Die ersten Schritte mit git

Hier sind die ersten Schritte, um mit git zu beginnen, mit einer ausführlichen Erklärung zu jedem Schritt und einer Möglichkeit, diese zu überprüfen:

1. **Installation**: Installiere git auf deinem Computer, falls es noch nicht installiert ist. Besuche die offizielle git-Website und folge den Installationsanweisungen für dein Betriebssystem.

   Überprüfung: Öffne die Kommandozeile oder das Terminal und gib den Befehl `git --version` ein. Die installierte Version von git sollte angezeigt werden.

2. **Konfiguration**: Konfiguriere git mit deinem Namen und deiner E-Mail-Adresse. Das ist wichtig, um Commits mit korrekten Informationen zu versehen.

   ```
   git config --global user.name "Dein Name"
   git config --global user.email "deine@email.com"
   ```

   Überprüfung: Gib den Befehl `git config --global --list` ein. Deine konfigurierten Benutzerinformationen sollten angezeigt werden.

3. **Repository erstellen**: Erstelle ein neues lokales git-Repository für dein Projekt. Navigiere zu dem Verzeichnis, in dem sich dein Projekt befindet, und führe den Befehl `git init` aus.

   Überprüfung: Navigiere zu deinem Projektverzeichnis und gib den Befehl `ls -a` (Linux/Mac) oder `dir` (Windows) ein. Ein verstecktes `.git`-Verzeichnis sollte angezeigt werden, das darauf hinweist, dass das Repository initialisiert wurde.

4. **Dateien hinzufügen**: Füge Dateien zu deinem Repository hinzu, indem du sie zum Staging-Bereich hinzufügst. Verwende den Befehl `git add <Dateiname>` oder `git add .` (Punkt), um alle Dateien im aktuellen Verzeichnis hinzuzufügen.

   Überprüfung: Gib den Befehl `git status` ein. Die hinzugefügten Dateien sollten im Staging-Bereich angezeigt werden.

5. **Commit erstellen**: Erstelle einen Commit mit den hinzugefügten Dateien und einer aussagekräftigen Commit-Nachricht. Verwende den Befehl `git commit -m "<Commit-Nachricht>"`.

   Überprüfung: Gib den Befehl `git log` ein. Der erstellte Commit und die zugehörigen Informationen sollten angezeigt werden.

6. **Feature-Branch erstellen**: Erstelle einen neuen Branch für die Entwicklung eines neuen Features oder einer neuen Funktion. Verwende den Befehl `git branch <Branch-Name>` gefolgt von `git checkout <Branch-Name>`, um auf den neuen Branch zu wechseln.

   Überprüfung: Gib den Befehl `git branch` ein. Der neu erstellte Branch sollte in der Liste der Branches angezeigt werden, und ein Sternchen (*) sollte neben dem aktuellen Branch stehen.

7. **Änderungen vornehmen**: Mache deine gewünschten Änderungen an deinem Code oder füge neue Funktionen hinzu. Verwende einen Texteditor oder eine Entwicklungsumgebung, um die Änderungen vorzunehmen.

   Überprüfung: Überprüfe den geänderten Code und stelle sicher, dass die gewünschten Änderungen oder Funktionen implementiert wurden.

8. **Commit auf dem Feature-Branch**: Füge die geänderten Dateien dem Staging-Bereich hinzu und erstelle einen Commit auf dem Feature-Branch mit einer aussagekräftigen Commit-Nachricht.

Überprüfung: Gib den Befehl `git log` ein. Der neue Commit sollte auf dem Feature-Branch angezeigt werden.

9. **Zurück zum Haupt-Branch**: Wechsle zurück zum Haupt-Branch (normalerweise als "master" bezeichnet), um den Code deines Feature-Branches mit dem Haupt-Branch zusammenzuführen. Verwende den Befehl `git checkout master`.

   Überprüfung: Gib den Befehl `git branch` ein. Der Haupt-Branch sollte als aktueller Branch angezeigt werden.

10. **Merge des Feature-Branches**: Führe den Merge des Feature-Branches in den Haupt-Branch durch, um deine Änderungen zu integrieren. Verwende den Befehl `git merge <Feature-Branch-Name>`, um den Merge durchzuführen.

    Überprüfung: Überprüfe den Code im Haupt-Branch und stelle sicher, dass die Änderungen aus dem Feature-Branch erfolgreich zusammengeführt wurden.

11. **Remote-Repository hinzufügen**: Füge ein Remote-Repository hinzu, um dein lokales Repository mit einem entfernten Repository (z.B. auf GitHub) zu verknüpfen. Verwende den Befehl `git remote add origin <Repo-URL>`, wobei `<Repo-URL>` die URL des entfernten Repositories ist.

    Überprüfung: Gib den Befehl `git remote -v` ein. Der Name und die URL des Remote-Repositories sollten angezeigt werden.

12. **Push**: Übertrage deine Commits in das Remote-Repository mit dem Befehl `git push -u origin master`. Das `-u` sorgt dafür, dass du beim nächsten Push nur noch `git push` verwenden kannst.

    Überprüfung: Gehe zum Remote-Repository (z.B. auf GitHub) und überprüfe, ob deine Commits und Dateien dort erscheinen.