**Bericht zum Unterrichtsprojekt**

|  |
| --- |
| **FAKULTÄT FÜR PHILOLOGIE** |
| **Sprachwissenschaftliches Institut** |

**Teilprojekt 1 von 1**

[‚1‘ bei 2-Fach-Master]

|  |  |
| --- | --- |
| **Name** | **Chandler** |
| **Vorname** | Christopher |
| **Matrikelnummer** |  |
|  |  |
| **Veranstaltungsnummer** | 050004 |
| **Lehrveranstaltungstitel** | Methoden der Linguistik |
| **Semester** | Sommersemester 2024 |
|  |  |
| **Seminar/Kurssitzungen am** | 01.07.2024 |

# 1. Erste Lehreinheit

## 1.1 Thema der Sitzung

* Einführung in die Versionsverwaltung mit Git

## 1.2 Spezifische Inhalte der Sitzung

### Teil 1: Git Einführung

* **Was ist Versionsverwaltung?**
  + Definition und Nutzen von Versionskontrollsystemen (VCS)
  + Unterschiede zwischen lokaler und zentraler Versionsverwaltung
* **Historie von Git**
  + Entstehung und Entwicklung von Git
  + Vergleich mit anderen Versionskontrollsystemen
* **Installation und Nutzung von Git**
  + Anleitung zur Installation von Git (Mac, Linux, Windows)
  + Nutzung der Kommandozeile und grafischen Benutzeroberflächen (GUIs)

### Teil 2: Git Grundlagen

* **Snapshots vs. Diffs**
  + Git speichert Daten als Schnappschüsse des gesamten Dateisystems
* **Lokale Operationen**
  + Schnelle und effiziente lokale Operationen
* **Sicherstellung der Datenintegrität**
  + Verwendung von Prüfsummen zur Datensicherung
* **Grundsätzliche Zustände von Dateien**
  + Committed: Daten sind in der lokalen Datenbank gespeichert
  + Modified: Dateien wurden geändert, aber noch nicht zur Staging-Area hinzugefügt
  + Staged: Dateien sind für den nächsten Commit vorgemerkt

### Teil 3: Git Branching

* **Erstellung und Verwaltung von Branches**
  + git branch zum Erstellen neuer Branches
  + git checkout zum Wechseln zwischen Branches
* **Merging von Branches**
  + git merge zum Zusammenführen von Branches
  + Konfliktlösung beim Merging
* **Best Practices für Branching**
  + Verwendung von Feature-Branches
  + Regelmäßiges Zusammenführen von Änderungen aus dem Hauptbranch (Master/Main)

### Teil 4: Git GUI Software

* **Nutzung von GUIs**
  + Vorteile und Einsatzgebiete von grafischen Benutzeroberflächen
* **Beliebte Git-GUI-Tools**
  + SourceTree: Installation, Einrichtung und Nutzung
  + Git in PyCharm: Integration und Nutzung
  + GitHub Desktop: Webseite kurz zeigen
* **Praktische Beispiele**
  + Erstellung und Verwaltung von Repositories mit GUIs
  + Verfolgung und Speicherung von Änderungen mit GUIs
  + Branching und Merging mit GUIs

## 1.3 Lernziele

* Verständnis der Grundlagen der Versionsverwaltung und ihrer Bedeutung
* Fähigkeit, grundlegende Git-Befehle anzuwenden (init, clone, add, commit, push, pull)
* Fähigkeit, Branches zu erstellen, zu wechseln und zusammenzuführen
* Vertrautheit mit der Nutzung von grafischen Benutzeroberflächen zur Verwaltung von Git-Repositories
* Anwendung von Best Practices für die Zusammenarbeit mit Git (regelmäßiges Pushen und Pullen)​

## 1.4 Sitzungsplanung und -strukturierung

|  |  |  |  |
| --- | --- | --- | --- |
| Lektion | Mittel/Materialien | Lernziele | Zeitangaben in Minuten |
| Teil 1: Einführung | Jupyter notebook  Quiz | Verständis, was Git ist und warum das wichtig ist. | 10 5 |
| Teil 2: Grundlagen | Jupyter notebook  Quiz | Wissen, wie Git funktioniert. | 20 5 |
| Teil 3: Branching | Jupyter notebook  Quiz | Wissen, wie man Branches erstellt | 15 5 |
| Teil 4: Gui | Jupyter notebook  Pycharm  Source tree | Wissen, wie man Git in IDEs verwenden kann. | 10 |
|  |  |  | Gesamtzeit: Ca. 70 |

Am Ende soll es Zeit für Fragen, Kritik, Anregung, etc. geben.

## 1.5 Unterrichtsmaterialien

* Jupyter Notebook
* Pdf
* HTML
* Moodle Quizes

## 1.6 Protokoll

(Beschreiben Sie hier den Sitzungsverlauf)

[Freier Text]

## 1.7 Selbstbewertung

(Was war gut? Was hat nicht so gut geklappt?

Was möchte ich beim nächsten Mal anders machen? Was habe ich selbst dabei gelernt?)

[Freier Text]

## 1.8 Evaluation

[Freier Text]