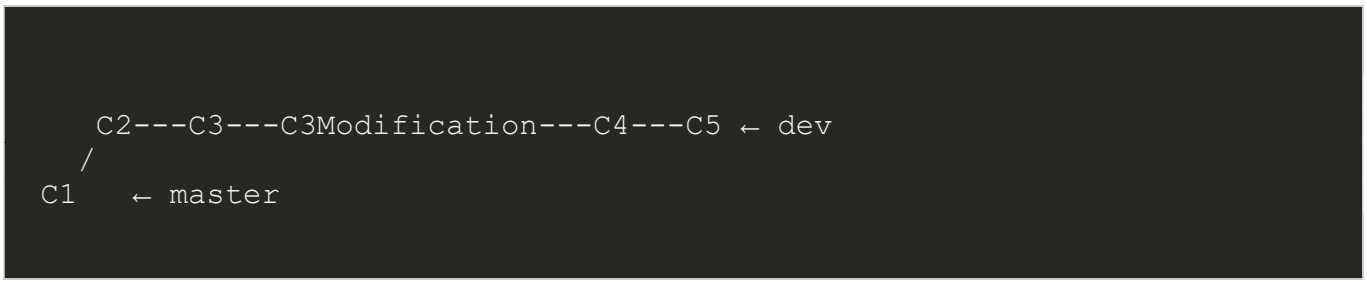
- Avec interactive rebase : **git rebase -i** rassemble tous les commits en un seul commit dont on peut réécrire le message. Solution souple, qui donne la main à travers un éditeur et permet de controler tout ce qui se déroule pendant le processus. Plus lent que git merge --squah
- Avec **git merge --squash** : même finalité, mais plus automatique que git rebase -i . En une ligne de commande, le processus est effectué.

  Nous aborderons ici git merge --squash.

TP1- Fusionner des commits avec git merge --squash

Etape 1 : préparation de l'environnement

Objectif : fabriquer des commits intermediaires



Créer une local repository dans un répertoire de travail

```
git init
```

********** Sur la branch master

Créer un fichier A-Commit.txt

```
touch A-Commit1.txt
git add .
git commit -m "C1"
```

A partir de master, créer la branch dev

```
git branch dev
```

Passer sur la branch dev

```
git checkout dev
```

Créer un fichier B-Commit2.txt

```
touch B-Commit2.txt
git add .
git commit -m "C2"
```

Créer un fichier C-Commit3.txt

```
touch C-Commit3.txt
git add .
git commit -m "C3"
```

Modifier le contenu du fichier C-Commit3.txt

```
nano C-Commit3.txt
# Ajouter du texte
git add .
git commit -m "C3-Modification"
```

Créer un fichier D-Commit4.txt

```
touch D-Commit4.txt
git add .
git commit -m "C4"
```

Créer un fichier E-Commit5.txt

```
touch E-Commit5.txt
git add .
git commit -m "C5"
```

Notre environnement est prêt. Il ne reste qu'à fusionner ces commits intermédiaires

Nous avons terminé la préparation de notre environnement

Vérifier l'historique de notre branch dev

```
git log --oneline
```

Etape 2 : procéder aux merges avec --squash

Objectif : fusionner les commits intermédiaires en un seul commit

```
C2---C3---C3Modification---C4---C5 ← dev ← devtarget →C6
```

Déroulement

Créer la branch devtarget à partir de master

```
git checkout master
git branch devtarget
```

On se place dans la branch devtarget

```
git checkout devtarget
```

Lancer la commande de merge de la branche dev avec l'option --squash

```
git merge --squash dev
```

ou pour lancer le commit en même temps

```
git merge --squash dev && git commit
```

Si vous n'avez pas effectué le commit à l'étape précédente, vous pouvez lancer le commit maintenant (C6)

```
git commit -am "C6-Squashing all commits"
```

Vérifier la log de la branche devtarget

```
git log --oneline
```