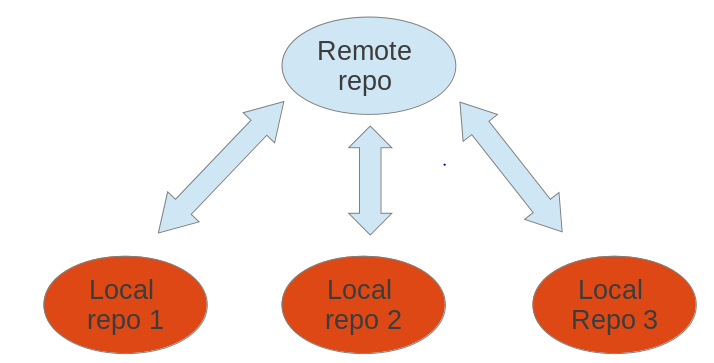
## http://www.vogella.com/tutorials/Git/article.html

## [19. Remote repositories](http://www.vogella.com/tutorials/Git/article.html#remotes)

### [What are remotes?](http://www.vogella.com/tutorials/Git/article.html#remotes_definition)

Git allows that you can synchronize your repository with more than one remote repository.

In the local repository you can address each remote repository by a shortcut. This shortcut is simply called remote. Such a remote repository point to another remote repository that can hosted on the Internet, locally or on the network.



### [Bare repositories](http://www.vogella.com/tutorials/Git/article.html#bareremotes_definition)

A remote repository on a server typically does not require a working tree. A Git repository without a working tree is called a bare repository. You can create such a repository with the   
--bare option.

# create a bare repository

git init --bare

As this is officially not supported, you should prefer cloning a repository with the --bareoption.

### [Cloning a repository](http://www.vogella.com/tutorials/Git/article.html#remotes_cloneoperation) This is used to target an existing repository and create an copy of the target repository.

### If you clone a repository, Git implicitly creates a *remote* named *origin*.

git clone git@github.com:whatever folder-name

### [Rename remote repositories](http://www.vogella.com/tutorials/Git/article.html#remote_rename)

### By default remote name will be the orgin.To rename an existing remote repository use the git remote rename command. This is demonstrated by the following listing.

# rename the existing remote repository from

# github\_http to github\_testing

git remote rename github\_http github\_testing

### If you create a Git repository from scratch with the git init command, the *origin* remote is not created

### [Adding a remote repository](http://www.vogella.com/tutorials/Git/article.html#adding-a-remote-repository) You created a new Git repository from scratch earlier. Use the following command to add a remote to your new bare repository using the *origin* name.

# add ../remote-repository.git with the name origin

git remote add origin ../remote-repository.git

### [Synchronizing with remote repositories](http://www.vogella.com/tutorials/Git/article.html#remotes_remote_synchronize)

You can synchronize your local Git repository with remote repositories. These commands are covered in detail in later sections but the following command demonstrates how you can send changes to your remote repository.

# do some changes

echo "I added a remote repo" > test02

# commit

git commit -a -m "This is a test for the new remote origin"

# to push use the command:

# git push [target]

# default for [target] is origin

git push origin

### [Show the existing remotes](http://www.vogella.com/tutorials/Git/article.html#remotes_showremote)

# show the details of the remote repo called origin

git remote show origin

# show the existing defined remotes

git remote

# show details about the remotes

git remote -v

### [Push changes to another repository](http://www.vogella.com/tutorials/Git/article.html" \l "cloneremotes_push)

The git push command allows you to send data to other repositories. By default it sends data from your current branch to the same branch of the remote repository.

### [Pull changes from a remote repository](http://www.vogella.com/tutorials/Git/article.html#cloneremotes_pull)

The git pull command allows you to get the latest changes from another repository for the current branch.

**20-25 in Home**

### [Git blame command](http://www.vogella.com/tutorials/Git/article.html#using-the-git-blame-command) The git blame command allows you to see which commit and author modified a file on a per line base. That is very useful to identify the person or the commit which introduced a change.

### [git stash command](http://www.vogella.com/tutorials/Git/article.html#stash_usage1) : (Explore)

Git provides the git stash command which allows you to record the current state of the working directory and the staging area and to revert to the last committed revision.

### [Removing untracked files](http://www.vogella.com/tutorials/Git/article.html#gitclean_command) :

Used to remove untracked files in working tree.

Git clean –dry-run : it shows what happens if we clean a git (shows how many files will be delete)

Git clean –n : it shows what happens if we clean a git (shows how many files will be delete)

Git clean –f : used to delete untracked file.

Git clean –fdx : used to delete untracked directory.

-x : remove the hidden file.

-d : delete new directory.

### [Remove staged changes from the staging area](http://www.vogella.com/tutorials/Git/article.html#undochanges_reset) :

This is exactly opposite to the git add , this will remove the files from staging area to avoid changes to be included in next commit.

Git reset [filename] : remove file from stage. [this used to revert new file from staging area]

Git reset Head [filename] : this will used to revert changed file from staging area based on last commit.

Git reset –hard : command makes the working tree exactly match HEAD.

**Start from 40**