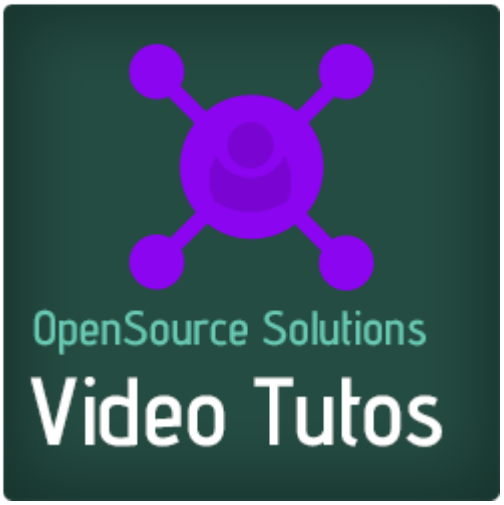
Tiki and Git workflow

Tiki and Git workflow

 Bernard Sfez - 2026-02-09 20:33



In this video I'll talk about:

- Tiki installations options
- How to install Git on your computer
- How to pull a Tiki branch from Git without all the history
- How to Upgrade, how to check status, etc.
- Git Workflow for developers
- What is changing for SVN Tiki contributors
- How to create your fork and setup your branch
- How to commit and submit your changes
- How to create a merge request and submit it for approval
- How to keep your branch up-to-date with the rest of the code
- Cherry-Picking (backport)
- Edit your commit message before push

Hello internet I'm Bernard Sfez a Tiki specialist.
In this video we will view together how to use Git with Tiki.

But first a quick tip about Tiki.

You may need information and even diagnostic about your server configuration and setting where you installed a Tiki. Tiki developers integrated a standalone script that checks over 100 different things and provides contextual feedback.
Log as admin in your Tiki, go to your Control Panels and in the Admin Navbar under the item Tools click on Server Check.

If you find this tip useful and like my video just click on the like button and subscribe to my channel. If you have a tip you want to share with the other Tikiers, add it in the comments of that

video.

Now let's start this Tiki Express Tutorial about Tiki and Git with the participation Fabio Montefuscolo a great guy, Tiki developer that was very involved to move the Tiki community to Git and in fact wrote all the documentation about Tiki and Git. This documentation I used for this tutorial is available at : https://dev.tiki.org/Git
This video is divided in two.
The first section for non-core developers and the second for Tiki developers.
While I encourage you to watch both you can stop at the first section or directly jump to the second.

In the first section

- Tiki installations options
- How to install Git on your computer
- How to pull a Tiki branch from Git without all the history
- How to Upgrade, how to check status, etc.

The second section is about the Git workflow for developers.

- Git Workflow for developers
- What is changing for SVN Tiki contributorsHow to create your fork and setup your branch
- How to commit and submit your changes
- How to create a merge request and submit it for approval
- How to keep your branch up-to-date with the rest of the code
- Cherry-Picking (backport)
- Edit your commit message before push

!!Tiki installations options
Using Git is one of the options as well as SVN but, we encourage non-developers to install Tiki using the regular packages released on a 3 month basis and available at : https://tiki.org/download or if you need a more updated version you can use Tiki pre-release tarball continually built every six hours at : https://dev.tiki.org/Daily-Build

Some users will prefer to use a continuously updated version of Tiki and there are valid reason for that. Because they need a fresh or beta feature that is evolving, because they prefer to integrate fixes as they come, etc. Using Git to update continuously your Tiki is possible a bit like we previously were launching the php program "svnup.php" in the Doc Devtools folder or using the shell with the command "svn update".
So let’s install Git on your computer

!!How to install Git on your computer
Let’s check first if git is not already installed using your shell, terminal on your local or using your ssh access for a remote server. If you don’t know what I’m talking about it the sign you should stop here as using git require minimum of knowledge.

This tutorial and the information given are developers or potential developers oriented and if this is not your case you really can and should install Tiki using the regular released package at tiki.org/download and everything will be fine.
So in your Terminal, using :

```
git —version
```

you will see what version of git your system is running. It is running on my computer and I can see the version but If none is running on yours then you need to install git using the link I give under that video at atlassian dot com and use the corresponding installer or package for your system.After the installation run again

```
git —version
```

to confirm your git is installed properly.
!!How to pull a Tiki branch from Git without all the history
Once this is done you may use Git on you system using any GUI available or even from inside your IDE like I do. For this tutorial I'll continue using my terminal. It is important to understand the command and the workflow then you can use any tool you prefer.So having git installed let's clone, create a working copy of Tiki from the official Tiki repository.The Tiki repository is available at GitLab: https://gitlab.com/tikiwiki/tikiThe complete package is a little less than 3Gb and contain all the history and commits done in Tiki. You can bring it all or just a part depending of your needs. To clone a specific branch of Tiki without all the history you need to had parameters to narrow the download. For example, if you want to create a Tiki 20 with minimum of history inside a folder named "mytiki20" you can type in your terminal:

```
git clone --branch=20.x --depth=1 https://gitlab.com/tikiwiki/tiki.git mytiki20
```

After a few minutes your working copy of Tiki will be ready to be setup and connected to the database as usual.

!!How to Upgrade, how to check status, etc.
While you have your Tiki installed the Tiki repository will continue to change and be updated. To view the differences between your working copy and the remote repo you need to fetch the changes and to indicate what you want to compare and with what. Let me show you how. I type

```
git fetch
```

, then to be sure I use the right name I do a

```
git branch -a
```

to see all the available branch and to see on what branch am I. Here I'm on 20.x. Oh and by default git use VIM editor that require you to type "q" to quit the editor. Once I have all the information I can use the diff command to do the comparison:

```
git diff 20.x origin/20.x
```

I can see that something new was committed and I can update my working copy using

```
```

```
git pull
```

I can also blindly fetch change and update without checking by doing git fetch and then

```
git pull
```

That’s it for this first part, this is very very basic but I hope it will give you enough leads to start learning and using Git with Tiki.
Let’s start now the second part of the video that will give more details and valuable information for the Tiki contributors.

!!Git Workflow
Let’s review this in details. Git is Distributed version control system and is being used to implement additional contributions to millions of projects around the world. Some of you use it already and there are plenty of information about Git on the internet but remain essential to understand that using Git any developer will be able to submit his changes to the Tiki project and that the modified code will be available for everyone to see, compare and test before being merged to the corresponding branch of Tiki.

!!What is changing for SVN Tiki contributors
With Git on Tiki you have the Tiki repository (upstream), your fork of the project and your working copy.You will push your changes to your fork and then submit your changes for approval.Once approved, your modification will be merged from your fork to the Tiki repository.
To summarise, you need create your GitLab user fork from the Tiki repository.
You create your working copy from your fork on your local computer.
You code and edit on your local working copy.
You commit also on your local working copy.
Once you’re done you push your changes to your fork.
Then you create a new merge request and participate to the approval process.That the process, let’s do it now.
How to create your fork and setup your branch
Log or Sign In to GitLab and go to the Tiki project.
Click on the Fork button at the top of the page.I have already a fork so it won’t create one but the first time it should take between 10 to 20mn.
Now from my own repository I will create a local clone.
From this clone that contain all the Tiki versions history and branch I’ll be able to checkout a working copy for any branch it contains. By default it is set to master (trunk) I’ll checkout (switch) to 20.x by typing :

```
git checkout 20.x
```

And by the way I can see if everything is up to date with the original 20.x branch or something was changed. Here I can see they were a commit on a file. As we’ve seen previously I can update my working copy doing git fetch then git pull.Now let’s pretend I want to fix something. I will create a different checkout using a name that mean something so other developers can see,

understand what this is about.

```
git checkout -b fixing-something
```

Doing this I created a branch on my local so I can start editing files.
!!How to commit and submit your changes
Now I can modify a file of my working copy.I will edit the README file.
Saved.
I check with git status and I can see I got a modified file inside my working copy but not yet in the staging area. It is displayed in red.

```
git add README
```

will add this file to the staging area.

```
git status
```

again and now the file name is displayed in green.
I commit the file to my local repo and add a commit message, again but important, this is local only, no change is sent yet to our remote fork repository. I'll use

```
git commit -m "[FIX] This is my fix of the README file"
```

tagged as per Tiki convention.
I enter git log and I can see the last committed files and the last one is the one I just committed with its hash number. We will see later how this hash number will be use for a cherry-picking, to add this modification onto different branch.
Now I can push my branch with my changes to my fork at the point of origin of my local repo, the remote repo at GitLab using:

```
git push -u origin fixing-something
```

You will have the important information displayed and here everything worked fine. With git and GitLab you'll have help all along the way. Here you can see they are already foreseen you should create a merge request and are giving some hints. Back to GitLab I do a refresh and I can see an activity update about the commit I just pushed. In the branch selector my new branch is here too.

!!How to create a merge request and submit it for approval
To create a merge request I click on the Create Merge Request green button. It open a form where I can see fields I can fill to submit my commit for approval to the rest of the community. The approval process is not automatic, other developers may discuss your commit suggest modification and approve so it is merged into the main code.

!!How to keep your branch up-to-date with the rest of the code
As we understand this process is not done as you submit your request for a merge, the code of Tiki may change. In GitLab there is a Mirroring repositories that will keep your branch up to date.

On your fork you go to Settings, Repository where it says Mirroring repositories click on the expand button in the field "Git repository URL" paste the URL repository of the Tiki project. It will update right away and then on a regular basis.

!!Cherry-Picking (backport)
There is no backport in Git and there is cherry picking. Cherry picking in Git means to choose a commit from one branch and apply it onto another. This is in contrast with other ways such as merge and rebase which normally apply many commits onto another branch.

To do a cherry picking you need the hash of the commit you want to apply onto another branch. You can find it at GitLab right next to the commit line or in your git log in the branch you committed it.Let cherry-picking (backport) our change from branch 20.x to 19.x.
I get the hash number of my commit from my log :

```
git log
```

I copy it. The I checkout branch 19.x on my local repo to set the working copy.And here I type

```
git cherry-pick and paste the hash number
```

Of course my test hasn’t been pushed back onto the remote repo so I can’t complete and type enter, but you got the idea.

!!Edit your commit message before push
The Tiki community keep trace of backport prepending the commit message with the SourceForge revision number. There is not yet a mechanism to help with this and you will need to manually edit and amend your commit message before pushing it at the moment I published that tutorial.

While your commit hasn’t being pushed yet you can edit the commit message and add the SourceForge revision number to the commit message. To do so enter:

```
git commit --amend
```

From here it will launch vi editor or any other editor if you changed git default setting and open your pending commit list. Here I click i to edit the text. I add a fake revision number and save on quit :wq
I check it is done using git log
All good.
!!That’s it for this video tutorial.There are several other git command that may be useful

```
git branch -r
git log -p
git log -p myfile
git rebase
git reset
```

```
git reset --hard
git remote show origin
git diff
git revert
git blame my_file
git config --global core.editor "nano --wait"
```

as well as plenty of parameters or even the shared option described on the Tiki dev documentation pages we saw previously.
You have plenty of documentation online and you'll more than you need depending the case you are trying to solve. It is too big for an Tiki Express Tutorial so google it.

Also your IDE like phpstorm integrate Git interface and commands so may be I will do a video about this one soon.

Please, if you like that video click on the like icon and share everywhere you think it should be.

If you don't want to miss my next tutorial and want to be notified when I publish a new video just click on the bell, the subscribe button of my youtube channel.

Thanks again for watching and
may the power of Tiki be with

Interesting links:
https://www.bernardsfez.com
https://doc.tiki.org/Server-Checkhttps://tiki.org/download
https://dev.tiki.org/Daily-Buildhttps://dev.tiki.org/Git
https://www.atlassian.com/git/tutorials/install-githttps://gitlab.com/tikiwiki/tiki
https://dev.tiki.org/Backport-changes-using-Git
https://towardsdatascience.com/10-git-commands-you-should-know-df54bea1595c