

Git Nachdenkzettel

1. Was genau sind die Gründe um Gitlab zu verwenden?

- Perfekt für Teamarbeit, da jeder eigenständig an dem Projekt arbeiten kann
- Teammitglieder können schnell und simpel ihren Code hochladen und den von anderen gefertigten Code pullen
- Bietet gute Einsicht darin, wer an welchen Teilen der Software gearbeitet hat

2. Welche Daten gehören (nicht) ins Repo?

- .java Files
- .xml Files
- .Jason Files
- Bilddateien für Gameprojekte
- Musikdateien für Projekte
- .project Files (Endung je nach Tool)
- UML Modelle
- Zeichnungen
- Notizen
- Dokumentation
- Configurationsfiles
- Kapitel eines Buches
- Eine Bachelorarbeit
- Passwörter für Cloud-Services
- Passwörter für lokale Services (Self-hosted)
- logfiles
- Messdaten vom Profiling

3. Was soll der „Mist“ mit den Stages (dass add/commit nur lokal wirken)?

- Code kann reviewt werden, bevor er ins Repository gepusht wird. Syntaxfehler etc. "zerstören" somit nicht gleich das gesamte Projekt.
- Jeder Entwickler kann unabhängig voneinander seinen Code testen und bearbeiten

4. Würden Sie in einer Firma Gitlab selber hosten oder GIT als Service im Netz? Begründung.

- Bei einem kleineren Team ist es praktischer, GIT im Netz zu nutzen, da es weniger kostet
- Wenn jedoch ein großes Team mit einem vertraulichem Projekt genug Ressourcen hat, ist es besser, GitLab selbst zu hosten, da man dann selbst Anpassungen an GitLab machen kann.

5. Verwenden Sie Branches im Projekt oder arbeiten alle Teammitglieder auf dem Master Branch? Zeigen Sie Vor- und Nachteile der Verfahren

Mehrere Branches

Vorteile:

- Vermeidung von Merge-Konflikten, da Code isoliert voneinander ist
- Bessere Aufgabenteilung parallele Entwicklungsarbeiten
- Leichtere Fehlerbehebung

Nachteile:

- Schwieriger, alle Branches in Master-Branch ohne Konflikte zu integrieren
- Unübersichtlicher und zeitaufwändiger, als alles auf einem Master-Branch zu haben

Master-Branch

Vorteile:

- Simpler und übersichtlicher
- Einfacher für Teammitglieder eine gute Vorstellung davon zu bekommen, wer an was arbeitet
- Code im Master-Branch ist in der Regel stabil, d.h. es geht schnell das Programm zu veröffentlichen

Nachteile:

- Mehrere Leute arbeiten gleichzeitig am Projekt und Veränderungen am Code können leicht zu Konflikten führen
- Code kann nicht auf separatem Branch getestet werden, wodurch unerwünschte Auswirkungen auf den Code nicht verhindert