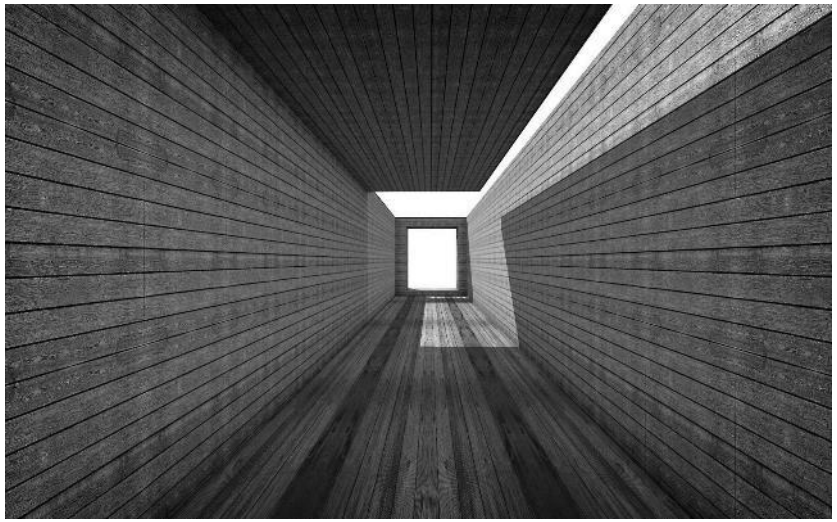
Bischöfliches Institut Büllingen

Mini-Projekt - Multifunktionsuhr

Auswertung & Verbesserungsvorschläge



Auswertung

Qualität bedeutet, dass der Kunde und nicht die Ware zurück kommt.

Hermann Tietz

Auswertung

Teamarbeit

- Gute Organisation
- Gute Mitarbeit

Struktur des Kode

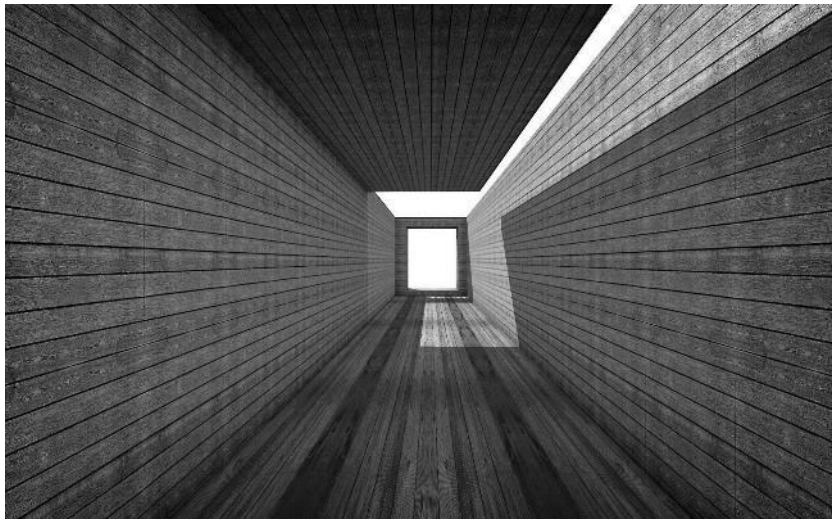
- Gut lesbarer Kode
- Funktion pro Feature
- Mehr Kommentare

Funktionalität

- Mehrsprachigkeit!

Qualität des Kode

- Verwaltung der Variablen
- Globalen Variablen vs. Lokale Variablen



Verbesserungs-vorschläge

Kleine Dinge sind verantwortlich für große Veränderungen.

Paolo Coelho

Bischöfliches Institut Büllingen

Verbesserungsvorschläge

Globale Variablen vs. Lokale Variablen

- Wann sind globale Variablen notwendig?
- Wann sind lokale Variablen besser?
- Bei globalen Variablen immer angeben!

Die Funktion `str()` nutzen, wenn nötig

- Unnötige Aufrufe entfernen

Verbesserungsvorschläge

Nutzung von `datetime`

- Die Objekte der Klasse `datetime` hat nützliche Attribute, um mit Zahlenwerte zu rechnen:
  - `year`
  - `month`
  - `day`
  - `hour`
  - `minute`
  - `second`
  - `microsecond`

Verbesserung der Funktion `set_alarm()`

- Abbruch des Alarms -> Kein Alarm
- Wie kann ein Alarm besser abgebrochen werden?
  - Unmittelbare Beendigung des Threads
  - Kein Alarm ertönt

Verbesserungsvorschläge

Problem in der Funktion `set_alarm()`

```
total_seconds_now = ((int(hours_now) * 60) * 60) + (int(minutes_now) * 60) +
seconds_now

if hours < hours_now:
    total_seconds_next_day = ((24*60)*60)-total_seconds_now
    time_to_alarm = total_seconds_next_day+total_seconds_alarm

    sleep(time_to_alarm)
else:
    time_to_alarm = total_seconds_alarm-total_seconds_now
```

Dieser Test ist nicht ausreichend. Der Test würde besser auf die berechneten Sekunden gemacht.

Verbesserungsvorschläge

Verbesserung der Funktion `set_timer()`

- Die Konvertierung der Timerzeit (in Ganzzahlen) sollte vor dem Aufruf von `set_timer()` geschehen
  - Warum?
- Umwandlung der Zahlenwerte ohne Zeichenketten

```
hours = str(seconds / 3600).split(".")[0]
seconds -= int(hours) * 3600
minutes = str(seconds / 60).split(".")[0]
seconds -= int(minutes) * 60
```

- Nutzung der Funktion `just()`

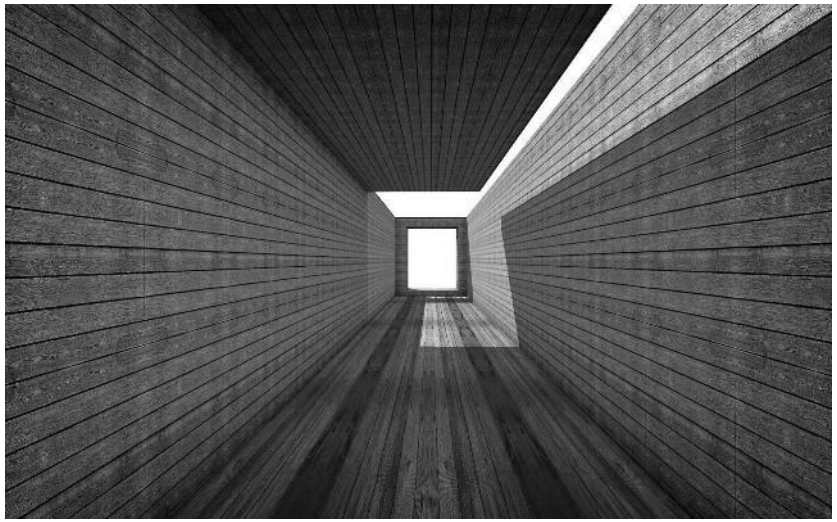
Verbesserung der Funktion `stopwatch()`

- Umwandlung der Zahlenwerte ohne Zeichenketten
- Nutzung der Funktion `just()`

Verbesserungsvorschläge

Verbesserung des Hauptkode

- Wenn das Fenster geschlossen wird, müssen alle Threads geschlossen werden
  - Es erscheint eine Fehlermeldung
  - Wie kann man die Threads korrekt beenden?
- Die Funktion „SET_TIME“ ist nicht mehr nötig
- Funktion index() muss in den try-Block gesetzt werden.



Vorgehensweise

Gut geplant, ist halb getan.

Volkstümlicher Spruch

Wichtige Konzepte von GitHub

Parallele Entwicklung verbessern

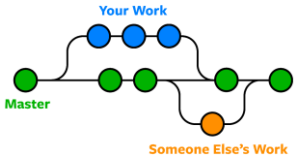
Branches

Ist bekannt

Issue

Datenbank zum Auflisten der Bugs & Wünsche

Feature Branch / Master Branch



Feature branch = 1 Branch pro Feature

Keine Entwiklung auf dem Master Branch

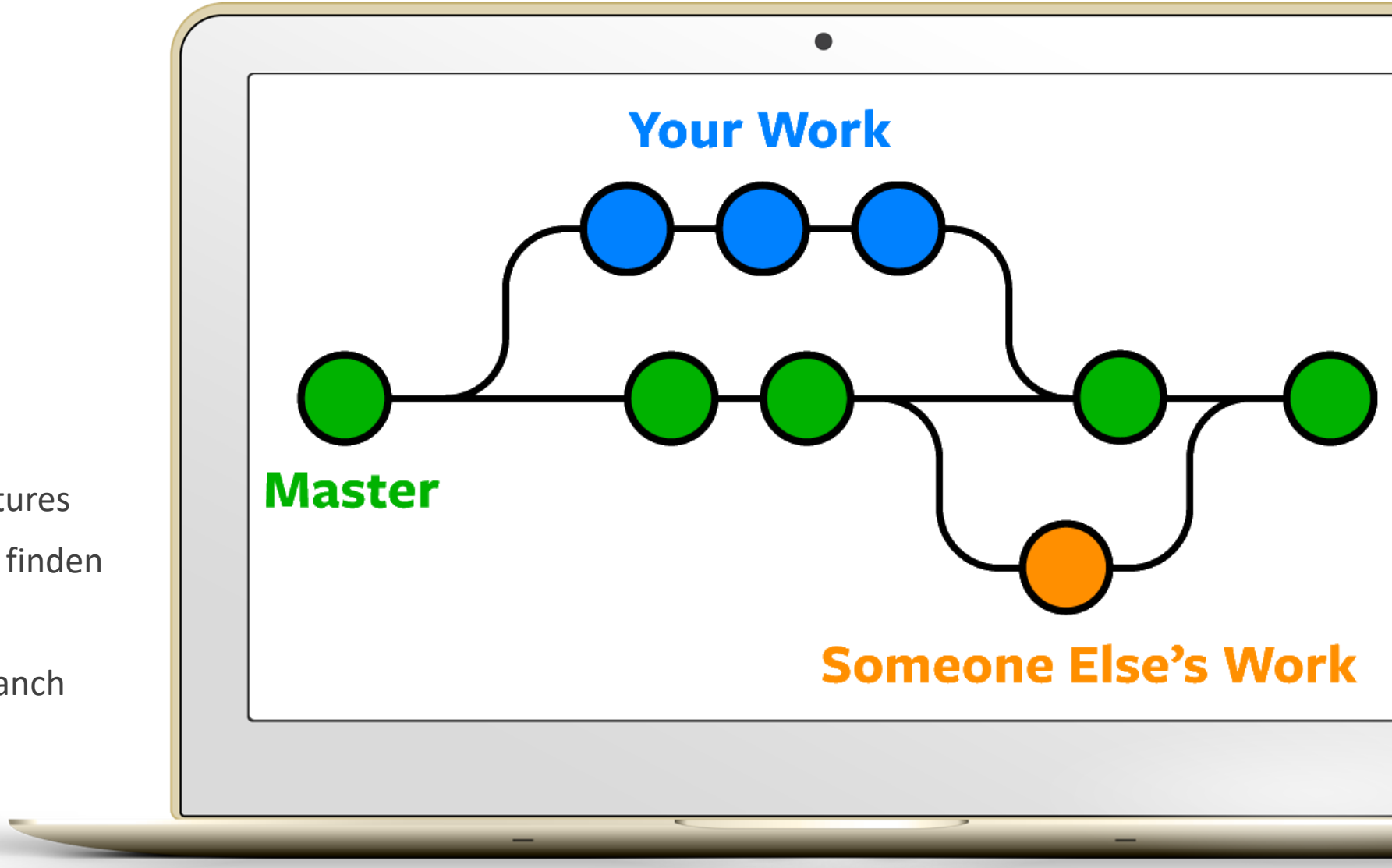
Pull Request

Integration der Entwicklung in den Master Branch

Feature Branch

Warum?

- Anzahl der Merge & Konflikte reduzieren
- Ungestörte Entwicklung der Features
- Commits für ein Feature einfach finden
- Teillieferung von Features
- Änderungen aus dem Master Branch können genutzt werden



Feature Branch - Ablauf

- Master Branch aktualisieren

```
git checkout master
git pull –ff-only (origin master)
```

- Feature Branch anlegen & bearbeiten

```
git checkout –b feature-A
git add *
git commit –m “Beschr.”
```

- Feature Branch im Remote Repository speichern

```
git push –set-upstream origin feature-A
```

- Feature Branch in Master Branch übertragen
  - Pull Request anlegen
  - Merge geschieht über den Pull Request

- Master Branch in den Feature Branch übernehmen

```
git checkout master
git pull --ff-only
git checkout feature-A
git merge –no-ff master
```

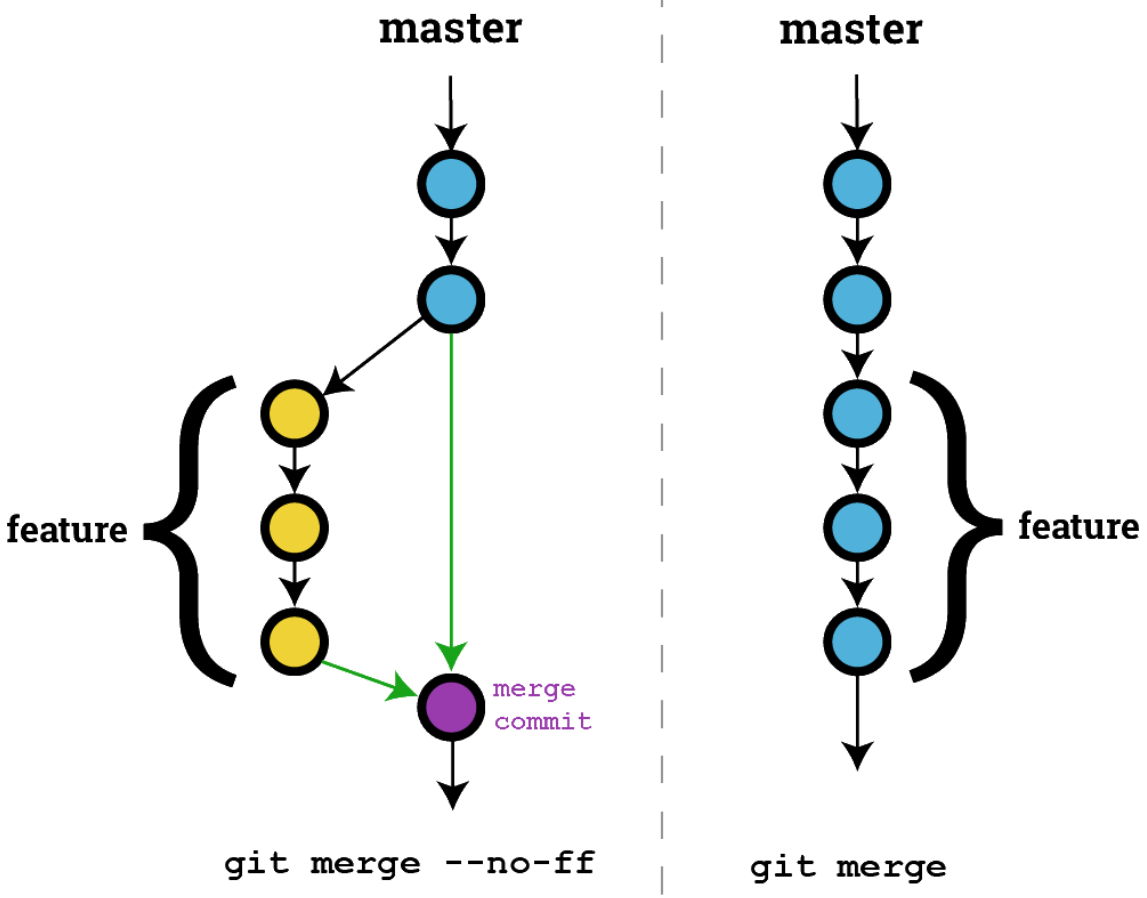
Damit kein Merge durchgeführt wird (unbewusste Änderungen auf dem Master Branch

Das ist nur beim ersten Mal nötig, um den Branch im Remote Repository anzulegen. Danach kann `git push` genutzt werden

Das wird nötig beim Pull Request, wenn es einen Konflikt zwischen dem Feature Branch und Master Branch entstanden ist (parallele Entwicklung am gleichen Kode).

Siehe nächstes Slide

Fast-Forward verhindern `--noff`

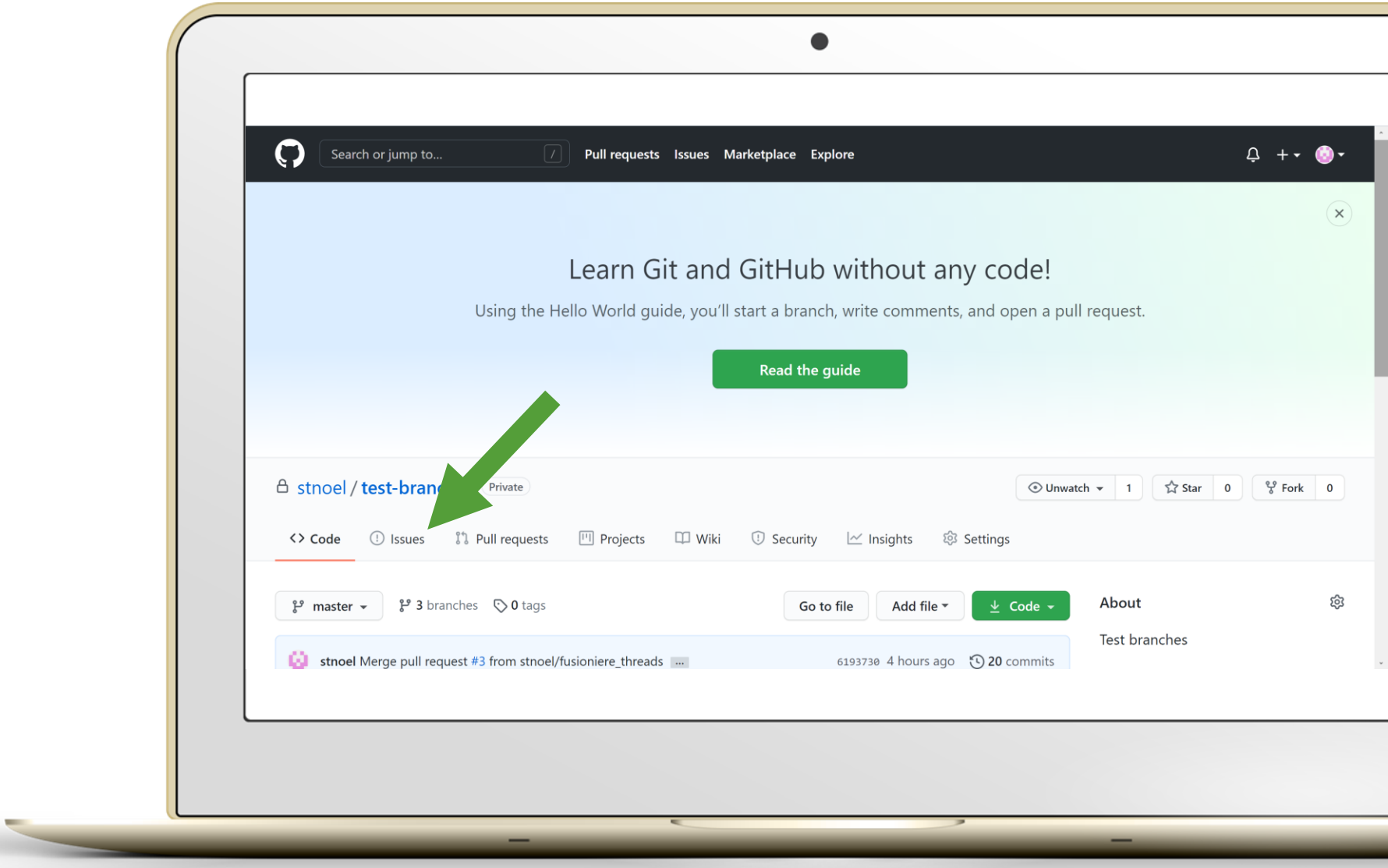


- Merge in den Master Branch (s. links)
  - Commits vom Feature Branch bleiben im Feature Branch
  - Es entsteht ein zusätzliches Commit
- Merge in den Feature Branch (Konflikt Auflösung)
  - Es wird vermieden, dass der Feature Branch und der Master Branch fusionieren

Issue

Was ist das?

- Datenbank in GitHub
- Informationen
  - Titel
  - Beschreibung
  - Verantwortliche Person
  - Milestone
  - Verbindung Pull Requests



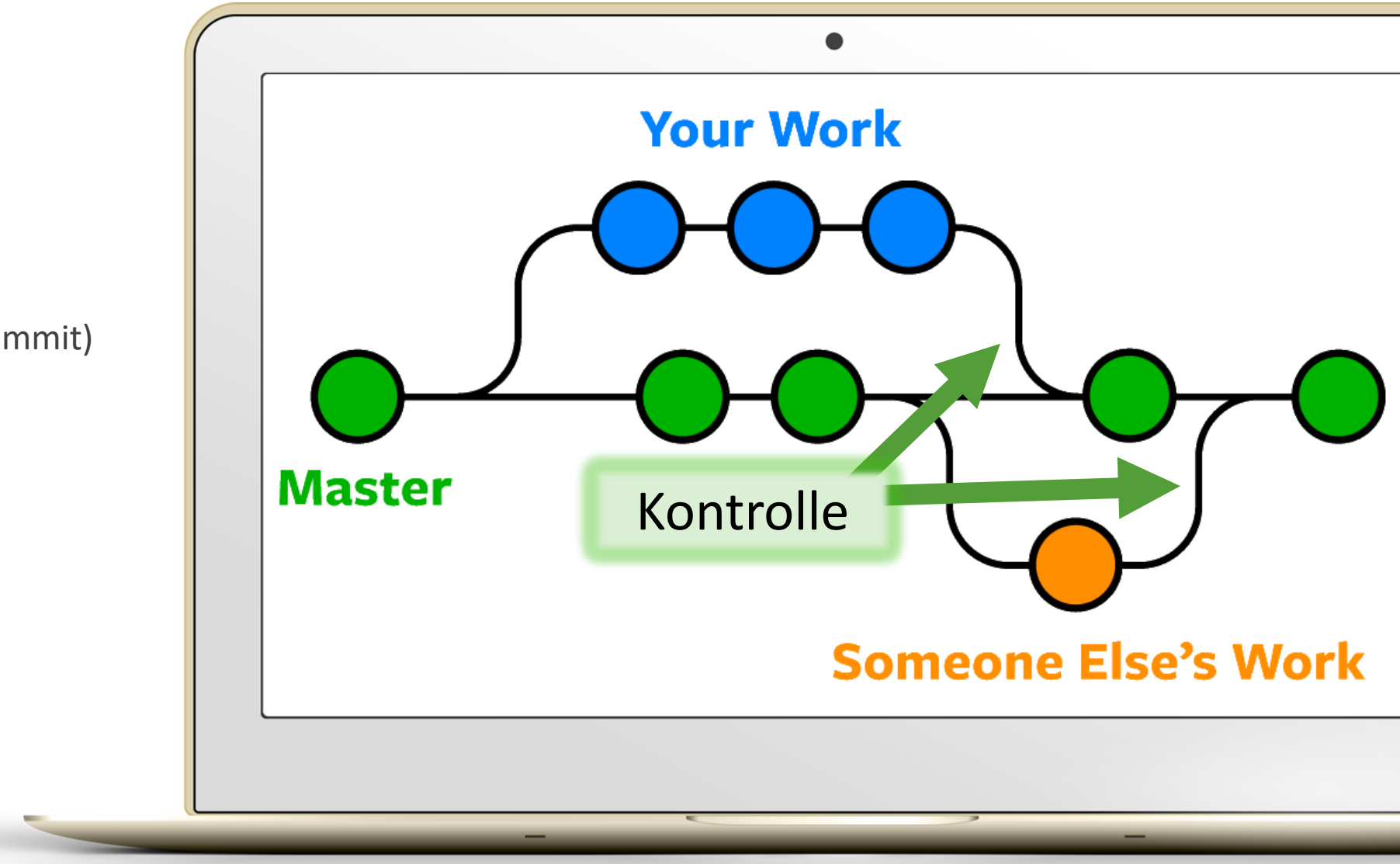
Pull Request

Sequenz

- Pull Request in GitHub
  - Link zu Issue!
- Administrator validiert
- Kommentare & Korrekturen (commit)
- Merge über GitHub

Warum?

- Qualitätskontrolle



Nützliche `git` Kommandos

- Speicherung der Logindaten:
  - `git config credential.helper manager`
  - `git config user.name <GitHub-Login>`
- Automatische Wahl des Branches für Pull/Push
  - `git checkout **--track** origin/feature-A`
  - `git branch –u origin/feature-B` (auf dem lokalen Branch)