

Gitting Started

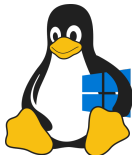
Practical for the git course

K. Müller¹ J. Verzijden²

¹MasterCLASS
E.T.S.V. Scintilla

²Scintilla Operator Team
E.T.S.V. Scintilla

June 27, 2023



Let's imagine...

Alice and Bob (you and your friend) want to make a portfolio website.



1 - Prerequisite: Installing the CLI

• Windows

- Using *winget*:

```
winget install --id Git.Git -e --source winget
```

- Using an installer or portable version:

<https://git-scm.com/downloads/win>

• Linux

- Don't, you already have it (-:
- Through your package manager:

- `apt install git`
- `dnf install git`
- `pacman -S git`

• MacOS

- Using Homebrew:
 - `brew install git`
- Install Xcode, that ships with Git

2 - Installation successful?

```
johanv@my-machine: ~$ git
usage: git [-v | --version] [-h | --help] [-C <path>] [-c <name>=<value>]
          [--exec-path[=<path>]] [--html-path] [--man-path] [--info-path]
          [-p | --paginate | -P | --no-pager] [--no-replace-objects] [--bare]
          [--git-dir=<path>] [--work-tree=<path>] [--namespace=<name>]
          [--super-prefix=<path>] [--config-env=<name>=<envvar>]
          <command> [<args>]
```

These are common Git commands used in various situations:

start a working area (see also: `git help tutorial`)

<code>clone</code>	Clone a repository into a new directory
<code>init</code>	Create an empty Git repository or reinitialize an existing one

work on the current change (see also: `git help everyday`)

<code>add</code>	Add file contents to the index
<code>mv</code>	Move or rename a file, a directory, or a symlink
<code>restore</code>	Restore working tree files
<code>rm</code>	Remove files from the working tree and from the index

examine the history and state (see also: `git help revisions`)

<code>bisect</code>	Use binary search to find the commit that introduced a bug
<code>diff</code>	Show changes between commits, commit and working tree, etc
<code>grep</code>	Print lines matching a pattern
<code>log</code>	Show commit logs

3 - Configuration

How:

- Using an editor:
`git config --global --edit`
- Using commands:
`git config --global user.name 'Johan Verzijden'`

What:

- Name (*user.name*)
- Email address (*user.email*)
- Default branch name (*init.defaultBranch*)
- Fast-forward (*pull.ff*)

Read `git help config` or `man git config` for all configuration options

4 - Starting a new project

Make a new directory and do a git init.

```
~$ mkdir my-portfolio  
~$ cd my-portfolio  
~/my-portfolio$ git init
```

5 - Making changes

Make some basic files, for instance, index.html:

```
<!DOCTYPE html>
<html>
  <head>
    <link rel="stylesheet" href="style.css">
    <title>Portfolio of Johan</title>
  </head>


  <body>
    <h1>Hello World!</h1>
  </body>
</html>
```

5 - Making changes

And also style.css:

```
body {  
    background-color: slategray;  
}  
h1 {  
    text-align: center;  
}
```


6 - Do your first commits



```
~/my-portfolio$ git add index.html style.  
css  
~/my-portfolio$ git commit -m 'Initial  
commit'
```

7 - Making changes (second commit)

Add some welcome text in your index.html:

```
<!DOCTYPE html>
<html>
  <head>
    <link rel="stylesheet" href="style.css">
    <title>Portfolio of Johan</title>
  </head>


  <body>
    <h1>Johan Verzijden</h1>
    <p>
      Welcome to my portfolio. Here you can find my
      projects.
    </p>
  </body>
</html>
```

8 - Do your second commit



```
~/my-portfolio$ git commit -am 'Add  
    heading and welcome text'  
# See the current git log:  
~/my-portfolio$ git log
```

9 - Create a new branch and switch to it



```
~/my-portfolio$ git switch -c add-git-  
project-info  
#### Alternatives  
~/my-portfolio$ git branch add-git-  
project-info  
~/my-portfolio$ git checkout add-git-  
project-info
```

10 - Make change in the separate branch

Adding something to the index.html:

```
<div class="projects">
  <div class="project-1">
    <h2>Learning about Git</h2>
    <p>Tonight I was at a great course given
      by two members of E.T.S.V. Scintilla,
      in which I learned how to use Git for
      my projects.</p>
  </div>
</div>
```

11 - Commit and switch back



```
# Add info about your latest project
~/my-portfolio$ git commit -am 'Add info
    about latest project'
```

12 - Make another change

Adding something to the index.html:

```
<div class="projects">
  <div class="project-1">
    <h2>Learning about Git</h2>
    <p>Tonight I was at a great course given
      by two members of E.T.S.V. Scintilla,
      in which I learned how to use Git for
      my projects.</p>
  </div>
  <div class="project-2">
    <h2>Learning about Git</h2>
    <p>Tonight safat for my projects.</p>
  </div>
</div>
```

13 - Commit and reset



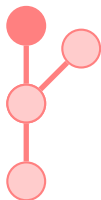
```
# Add info about your latest project
~/my-portfolio$ git commit -am 'Add
    info about latest project'
# Add info about another project
~/my-portfolio$ git commit -am 'Add
    info about another project'
# Oh wait, I don't want this in this
    branch. Let's go back
~/my-portfolio$ git log
~/my-portfolio$ git reset <previous
    commit hash>
~/my-portfolio$ git switch main
```


14 - Another change on the main branch

Adding something to the index.html:

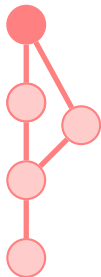
```
<div class="contact-details">  
  <p>If you have any questions about my  
    projects, you can contact me on my email  
    address</p>  
</div>
```

15 - Commit on main branch



```
# Add contact details
~/my-portfolio$ git commit -am 'Add
    contact details'
```

16 - Merge the two branches



```
~/my-portfolio$ git merge add-git-project  
-info  
# Merge conflict!
```

17 - Resolve the merge conflict

```
# Go into the file and resolve the conflict
~/my-portfolio$ git add index.html
~/my-portfolio$ git merge --continue
```

18 - Change some color

Adding something to the style.css:

```
body {  
    color: slategray;  
    background-color: slategray;  
}
```

19 - Change the text colour and commit



```
# Make the text and background colour the  
    same  
# (This is on purpose, will become clear  
    later)  
~/my-portfolio$ git commit -am 'Give the  
    text a great colour'
```

20 - Create repository on Gitlab

Your work

- Projects
- Groups
- Issues
- Merge requests
- To-Do List
- Milestones
- Snippets
- Activity
- Environments Dashboard
- Operations Dashboard
- Security

Your work > Projects

Welcome to GitLab

Faster releases. Better code. Less pain.

Create a project

Projects are where you store your code, access issues, wiki and other features of GitLab.

Create a group

Groups are the best way to manage projects and members.

Explore public projects

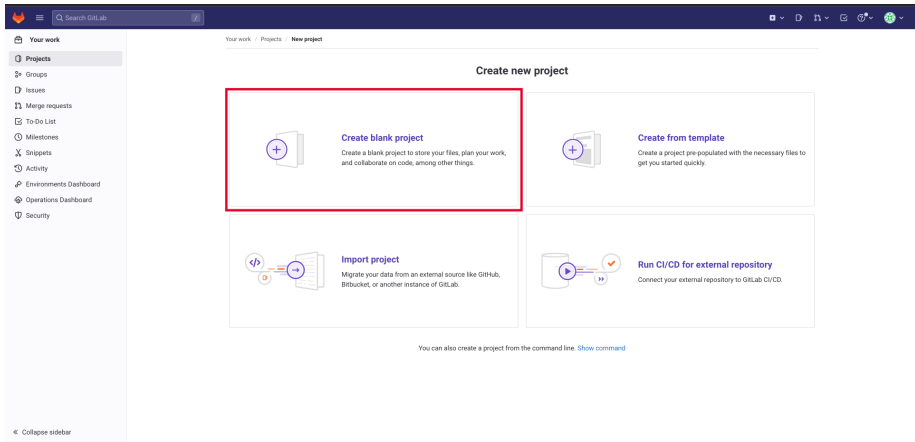
Public projects are an easy way to allow everyone to have read-only access.

Learn more about GitLab

Take a look at the documentation to discover all of GitLab's capabilities.

< Collapse sidebar

Create repository on Gitlab



The screenshot shows the GitLab interface for creating a new project. The left sidebar contains navigation options: Your work, Projects, Groups, Issues, Merge requests, To-Do List, Milestones, Snippets, Activity, Environments Dashboard, Operations Dashboard, and Security. The main content area is titled 'Create new project' and features four options:

- Create blank project**: Create a blank project to store your files, plan your work, and collaborate on code, among other things. This option is highlighted with a red box.
- Create from template**: Create a project pre-populated with the necessary files to get you started quickly.
- Import project**: Migrate your data from an external source like GitHub, Bitbucket, or another instance of GitLab.
- Run CI/CD for external repository**: Connect your external repository to GitLab CI/CD.

At the bottom, a note states: "You can also create a project from the command line. [Show command](#)"

Create repository on Gitlab

The screenshot shows the GitLab interface for creating a new project. The left sidebar contains navigation options like 'Your work', 'Projects', 'Groups', 'Issues', 'Merge requests', 'To-Do List', 'Milestones', 'Snippets', 'Activity', 'Environments Dashboard', 'Operations Dashboard', and 'Security'. The main content area is titled 'Create blank project' and includes a sub-header 'Create blank project' with a description: 'Create a blank project to store your files, plan your work, and collaborate on code, among other things.'

Three red circles with numbers 1, 2, and 3 are overlaid on the form to highlight specific fields:

- 1** Points to the 'Project name' field, which contains the text 'My awesome project'. Below the field is a note: 'Must start with a lowercase or uppercase letter, digit, emoji, or underscore. Can also contain dots, pluses, dashes, or spaces.'
- 2** Points to the 'Visibility Level' section. It has a dropdown menu set to 'Private'. Below the dropdown are three radio button options: 'Internal' (selected), 'Public', and 'Private'. Each option has a brief description of its access level.
- 3** Points to the 'Project Configuration' section. It contains two checkboxes: 'Initialize repository with a README' (checked) and 'Enable Static Application Security Testing (SAST)' (unchecked). Below the SAST checkbox is a link to 'Learn more'.

At the bottom of the form are two buttons: 'Create project' (in blue) and 'Cancel'.

Create repository on Gitlab

The screenshot shows the GitLab web interface for a newly created repository named "My awesome project". The interface includes a dark blue header with the GitLab logo and search bar, and a left sidebar with navigation options like "Project information", "Issues", "Merge requests", "CI/CD", "Security and Compliance", "Deployments", "Packages and registries", "Infrastructure", "Monitor", "Analytics", "Wiki", "Snippets", and "Settings".

The main content area displays a notification: "Project 'My awesome project' was successfully created." Below this, the project name "My awesome project" is shown with a project ID of "9405". There are buttons for "Invite your team", "Clone", "Upload File", "New file", "Add README", "Add LICENSE", "Add CHANGELOG", "Add CONTRIBUTING", "Add Wiki", and "Configure Integrations".

The "Invite your team" section includes the text "Add members to this project and start collaborating with your team." and a "Invite members" button.

The "The repository for this project is empty" section provides instructions on how to get started, including cloning the repository or starting with existing files. It also includes "Command line instructions" for setting up Git globally and creating a new repository.

```
git config --global user.name "John Doe"
git config --global user.email "john@doe@example.com"

Create a new repository

git clone https://gitlab.utwente.nl/<number>/my-awesome-project.git
cd my-awesome-project
git switch -c main
touch README.md
git add README.md
git commit -m "add README"
git push -u origin main

Push an existing folder
```

21 - Set up remote tracking locally

```
~/my-portfolio$ git remote add origin https  
://gitlab.utwente.nl/<s-number>/<project-  
name>.git
```

22 - Push to remote

```
~/my-portfolio$ git push -u origin main
```

23 - Look at repository on Github/Gitlab

The screenshot shows the GitLab interface for a repository named "My awesome project". The left sidebar contains navigation options: Project information, Repository, Issues, Merge requests, CI/CD, Security and Compliance, Deployments, Packages and registries, Infrastructure, Monitor, Analytics, Wiki, Snippets, and Settings. The main content area displays the repository details, including the project name, ID (1405), and statistics (5 Commits, 1 Branch, 0 Tags, 0 Bytes Project Storage). A recent commit is highlighted: "Merge branch 'add-learning-git-project'" by Johan Verzijden, authored 1 week ago. Below this, there are buttons for "Add README", "Add LICENSE", "Add CHANGELOG", "Add CONTRIBUTING", "Enable Auto DevOps", "Add Kubernetes cluster", "Set up CI/CD", and "Add Wiki". A table lists the files in the repository:

Name	Last commit	Last update
index.html	Merge branch 'add-learning-git-project'	1 week ago
style.css	Initial commit	1 week ago

24 - Fork from neighbour

The screenshot shows the GitLab interface for a repository named "My awesome project". The top navigation bar includes a search field and various utility icons. The left sidebar contains a list of repository features like "Project information", "Repository", "Issues", "Merge requests", "CI/CD", "Security and Compliance", "Deployments", "Packages and registries", "Infrastructure", "Monitor", "Analytics", "Wiki", "Snippets", and "Settings". The main content area displays the repository details, including the name "My awesome project", project ID "1465", and statistics for commits, branches, tags, and storage. A "Fork" button is highlighted with a red box. Below this, a recent commit is shown with a diff view, and a table of files is displayed.

My awesome project

Project ID: 1465

5 Commits 1 Branch 0 Tags 0 Bytes Project Storage

My awesome project

Project ID: 1465

5 Commits 1 Branch 0 Tags 0 Bytes Project Storage

merge branch 'add-learning-git-project'

Johan Verzijden authored 1 week ago

main my-awesome-project

Find file Web IDE Clone

Add README Add LICENSE Add CHANGELOG Add CONTRIBUTING Enable Auto DevOps Add Kubernetes cluster Set up CI/CD Add Wiki

Configure Integrations

Name	Last commit	Last update
index.html	Merge branch 'add-learning-git-project'	1 week ago
style.css	Initial commit	1 week ago

Fork from neighbour

My awesome project

My awesome project > Fork project

Fork project

A fork is a copy of a project. Forking a repository allows you to make changes without affecting the original project.

Project name

My awesome project

Must start with a lowercase or uppercase letter, digit, emoji, or underscore. Can also contain dots, pluses, dashes, or spaces.

Project URL

https://gitlab.utwente.nl/ **Select a namespace**

Project slug

my-awesome-project

Want to organize several dependent projects under the same namespace? [Create a group](#)

Project description (optional)

Visibility level

Private
Project access must be granted explicitly to each user. If this project is part of a group, access will be granted to members of the group.

Internal
The project can be accessed by any logged in user.

Public
The project can be accessed without any authentication.

Fork project **Cancel**

My awesome project

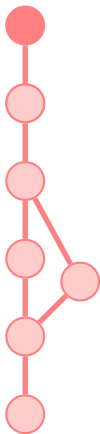
- Project information
- Repository
- Issues
- Merge requests
- CI/CD
- Security and Compliance
- Deployments
- Packages and registries
- Infrastructure
- Monitor
- Analytics
- Wiki
- Snippets
- Settings

Collapse sidebar

Clone fork to local


```
~$ git clone https://gitlab.utwente.nl/<number>/<forked-project-name>  
~$ cd <forked-project-name>
```


25 - Revert last commit to fix the text colour



```
~/<forked-project-name>$ git revert <  
commit hash>
```

26 - Do a different commit, push to remote again



```
# Change text to a different colour
~/<forked-project-name>$ git add .
~/<forked-project-name>$ git commit -m '
    Change the text colour to something
    amazing'
~/<forked-project-name>$ git push
```

27 - Open merge request

The screenshot shows the GitLab interface for a merge request. The left sidebar contains navigation options: Project information, Repository, Issues, Merge requests, CI/CD, Security and Compliance, Deployments, Packages and registries, Infrastructure, Monitor, Analytics, Wiki, Snippets, and Settings. The main content area displays the merge request details for 'My awesome project fork' (Project ID: 9407). It shows a commit titled 'Change the text colour to something amazing' by Johan Verzijden, authored 1 minute ago. The merge request is currently 'Unverified' with a SHA of 40fccf80. The source branch is 'main' in the 'my-awesome-project-fork' repository. A 'Create merge request' button is highlighted with a red box. Below the commit information, there are several checkboxes for adding features like DevOps enabled, README, LICENSE, CHANGELOG, CONTRIBUTING, Kubernetes cluster, and Wiki. At the bottom, a table lists the files involved in the merge request.

Name	Last commit	Last update
index.html	Merge branch 'add-learning-gli-project'	1 week ago
style.css	Change the text colour to something amazing	1 minute ago

Open merge request

The screenshot shows the GitLab interface for creating a new merge request. The left sidebar contains navigation options for the project. The main content area is titled 'New merge request' and includes the following sections:

- Title (required):** A text input field containing 'Change the text colour to something amazing'. Below it is a checkbox for 'Mark as draft'.
- Description:** A rich text editor with a toolbar and a text area containing 'Describe the goal of the changes and what reviewers should be aware of.' Below the editor is a link for 'Add description templates'.
- Assignees:** A dropdown menu set to 'Unassigned' and a red 'Assign to me' button.
- Reviewers:** A dropdown menu set to 'Unassigned' and a link for 'Approval rules'.
- Milestone:** A dropdown menu set to 'Select milestone'.
- Labels:** A section for adding labels.

Open merge request

The screenshot shows the GitLab web interface for a project named "My awesome project fork". The left sidebar contains navigation links for Project information, Repository, Issues, Merge requests, CI/CD, Security and Compliance, Deployments, Packages and registries, Infrastructure, Monitor, Analytics, Wiki, Snippets, and Settings. The main content area is titled "Creating merge request" and includes sections for "Approvals are optional", "Milestone" (with a "Select milestone" dropdown), "Labels" (with a "Labels" dropdown), "Merge request dependencies" (with a text input field for URLs and a note that references should be in the form of path/to/project/merge_request_id), "Merge options" (with a checkbox for "Squash commits when merge request is accepted"), and "Contribution" (with a checkbox for "Allow commits from members who can merge to the target branch"). At the bottom of the form, there are two buttons: "Create merge request" (highlighted with a red box) and "Cancel". Below the form, there is a "Commits" section showing 1 commit from 20 Jun, 2023, with a commit message "Change the text colour to something amazing" by Johan Verzijden, 6 days ago, and a commit hash of 48fccf88.

28 - Review and accept merge request

The screenshot displays the GitLab interface for a Merge Request. The title is "Change the text colour to something amazing". The "Changes" tab is selected and highlighted with a red box. The status is "Ready to merge!". A "Merge" button is highlighted with a red box. The "Activity" section shows a comment input field. The right sidebar contains metadata such as "Assignees", "Reviewers", "Labels", "Milestone", "Time tracking", "Lock merge request", "Notifications", and "1 Participant".

My awesome project > Merge requests > #1

Change the text colour to something amazing

Open · requested to merge /my-awesome-project into main · 6 days ago

Overview · Commits · Pipelines · **Changes**

0 0

Approval is optional

Ready to merge!

Squash commits Edit commit message

The source branch is [2 commits behind](#) the target branch. 1 commit and 1 merge commit will be added to main.

Merge

Activity

Sort or filter

Write Preview

Write a comment or drag your files here...

Supports Markdown. For quick actions, type `/`

Comment Close merge request

Mark as done

0 Assignees None - assign yourself

0 Reviewers None - assign yourself

Labels None

Milestone None

Time tracking No estimate or time spent

Lock merge request Unlocked

Notifications

1 Participant

Reference: s2235854/my-awesom...
Source branch: main

29 - Look at Blame

```
~/my-repo $ git blame index.html
```

Advanced topics

- Stash
- Rebase
- Cherry-pick

30 - Common Git Commands - Cheat sheet

git ...

- `init`
- `add [<filenames>]`
- `commit [-m '<message>']`
- `switch [-c] <branch name>`
- `log [--graph --oneline]`
- `push`
- `pull`
- `fetch`
- `clone <path/URL>`
- `blame <file> [--color-by-age]`
- `revert <commit>`
- `reset [--hard] <commit>`

Sources for future reference

- `git help <command>`
- `man gittutorial`
- gitimmersion.com

Think of a question later on? Feel free to reach out to us!

MasterCLASS

masterclass@scintilla.utwente.nl

Kasper Müller

kasperm@scintilla.utwente.nl

Johan Verzijden

johanv@scintilla.utwente.nl