



ZooRoyal IT

Git

Sebastian Knott

10. April 2018

Git

2018-04-10

ZooRoyal IT

Git

Sebastian Knott

10. April 2018

Überblick



Git unter der Haube

- Datenhaltung in Git
- History
- Branch
- Merge
- Rebase

Arbeiten mit Git

- Repositories organisieren
- Git Cli



Git

2018-04-10

└ Überblick



Git unter der Haube
Datenhaltung in Git
History
Branch
Merge
Rebase
Arbeiten mit Git
Repositories organisieren
Git Cli



Git unter der Haube

Datenhaltung in Git

History

Branch

Merge

Rebase

Arbeiten mit Git

Repositories organisieren

Git Cli



2018-04-10

Git

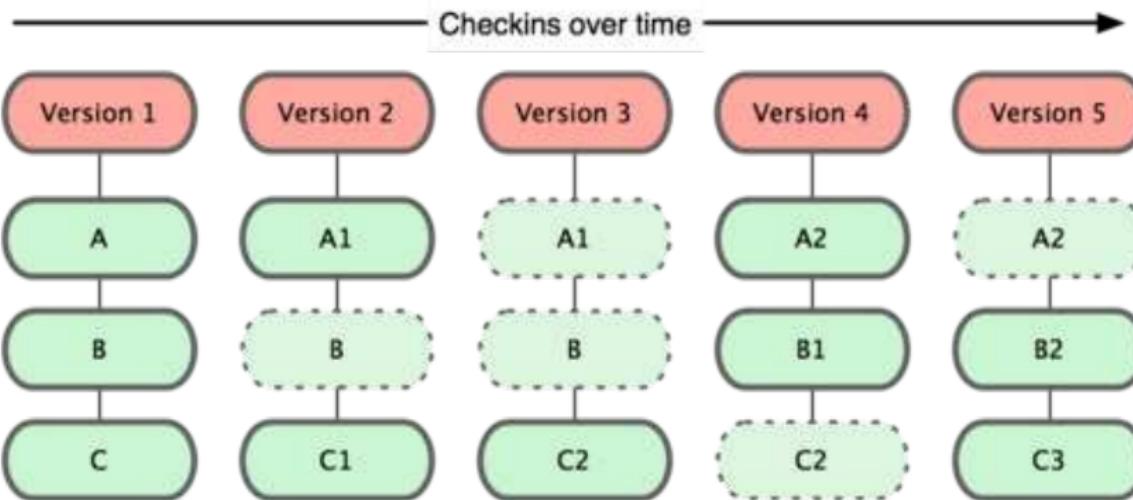
- └ Git unter der Haube
 - └ Datenhaltung in Git
 - └ Überblick



Git unter der Haube
Datenhaltung in Git
History
Branch
Merge
Rebase
Arbeiten mit Git
Repositories organisieren
Git Cli

Datenhaltung in Git

Versionen und Dateien

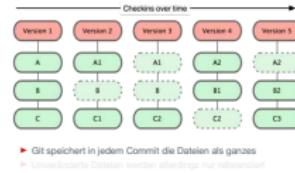


- ▶ Git speichert in jedem Commit die Dateien als ganzes
- ▶ Unveränderte Dateien werden allerdings nur referenziert

2018-04-10

Git

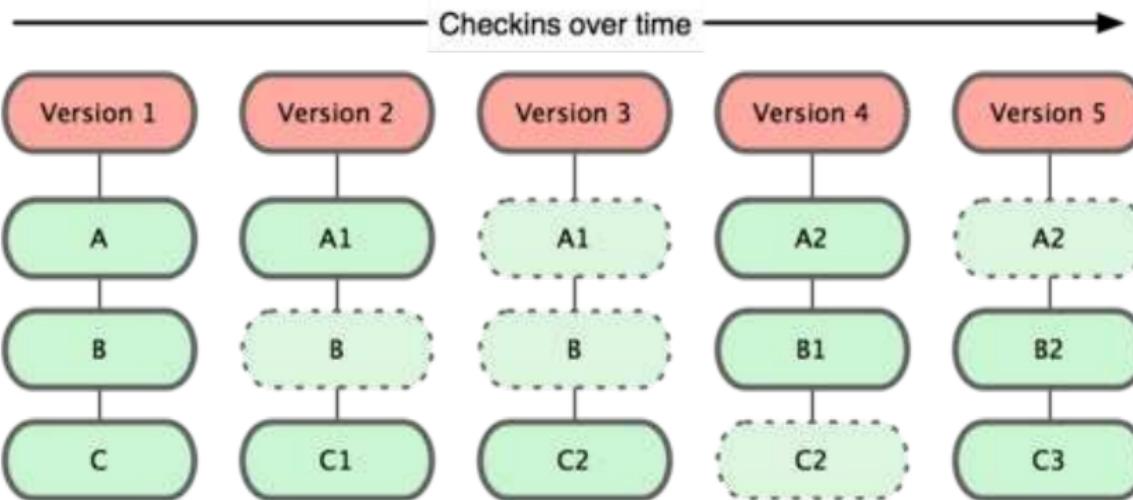
└ Git unter der Haube
 └ Datenhaltung in Git
 └ Datenhaltung in Git



1. Entspricht cp -r quelle ziel

Datenhaltung in Git

Versionen und Dateien

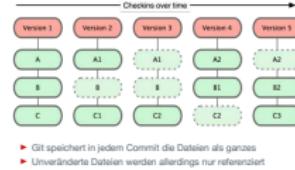


- ▶ Git speichert in jedem Commit die Dateien als ganzes
- ▶ Unveränderte Dateien werden allerdings nur referenziert

2018-04-10

Git

└ Git unter der Haube
 └ Datenhaltung in Git
 └ Datenhaltung in Git



1. Entspricht `cp -r quelle ziel`

Fragen

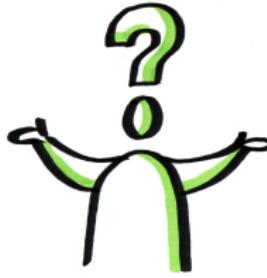
Fragen?



Git

- └ Git unter der Haube
 - └ Datenhaltung in Git
 - └ Fragen

2018-04-10



Fragen?

Datenhaltung in Git

Git Objekte

Git kennt vier Typen von Objekten

blobs Dateien

trees Verzeichnisse

commit Momentaufnahmen des Workspace

tags signierte Namen von commits

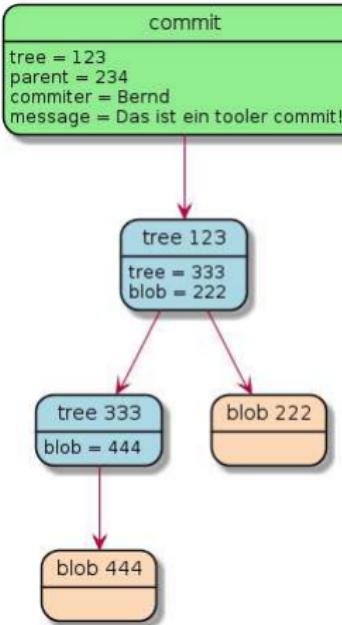
Alle Objekte werden mit einem SHA-1 Hash ihres Inhalts identifiziert



2018-04-10

Git

└ Git unter der Haube
 └ Datenhaltung in Git
 └ Datenhaltung in Git



Datenhaltung in Git

Git Objekte

Git kennt vier Typen von Objekten

blobs Dateien

trees Verzeichnisse

commit Momentaufnahmen des Workspace

tags signierte Namen von commits

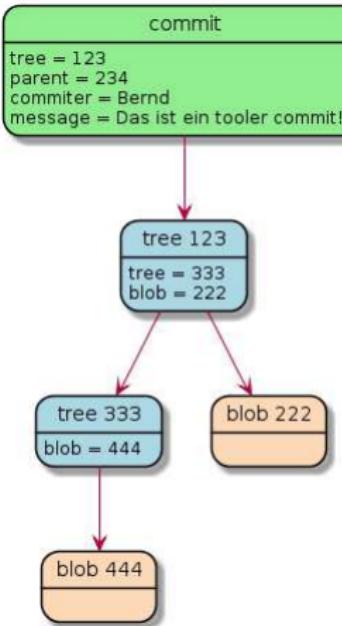
Alle Objekte werden mit einem SHA-1 Hash ihres Inhalts identifiziert



Git

└ Git unter der Haube
 └ Datenhaltung in Git
 └ Datenhaltung in Git

2018-04-10



Fragen

Fragen?



Git

- └ Git unter der Haube
 - └ Datenhaltung in Git
 - └ Fragen

2018-04-10



Aufgabe: Verzeichnisbaum anzeichnen und neuen Git Commit mit Verzeichnissen und Daten konstruieren lassen.



Fragen?



Git unter der Haube

Datenhaltung in Git

History

Branch

Merge

Rebase

Arbeiten mit Git

Repositories organisieren

Git Cli



2018-04-10

Git

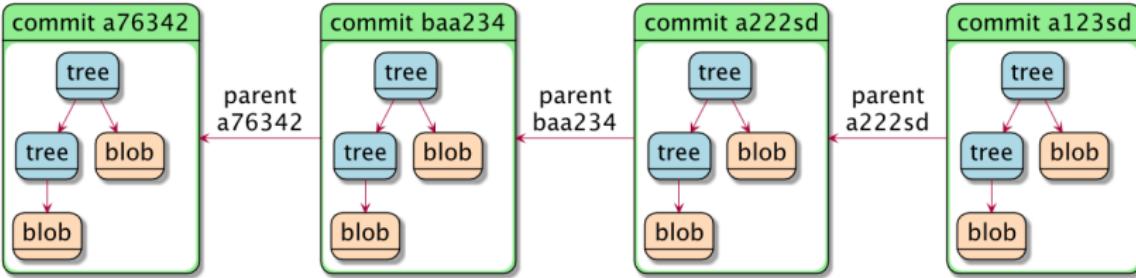
- └ Git unter der Haube
 - └ History
 - └ Überblick



Git unter der Haube
Datenhaltung in Git
History
Branch
Merge
Rebase
Arbeiten mit Git
Repositories organisieren
Git Cli

History

Verkettung von commits

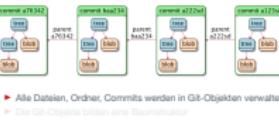


- ▶ Alle Dateien, Ordner, Commits werden in Git-Objekten verwaltet
- ▶ Die Git-Objekte bilden eine Baumstruktur

2018-04-10

Git

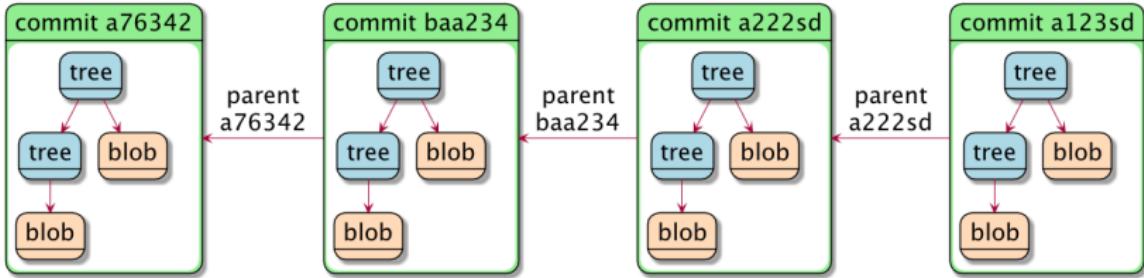
└ Git unter der Haube
 └ History
 └ History



1. Aufgabe: Commit anzeichnen. Eine Datei verändert. Neuen Commit anzeichnen lassen.

History

Verkettung von commits

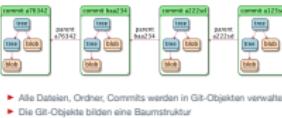


- ▶ Alle Dateien, Ordner, Commits werden in Git-Objekten verwaltet
- ▶ Die Git-Objekte bilden eine Baumstruktur

2018-04-10

Git

└ Git unter der Haube
 └ History
 └ History



1. Aufgabe: Commit anzeichnen. Eine Datei verändert. Neuen Commit anzeichnen lassen.

Fragen

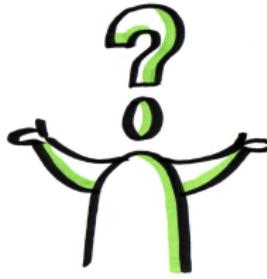
Fragen?



Git

- └ Git unter der Haube
 - └ History
 - └ Fragen

2018-04-10



Fragen?

Überblick



Git unter der Haube

Datenhaltung in Git

History

Branch

Merge

Rebase

Arbeiten mit Git

Repositories organisieren

Git Cli



2018-04-10

Git

└─Git unter der Haube
 └─Branch
 └─Überblick

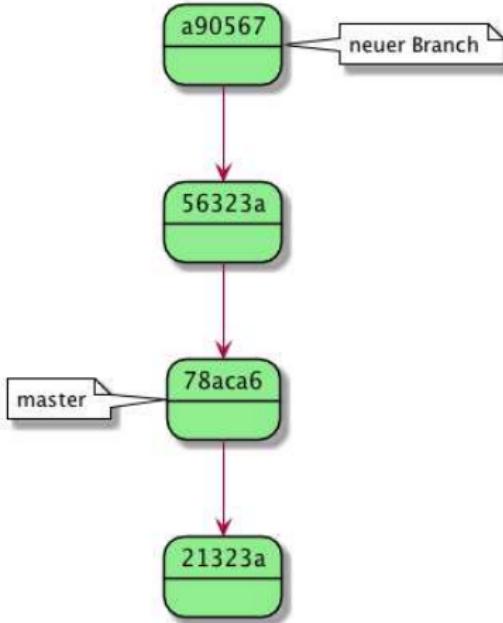


Branches

Organisation von Commits

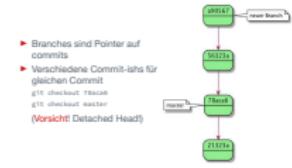


- ▶ Branches sind Pointer auf commits
- ▶ Verschiedene Commit-Isxs für gleichen Commit
`git checkout 78aca6`
`git checkout master`
(Vorsicht! Detached Head!)



2018-04-10

Git
└─ Git unter der Haube
 └─ Branch
 └─ Branches

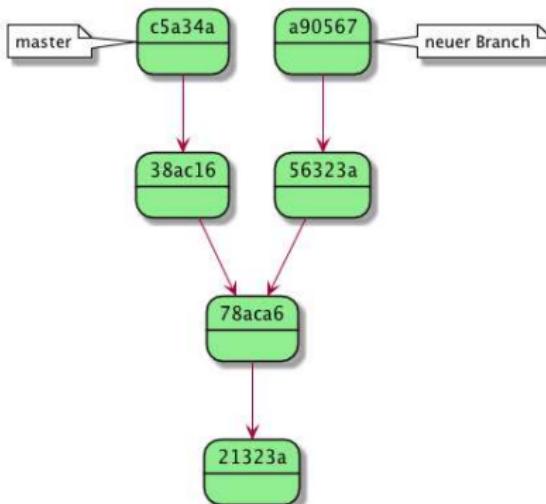


Branches

Organisation von Commits



Neue commits in einem Branch
verschieben den Pointer



2018-04-10

Git

└ Git unter der Haube
 └ Branch
 └ Branches



- Aufgabe1: Git Commit anzeichnen. Weitere Branches erzeugen lassen. Weitere Commits erzeugen lassen. Einen Branch resetten.
- Hashes verändern sich nicht!

Fragen

Fragen?



Git

└ Git unter der Haube
 └ Branch
 └ Fragen

2018-04-10



Fragen?



Git unter der Haube

Datenhaltung in Git

History

Branch

Merge

Rebase

Arbeiten mit Git

Repositories organisieren

Git Cli



2018-04-10

Git

- └ Git unter der Haube
 - └ Merge
 - └ Überblick



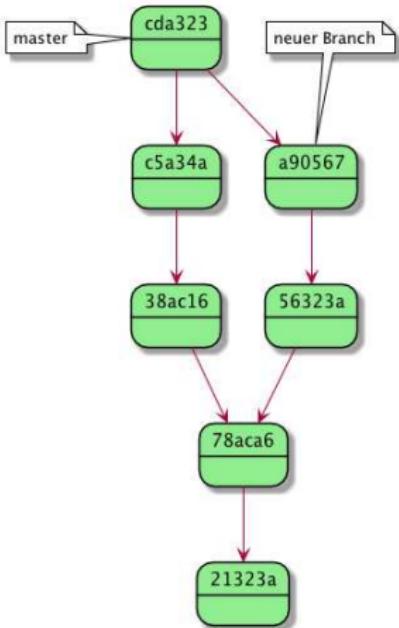
Git unter der Haube
Datenhaltung in Git
History
Branch
Merge
Rebase
Arbeiten mit Git
Repositories organisieren
Git Cli

Merge

Zusammenführen von zwei Branches



- ▶ Neuer Branch → master
- ▶ Quellbranch bleibt bestehen
- ▶ Pointer im Quellbranch wird nicht bewegt
- ▶ Neuer commit (`cda323`) hat zwei Parents
- ▶ `git branch -d neuer Branch` entfernt nur den Pointer, nicht die commits



2018-04-10

Git

└ Git unter der Haube
 └ Merge
 └ Merge



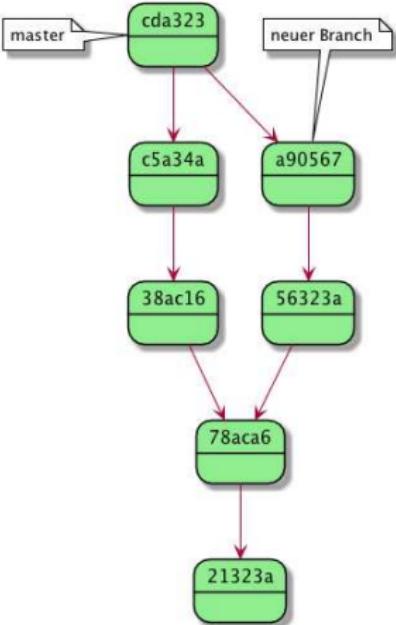
1. Mit HEAD 1 wird uneindeutig. HEAD¹ gibt Parent an

Merge

Zusammenführen von zwei Branches



- ▶ Neuer Branch → master
- ▶ Quellbranch bleibt bestehen
- ▶ Pointer im Quellbranch wird nicht bewegt
- ▶ Neuer commit (`cda323`) hat zwei Parents
- ▶ `git branch -d neuer Branch` entfernt nur den Pointer, nicht die commits



2018-04-10

Git

└ Git unter der Haube
 └ Merge
 └ Merge

- ▶ Neuer Branch → master
- ▶ Quellbranch bleibt bestehen
- ▶ Pointer im Quellbranch wird nicht bewegt
- ▶ Neuer commit (neuer) hat zwei Parents
- ▶ `git branch -d neuer Branch` entfernt nur den Pointer, nicht die commits



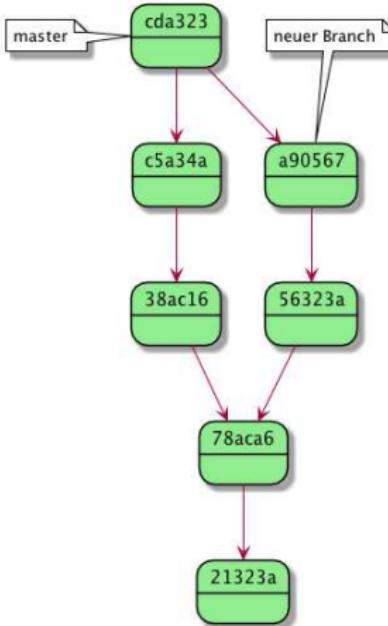
1. Mit HEAD 1 wird uneindeutig. HEAD^1 gibt Parent an

Merge

Zusammenführen von zwei Branches



- ▶ Neuer Branch → master
- ▶ Quellbranch bleibt bestehen
- ▶ Pointer im Quellbranch wird nicht bewegt
- ▶ Neuer commit (`cda323`) hat zwei Parents
- ▶ `git branch -d neuer Branch` entfernt nur den Pointer, nicht die commits



2018-04-10

Git

└ Git unter der Haube
 └ Merge
 └ Merge



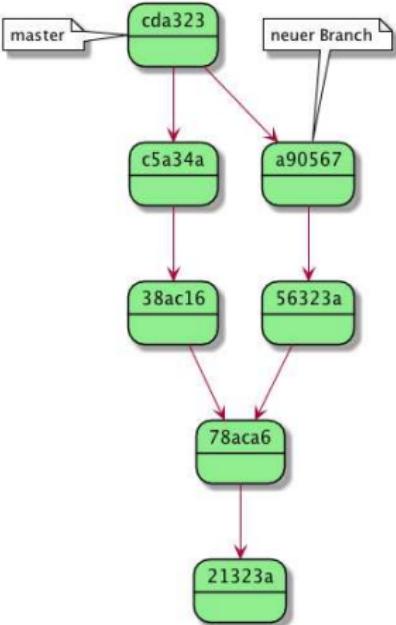
1. Mit HEAD 1 wird uneindeutig. HEAD¹ gibt Parent an

Merge

Zusammenführen von zwei Branches



- ▶ Neuer Branch → master
- ▶ Quellbranch bleibt bestehen
- ▶ Pointer im Quellbranch wird nicht bewegt
- ▶ Neuer commit (`cda323`) hat zwei Parents
- ▶ `git branch -d neuer Branch` entfernt nur den Pointer, nicht die commits

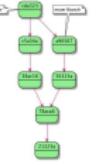


2018-04-10

Git

└ Git unter der Haube
 └ Merge
 └ Merge

- ▶ Neuer Branch → master
- ▶ Quellbranch bleibt bestehen
- ▶ Pointer im Quellbranch wird nicht bewegt
- ▶ Neuer commit (`cda323`) hat zwei Parents
- ▶ `git branch -d neuer Branch` entfernt nur den Pointer, nicht die commits



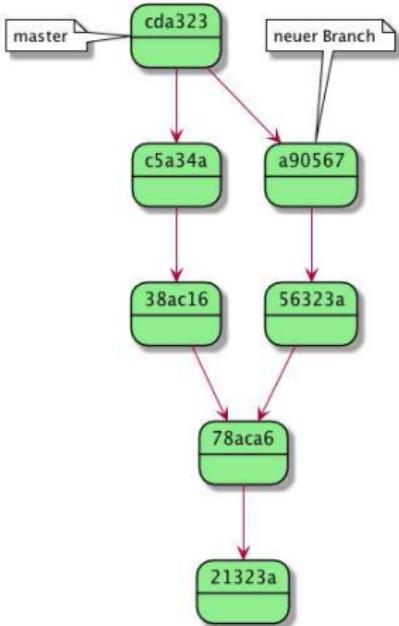
1. Mit HEAD 1 wird uneindeutig. HEAD^1 gibt Parent an

Merge

Zusammenführen von zwei Branches



- ▶ Neuer Branch → master
- ▶ Quellbranch bleibt bestehen
- ▶ Pointer im Quellbranch wird nicht bewegt
- ▶ Neuer commit (`cda323`) hat zwei Parents
- ▶ `git branch -d neuer Branch` entfernt nur den Pointer, nicht die commits

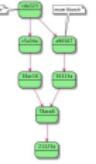


2018-04-10

Git

└ Git unter der Haube
 └ Merge
 └ Merge

- ▶ Neuer Branch → master
- ▶ Quellbranch bleibt bestehen
- ▶ Pointer im Quellbranch wird nicht bewegt
- ▶ Neuer commit (`cda323`) hat zwei Parents
- ▶ `git branch -d neuer Branch` entfernt nur den Pointer, nicht die commits



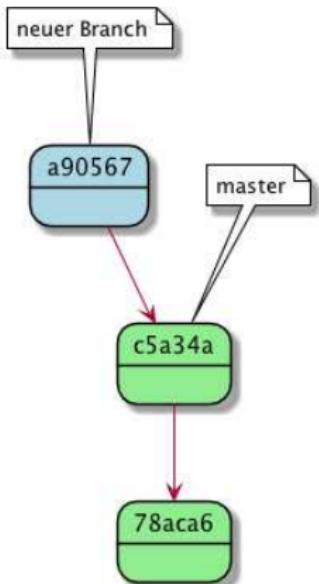
1. Mit HEAD 1 wird uneindeutig. HEAD^1 gibt Parent an

Merge

Mergen mit Fast Forward



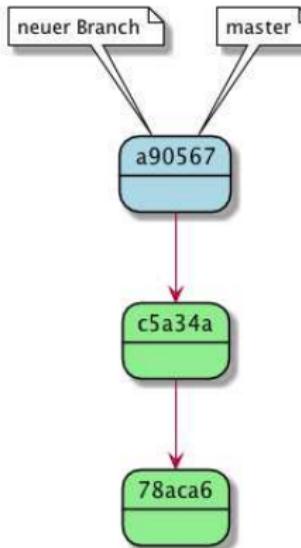
Vorher



Entspricht Rebbase +
Reset master

- ▶ Pointer wird weiter geschoben
- ▶ Nur ein Parent
- ▶ Lineare History
- ▶ Git Default

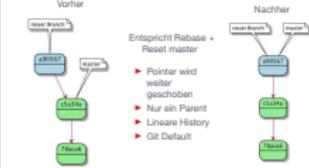
Nachher



2018-04-10

Git

└─ Git unter der Haube
 └─ Merge
 └─ Merge



Fragen

Fragen?



2018-04-10
Git
└ Git unter der Haube
 └ Merge
 └ Fragen



Fragen?

Überblick



Git unter der Haube

Datenhaltung in Git

History

Branch

Merge

Rebase

Arbeiten mit Git

Repositories organisieren

Git Cli



2018-04-10

Git

- └ Git unter der Haube
 - └ Rebases
 - └ Überblick



Git unter der Haube
Datenhaltung in Git
History
Branch
Merge
Rebase
Arbeiten mit Git
Repositories organisieren
Git Cli

Rebase

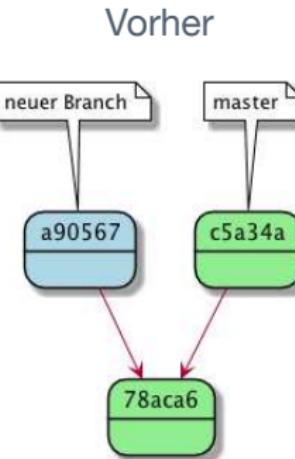
Was passiert bei einem Rebase



Git

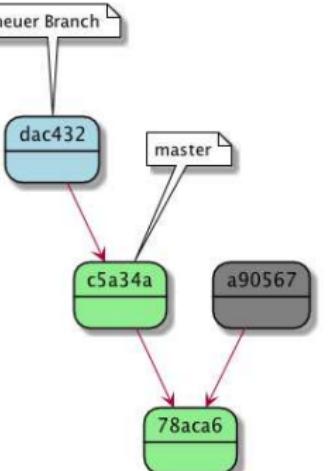
└ Git unter der Haube
 └ Rebbase
 └ Rebbase

2018-04-10



- ▶ Commit wird kopiert und Parent gesetzt
- ▶ master Pointer wird **nicht** verschoben
- ▶ Lineare History

Nachher



Rebase

Interaktive Rebases



Rebase gibt einem weitere Möglichkeiten Commits umzuorganisieren.

Drop Einzelne Commits entfernen

Squash, Fixup Commits zusammenführen

Reword Einzelne Commit Messages ändern

Edit Einzelne Commits amenden

Operationen können beliebig kombiniert werden

Git

└ Git unter der Haube
 └ Rebbase
 └ Rebbase

2018-04-10

Rebase gibt einem weitere Möglichkeiten Commits umzuorganisieren.
Drop Einzelne Commits entfernen
Squash, Fixup Commits zusammenführen
Reword Einzelne Commit Messages ändern
Edit Einzelne Commits amenden
Operationen können beliebig kombiniert werden

Rebase

Interaktive Rebases



Rebase gibt einem weitere Möglichkeiten Commits umzuorganisieren.

Drop Einzelne Commits entfernen

Squash, Fixup Commits zusammenführen

Reword Einzelne Commit Messages ändern

Edit Einzelne Commits amenden

Operationen können beliebig kombiniert werden

Git

└ Git unter der Haube
 └ Rebbase
 └ Rebbase

2018-04-10

Rebase gibt einem weitere Möglichkeiten Commits umzuorganisieren.
Drop Einzelne Commits entfernen
Squash, Fixup Commits zusammenführen
Reword Einzelne Commit Messages ändern
Edit Einzelne Commits amenden
Operationen können beliebig kombiniert werden

Rebase

Interaktive Rebases



Rebase gibt einem weitere Möglichkeiten Commits umzuorganisieren.

Drop Einzelne Commits entfernen

Squash, Fixup Commits zusammenführen

Reword Einzelne Commit Messages ändern

Edit Einzelne Commits amenden

Operationen können beliebig kombiniert werden

Git

└ Git unter der Haube
 └ Rebbase
 └ Rebbase

2018-04-10

Rebase gibt einem weitere Möglichkeiten Commits umzuorganisieren.
Drop Einzelne Commits entfernen
Squash, Fixup Commits zusammenführen
Reword Einzelne Commit Messages ändern
Edit Einzelne Commits amenden
Operationen können beliebig kombiniert werden

Rebase

Interaktive Rebases



Rebase gibt einem weitere Möglichkeiten Commits umzuorganisieren.

Drop Einzelne Commits entfernen

Squash, Fixup Commits zusammenführen

Reword Einzelne Commit Messages ändern

Edit Einzelne Commits amenden

Operationen können beliebig kombiniert werden

Git

└ Git unter der Haube
 └ Rebbase
 └ Rebbase

2018-04-10

Rebase gibt einem weitere Möglichkeiten Commits umzuorganisieren.
Drop Einzelne Commits entfernen
Squash, Fixup Commits zusammenführen
Reword Einzelne Commit Messages ändern
Edit Einzelne Commits amenden
Operationen können beliebig kombiniert werden

Rebase

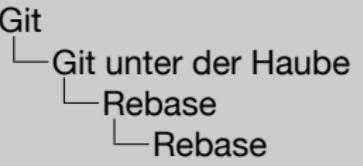
Mächtiges Rebase



Mit großer Macht kommt große Verwirrung

- ▶ Vorsicht vor Branches, auf denen andere Entwickler Arbeiten
- ▶ Überkomplexe Branches machen Rebases extrem schwer
- ▶ Mitdenken erforderlich

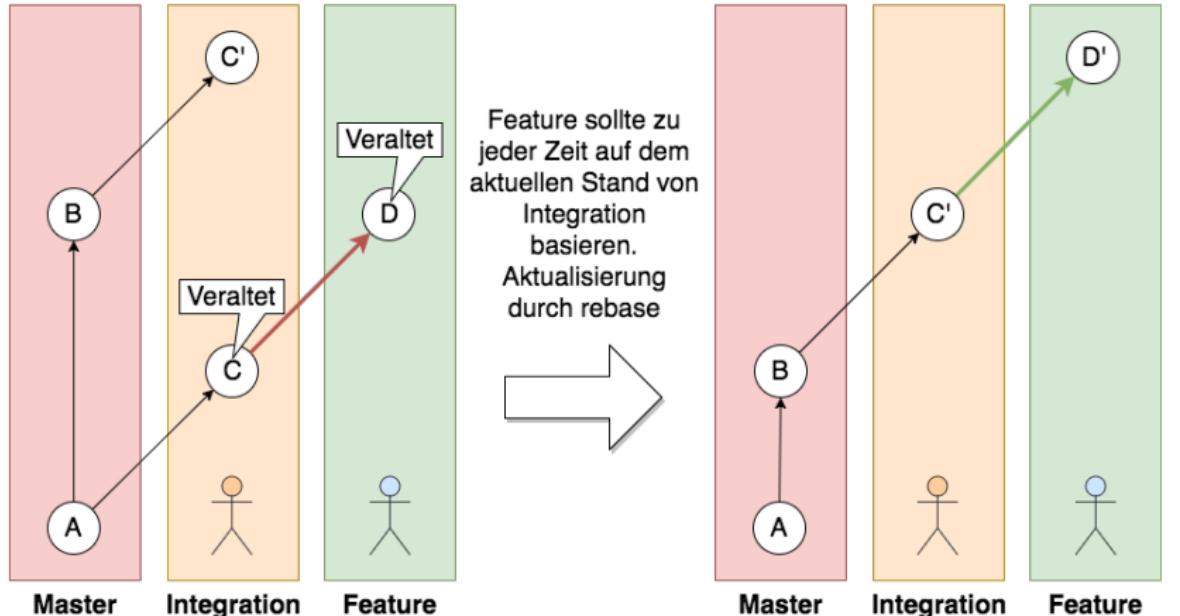
2018-04-10



► Vorsicht vor Branches, auf denen andere Entwickler Arbeiten
► Überkomplexe Branches machen Rebases extrem schwer
► Mitdenken erforderlich

Rebase

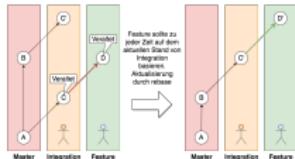
Veraltete Stände



2018-04-10

Git

└ Git unter der Haube
 └ Rebase
 └ Rebase



Fragen

Fragen?



2018-04-10
Git
└ Git unter der Haube
 └ Rebbase
 └ Fragen



Fragen?



Git unter der Haube

Datenhaltung in Git

History

Branch

Merge

Rebase

Arbeiten mit Git

Repositories organisieren

Git Cli



2018-04-10

Git

Arbeiten mit Git

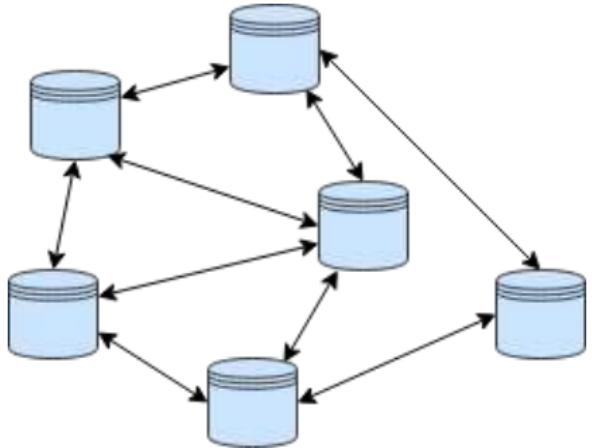
Repositories organisieren

Überblick



Repositories organisieren

Repositories beliebig verknüpfen

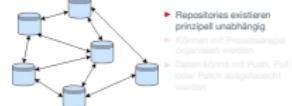


- ▶ Repositories existieren prinzipiell unabhängig
- ▶ Können mit Prozesseregel organisiert werden
- ▶ Daten können mit Push, Pull oder Patch ausgetauscht werden

2018-04-10

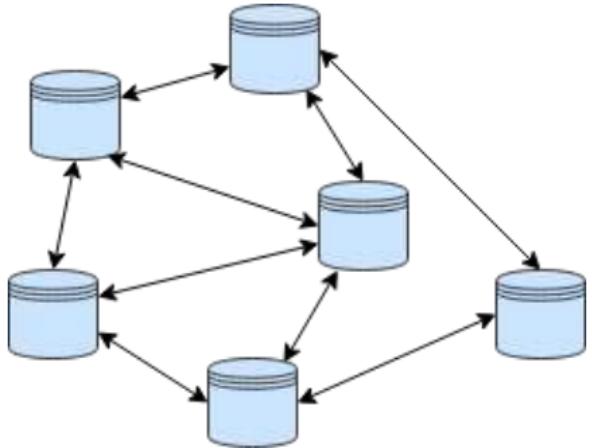
Git

Arbeiten mit Git
└ Repositories organisieren
 └ Repositories organisieren



Repositories organisieren

Repositories beliebig verknüpfen

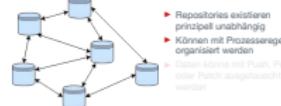


- ▶ Repositories existieren prinzipiell unabhängig
- ▶ Können mit Prozesseregel organisiert werden
- ▶ Daten können mit Push, Pull oder Patch ausgetauscht werden

2018-04-10

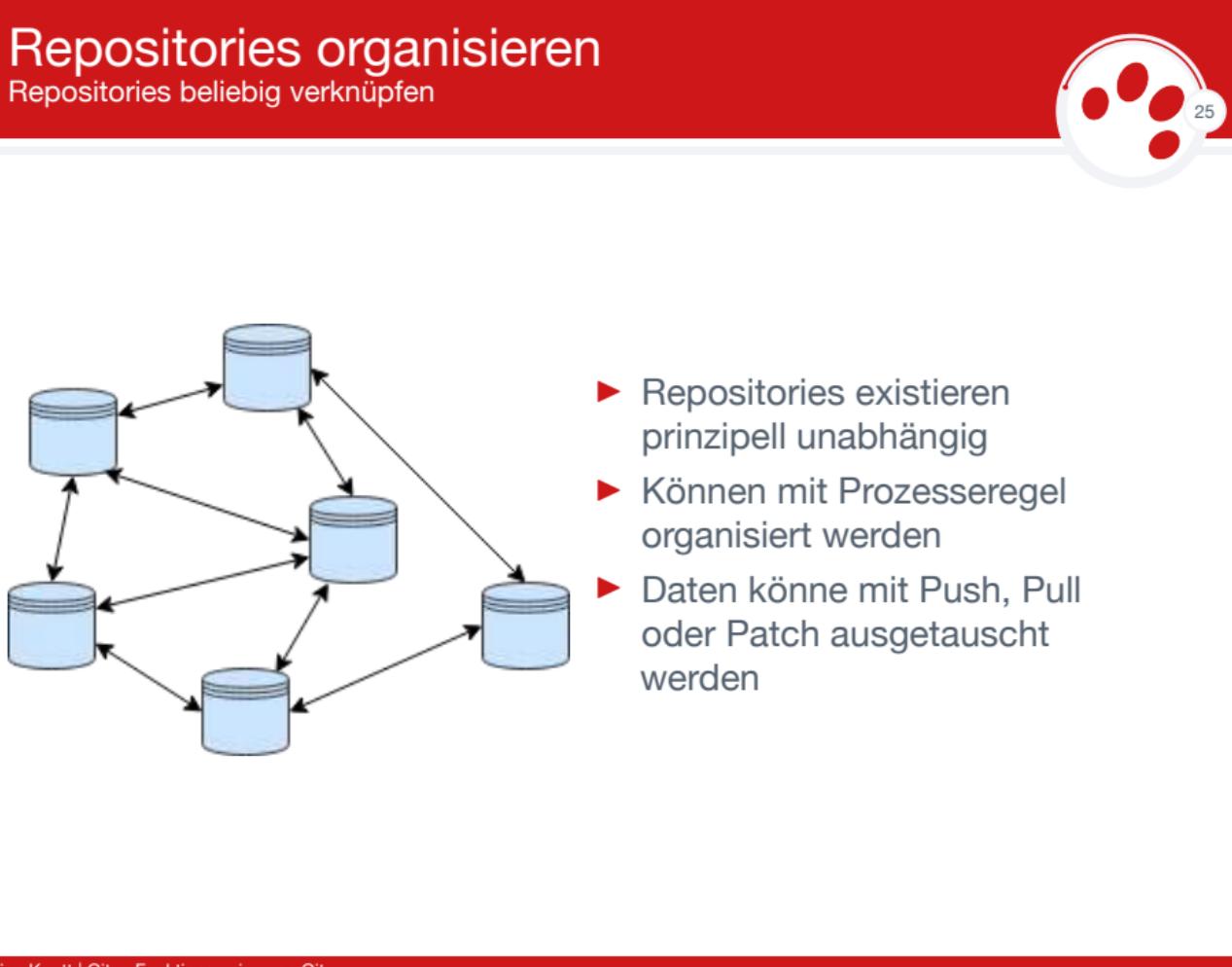
Git

Arbeiten mit Git
└ Repositories organisieren
 └ Repositories organisieren



Repositories organisieren

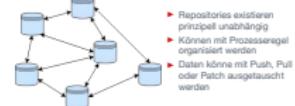
Repositories beliebig verknüpfen



Git

Arbeiten mit Git
Repositories organisieren
Repositories organisieren

2018-04-10





Git bietet einige Protokolle zur Datenübertragung

- ▶ git
- ▶ file
- ▶ ssh
- ▶ http
- ▶ mailto

Git

- └ Arbeiten mit Git
 - └ Repositories organisieren
 - └ Repositories organisieren

2018-04-10

Git bietet einige Protokolle zur Datenübertragung

- ▶ git
- ▶ file
- ▶ ssh
- ▶ http
- ▶ mailto

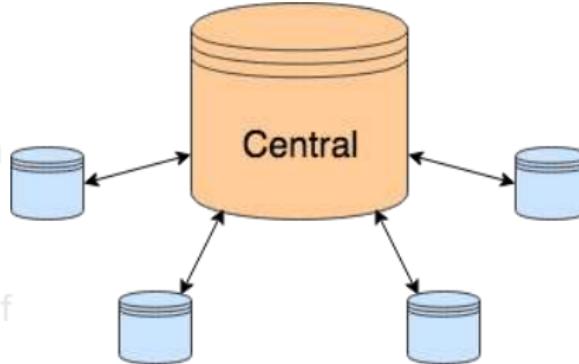
Repositories organisieren

Zentrales Repository



Zentrales Repository

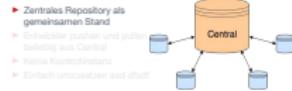
- ▶ Zentrales Repository als gemeinsamen Stand
- ▶ Entwickler pushen und pullen beliebig aus Central
- ▶ Keine Kontrollinstanz
- ▶ Einfach umzusetzen asd dfsdf



Git

- └ Arbeiten mit Git
- └ Repositories organisieren
- └ Repositories organisieren

2018-04-10



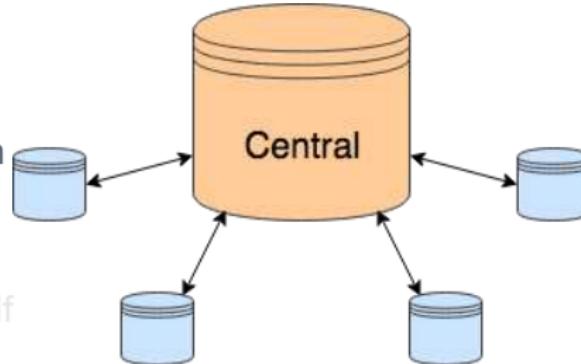
Repositories organisieren

Zentrales Repository



Zentrales Repository

- ▶ Zentrales Repository als gemeinsamen Stand
- ▶ Entwickler pushen und pullen beliebig aus Central
- ▶ Keine Kontrollinstanz
- ▶ Einfach umzusetzen asd dfsdf



Git

- Arbeiten mit Git
- Repositories organisieren
- Repositories organisieren

2018-04-10

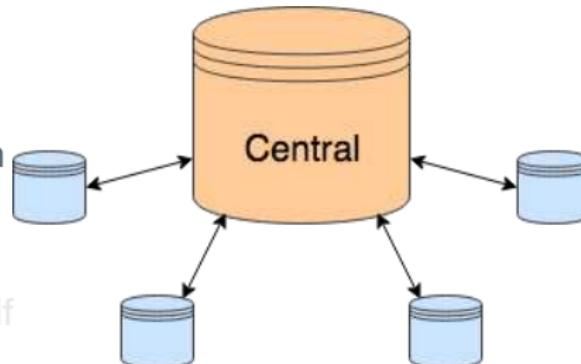


Repositories organisieren

Zentrales Repository



- ▶ Zentrales Repository als gemeinsamen Stand
- ▶ Entwickler pushen und pullen beliebig aus Central
- ▶ Keine Kontrollinstanz
- ▶ Einfach umzusetzen asd dfsdf



Git

- └ Arbeiten mit Git
- └ Repositories organisieren
- └ Repositories organisieren

2018-04-10



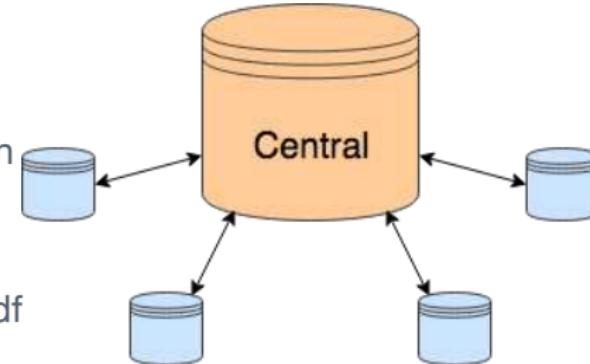
Repositories organisieren

Zentrales Repository



Zentrales Repository

- ▶ Zentrales Repository als gemeinsamen Stand
- ▶ Entwickler pushen und pullen beliebig aus Central
- ▶ Keine Kontrollinstanz
- ▶ Einfach umzusetzen asd dfsdf



Git

Arbeiten mit Git
└ Repositories organisieren
 └ Repositories organisieren

2018-04-10

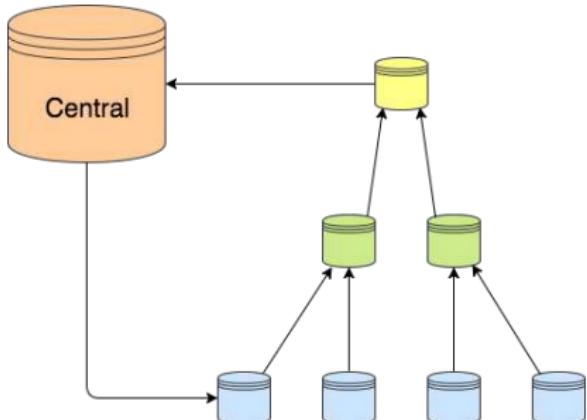
- ▶ Zentrales Repository als gemeinsamen Stand
- ▶ Entwickler pushen und pullen beliebig aus Central
- ▶ Keine Kontrollinstanz
- ▶ Einfach umzusetzen asd dfsdf

Repositories organisieren

Zentrales Repository



- ▶ Verteilte Verantwortung
(Network of Trust)
- ▶ Verwalter betrachten
verschiedene Scopes
- ▶ Hochflexibel



2018-04-10

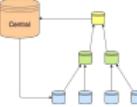
Git

- └ Arbeiten mit Git
 - └ Repositories organisieren
 - └ Repositories organisieren

Verteilte Verantwortung
(Network of Trust)

Central verantwortet verschiedene Scopes

Hochflexibel

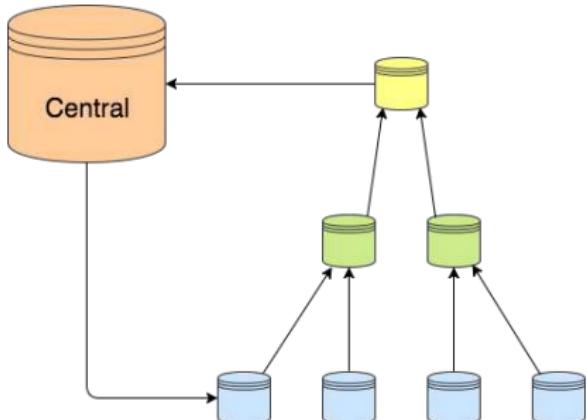


Repositories organisieren

Zentrales Repository



- ▶ Verteilte Verantwortung (Network of Trust)
- ▶ Verwalter betrachten verschiedene Scopes
- ▶ Hochflexibel

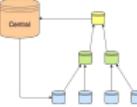


2018-04-10

Git

- └ Arbeiten mit Git
 - └ Repositories organisieren
 - └ Repositories organisieren

- ▶ Verteilte Verantwortung (Network of Trust)
- ▶ Verwalter betrachten verschiedene Scopes

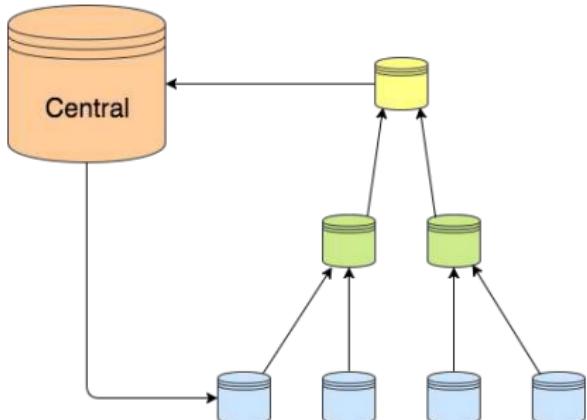


Repositories organisieren

Zentrales Repository



- ▶ Verteilte Verantwortung (Network of Trust)
- ▶ Verwalter betrachten verschiedene Scopes
- ▶ Hochflexibel

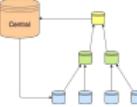


2018-04-10

Git

- └ Arbeiten mit Git
 - └ Repositories organisieren
 - └ Repositories organisieren

- ▶ Verteilte Verantwortung (Network of Trust)
- ▶ Verwalter betrachten verschiedene Scopes
- ▶ Hochflexibel

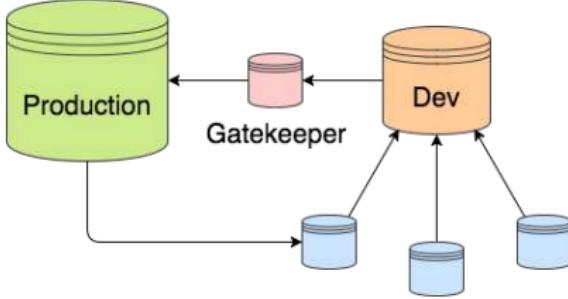


Repositories organisieren

Zentrales Repository



- ▶ Continuous integration
- ▶ Automatische Tests
- ▶ Abnahme
- ▶ Einheitlicher Entwicklungsstand
- ▶ Releases



2018-04-10

Git

- └ Arbeiten mit Git
 - └ Repositories organisieren
 - └ Repositories organisieren

- ▶ Continuous integration
- ▶ Automatische Tests
- ▶ Abnahme
- ▶ Einheitlicher Entwicklungsstand
- ▶ Releases



Fragen

Fragen?



2018-04-10

Git

- Arbeiten mit Git
 - Repositories organisieren
 - Fragen



Fragen?

Überblick



Git unter der Haube

Datenhaltung in Git

History

Branch

Merge

Rebase

Arbeiten mit Git

Repositories organisieren

Git Cli



2018-04-10

Git
└ Arbeiten mit Git
 └ Git Cli
 └ Überblick



Git unter der Haube
Datenhaltung in Git
History
Branch
Merge
Rebase
Arbeiten mit Git
Repositories organisieren
Git Cli

Git CLI

Warum CLI?



- ▶ Use Git the Git way
- ▶ Umfangreicher Werkzeugkofer
- ▶ Schnell
- ▶ Zielsicher
- ▶ Stabil

```
git reflog | cat
d5f470e HEAD@{0}: commit: Git strukturen halb durch
3518d7b HEAD@{1}: reset: moving to HEAD
3518d7b HEAD@{2}: commit: Mockery Farbcodes in Listings geändert
ec41e65 HEAD@{3}: commit: Gitvortrag angefangen
4659920 HEAD@{4}: commit (amend): Komprimierter Vortrag. Bereinigtes leeres Projekt
c84a2c6 HEAD@{5}: commit: Komprimierter Vortrag. Bereinigtes leeres Projekt
1716399 HEAD@{6}: commit: Initial commit
9ac8809 HEAD@{7}: reset: moving to HEAD
9ac8809 HEAD@{8}: commit: Mockery Vortrag
d846250 HEAD@{10}: commit (amend): UnitTest vorversion; FeatherTemplateZR release
724716a HEAD@{11}: commit (amend): UnitTest vorversion; FeatherTemplateZR release
7fdc995 HEAD@{12}: commit (amend): UnitTest vorversion; FeatherTemplateZR release
1a0514e HEAD@{13}: commit (amend): UnitTest vorversion; FeatherTemplateZR release
d854940 HEAD@{14}: commit (initial): Initial Commit

git status
On branch master
Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git checkout -- <file>..." to discard changes in working directory)

    modified:   Git/Git.pdf
    modified:   Git/Git.tex

Untracked files:
  (use "git add <file>..." to include in what will be committed)

    Git/Images/git_logo.png

no changes added to commit (use "git add" and/or "git commit -a")
```

2018-04-10
Git
└ Arbeiten mit Git
 └ Git Cli
 └ Git CLI

- ▶ Use Git the Git way
- ▶ Umfangreicher Werkzeugkofer
- ▶ Schnell
- ▶ Zielsicher
- ▶ Stabil



Git CLI

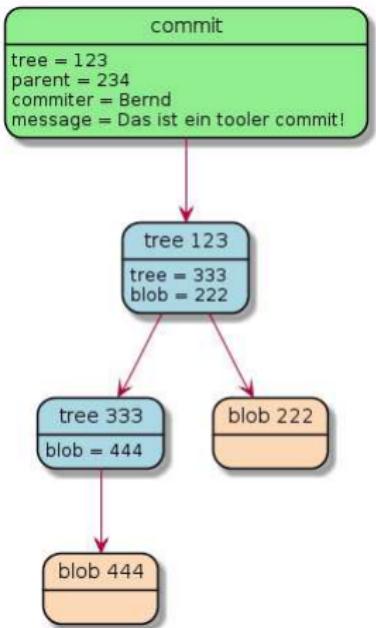
Adressierung von Git-Objekten



commit-ish

A commit object or an object that can be recursively dereferenced to a commit object.

- ▶ a commit object
- ▶ a tag object that points to a commit object



2018-04-10

Git

Arbeiten mit Git
└ Git Cli
 └ Git CLI

commit-ish
A commit object or an object that can be recursively dereferenced to a commit object.

- ▶ a commit object
- ▶ a tag object that points to a commit object



Git CLI

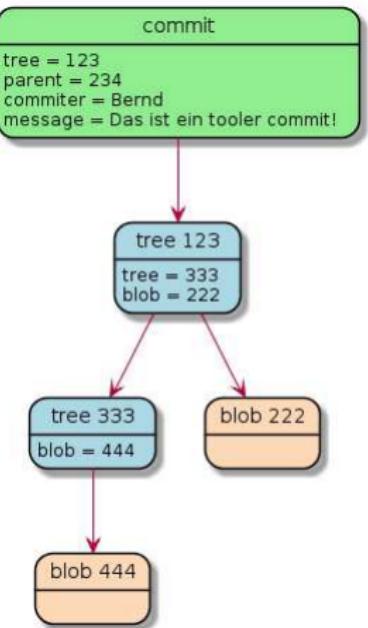
Adressierung von Git-Objekten



tree-ish

A tree object or an object that can be recursively dereferenced to a tree object.

- ▶ a commit-ish
(tree of top directory)
- ▶ a tree object
- ▶ a tag object that points to a tree object



2018-04-10

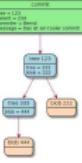
Git

Arbeiten mit Git
└ Git Cli
 └ Git CLI

tree-ish

A tree object or an object that can be recursively dereferenced to a tree object.

- ▶ a commit-ish
(tree of top directory)
- ▶ a tree object
- ▶ a tag object that points to a tree object



Git CLI

Adressierung von Git-Objekten



commit-ish

	Example
<sha1>	dae86e1950b1277e545cee180551750029cf735
<describeOutput>	v1.7.4.2-679-g3bee7fb
<refname>	master, heads/master, refs/heads/master
<refname>@{<date>}	master@{yesterday}, HEAD@{5 minutes}
<refname>@{<n>}	master@{1}
@{<n>}	@{1}
@{-<n>}	@{-1}
<refname>@{upstream}	master@{upstream}, @{u}
<rev>^	HEAD^, v1.5.1^0
<rev>~<n>	master~3
<rev>^{<type>}	v0.99.8^{commit}
<rev>^{}{}	v0.99.8^{}{}
<rev>^{/<text>}	HEAD^{/fix nasty bug}
:<text>	:/fix nasty bug

► Git Dokumentation

Git

Arbeiten mit Git
└─ Git Cli
 └─ Git CLI

2018-04-10

commish
commit
describeOutput
refname
refname@{date}
refname@{n}
ref@{1}
ref@{-1}
ref@{upstream}, @{u}
HEAD^, v1.5.1^0
master~3
v0.99.8^{commit}
v0.99.8^{}{}
HEAD^{/fix nasty bug}
:/fix nasty bug

Fragen

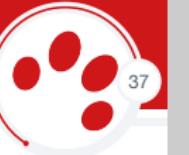
Fragen?



Fragen?

2018-04-10
Git
└ Arbeiten mit Git
 └ Git Cli
 └ Fragen





git reflog

► Git Dokumentation

- Historie der letzten Git-Operationen
- Commit-ish benutzen um zum Stand zurück zu kehren

```
git reset --hard HEAD@{5}
```

```
d5f470e (HEAD -> master) HEAD@{0}: commit: Gitvortrag angefangen
3518d7b HEAD@{1}: reset: moving to HEAD
3518d7b HEAD@{2}: commit: Mockery Farbcodes
ec41e65 HEAD@{3}: commit: Gitvortrag angefangen
4659920 HEAD@{4}: commit (amend): Komprimierter Vortrag
c84a20c HEAD@{5}: commit: Komprimierter Vortrag
1716399 HEAD@{6}: commit: Initial commit
9ac8809 HEAD@{7}: reset: moving to HEAD
9ac8809 HEAD@{8}: reset: moving to HEAD
9ac8809 HEAD@{9}: commit: Mockery Vortrag
d846250 HEAD@{10}: commit (amend): UnitTest vorbereitet
724716a HEAD@{11}: commit (amend): UnitTest vorbereitet
7fdc995 HEAD@{12}: commit (amend): UnitTest vorbereitet
1a0514e HEAD@{13}: commit: UnitTest vorbereitet
d854940 HEAD@{14}: commit (initial): Initial commit
(END)
```

2018-04-10

Git
└ Arbeiten mit Git
 └ Git Cli
 └ Reflog



Status

Status des aktuellen Workspace

git status

► Git Dokumentation

- ▶ Working mode
(Rebasing, Merging ...)
- ▶ Branch, Remotes
- ▶ Staged, Unstaged

git status

git status -sb #aka STFU-Mode



```
git status
On branch master
Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git checkout -- <file>..." to discard changes in working directory)

        modified:   Git/Git.pdf
        modified:   Git/Git.tex

Untracked files:
  (use "git add <file>..." to include in what will be committed)

        Git/Images/git_client.png
        Git/Images/git_logo.png
        Git/Images/git_reflog.png

no changes added to commit (use "git add" and/or "git commit")

git status -sb
## master
M Git/Git.pdf
M Git/Git.tex
?? Git/Images/git_client.png
?? Git/Images/git_logo.png
?? Git/Images/git_reflog.png
```

2018-04-10

Git
└ Arbeiten mit Git
 └ Git Cli
 └ Status

```
git status
Working mode
(Rebasing, Merging ...)
Branch, Remotes
Staged, Unstaged

git status -sb #aka STFU-Mode
```

Add

Dateien stagieren

git add

► Git Dokumentation

- ▶ Fügt unbekannte Dateien zum index hinzu
- ▶ Staged Änderungen an bekannten Datien

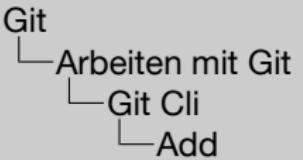
```
git add NewFile.php
```

```
git add --all
```

```
git add -i #Interactive
```



2018-04-10



```
git status
HEAD detached at origin/master
Changes to be committed:
  (use "git reset HEAD <file>..." to unstage)

    new file:   .idea/inspectionProfiles/profiles_settings.xml

Changes not staged for commit:
  (use "git add/rm <file>..." to update what will be committed)
  (use "git checkout -- <file>..." to discard changes in working directory)

    deleted:   .idea/inspectionProfiles/profiles_settings.xml
    modified:  build/config/phpcs/ZooroyalDefault/ruleset.xml

git add .idea/inspectionProfiles/profiles_settings.xml

git status
HEAD detached at origin/master
Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git checkout -- <file>..." to discard changes in working directory)

    modified:  build/config/phpcs/ZooroyalDefault/ruleset.xml

no changes added to commit (use "git add" and/or "git commit -a")

git add -i
      staged      unstaged path
  1: unchanged       +@-/build/config/phpcs/ZooroyalDefault/ruleset.xml

*** Commands ***
  1: status     2: update    3: revert    4: add untracked
  5: patch     6: diff       7: quit      8: help
What now? 
```



Remove/Move

Dateien entfernen und verschieben



2018-04-10

Git

Arbeiten mit Git
└─ Git Cli
 └─ Remove/Move

git rm / git mv

► [Git Dokumentation](#)

► Git-Historie pflegen

```
git rm NewFile.php  
git mv NFile.php NewFile.php
```

```
touch test.txt  
git add test.txt  
git mv test.txt bla.txt  
git rm bla.txt -f  
rm 'bla.txt'
```



Checkout

Branches und Dateien auschecken

git checkout

[► Git Dokumentation](#)

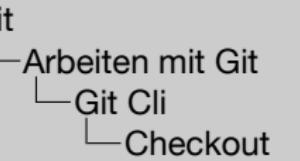
Stellt Git-Objekte im Workspace bereit

```
# checkout branch  
git checkout master  
# create and checkout branch  
git checkout -b new_Branch  
# checkout remote branch at date  
git checkout origin/master@{  
    yesterday}  
# checkout single file  
git checkout <commit-ish> -- <  
    file_path>  
# checkout last used branch  
git checkout - # alt: @{-1}
```



```
apple ~ % cd /git_repositories/zooroyalweb  
apple ~/git_repositories/zooroyalweb % git checkout @{yesterday}  
M build/config/phpcs/ZooroyalDefault/ruleset.xml  
Note: checking out '@{yesterday}'.  
  
You are in 'detached HEAD' state. You can look around, make experimental  
changes and commit them, and you can discard any commits you make in this  
state without impacting any branches by performing another checkout.  
  
If you want to create a new branch to retain commits you create, you may  
do so (now or later) by using -b with the checkout command again. Example:  
  
git checkout -b <new-branch-name>  
  
HEAD is now at fa9e2faad zwischenstand mit Mockery  
  
apple ~ % cd /git_repositories/zooroyalweb  
apple ~/git_repositories/zooroyalweb % git checkout UnitTestPOC  
M build/config/phpcs/ZooroyalDefault/ruleset.xml  
Previous HEAD position was fa9e2faad zwischenstand mit Mockery  
Switched to branch 'UnitTestPOC'  
Your branch is ahead of 'origin/UnitTestPOC' by 2 commits.  
(use "git push" to publish your local commits)  
  
apple ~ % cd /git_repositories/zooroyalweb  
apple ~/git_repositories/zooroyalweb % git checkout origin/UnitTestPOC  
M build/config/phpcs/ZooroyalDefault/ruleset.xml  
Note: checking out 'origin/UnitTestPOC'.  
  
You are in 'detached HEAD' state. You can look around, make experimental  
changes and commit them, and you can discard any commits you make in this  
state without impacting any branches by performing another checkout.  
  
If you want to create a new branch to retain commits you create, you may  
do so (now or later) by using -b with the checkout command again. Example:  
  
git checkout -b <new-branch-name>  
  
HEAD is now at 960cdaf8 POC zum UnitTest für Plugins  
apple ~ % cd /git_repositories/zooroyalweb  
apple ~/git_repositories/zooroyalweb % git checkout - # alt: @{-1}
```

2018-04-10



git checkout

Stellt Git-Objekte im Workspace bereit

```
# checkout branch  
git checkout master  
# create and checkout branch  
git checkout -b <new-branch-name>  
# checkout remote branch at date  
git checkout origin/master@{  
    yesterday}  
# checkout single file  
git checkout <commit-ish> -- <  
    file_path>  
# checkout last used branch  
git checkout - # alt: @{-1}
```

Log

Historie des aktuellen Workspaces

git log

► Git Dokumentation

- Zeigt die Historie
- Verschiedene Ansichten
- Durchsuchen

```
git log
# detailed view
git log -p
# short view with graph
git log --graph --online
# grep message for pattern
git log --grep=Bugfix
# show commits since date
git log --since=yesterday
```



Git

Arbeiten mit Git
└ Git Cli
└ Log

2018-04-10

```
* 1b10bb4c75 (HEAD, tag: 1.6.6, origin/master, origin/HEAD) Merge pull request #1748 from ZooRoyal/DEV-2046
| + 2abf15bb64 (origin/DEV-2048) remove button from french rip page and add some changes
| + 6b42836d1 (tag: 1.6.5) Merge pull request #1746 from ZooRoyal/DEV-2045
| + | 8b9aa00e2 (origin/DEV-2048) DEV-2045 Remove FR from findologic check
| + | aca1006a8 Merge pull request #1747 from ZooRoyal/DEV-2046
| + |
| + | 752b8e7cc (origin/DEV-2046) DEV-2046 sitemap export zooroyal.fr removed
| + | 8253821a2 (tag: 1.6.4) Merge pull request #1739 from ZooRoyal/DEV-2016-2
| + | d8114fec9 DEV-2016-2 Add FR host; add 410 gone header
| + | 254aa0f14 DEV-2016-2 Add FR host; add 410 gone header
| + | 527d66534 (origin/DEV-2016-2) DEV-2016-2 Add rip page text; add shopware routing
| + | 6c63cccfb DEV-2016-2 Add rip page text; add shopware routing
| + | 829714ab6 DEV-2016 Wippage und Shellspace zum generieren von data-uri
| + | ea8a4def7 DEV-2016 Analog zu ZHreFlag die aktiven Shops in einer Klassenvariable definiert
| + | 6c8e58577 DEV-2016 Französische Texte entfernt
| + | ea434d5f2 DEV-2016 Modal Anzeige für FR Kunden auf DE und AT verhindert
| + | b275dd9e6 DEV-2016 Meta-Link hrefLang für FR unterbunden
| + |
| + | 55b6734b6 Merge pull request #1735 from ZooRoyal/DEV-2027
| + | 5df0947fe (origin/DEV-2027) DEV-2027 find_php_to_check respects excludes in diff build
| + | 4da03dc09 (tag: 1.6.3) Merge pull request #1732 from ZooRoyal/DEV-2017
| + | 183d602ea (origin/DEV-2017) DEV-2017 type mismatch fixed; probably, codestyle
| + | 9720b3961 DEV-2017 only apply voucher total value check to percental vouchers, some code cleanup
| + | 82057982a DEV-2017 basket total patch ported from WFS
| + | 23d219d6f DEV-2017 order export to pixi DE Linetem fix from RL2016/DEV-1591
| + | 0a3bea041 DEV-2017 voucher plugin tweaks for souque-voucher from weinfreunde (WFS-598)
```

git log

- Zeigt die Historie
- Verschiedene Ansichten
- Durchsuchen

git log
detailed view
git log
short view with graph
git log --graph --online
grep message for pattern
git log --grep=Bugfix
show commits since date
git log --since=yesterday



Diff

Unterschiede zwischen Commit-Ishts



git diff

► Git Dokumentation

- ▶ Zeigt Unterschiede zwischen Commit-IsEs
 - ▶ Durchsuchen

git diff

```
# only names of unmerged  
git diff --name-only --diff-filter  
    =U  
# changed chars only  
git diff --word-diff  
# ignore whitespace  
git diff HEAD~4..HEAD~2 -b
```

```
git diff HEAD -- autoload.php | cat
-- git -g a/autoload.php b/autoload.php
index f7af6f58e..2f783b3ea 100755
-- a/autoload.php
++ b/autoload.php
@ -28,8 +28,9 @@ use Composer\Autoload\ClassLoader;
/**+
 * @var ClassLoader $loader
 */
$loader = require __DIR__ . '/vendor/autoload.php';
$loader = require __DIR__ . '/vendor/autoload.php';

AnnotationRegistry::registerLoader(array($loader, 'loadClass'));

return $loader;
```

Tools:

```
git diff HEAD -b -- autoload.php | cat
-- git -g a/autoload.php b/autoload.php
index f7af6f58e..2f783b3ea 100755
-- a/autoload.php
++ b/autoload.php
@ -33,3 +33,4 @@ $loader = require __DIR__ . '/vendor/autoload.php';
AnnotationRegistry::registerLoader(array($loader, 'loadClass'));

return $loader;
```

Everyday Git in Twenty commands

```
git diff HEAD --name-only | cat
-- build/config/phpcs/ZooroyalDefault/ruleset.xml
```

All these commands are test

Remote sensitive data from

Sync with remote, over-write

All the changes from

Git

- Arbeiten mit Git
 - Git Cli
 - Diff

2018-04-10



Clean

Workspace aufräumen



git clean

► Git Dokumentation

- ▶ Entfernt nicht indizierte Dateien
- ▶ .gitignore wird beachtet

```
git clean  
# deletes directories too  
git clean -df
```

```
apple ~ % cd /git_repositories/zooroyalweb  
apple ~ % git status  
# On branch UnitTestPOC  
# Your branch is ahead of 'origin/UnitTestPOC' by 1 commit.  
#   (use "git push" to publish your local commits)  
#   M build/config/phpcs/ZooroyalDefault/ruleset.xml  
#   ?? test.php  
#       Stash changes before run  
apple ~ % touch test.php  
apple ~ % git status  
# On branch UnitTestPOC  
# Your branch is ahead of 'origin/UnitTestPOC' by 1 commit.  
#   (use "git push" to publish your local commits)  
#   M build/config/phpcs/ZooroyalDefault/ruleset.xml  
#   ?? test.php  
#       Stash changes before run  
apple ~ % git clean -df  
Removing test.php  
apple ~ % git status  
# On branch UnitTestPOC  
# Your branch is ahead of 'origin/UnitTestPOC' by 1 commit.  
#   (use "git push" to publish your local commits)  
#   M build/config/phpcs/ZooroyalDefault/ruleset.xml  
apple ~ %
```

2018-04-10

Git
└ Arbeiten mit Git
 └ Git Cli
 └ Clean



Fetch

Informationen über Remotes einholen



45

git fetch

[Git Dokumentation](#)

Bezieht Refs von
Remote-Reposioties

```
git fetch
# fetches a specific ref for
branch
git fetch origin master:refs/
remotes/origin/mymaster
```

```
git fetch
remote: Counting objects: 673, done.
remote: Compressing objects: 100% (211/211), done.
remote: Total 673 (delta 426), reused 573 (delta 383), pack-reused 33
Receiving objects: 100% (673/673), 235.07 KB | 1.44 MB/s, done.
Resolving deltas: 100% (429/429), completed with 242 local objects.
From github.com:ZooRoyal/zooroyalweb
 * [new branch]      A80-2          -> origin/A80-2
 * [new branch]      A80-34         -> origin/A80-34
 69778c95a...6bf6af530 DEV-1342       -> origin/DEV-1342
 df295eaac...eb4fb8b309 DEV-1492       -> origin/DEV-1492
 * [new branch]      DEV-1492-neu     -> origin/DEV-1492-neu
 7b9b7305d..413ae9857 DEV-1575       -> origin/DEV-1575
 3f034fbf6..44a12c9d6 DEV-1575-neu   -> origin/DEV-1575-neu
 7beeed02e..91dcf412a DEV-1697       -> origin/DEV-1697
 * [new branch]      DEV-1966       -> origin/DEV-1966
 257853203..be76f3c48 DEV-1978       -> origin/DEV-1978
 + 7c25c8e1b...2ca6279be DEV-2008     -> origin/DEV-2008 (forced update)
 ab2fe1c8c..beceb800f DEV-2037       -> origin/DEV-2037
 * [new branch]      DEV-2045       -> origin/DEV-2045
 * [new branch]      DEV-2046       -> origin/DEV-2046
 * [new branch]      DEV-2048       -> origin/DEV-2048
 3de648a6d..f7539c595 RL2017        -> origin/RL2017
 0857ead2..ca4ff99302 SSR-14        -> origin/SSR-14
 * [new branch]      ZOO-8         -> origin/ZOO-8
 8253821a2..1b16b4c75 master        -> origin/master
 * [new branch]      master-vagrantfix -> origin/master-vagrantfix
 * [new tag]          1.6.4          -> 1.6.4
 * [new tag]          1.6.6          -> 1.6.6
 * [new tag]          1.6.5          -> 1.6.5
```

2018-04-10

Git
└ Arbeiten mit Git
 └ Git Cli
 └ Fetch

git fetch
Bezieht Refs von
Remote-Reposioties

git fetch
Fetches a specific ref for
branch
git fetch origin master:refs/
remotes/origin/mymaster



Merge

Zwei Branches zusammen führen

git merge

► Git Dokumentation

- ▶ Führt zwei Branches zusammen
- ▶ Merge oder Fast-Forward
- ▶ Interaktiv

```
# Pause before commit
git merge --no-commit Quellbranch
# Abort ongoing merge
git merge --abort
# Continue paused merge
git merge --continue
# Force non Fast-Forward
git merge -no-ff Quellbranch
```



46

Git

Arbeiten mit Git
└─ Git Cli
 └─ Merge

2018-04-10

```
apple:~/git_repositories/zooroyalweb P UnitTestPOC ①> git merge --no-commit origin/master
Automatic merge went well; stopped before committing as requested
apple:~/git_repositories/zooroyalweb P UnitTestPOC ①> git status -sb
# UnitTestPOC...origin/UnitTestPOC [ahead 1]
* assets/src/js/modal-i18n.js
  build/bin/copy_paste_detect.php
  build/bin/find_php_to_check.sh
  engine/Shopware/Plugins/Local/Frontend/ZRHealthCheck/Components/HealthCheck/Check/F
  engine/Shopware/Plugins/Local/Frontend/ZRHreflang/Bootstrap.php
  engine/Shopware/Plugins/Local/Frontend/ZRJsonREST/Controllers/Frontend/ZRJsonREST.p
  engine/Shopware/Plugins/Local/Frontend/ZRSitemap/Bootstrap.php
  engine/Shopware/Plugins/Local/Frontend/ZRVoucherPromotion/Bootstrap.php
  engine/Shopware/Plugins/Local/Frontend/ZRVoucherPromotion/Models/ZRBasket.php
  engine/Shopware/Plugins/Local/Frontend/ZRVoucherPromotion/Views/backend/zr_voucher_
  window.js
  landing/r1p_page/r1p_page-fr.html
  pixishopconnection/system/application/views/export_pix_order.php
  resources/maintenance/wait-page-fr.html
  resources/scripts/CVS_checkout_all.sh
  resources/scripts/data_uri_generator.sh
  shopware.php
apple:~/git_repositories/zooroyalweb P UnitTestPOC ①> git commit
[UnitestPOC 2a680f3fe] Merge remote-tracking branch 'origin/master' into UnitestPOC
apple:~/git_repositories/zooroyalweb P UnitTestPOC ③1>
```

git merge

- ▶ Führt zwei Branches zusammen
- ▶ Merge oder Fast-Forward
- ▶ Interaktiv

```
# Pause before commit
git merge --no-commit Quellbranch
# Abort ongoing merge
git merge --abort
# Continue paused merge
git merge --continue
# Force non Fast-Forward
git merge -no-ff Quellbranch
```



Pull

Daten aus dem anderem Branch integrieren

git pull

► Git Dokumentation

- ▶ Integriert Daten aus anderem Branch
- ▶ Fetch und merge

```
# fetch + merge FETCH_HEAD
git pull
# pull ref from remote
git pull origin master
# fetch + rebase instead
git pull --rebase master
# Config always rebase
git config --global pull.rebase
    true
```



2018-04-10

Git
└ Arbeiten mit Git
 └ Git Cli
 └ Pull

```
git status -sb
## master...origin/master [behind 2]
?? changes.txt

git add .
git commit -m "CHAAAAAAANGES"
[master f87be494b] CHAAAAAAANGES
1 file changed, 0 insertions(+), 0 deletions(-)
create mode 100644 changes.txt

git pull --rebase origin master
From github.com:ZooRoyal/zooroyalweb
 * branch            master      -> FETCH_HEAD
First, rewinding head to replay your work on top of it...
Applying: CHAAAAAAANGES

git log -3 --oneline --graph --source --color | cat
* 7efleeeb HEAD CHAAAAAAANGES
* 1b16b4c75 HEAD Merge pull request #1748 from ZooRoyal/DEV-2048
|\
| * 2abf15bb4 HEAD remove button from french rip page

git log -3 --oneline --graph --source --color | cat
```

git pull

- ▶ Interpret Daten aus anderem Branch
- ▶ Fetch and merge

```
git fetch + merge FETCH_HEAD
git pull ref from remote
git pull --rebase master
git pull --rebase --track
git pull --rebase master
git config always-rebase
git config --global pull.rebase
true
```

Push

Daten in anderen Branch übertragen



2018-04-10

git push

[Git Dokumentation](#)

Remote-Repository-Daten
schreiben oder löschen

```
# force push to origin/branch
git push -f origin branch
# delete remote branch
git push origin :branch
# protected force push
git push --force-with-lease
```

```
git log --color -3 --oneline --graph origin/UnitTestPOC | cat
* 960cdaf0 POC zum UnitTest für Plugins
* 95c462aed Merge pull request #1731 from ZooRoyal/DEV-1971-Anpassungen
| \
| * 9f05e4140 Anpassungen an idea-settings und phpcs ruleset
|
Forced push but still ens
git log --color -3 --oneline --graph UnitTestPOC | cat
* fa9e2faad zwischenstand mit Mockery
* 960cdaf0 POC zum UnitTest für Plugins
* 95c462aed Merge pull request #1731 from ZooRoyal/DEV-1971-Anpassungen
| \
Show how many lines do
git status -sb
## UnitTestPOC...origin/UnitTestPOC [ahead 1]
git push
Counting objects: 14, done.
Delta compression using up to 8 threads.
Compressing objects: 100% (11/11), done.
Writing objects: 100% (14/14), 2.41 KiB | 823.00 KiB/s, done.
Total 14 (delta 6), reused 0 (delta 0)
remote: Resolving deltas: 100% (6/6), completed with 6 local objects.
To github.com:ZooRoyal/zooroyalweb.git
  960cdaf0..fa9e2faad  UnitTestPOC -> UnitTestPOC
an ent
```

Git

Arbeiten mit Git
└─ Git Cli
 └─ Push

git push Remote-Repository-Daten schreiben oder löschen

Force push to origin/branch
git push -f origin branch
delete remote branch
git push origin :branch
protected force push
git push --force-with-lease



Revert

Änderungen mit sauberer History rückgängig



git revert [commit-ish](#)
Erzeugt einen neuen Commit der
Änderungen eines anderen
Commits rückgängig macht

undo <commit-ish>
git revert <commit-ish>
undo merge
git revert -m 1 <commit-ish>

git revert

[Git Dokumentation](#)

Erzeugt einen neuen Commit der
Änderungen eines anderen
Commits rückgängig macht

```
# undo <commit-ish>
git revert <commit-ish>
# undo merge
git revert -m 1 <commit-ish>
```

```
git status -sb
## UnitTestPOC...origin/UnitTestPOC 013 0 lines, 0 files changed, 0 insertions(+), 0 deletions(-)

touch test.txt
git add test.txt
git shortlog
git commit -m "broken commit"
[UnitTestPOC e0a2e4039] broken commit
1 file changed, 0 insertions(+), 0 deletions(-)
create mode 100644 test.txt

git log -1 | cat
commit e0a2e4039d9298840ab35738a2e529d21c5bea98
Author: Sebastian Knott <s.knott@zooroyal.de>
Date:   Fri Jan 26 13:33:36 2018 +0100

    broken commit
git log --author='Your Name' --since='2 days ago' --oneline --color | cat
git revert e0a2e4039d9298840ab35738a2e529d21c5bea98
[UnitTestPOC 1d46c6c12] Revert "broken commit"
1 file changed, 0 insertions(+), 0 deletions(-)
delete mode 100644 test.txt

git log -2 --oneline --color | cat
1d46c6c12 Revert "broken commit"
e0a2e4039 broken commit
```

Branch

Branches verwalten



git branch

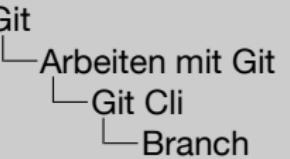
► Git Dokumentation

- Durchsuchen
- Hinzufügen
- Löschen

```
# list merged branchesNN
git branch --merged master
# delete local MyBranch
git branch -d MyBranch
# list info of local branches
git branch -vv
```

```
git branch -vv
git branch -vv --color | cat
git branch -D master
git branch --merged master
```

2018-04-10

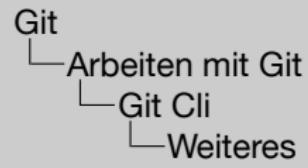


```
# list merged branchesNN
git branch --merged master
# delete local MyBranch
git branch -d MyBranch
# list info of local branches
git branch -vv
```



- ▶ Einsteigerkurs von GitHub
- ▶ Roger Dudlers Git Guide
- ▶ Git Tricks

2018-04-10



▶ Einsteigerkurs von GitHub
▶ Roger Dudlers Git Guide
▶ Git Tricks

 *Vielen
Dank!*

Danke für eure Aufmerksamkeit!

2018-04-10

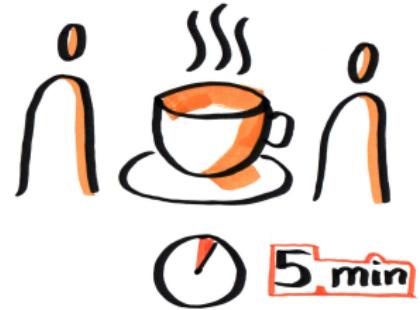
Git
└ Arbeiten mit Git
 └ Git Cli



Danke für eure Aufmerksamkeit!

Pause

5 Minuten Pause



Git

2018-04-10

└ Pause



Pause

10 Minuten Pause



Git

2018-04-10

└ Pause



Pause
Pause



Git

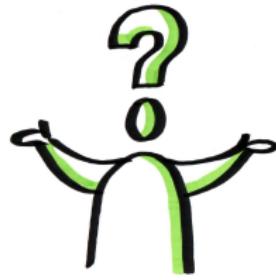
2018-04-10

└ Pause



Fragen

Fragen?



Fragen?

Git

2018-04-10

└ Fragen

