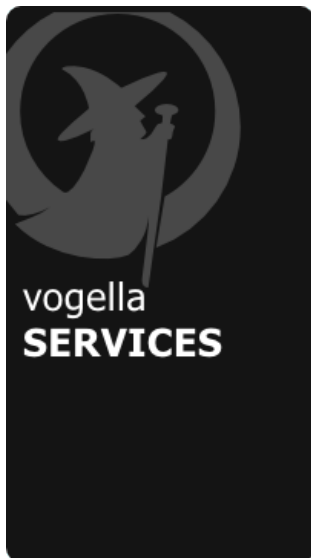




- 
- [Tutorials](#)
  - [Android Programming](#)
  - [Eclipse IDE](#)
  - [Eclipse RCP/Plug-ins](#)
  - [Version Control](#)
  - [Web](#)
  - [Java](#)
  - [Technology](#)
  - [Google](#)
  - [Algorithms](#)
  - [Design Pattern](#)
- [Services](#)
  - [Android Development Training](#)
    - [Android Professional](#)
    - [Android Testing and Appl. Optimization](#)
  - [Eclipse Development Training](#)
    - [Eclipse RCP](#)
    - [Eclipse IDE Expert](#)
    - [Eclipse RCP Migration](#)
    - [Contributing to Eclipse](#)
    - [Eclipse IDE Plug-in Development](#)
    - [Maven/Tycho](#)
    - [Eclipse RCP with Java EE and Spring](#)
    - [oXygen in Eclipse](#)
    - [NatTable](#)
  - [Git Training](#)
  - [Groovy Training](#)
  - [Gradle Training](#)
  - [Java Training](#)
  - [Spring Workshop](#)
  - [Expert Consulting](#)
- [Products](#)
  - [saneclipse](#)
  - [CodeModify](#)
  - [PreferenceSpy](#)
- [Books](#)
  - [Contributing to the Eclipse Project](#)
  - [Eclipse RCP](#)
  - [Eclipse IDE](#)
  - [Git](#)
  - [Android SQLite and ContentProvider](#)
- [Company](#)
  - [About us](#)
  - [People at vogella](#)
  - [Reference Customers](#)
  - [Blog](#)
  - [Jobs](#)
  - [FAQ](#)
  - [Legal](#)
- [Donate](#)
- [Contact us](#)







[Training Books](#)

**FOLLOW  
ME ON**



**FOLLOW  
ME ON**



**GET TRAINING**



**PURCHASE BOOKS**



**SUPPORT FREE TUTORIAL**

# Git hosting on GitHub, Bitbucket or on your own server - Tutorial

**Lars Vogel**

Version 5.5

Copyright © 2009, 2010, 2011, 2012, 2013, 2014 vogella GmbH

14.12.2014

## Hosting your Git repositories

This tutorial explains how to use GitHub, Bitbucket as hosting provider for your Git repository and how you can install a Git server on your own machine.

---

## Table of Contents

[1. Git Hosting Provider](#)

[2. Authentication via SSH](#)

[2.1. The concept of SSH](#)

[2.2. SSH key pair generation](#)

[3. GitHub](#)

[3.1. What is GitHub?](#)

[3.2. Create repository in GitHub](#)

[3.3. Merging pull request at GitHub](#)

[4. Bitbucket](#)

[4.1. What is Bitbucket?](#)

[4.2. Creating a repository](#)

[5. Own Git server](#)

[5.1. Hosting your own Git server](#)

[5.2. Give write access to a Git repository](#)

[5.3. Security setup for the git user](#)

[6. Git series](#)

[7. Get the Book](#)

[8. About this website](#)

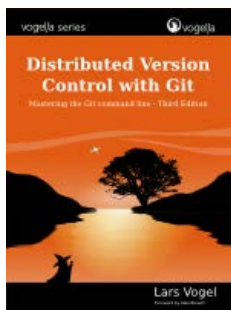
[8.1. Donate to support free tutorials](#)

[8.2. Questions and discussion](#)

[8.3. License for this tutorial and its code](#)

[9. Links and Literature](#)

[9.1. vogella Resources](#)



[Get the book](#)

# 1. Git Hosting Provider

Git allows you to host your own Git server. Instead of setting up your own server, you can also use a hosting service. The most popular Git hosting sites are GitHub and Bitbucket. Both offer free hosting with certain limitations.

## 2. Authentication via SSH

### 2.1. The concept of SSH

Most Git (and Gerrit) servers support SSH based authentication. This requires a *SSH key pair* for automatic authentication.

An SSH key pair consists of a public and private key. The public key is uploaded to the application you want to authenticate with. The application has no access to the private key. If you interact with the hosting provider via the ssh protocol, the public key is used to identify a user who encrypted the data during communication with the corresponding private key.

### 2.2. SSH key pair generation

To create an SSH key under Linux (or Windows / Mac with OpenSSH installed) switch to the command line and execute the following commands. The generated SSH key is by default located in the `.ssh` directory of the user home directory. Ensure that you backup existing keys in this directory before running the following commands.

```
# Switch to your .ssh directory
cd ~/.ssh

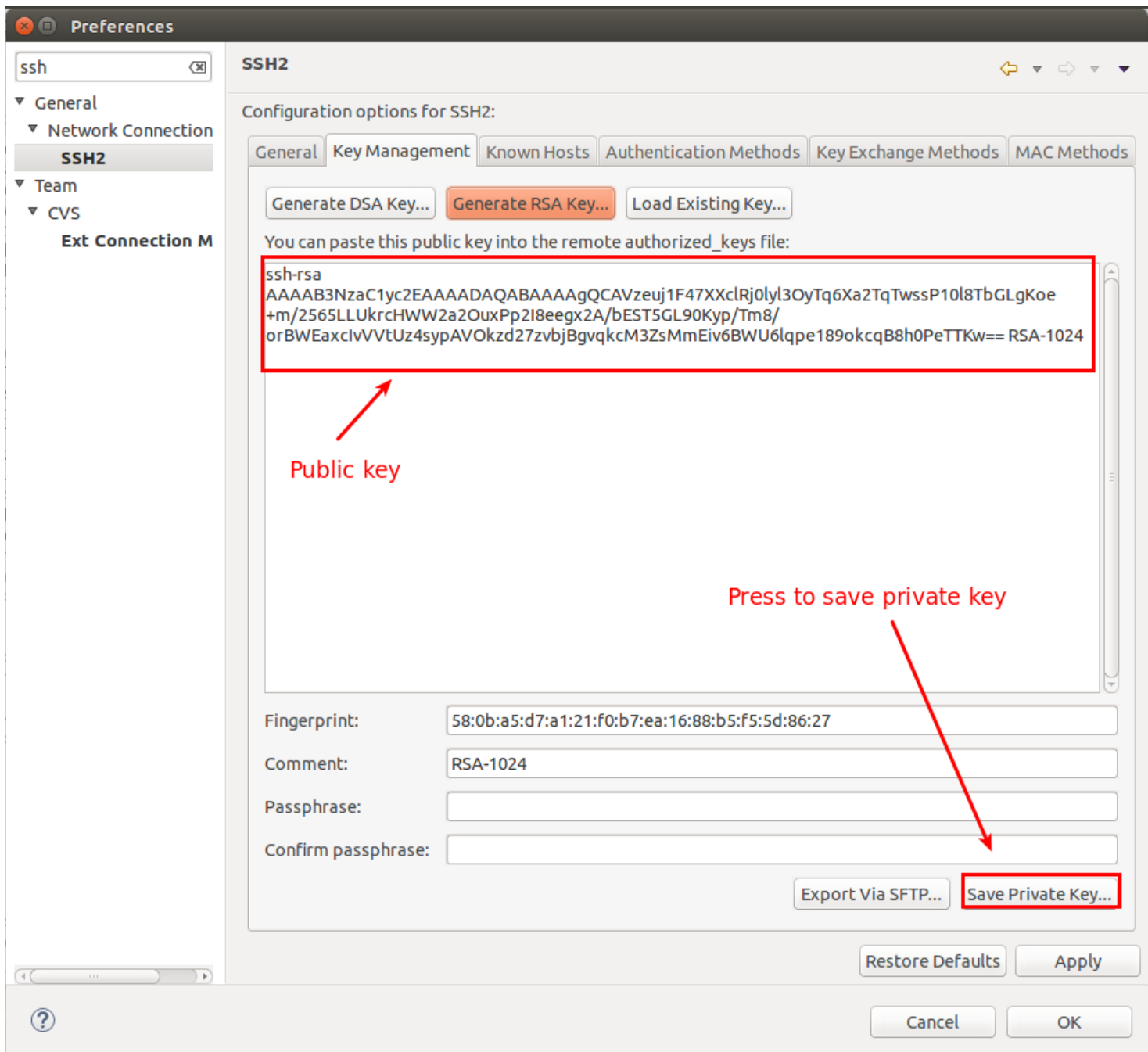
# If the directory
# does not exist, create it via:
# mkdir .ssh

# Manually backup all existing content of this dir!!!

# Afterwards generate the ssh key
ssh-keygen -t rsa -C "your_email@youremail.com"

# Press enter to select the default directory
# You will be prompted for an optional passphrase
# A passphrase protects your private key
# but you have to enter it manually during ssh operations
```

The Eclipse IDE allows you to create an SSH key pair via Window → Preferences → General → Network Connection → SSH2.



It is good practice to use a passphrase to protect your private key. It is also good practice to use operating system level permission settings to ensure that only the owning user can access the `~/.ssh` folder and its content.

**Note**

In the above `ssh-keygen` command the `-C` parameter is a comment. Using your email is good practice so that someone looking at your public key can contact you in case they have questions. Including the email enables system administrators to contact the person in case of questions.

The result will be two files, `id_rsa` which is your private key and `id_rsa.pub` which is your public key.

You find more details for the generation of an SSH key on the following webpages: [GitHub Help: description of SSH key creation](#) or [OpenSSH manual](#).

**Tip**

You can specify alternative key names with the `-f` parameter on the command line. This is helpful if you have multiple different repositories and you want to have a different key for each one. For example, you can name your SSH keys in domain name format, e.g., `eclipse.org` and `eclipse.org.pub` as well as `github.com` and `github.com.pub`.

You need additional configuration in the `~/.ssh/config` file, because only the `id_rsa` will be picked up by default. The following code shows an example.

```
Host *.eclipse.org
  IdentityFile ~/.ssh/eclipse.org

Host *.github.com
  IdentityFile ~/.ssh/github.com
```

## 3. GitHub

### 3.1. What is GitHub?

GitHub is a popular hosting provider for Git repositories. GitHub provides also additional services around these repositories, for example an issue tracker for each repository, build server integration and more.

GitHub supports that repositories can be cloned to a new Git repository hosted at Github. GitHub uses the term *fork* or *forking* for creating such clones.

GitHub provides free hosting for publicly visible Git repositories. A public repository can be cloned by other people at any point in time.

If the repository should not be visible to everyone, Git allows to create private repositories, but you must pay for this service a monthly rate. Private repository allows you to specify the people which have access to the repository and to define their access rights.

GitHub can be found under the following URL.

[GitHub](https://github.com)

If you create an account at GitHub, you can create a repository. After creating a repository at GitHub, you will get a description of all the commands you need to execute to upload your project to GitHub. Follow the instructions below.

These instructions will be similar to the following commands.

```
# global setup:
# set up git
git config --global user.name "Your Name"
git config --global user.email your.email@gmail.com

# next steps for a new repository
mkdir gitbook
cd gitbook
git init
touch README
git add README
git commit -m 'first commit'
git remote add origin git@github.com:vogella/gitbook.git
git push -u origin master

# alternatively for an existing Git repo
# add remote and push
cd existing_git_repo
git remote add origin git@github.com:vogella/gitbook.git
git push -u origin master
```




GitHub allows you to use SSH based or HTTPS based authentication to access your repositories. To clone, pull or fetch from a public available repository no authentication is required.

### 3.2. Create repository in GitHub

Once you create a user at Github, you can create a new public repository. For example the following screenshots demonstrate the creation of the *de.vogella.git.github* repository.

The screenshot shows the GitHub homepage for user 'vogella'. At the top, there's a navigation bar with 'Explore', 'Gist', 'Blog', and 'Help'. Below it, a 'News Feed' tab is selected. The main content area features a 'GitHub Bootcamp' section with four numbered steps: 1. Set Up Git, 2. Create A Repository, 3. Fork a Repository, and 4. Be social. A red arrow points from the text 'Press to create new repository' to the 'Create A Repository' step. Below the bootcamp section, there's a message 'You've been added to the [organization]' and a 'Your Repositories (57)' section with a 'New repository' button highlighted in red.

**Owner** **Repository name**

PUBLIC   **vogella** /  

Great repository names are short and memorable. Need inspiration? How about **massive-octo-tyrion**.

**Description (optional)**

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

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

☐ **Initialize this repository with a README**  
This will allow you to `git clone` the repository immediately.

Add .gitignore: **None**

**Create repository**

After creation of your new repository GitHub displays the information what you have to do if you want to connect to this repository via the command line. As we are going to use EGit you can ignore this information.

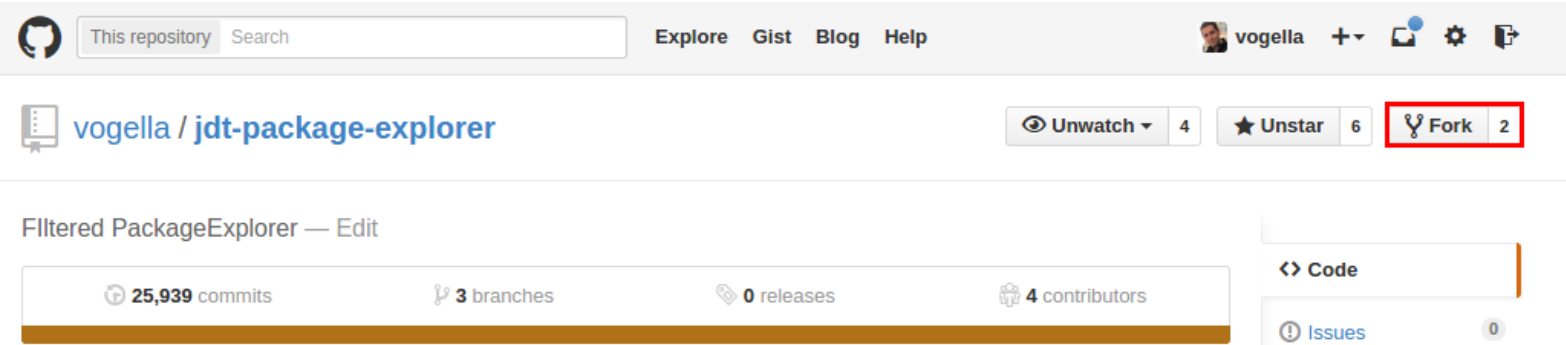
### 3.3. Merging pull request at GitHub



GitHub uses *Pull requests* for contributions.

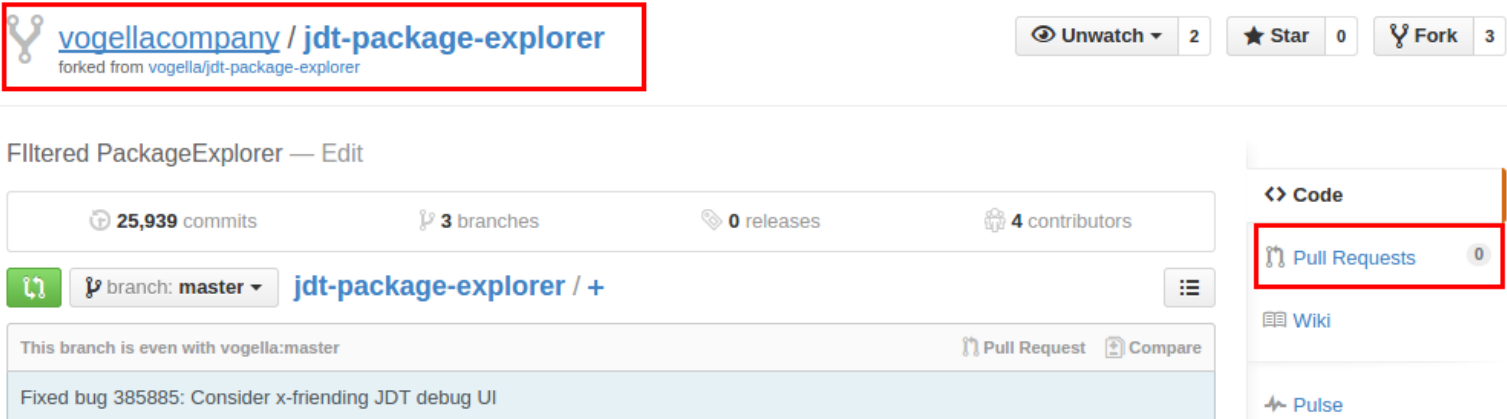
The typical workflow in GitHub is to fork a repository, create changes in your fork and send a pull request to the origin repository via the GitHub webinterface.

GitHub makes it easy to fork a repository via its web interface. Simply click the Fork button of a repository of your choice.

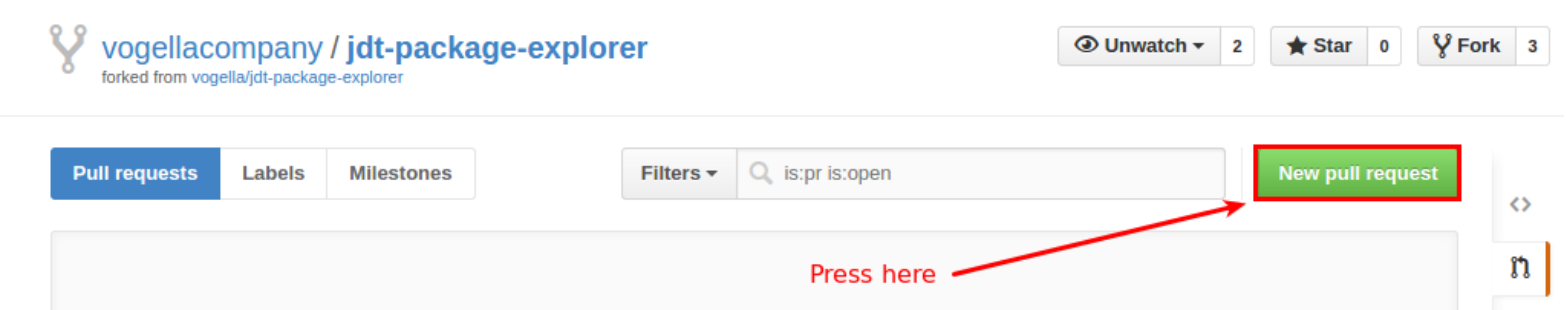


You can now clone this fork to your local development environment and push the changes to this fork at GitHub.

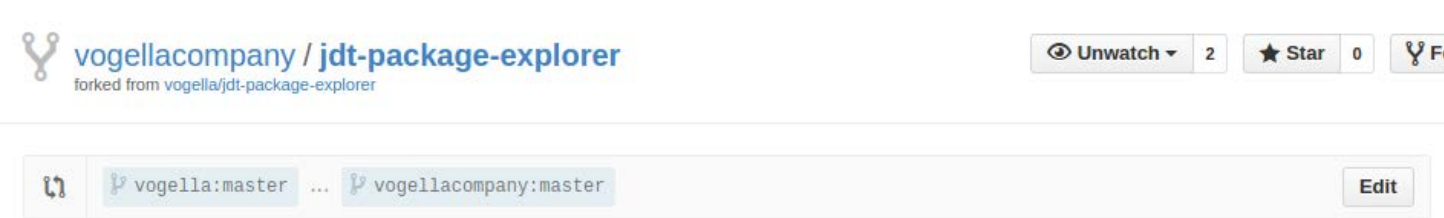
Afterwards you can create a pull request for the repository you forked from. Your repository is the fork as highlighted in the screenshot. Press Pull Requests to see existing and create new pull requests.



Click New pull request to create a new one.



On the next screen you can specify the direction of the pull request and the branches if you select the Edit button.



If the owner of the repository accepts your pull request your changes are integrated into the original repository.

## 4. Bitbucket

## 4.1. What is Bitbucket?

[Bitbucket](#) offers free hosting of public and private Git repositories.

Bitbucket allows unlimited public and private repositories. The number of participants for a free private repository is currently limited to 5 collaborators, i.e., if you have more than 5 developers which need access to a private repository you have to pay money to BitBucket.

## 4.2. Creating a repository

You need to create a user via the web interface of Bitbucket. After creating this user you can create new repositories via the web interface.

After creating a new repository on BitBucket, you can use the following instructions connect a local Git repository with the BitBucket repository.

These instructions will be similar to the following commands.

```
# Global setup:
# Set up git
git config --global user.name "Your Name"
git config --global user.email your.email@gmail.com

# Next steps for a new repository
mkdir gitbook
cd gitbook
git init
touch README
git add README
git commit -m 'first commit'
git remote add origin ssh://git@bitbucket.org/vogella/gitbook.git
git push -u origin master

# alternatively for an existing Git repo
# add remote and push
cd existing_git_repo
git remote add origin ssh://git@bitbucket.org/vogella/gitbook.git
git push -u origin master
```

## 5. Own Git server

### 5.1. Hosting your own Git server

As described before, you do not need a server. You can just use a file system or a public Git provider, such as GitHub or Bitbucket. Sometimes, however, it is convenient to have your own server, and installing it under Ubuntu is relatively easy.

First make sure you have installed the SSH tooling.

```
sudo apt-get install ssh
```

If you have not yet installed Git on your server, you need to do this too.

```
sudo apt-get install git-core
```

Create a new user and set a password for the Git system.

```
sudo adduser git
```

Now log in with your Git user and create a bare repository.

```
# Login to server
# to test use localhost
ssh git@IP_ADDRESS_OF_SERVER

# Create repository
git init --bare example.git
```

Now you can push to the remote repository.

```
mkdir gitexample
cd gitexample
git init
touch README
git add README
git commit -m 'first commit'
git remote add origin git@IP_ADDRESS_OF_SERVER:example.git
git push origin master
```

### 5.2. Give write access to a Git repository

The typical setup based on the created "git" user from above is that the public SSH key of each user is added to the ~/.ssh/authorized\_keys file of the "git" user. Afterwards everyone can access the system using the "git" user.

Alternatively you could use LDAP authentication or other special configurations.

### 5.3. Security setup for the git user

The Git installation provides a specialized shell, which can be assigned to the user. Typically this shell is located under in `/usr/bin/git-shell` and can be assigned to the user via the `/etc/passwd` configuration file to the Git user. If you assign this shell to the Git user, this user can also perform git commands which add safety to your Git setup.

## 6. Git series

This tutorial is part of a series about the Git version control system. See the other tutorials for more information.

- [Introduction to Git](#)
- [Hosting Git repositories at GitHub, Bitbucket or on your own server](#)
- [Typical workflows with Git](#)
- [EGit - Teamprovider for Eclipse](#)

## 7. Get the Book

This tutorial is part of a book available as [paper print](#) and electronic form for your [Kindle](#).

## 8. About this website

### 8.1. Donate to support free tutorials



Please consider a contribution if this article helped you. It will help to maintain our content and our Open Source activities.

### 8.2. Questions and discussion

Writing and updating these tutorials is a lot of work. If this free community service was helpful, you can support the cause by giving a tip as well as reporting typos and factual errors.

If you find errors in this tutorial, please notify me (see the [top of the page](#)). Please note that due to the high volume of feedback I receive, I cannot answer questions to your implementation. Ensure you have read the [vogella FAQ](#) as I don't respond to questions already answered there.

### 8.3. License for this tutorial and its code

This tutorial is Open Content under the [CC BY-NC-SA 3.0 DE](#) license. Source code in this tutorial is distributed under the [Eclipse Public License](#). See the [vogella License](#) page for details on the terms of reuse.

## 9. Links and Literature

[Git homepage](#)

[Video with Linus Torvalds on Git](#)

[Git on Windows](#)

**9.1. vogella Resources**

**TRAINING**

The vogella company provides comprehensive [training and education services](#) from experts in the areas of Eclipse RCP, Android, Git, Java, Gradle and Spring. We offer both public and inhouse training. Whichever course you decide to take, you are guaranteed to experience what many before you refer to as [“The best IT class I have ever attended”](#).

**SERVICE & SUPPORT**

The vogella company offers [expert consulting](#) services, development support and coaching. Our customers range from Fortune 100 corporations to individual developers.