Spickzettel: Git-Workflow mit zentralem Repository

1. Repository klonen (Erstinstallation)

```
git clone <url>             # Remote-Repository lokal klonen
cd <projektname>
```

2. Aktuelle Änderungen abrufen

```
git pull origin main        # Änderungen vom zentralen Repo holen
```

3. Neuer Feature-Branch erstellen

```
git checkout -b feature-xyz   # Neuen Branch für das Feature anlegen
```

4. Änderungen machen und committen

```
git status                    # Status der Änderungen prüfen
git add <datei>                # Datei für den Commit vormerken
git commit -m "Feature XYZ hinzugefügt"  # Commit erstellen
```

5. Branch mit Remote synchronisieren

```
git push origin feature-xyz    # Branch zum Remote-Repo pushen
```

6. Änderungen in `main` integrieren (nach Review)

1. **Zurück zum Hauptbranch wechseln**

```
git checkout main
git pull origin main           # Aktuelle Änderungen holen
```

2. **Feature-Branch mergen**

```
git merge feature-xyz          # Änderungen in main integrieren
```

3. **Branch löschen (optional)**

```
git branch -d feature-xyz      # Lokalen Branch entfernen
git push origin --delete feature-xyz  # Remote-Branch löschen
```

7. Änderungen von anderen Entwicklern holen

```
git fetch origin               # Änderungen abrufen (ohne Merge)
git merge origin/main          # Manuelles Mergen mit aktuellem Stand
```

8. Typische Fehler & Lösungen

- **Merge-Konflikt lösen**
  - Betroffene Datei öffnen, Konfliktstellen bereinigen.
  - Änderungen speichern, dann:

```
git add <datei>
git commit -m "Konflikt behoben"
```

- **Alten Stand wiederherstellen**

```
git reset --hard HEAD~1  # Letzten Commit verwerfen
```

Best Practices

- **Häufig pullen**, um Konflikte zu vermeiden.
- **Feature-Branches nutzen**, um Änderungen strukturiert zu halten.
- **Klar verständliche Commit-Messages schreiben.**
- **Pull Requests nutzen**, um Code-Reviews zu ermöglichen.