git im Kontext industrieller Softwareentwicklung

Arlo O‘Keeffe
mailtoarlo@gmail.com
2018-10-25


THIS IS GIT. IT TRACKS COLLABORATIVE WORK ON PROJECTS THROUGH A BEAUTIFUL DISTRIBUTED GRAPH THEORY TREE MODEL.
COOL. HOW DO WE USE IT?
NO IDEA. JUST MEMORIZE THESE SHELL COMMANDS AND TYPE THEM TO SYNC UP. IF YOU GET ERRORS, SAVE YOUR WORK ELSEWHERE, DELETE THE PROJECT, AND DOWNLOAD A FRESH COPY.

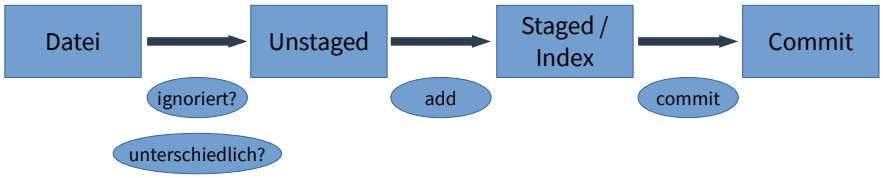
Ziel

- Know your tools
- Alle Fragen kennen und beantworten können

Begriffe

- Was ist eine untracked Datei?
- Was ist eine Änderung?
- Was ist die Staging?
- Was ist der Index?
- Was ist ein Commit?
- Was ist ein Hash?

Begriffe


Datei
Unstaged
Staged / Index
Commit
ignoriert?
unterschiedlich?
add
commit

Begriffe

- Was ist ein Branch?
- Was ist ein Tag?
- Was ist der HEAD?
- Was ist ein detached HEAD?
- Was ist HEAD^?
- Was ist HEAD^^?
- Was ist HEAD~3?

Begriffe

- Was ist ein Checkout?
- Was ist ein Clone?
- Was ist ein Repository?
- Was ist ein bare Repository?
- Was ist ein Remote?

Begriffe

- Was ist ein Stash?
- Was ist eine Hook?
- Was ist ein Submodule?
- Was ist ein Pull Request?

Aktionen

- Wie erstelle ich ein neues Repository?
- Wie sehe ich lokale Änderungen?
- Wie stage ich eine Änderung?
- Wie stage ich nur einen Teil meiner Änderungen?
- Wie commite ich meine Änderungen?
- Wie stage und commite ich eine Änderung?

9

* Wie erstelle ich ein neues Repository?
  * Was macht `init`?
* Wie sehe ich lokale Änderungen?
  * Was macht `status`?
* Wie stage ich eine Änderung?
  * Was macht `add`?
  * Was macht `add --all`?
  * Was macht `add .`?
* Wie stage ich nur einen Teil meiner Änderungen?
  * Was macht `add --interactive`?
* Wie commite ich meine Änderungen?
  * Was macht `commit`?
  * Was macht `commit -m`?
* Wie stage und commite ich eine Änderung?
  * Was macht `commit -am`?

Aktionen

- Wie ignoriere ich bestimmte Dateien?
- Wie ignoriere ich einen ganzen Ordner?
- Wie de-ignoriere ich bestimmte Dateien in einem ignorierten Ordner?
- Wie füge ich trotz .gitignore eine Datei hinzu?

10

* Wie ignoriere ich bestimmte Dateien?
  * `*.log`
* Wie ignoriere ich einen ganzen Ordner?
  * `/.settings/`
* Wie de-ignoriere ich bestimmte Dateien in einem ignorierten Ordner?
  * `!/.settings/user.setup`
* Wie füge ich trotz .gitignore eine Datei hinzu?
  * Was macht `add --force`?

Aktionen

- Wie erstelle ich einen Branch?
- Wie lösche ich einen Branch?
- Wie wechsel ich den aktuellen Branch?
- Wie sehe ich den Unterschied zwischen zwei Branches?
- Wie merge ich zwei Branches?
- Wie löse ich einen Merge-Konflikt?
- Wie erstelle ich einen Tag?

11

* Wie erstelle ich einen Branch?
  * Was macht `branch`?
* Wie lösche ich einen Branch?
  * Was macht `branch -d`?
* Wie wechsel ich den aktuellen Branch?
  * Was macht `checkout`?
* Wie sehe ich den Unterschied zwischen zwei Branches?
  * Was macht `diff`?
* Wie merge ich zwei Branches?
  * Was macht `merge`?
  * Was macht `merge --ff`?
* Wie löse ich einen Merge-Konflikt?
* Wie erstelle ich einen Tag?
  * Was macht `tag`?

Aktionen

<<<<<<<
Mine. Changes made on the branch that is being merged into. In most cases, this is the branch that I have currently checked out (i.e. HEAD).
|||||||
The common ancestor version.
=======
Theirs. Changes made on the branch that is being merged in. This is often a feature/topic branch.
>>>>>>>

Aktionen

- Wie mache ich eine Änderung rückgängig?
- Wie ändere ich eine Commit Message?

* Wie mache ich eine Änderung rückgängig?
  * Was macht `reset --soft`?
  * Was macht `reset --hard`?
* Wie ändere ich eine Commit Message?
  * Was macht `commit --amend`?
  * Was macht `rebase`?

Aktionen

- Wie hole ich mir ein externes Repository?
- Wie schiebe ich meine Änderungen in ein externes Repository?
- Wie aktualisiere ich den Zustand vom externen Repository?
- Wie hole ich mir die neusten externen Änderungen in mein Projekt?

14

* Wie hole ich mir ein externes Repository?
  * Was macht `clone`?
* Wie schiebe ich meine Änderungen in ein externes Repository?
  * Was macht `push`?
  * Was macht `push --force`?
* Wie aktualisiere ich den Zustand vom externen Repository?
  * Was macht `fetch`?
* Wie hole ich mir die neusten externen Änderungen in mein Projekt?
  * Was macht `pull`?
  * Was macht `pull --rebase`?

Aktionen

- Wie kann ich Änderungen zwischenspeichern?
- Wie kann ich Änderungen zwischenspeichern und den Index behalten?
- Wie kann ich einen einzelnen Commit eines anderen Branches übernehmen?
- Wie entferne ich alle Änderungen und unnötige Dateien (inkl. ignorierten)?

15

* Wie kann ich Änderungen zwischenspeichern?
  * Was macht `stash push|save`?
  * Was macht `stash pop|apply`?
* Wie kann ich Änderungen zwischenspeichern und den Index behalten?
  * Was macht `stash push --keep-index`?
* Wie kann ich einen einzelnen Commit eines anderen Branches übernehmen?
  * Was macht `cherry-pick`?
* Wie entferne ich alle Änderungen und unnötige Dateien (inkl. ignorierten)?
  * Was macht `clean`?

Aktionen

- Wie finde ich raus wer das commitet hat?
- Wie finde ich raus wann der Fehler eingeführt wurde?
- Wie füge ich unter Windows ein Executable Flag hinzu?
- Wie entferne ich nicht erreichbare Commits?
- Wie finde ich nicht erreichbare Commits?

16

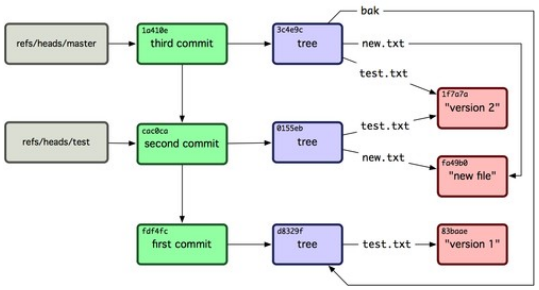
* Wie finde ich raus wer das commitet hat?
  * Was macht `blame`?
* Wie finde ich raus wann der Fehler eingeführt wurde?
  * Was macht `bisect`?
* Wie füge ich unter Windows ein Executable Flag hinzu?
  * Was macht `update-index --chmod=+x`
* Wie entferne ich nicht erreichbare Commits?
  * Was macht `gc`?
* Wie finde ich nicht erreichbare Commits?
  * Was macht `reflog`?

Prinzipien

- Was ist ein Graph?
- Was ist ein gerichteter Graph?
- Was ist ein azyklischer Graph?
- Was ist Erreichbarkeit?
- Was ist ein Baum?
- Was ist ein Object?
- Was ist eine Referenz?

- Object
  - Blob
  - Tree
  - Commit
- Referenz
  - Tag
  - Head / Branch
  - Remote

Prinzipien


refs/heads/master
1a410e
third commit
3c4e9c
tree
bak
new.txt
test.txt
1f7a7a
"version 2"
refs/heads/test
cac0ca
second commit
0155eb
tree
test.txt
new.txt
fa49b0
"new file"
fdf4fc
first commit
d8329f
tree
test.txt
83baae
"version 1"

Prinzipien

- Was ist ein Event-Sourced System?
- Was bedeutet es, dass git verteilt ist?
- Kann ich ein Git Repository zippen?
- Was ist porcelain?
- Was ist plumbing?

- Event-Sourced
  Die Summe aller Änderungen erzeugt den finalen Zustand
- Verteilt
  - Im ersten Moment alle Änderungen lokal
  - Unter unserer Kontrolle wann remote
  Schnell+Unabhängig (weil kein Netzwerk notwendig)
- Zip: Ja
- Plumbing?
  - Die Toolkit Kommandos (z.B. reflog)
- Porcelain?
  - Die „benutzerfreundlichen“ Kommandos (checkout, branch, remote)

Philosophie

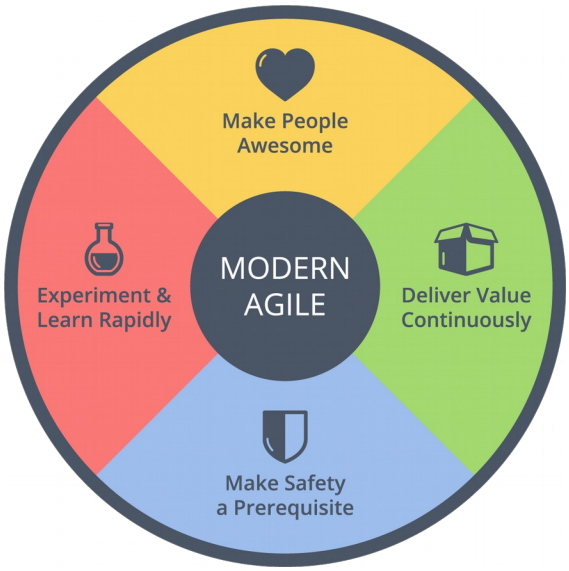
- Warum will man eine Versionskontrolle?
- Ist git die beste Versionskontrolle?
- Ist git benutzerfreundlich?
- Ist git besser als svn?
- Was ist Social Coding?

Kollaboration

- **Wie arbeitet man mit mehreren Leuten zusammen?**

So wenig Prozess / Workflow / Konventionen wie möglich.

Kollaboration



22

Vertrauen in seine Kollegen
Individuals and interactions over processes and tools
Working software over comprehensive documentation
Responding to change over following a plan

Kollaboration

- Welcher Workflow wird eingesetzt?
- Was sind sonstige Konventionen?
- Wie sieht der Prozess zwischen finalem Commit und Release zum End-User aus (aka Deploy)?
- Gibt es Reviews? Wer reviewed wen?
- Wo ist das Haupt-Repository gehostet?

23

* So wenig Prozess wie möglich. So viel wie nötig.
  *
  http://widgetsandshit.com/teddziuba/2011/12/process.html
* Was sind die Konventionen?
  * Commit Messages
  * lowercase Branch-Namen
  * master, develop, feature-* Branches
  * origin und upstream Remote
* Wie hoste ich ein Remote Repository?
  * git daemon
  * Smart HTTP
  * GitWeb
  * GitLab
  * GitBucket
  * https://git.wiki.kernel.org/index.php/GitHosting

Repository Management

- Was will man alles in einem Repository haben?
- Was gehört in eine .gitignore?
- Was gehört in eine gitattributes?
- Wie sorge ich dafür, dass in jedem Betriebssystem die richtigen Line Endings sind?

Branch Management

- Wie werden Branches benannt?
- Wofür werden Branches erstellt?
- Wie sieht der Lebenszyklus eines Branches aus?
- Wer macht was? Wer arbeitet wo?
- Sind Notfall-Patches notwendig?
- Müssen alte Releases supported werden?
- Was ist ein protected Branch?

Tag Management

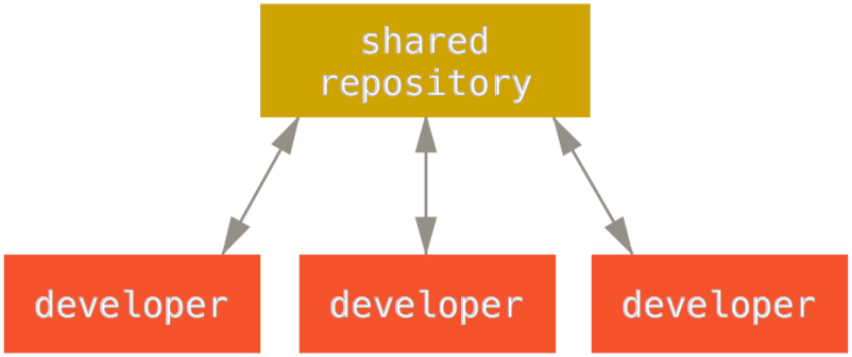
- Wie werden Tags benannt?
- Wofür werden Tags erstellt?
- Wer erstellt Tags?

Log / History Management

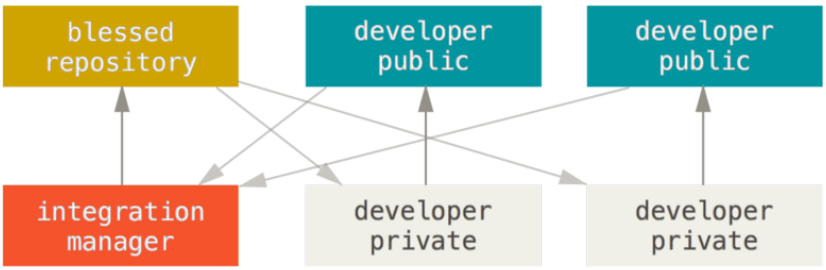
- Wie sehen Commit Messages aus?
- Interessiert mich das „Sausage Making“?
- Wie oft wird commited?
- Wird ge-squashed?
- Werden Branches gerebased oder wird Baseline gemerged?

- There are only two hard things in Computer Science: cache invalidation and naming things.-- Phil Karlton
- Separate subject from body with a blank line
- Limit the subject line to 50 characters
- Capitalize the subject line
- Do not end the subject line with a period
- Use the imperative mood in the subject line
- Wrap the body at 72 characters
- Use the body to explain what and why vs. how

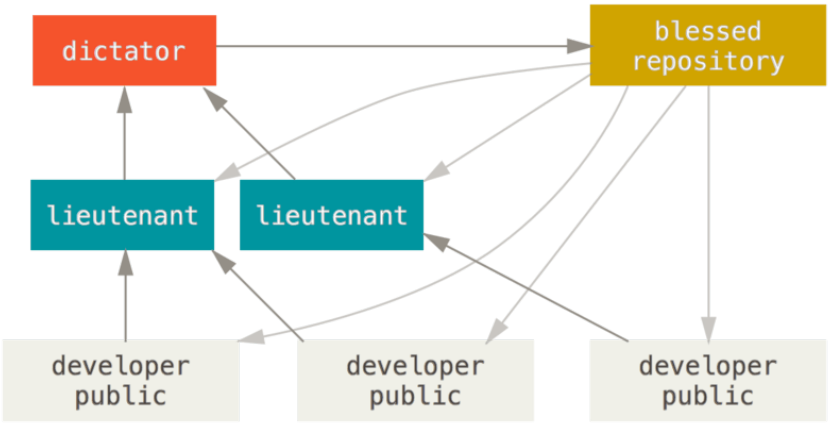
Kollaboration – Zentralisierter Workflow



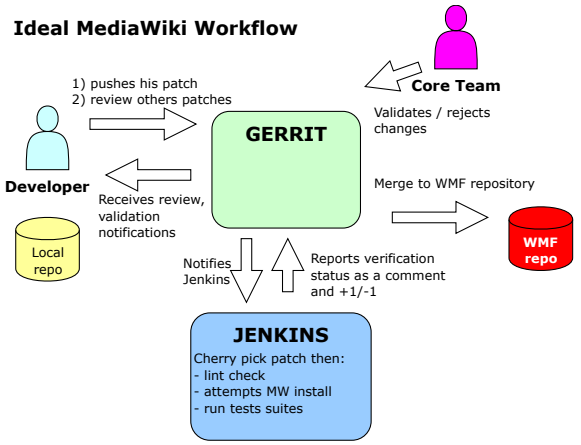
Kollaboration – Integration Manager



Kollaboration - Benevolent Dictator

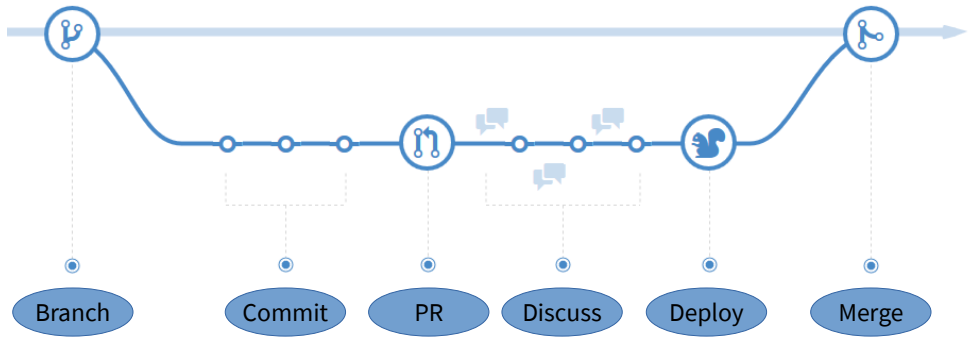


Kollaboration - Gerrit

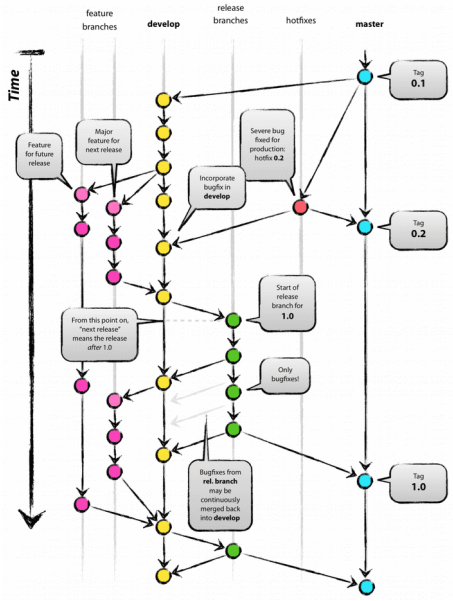


Jede Änderung wird gereviewed und muss von einer gewissen Anzahl Leuten freigegeben werden

Kollaboration - GitHub Flow



Kollaboration - Git Flow



Alltag

- Backups
- Unterschiedliche Clients (GUI vs. Nicht-GUI)
- Was ist die .gitconfig?
- Was ist autostash?
- Was ist eine globale .gitignore?
- Was sind Aliase?

Einstellungen

- user.name=Arlo O'Keeffe
- user.email=mailtoarlo@gmail.com
- core.autocrlf=true
- core.excludesfile=~/.gitignore_global
- color.ui=true
- push.default=simple
- rebase.autostash=true
- pull.ff=only
- pull.rebase=true
- alias.adog=log --all --decorate --oneline --graph

Automatisierung und Integration

- **Skripting via Bash / Batch**
- **JGIT**
- **Atlassian JIRA + BitBucket**
- **Fix / Close Commits**
- **Continuous Deployment**
  - .git Ordner nicht veröffentlichen


mailtoarlo@gmail.com
37