

git

Jan Popko

Python Advanced

Installation von Git

Download von: <https://git-scm.com/download/>

Installation von Git Bash

- Pfad wählen (z.B. [C:\Git](#))
- ob Desktop Symbol gewünscht ist
- Editor wählen (z.B. Atom, Notepad++)
- wählen ob Git auch von der Windows Kommandozeile ausgeführt werden kann
- OpenSSL library wählen
- Checkout as Windows-style
- MinTTY wählen
- Enable file system caching + Enable Git Credential Manager

Jan Popko

Python Advanced

Einstellen von GitHub

Gehe zu: github.com/

Username, Email, Passwort wählen

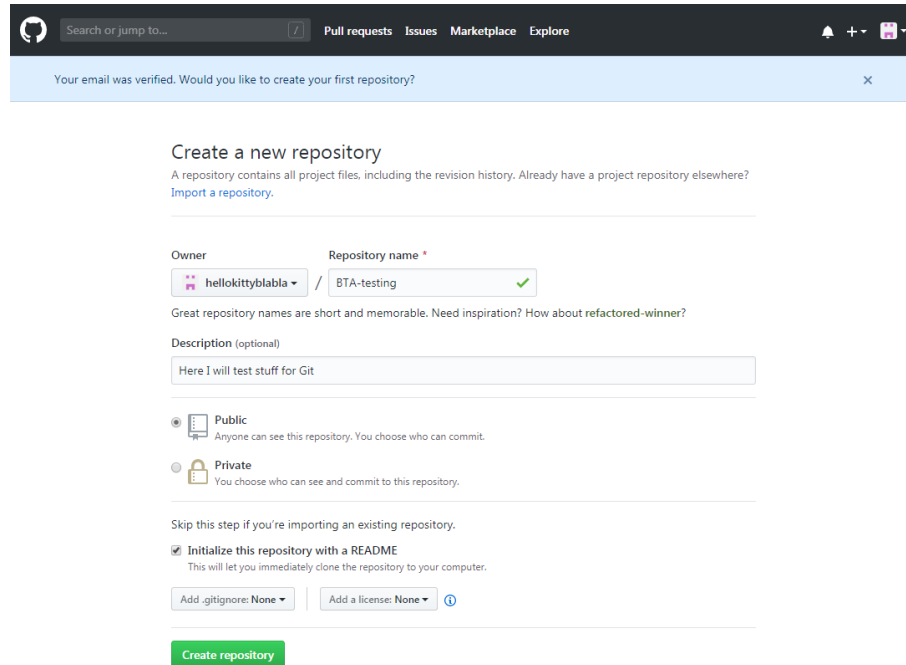
Free Plan wählen

Repository Namen Wählen

Repository erstellen

Jan Popko

Python Advanced



Search or jump to... Pull requests Issues Marketplace Explore

Your email was verified. Would you like to create your first repository?

Create a new repository

A repository contains all project files, including the revision history. Already have a project repository elsewhere? [Import a repository.](#)

Owner: helloworldblabla / Repository name: BTA-testing

Great repository names are short and memorable. Need inspiration? How about [refactored-winner](#)?

Description (optional)
Here I will test stuff for Git

☒ Public
Anyone can see this repository. You choose who can commit.

☐ Private
You choose who can see and commit to this repository.

Skip this step if you're importing an existing repository.

☒ Initialize this repository with a README
This will let you immediately clone the repository to your computer.

Add .gitignore: None Add a license: None

Create repository

Git Bash

TAB
STRG + Einfg
SHIFT + Einfg

- Autovervollständigung wenn möglich
- Kopieren
- Einfügen

Bewegen in den Verzeichnissen

- cd
- cd <Folder>
- cd ..
- cd c

- C:\Users\<User>
- Geht zum Verzeichnis <Folder>
- Gehe in den übergeordneten Ordner
- Geht zu C:\

Ansehen und Bearbeiten von Dateien und Ordnern

- ls
- mkdir <Folder>
- touch <File>

- zeigt alle Ordner und Dateien im derzeitigen Ordner an
- erstellt Ordner
- erstellt Datei, wenn nicht vorhanden

Jan Popko

Python Advanced

Git Repository erstellen

Konfiguration des Usernamens:

```
git config --global user.name '<github username>'
```

Konfiguration der Emailadresse:

```
git config --global user.email <email>
```

Einfügen des Repositories von GitHub

```
git clone <github url - siehe nächste Seite>
```

GitHub Link

Link von GitHub für Repository:

hellokittyblabla / BTA-testing

Unwatch 1 Star 0 Fork 0

Code Issues 0 Pull requests 0 Actions Projects 0 Wiki Security Insights Settings

Here I will test stuff for Git [Edit](#)

[Manage topics](#)

1 commit 1 branch 0 packages 0 releases 1 contributor

Branch: master New pull request Create new file Upload files Find file Clone or download

hellokittyblabla Initial commit

README.md Initial commit

README.md

BTA-testing

Here I will test stuff for Git

Clone with HTTPS Use SSH

Use Git or checkout with SVN using the web URL.

<https://github.com/hellokittyblabla/BTA->

Open in Desktop Download ZIP

Jan Popko

Python Advanced

Upload zu GitHub

Atom:

File → Open Folder → 'Git Repository Ordner'

Erstelle 'test.txt'

GitBash:

Gibt an welchen Status die Dateien im Rep haben:

```
git status
```

test.txt wird zum Rep hinzugefügt:

```
git add test.txt
```

(git add *.txt) alle txt Dateien werden dem Rep hinzugefügt

(git add .) alle Dateien werden dem Rep hinzugefügt

Muß aber noch bestätigt werden:

```
git commit -m 'comment to commit' test.txt
```

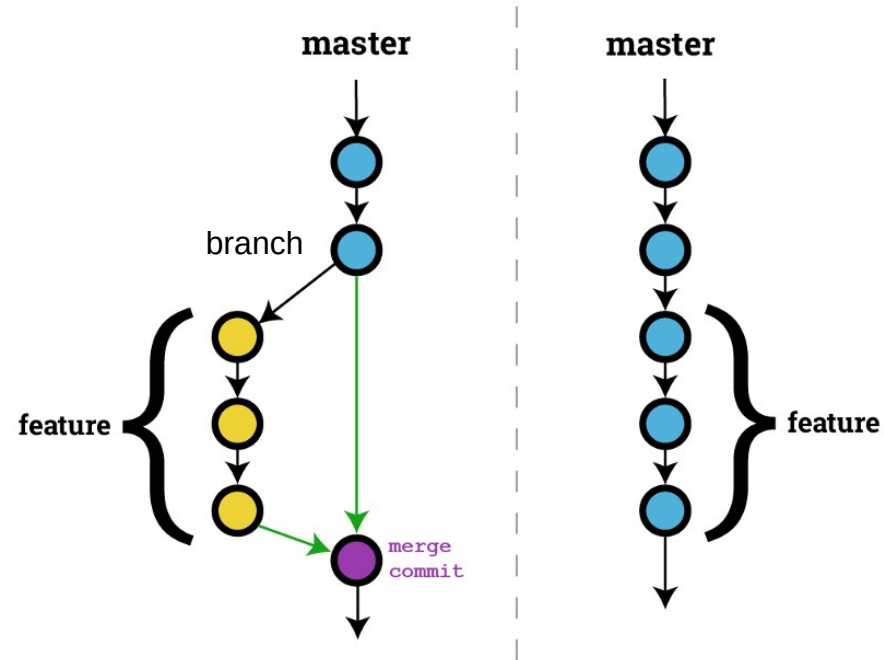
Upload zu GitHub:

```
git push -u origin master
```

Jan Popko

Python Advanced

Git Workflow



Jan Popko

Python Advanced

Setup Branch

GitBash:

Neuen Branch erstellen:
`git branch <branchname>`

Alle Branches anzeigen:
`git show-branch`

Zum neuen Branch wechseln:
`git checkout <branchname>`

Publish subbranch in GitHub:
`git push --set-upstream origin <branchname>`

Branch löschen (Lokal)
`git branch -d <localBranchName>`

Branch löschen remote (auf GitHub)
`git push origin --delete <remoteBranchName>`

Jan Popko

Python Advanced

Merge Branch + Nützliches

GitBash:

Branch mit dem aktuellen zusammenfügen:
`git merge <branchname>`

Alle commits anzeigen:
`git reflog`

Die commit ID wird zurückgenommen (kann mit reflog gefunden werden)
`git revert -m 1 <commitID>`

Datei:
Alle hier angegebenen Ordner und Dateien werden ignoriert.
`.gitignore`