

Abgezweigte Git-Banches

Wenn sich Ihr lokales `main`-Branch und der entfernte `origin/main`-Branch in Git voneinander entfernt haben —mit 74 bzw. 70 verschiedenen Commits—bedeutet dies, dass sie einen gemeinsamen Vorfahren haben, sich seitdem jedoch einzigartige Commits entwickelt haben. Ihr Ziel ist es, Ihren lokalen `main`-Branch so zu ändern, dass er mit `origin/main` übereinstimmt, indem Sie Ihren lokalen Branch auf den gemeinsamen Vorfahren (den “Elternteil beider”) zurücksetzen und dann `git pull` verwenden, um die entfernten Änderungen zu übernehmen und Ihre lokalen abweichenden Commits zugunsten der entfernten zu verwerfen. Hier sind die Schritte, um dies zu erreichen.

Schritte zur Behebung der Divergenz

1. **Die neuesten Remote-Änderungen abrufen** Stellen Sie sicher, dass Ihr lokales Repository die neuesten Informationen vom Remote-Repository hat. Dies aktualisiert Ihren Verweis auf `origin/main`, ohne Ihren lokalen `main`-Branch zu ändern.

```
git fetch origin
```

2. **Den gemeinsamen Vorfahren-Commit finden** Verwenden Sie `git merge-base`, um den gemeinsamen Vorfahren-Commit zwischen Ihrem lokalen `main` und `origin/main` zu identifizieren. Dies ist der Punkt, an dem sich die Branches getrennt haben.

```
ancestor=$(git merge-base main origin/main)
```

Dieser Befehl speichert den Commit-Hash des gemeinsamen Vorfahren in der Variablen `ancestor`.

3. **Lokalen `main` auf den gemeinsamen Vorfahren zurücksetzen** Setzen Sie Ihren lokalen `main`-Branch auf den gemeinsamen Vorfahren-Commit zurück. Die Option `--hard` stellt sicher, dass sowohl der Branch-Pointer als auch Ihr Arbeitsverzeichnis aktualisiert werden, wodurch alle lokalen Commits und Änderungen nach diesem Punkt verworfen werden.

```
git reset --hard $ancestor
```

Achtung: Dieser Schritt wird alle nicht festgelegten Änderungen in Ihrem Arbeitsverzeichnis und Staging-Bereich sowie die 74 Commits, die einzigartig für Ihren lokalen `main` sind, verwerfen. Wenn Sie diese behalten möchten, erstellen Sie vorher einen Sicherungsbranch (siehe “Optionale Sicherung” unten).

4. **Die Remote-Änderungen ziehen** Ziehen Sie nun die Änderungen von `origin/main`. Da Ihr lokaler `main` jetzt beim gemeinsamen Vorfahren ist und `origin/main` 70 Commits voraus ist, wird dies Ihren lokalen `main` schnell nach vorne bringen, um mit `origin/main` übereinzustimmen.

```
git pull origin main
```

Danach wird Ihr lokaler `main`-Branch identisch mit `origin/main` sein, wobei alle lokalen abweichenden Commits verworfen wurden.

Vollständige Befehlssequenz

Hier ist die vollständige Sequenz der auszuführenden Befehle:

```
git fetch origin
ancestor=$(git merge-base main origin/main)
git reset --hard $ancestor
git pull origin main
```

Was passiert?

- **Nach Schritt 1:** Ihr lokales Repository kennt den neuesten Zustand von `origin/main`.
- **Nach Schritt 2:** Sie haben den gemeinsamen Vorfahren-Commit identifiziert (z.B. einen Hash wie `abc123`).
- **Nach Schritt 3:** Ihr lokaler `main` wurde auf `abc123` zurückgesetzt, wodurch die 74 einzigartigen lokalen Commits verloren gehen.
- **Nach Schritt 4:** `git pull` bringt `main` schnell nach vorne zu `origin/main`, wodurch die 70 Commits von der Remote-Branch übernommen werden.

Das Ergebnis ist, dass Ihr lokaler `main` genau mit `origin/main` übereinstimmt, wodurch Ihr Ziel, die Remote-Branch zu verwenden, erfüllt wird.

Optionale Sicherung

Wenn Sie sich nicht sicher sind, ob Sie Ihre lokalen Commits verwerfen möchten, und diese für später aufbewahren möchten, erstellen Sie vor dem Zurücksetzen einen Sicherungsbranch:

```
git branch backup-main
```

Führen Sie dies vor dem `git reset`-Befehl aus. Ihre 74 lokalen Commits bleiben auf `backup-main`, und Sie können sie später bei Bedarf erneut besuchen.

Wichtige Hinweise

- **Nicht festgelegte Änderungen:** Wenn Sie nicht festgelegte Änderungen (in Ihrem Arbeitsverzeichnis oder Staging-Bereich) haben, die Sie behalten möchten, stashen Sie diese vor dem Zurücksetzen:

```
git stash
```

Nach Abschluss des Prozesses können Sie sie mit `git stash pop` wiederherstellen.

- **Warum das funktioniert:** Das Zurücksetzen auf den gemeinsamen Vorfahren und das Ziehen stellt sicher, dass eine saubere schnelle Vorwärtsbewegung zu `origin/main` erfolgt, wodurch Merge-Konflikte vermieden werden, die durch direktes Ziehen mit abweichenden Branches entstehen könnten.

Durch die Durchführung dieser Schritte setzen Sie Ihren lokalen `main` erfolgreich so zurück, dass er mit `origin/main` übereinstimmt, wie gewünscht.