

## Tutorial: Git aufsetzen (Teil 2: git clone)

Torsten Groß  
2. Januar 2015

In einer Reihe von Tutorials bieten wir einen Überblick über die gebräuchlichsten Git-Befehle. Zunächst soll Git aufgesetzt werden, um mit einem neuen Projekt mit Versionskontrolle starten zu können. Wie das funktioniert, zeigt diese Artikelserie. Nach der Initialisierung (siehe Teil 1) wollen Entwickler Repositories nun mit `git clone` kopieren.

### git clone

Der Befehl `git clone` kopiert ein bestehendes Git-Repository. Das entspricht in etwa dem Kommando `svn checkout`, nur dass die "Arbeitskopie" ein eigenständiges Git-Repo ist – mit eigener Historie, eigenen Dateien und einer komplett vom Original-Repo isolierten Umgebung.

Komfortablerweise erzeugt das Klonen direkt eine Remote-Verbindung namens `origin`, die zurück auf das Original-Repository verweist. Dadurch ist die Interaktion mit dem zentralen Repo sehr einfach.

### Nutzung

```
git clone <repo>
```

Klont das Repository, das sich unter `<repo>` befindet, auf die lokale Maschine. Das Original-Repo kann sich im lokalen Dateisystem oder auf einer Remote-Maschine befinden, die via HTTP oder SSH zugänglich ist.

```
git clone <repo> <verzeichnis>
```

Klont das Repository, das sich unter `<repo>` befindet, in den Ordner namens `<verzeichnis>` auf der lokalen Maschine.

### Hinweise

Wenn ein Projekt bereits in einem zentralen Repository aufgesetzt ist, ist `git clone` der gebräuchlichste Weg, eine Entwicklungskopie zu ziehen. Wie `git init` ist das Klonen zumeist eine einmalige Operation: Hat ein Entwickler erst einmal eine Arbeitskopie, kann er alle Versionskontrolloperationen und die Zusammenarbeit über sein lokales Repo managen.

### Kollaboration zwischen Repos

Die Git-Idee einer "Arbeitskopie" unterscheidet sich sehr stark von der Arbeitskopie, die man erhält, wenn man Code aus einem SVN-Repo auscheckt. Anders als SVN macht Git keinen Unterschied zwischen Arbeitskopien und dem zentralen Repository – es sind vollwertige Git-Repos.

Dadurch unterscheidet sich auch die Zusammenarbeit fundamental. Während es bei SVN auf die Beziehung zwischen zentralem Repo und Arbeitskopie ankommt, basiert das Kollaborationsmodell von Git auf der Interaktion von Repository zu Repository. Statt Arbeitskopien in eine zentrale SVN-Repo einzuchecken, werden Commits von einer Repository zur anderen gepusht und gepullt.

Dabei gibt es jedoch keine Beschränkungen, bestimmten Git-Repos eine bestimmte Bedeutung zu geben. Es ist beispielsweise möglich, ein Repository zum "zentralen" Repo zu bestimmen und so den zentralisierten Workflow mit Git zu replizieren.

### Beispiel

Das Beispiel unten zeigt, wie ein Entwickler eine lokale Kopie eines zentralen Repositories von einem Server unter `beispiel.com` mit dem SSH-Nutzernamen Max bezieht:

```
git clone ssh://max@beispiel.com/path/to/mein-projekt.git
cd mein-projekt
```

Der erste Befehl initialisiert ein neues Git-Repository im Ordner `mein-projekt`. Dann kann der Entwickler mit `cd` ins Projekt gehen und anfangen, Dateien zu editieren, Kleinigkeiten zu committen und mit anderen Repositories zu interagieren.

Im [dritten Teil des Tutorials](#) besprechen wir dann die grundlegende Konfiguration.

### Weiterführende Infos: Ihr Partner für Git und Stash

Kennen Sie [Stash von Atlassian](#)? Stash bietet eine zentrale Lösung zum Management des gesamten distribuierten Codes: Hier kommen alle Git-Repositories im Unternehmen zusammen, hier finden Entwickler immer die letzte offizielle Version eines Projekts, hier können Projektverantwortliche Berechtigungen kontrollieren, um sicherzustellen, dass die richtigen Nutzer Zugriff auf den richtigen Code haben.

Möchten Sie mehr erfahren? Wir sind offizieller Vertriebspartner von Atlassian und einer der größten Atlassian Experts Partner weltweit. Gerne unterstützen wir Sie bei der Evaluierung, Lizenzierung und Adaption von Stash. Und wenn Sie [Atlassian-Lizenzen bei //SEIBERT/MEDIA kaufen](#), gewähren wir Ihnen einen Rabatt in Höhe von 10% der Lizenzkosten in Form von Beratungsleistungen. [Bitte sprechen Sie uns einfach an.](#)

[99 Argumente für Stash als Git-Repository-Manager](#)

[Branch-basierte Git-Workflows mit Stash adaptieren](#)

[Echte Integration: Das Zusammenspiel von JIRA, Stash und Bamboo](#)

[Interview: Die Vorteile von Git in der Software-Entwicklung und die Möglichkeiten von Stash](#)

[So funktioniert die Lizenzierung von Atlassian-Produkten](#)

Schlagwörter/Tags:

DVCS Git Software-Entwicklung Stash SVN Tutorial



0 Kommentare //S-Weblog DE

Anmelden

Empfehlen Tweet Teilen

Nach Ältesten sortieren



Die Diskussion starten...

ANMELDEN MIT

ODER MIT DISQUS EINLOGGEN



Name

Schreiben Sie den ersten Kommentar.

### Die wichtigsten Themen im Blog

//SEIBERT/MEDIA Firmenwikis Confluence JIRA Atlassian  
Software-Entwicklung Git Agile Intranet Google Cloud  
Creative Commons

Weitere Tags

### Abonnieren Sie unseren Newsletter

E-Mail\*

SENDEN

### Weblog für die Ohren



Unser Gesprächsformat rund um moderne Zusammenarbeit und Unternehmens-Software

Zur Übersicht

### Jobs bei //SEIBERT/MEDIA

Hier findest du eine Übersicht unserer aktuellen Job-Angebote

RSS-Feed abonnieren

Archiv

Unsere Autoren

Wir zwitschern auf Twitter

Unsere Videos

Unser Facebook-Freundeskreis

## Fragen? Nehmen Sie jetzt Kontakt mit uns auf!

Vorname	Nachname	Nachricht
<input type="text"/>	<input type="text"/>	
Firma	<input type="text"/>	
E-Mail *	Telefon	
<input type="text"/>	<input type="text"/>	<input type="text"/>


SENDEN


**WEBSITE** Wir stellen uns vor.

**INFOTHEK** Wir teilen unser Wissen.


**KONTAKT** Wir sind gespannt.


**IMPRESSUM**


 **RSS-Feed abonnieren**


 **Newsletter abonnieren**

 **Archiv**

 **Unsere Autoren**

 **Wir zwitschern auf Twitter**

 **Unsere Videos**

 **Unser Facebook-Freundeskreis**