

Gherkin Syntax textuell beschreiben

Vorbedingung → 'Given' und 'And' → app name descr.
 Event → 'When' → neue Task anlegen
 Nachbedingung → 'Then' → Wkt mit neuen Task
 *.feature - Datei

→ Nur Beschreibung des erwarteten Verhaltens
 ↳ Kontakte Funktionalität

→ Ausimplementieren des Tests in *.steps
 add-task.py → Schritte 'steps' der *.feature implementiert

↳ Fehlende Flkt. in Hauptprogramm umsetzen, sodass *step.py Test erfolgreich durchläuft.

Vorbedingung "add-task"
 → task manager app läuft
 → task name 'My Task',
 → task descr. 'My Descr.'

→ add-task → neue Task erstellen

Nachbedingung

→ task in einer Task list existiert
 ↳ name
 ↳ Beschreibung

Git Hooks



- Identifizieren welche Flkt. die sich wiederholen soll (ohne Gedacht sich als 'Hook')?
- Schreibe Script und lege es in /.git/hooks mit 'richtiger' Namen ab
- aktiviere **local** script ohne .sample Endung
- wenn für Team interessant → script Teilen und bei allen Mitgliedern in /.git/hooks ablegen

Server-side können Script ebenfalls unter /.git/hooks abgelegt werden
 z.B.: pre-receive auf Server vor Änderungen empfangen werden

Für uns

pre-commit

- läufe formatter
- läufe checkstyle
- läufe security check

pre-push

- bdd tests
- unit tests
- compliance 'header'

post-update

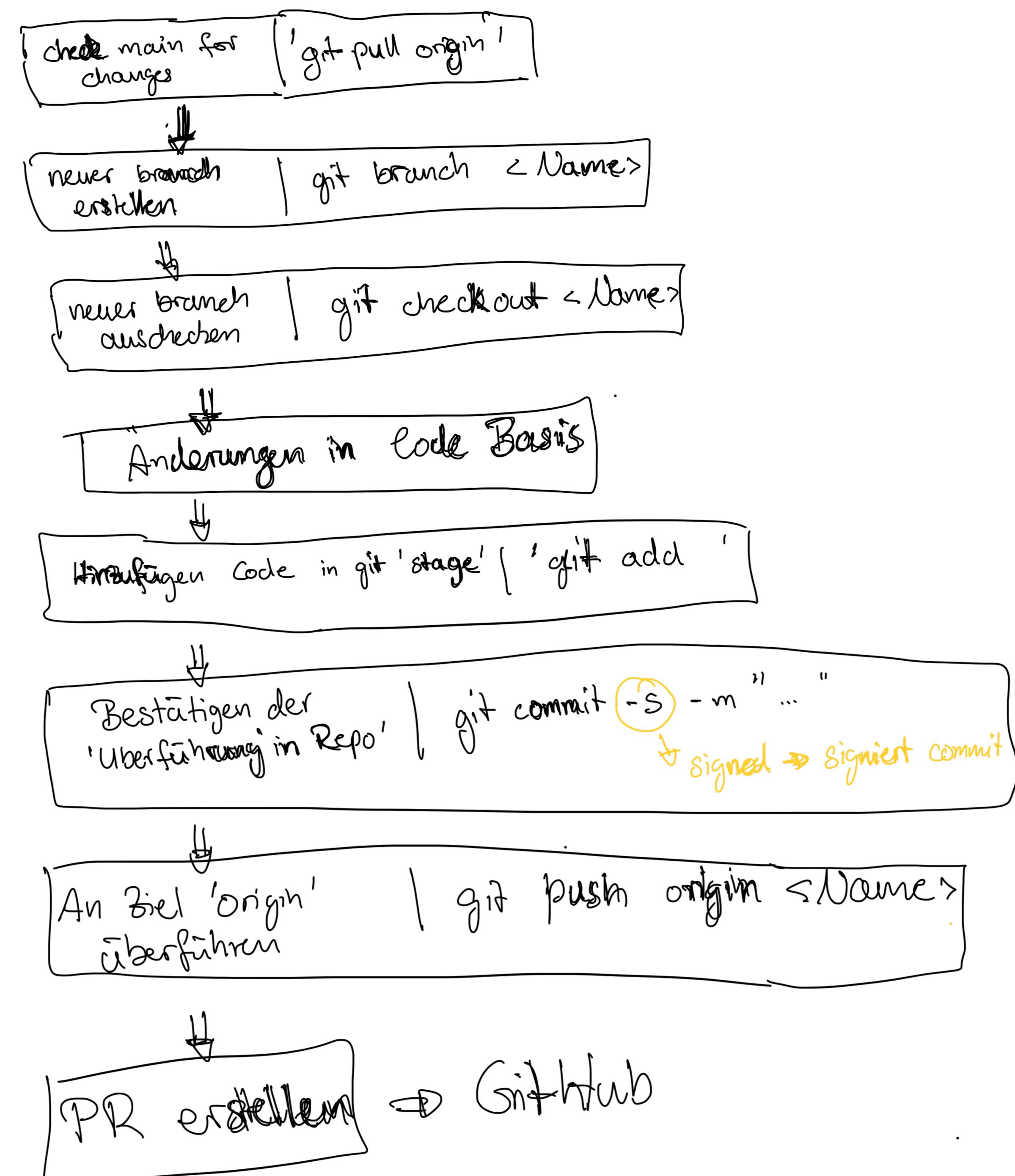
- SonarQube zu informieren
- informiere Build Server

dee → product

Git (Stages)

- unstaged → Bereich mit Dateien die bearbeitet werden
- staged → Dateien die für Bearbeitung 'bereit' sind
 - ↳ 'commit' → bestätigt
 - ↳ 'stashed' → Zwischenablage
- push
- ...

Best Practices Git Development



Tool 1

Tool 2

Tool 3

manuell ausgeführt → nicht optimal effizient

→ **Script.sh** schreiben um Tools nacheinander auszuführen → müssen immer trotzdem manuell ausführen

→ **git hooks** um in gewissen Git 'Stages' flkt. auszulösen → nicht bei allen Devs aktiv

→ **separate Config** um mittels pre-commit ausführung zu erzwingen