

[Return to developer documentation index](#)

This is informations on how you can get sources of Dolibarr project.

Contents

- [1 Get last version of sources by download](#)
- [2 Get last version of sources using a GIT anonymous access](#)
 - ◆ [2.1 Prerequisite](#)
 - ◆ [2.2 Clone process](#)
- [3 Get last version of sources using a full GIT access](#)
 - ◆ [3.1 Prerequisite](#)
 - ◆ [3.2 Clone process](#)
- [4 Update your repository](#)
- [5 Commit and publish your changes](#)
- [6 Add an external commit into current repository](#)
- [7 Solving a conflict](#)
- [8 Tagging sources](#)
- [9 Creating a new branch](#)
- [10 Merging a maintenance branch x.x into dev branch](#)
- [11 Statistics](#)

Get last version of sources by download

This method allows you to get sources of main development branch (called HEAD): This operation consist to get last version of Dolibarr files that are packaged each night into a tgz file. File is available with link:

[Last GitHub snapshot of Dolibarr source files.](#)

New files overwrite old ones. The inconvenient of this method is that you must download and uncompress package each time you want to get updates.

Get last version of sources using a GIT anonymous access

Prerequisite

First you must have a working install of Eclipse. **EGit** plugin must also have been installed. For this, use the **Find And Install process** into Eclipse menu, add the GIT update URL:

<http://download.eclipse.org/egit/updates>

Then choose to install **EGit**.

To get sources with goal to edit them, you need a GitHub write access to the GIT remote repository, and a GIT client (GIT command line tool with Linux, or the GIT client in Eclipse). See next chapter for this.

Clone process

First, enter value of the repository with reference sources into your GIT client tool.

- With Eclipse, choose "Clone GIT repository". Enter GIT URL of project to get.

GIT Url for Dolibarr official source is:

```
git://github.com/Dolibarr/dolibarr.git (read only)
or
git@github.com:Dolibarr/dolibarr.git (read and write access)
```

Your Eclipse GIT client will ask you to choose the branches you are interesting in. Branches not selected now could be selected later by doing a "Fetch" from the "Remotes" view, but try to choose only the branch you need. This will save you time and problems later.

Note: if you are behind a firewall, you must set Eclipse proxy credentials into Eclipse menu Windows - Preferences - General - Network connection and use the alternative URL:

```
https://gitlogin:gitpassword@github.com/Dolibarr/dolibarr.git
```

- With GIT on Command Line Interface:

First setup your git client:

```
git config --global user.name "Your Name"
git config --global user.email "user@domain.com"
git config --list
```

Then clone repository locally:

```
git clone git://github.com/Dolibarr/dolibarr.git dolibarr
```

Note: if you are behind a firewall, you must set your proxy information and use the alternative https URL:

```
git config --global http.proxy http://proxyuser:proxypass@proxyserver:proxyport
git clone https://gitlogin:gitpassword@github.com/Dolibarr/dolibarr.git dolibarr
```

Once this step is validated, your GIT client will download all files from remote GIT server. This may last several seconds or minutes.

Once download is finished, you must choose which branch to use to work on.

- With GIT on Command Line Interface;

```
git checkout develop
git checkout x.y
```

This will create local branch and make checkout automatically.

- With Eclipse, right click on "Branches - Remotes - The branch you want to duplicate locally".

FAQ_Get,update_GIT_project_sources

Choose "Create Branch" (and check the box "Checkout"). This will duplicate content of branch to work on it locally and will refresh your "Working Directory" to work on this branch.

Create Eclipse project:

If using Eclipse, once your branch is created locally, create a PHP project from assistant by choosing "**Create from existing project**" and select directory that is your GIT local repository (Do not choose to create a PHP project from scratch). Then right click on project from your Eclipse workspace and Choose "**Team - Share**". Select GIT and click on option "**Use or create repository in parent folder of project**". An alternate method is File->Import, Git->Project from Git, Select the Git repository and Next, Use New project Wizard -> Php project, enter the folder of you GIT local repository.

Then, you can setup Eclipse to avoid it to scan all project files for its build tools (Outline scanner, TODO scanner, syntax scanner...). This will avoid Eclipse to be too slow. For this, right click on project, choose Build Path and setup it like into following screenshot.

Get last version of sources using a full GIT access

Prerequisite

First you must have a working install of Eclipse. **EGit** plugin must also have been installed. For this, use the **Find And Install process** into Eclipse menu, add the GIT update URL:

<http://download.eclipse.org/egit/updates>

Then choose to install **EGit**.

You must before create an account on GitHub (<https://github.com/>) and upload a public RSA or DSA certificate into your account profile.

Clone process

Then process is same than getting sources with anonymous access.

URL to use for GIT clone is same:

```
git@github.com:Dolibarr/dolibarr.git
```

Note that a HTTP URL is also available but it's sometimes offline (so prefer using first one, it is required only if you are behind a firewall):

```
https://yourgitlogin@github.com/Dolibarr/dolibarr.git
```

Update your repository

To update your local workspace, right click on Eclipse project and select "Pull".

FAQ_Get,update_GIT_project_sources

If you made some changes into your Eclipse workspace on files that were changed into the GIT remote reference, your GIT client will make a merge automatically.

- If there is conflicts, the merge will mark all conflict files as "conflict" (See later to solve this state).
- If not, you will get all new version files including all your changes.

If you already have committed some files into your local repository, your workspace will be marked as "Merged". All you have to do is make a commit to validate your local repository as including your changes and changes made by others.

Commit and publish your changes

To see all changes you made into your working directory and waiting to be added into Git:

- With GIT Command Line Interface:

```
git status
git diff subdir/filename
```

To add changes made into your local repository into the GIT index, then commit this index :

- With Eclipse: Right click on file or directory to commit. Select file to add/remove/update into commit.
- With GIT Command Line Interface:

```
git commit -a -m "Commit text"
```

This will save add and changes into your local branch.

To push them on the GIT repository server,

- With Eclipse: right click on project and choose "**Team - Push to Upstream**".
- With GIT Command Line Interface:

```
git push
```

To cancel a commit:

- With GIT Command Line Interface:

```
git revert IdCommit or git revert HEAD (for last commit)
```

Add an external commit into current repository

Some developers may work and do commit into their own Git repository after forking the project. If you are interesting into getting this changes, you can import them with the following steps.

You need both Eclipse and GIT Command Line interface to achieve this:

- First, go into GIT setup view and right click on "**Remotes**" of your Eclipse Git working space and choose "**Create a remote...**"


```
git tag -d "x.y.z_YYYYMMDD"  
git push origin :x.y.z_YYYYMMDD # To validate delete on remote origin
```

Creating a new branch

With GIT Command Line Interface:

Create a clone (see previously) of the branch from where you want to create another branch, then go into directory and launch command:

```
git branch x.y  
git push origin x.y
```

With Eclipse:

Create a clone (see previously) of the branch from where you want to create another branch, with a checkout done. The go into view git. Choose brnahces, remote tracking and select start brancht. Right click and choose "Create branch". Then go into "Remotes - origins" and edit the entries to pull and push onto new branch.

Merging a maintenance branch x.x into dev branch

With Eclipse:

- Go onto the root directory of project with a checkout of the dev branch.
- Check fetch setup to be sure, this repository contains both information of dev branch and version x.x you want to merge.
- Right click onto root directory and choose "Team - Merge". Select "x.x". Click onto "Merge options - Commit", because we want to commit merged branch if there is no conflicts.
- Once merge is done, check everything is ok.
- Commit and push.

Statistics

With GIT Command Line interface:

To count number of changes developers have made between a version x and y:

- First, run git merge-base to know the id of last common commit between x and y

```
git merge-base origin/3.1 origin/develop
```

- Then run git log to have list of all changes from last common commit to last commit

```
git log id_found_at_previous_step..HEAD
```

- To get total number of added/deleted lines:

```
git log --ancestry-path --numstat --pretty="%H" id_found_at_previous_step..HEAD | awk 'NF==3 {plus+=
```

FAQ_Get,update_GIT_project_sources

To count number of changes finally between a version x and y (it differs from previous count because in previous count, a change can be done 2 times at two different moment to change differently. With second method, you count changes finally found if changes were ok at first try).

- Use same method than previously but instead of running git log, run git diff.

```
git diff -b -M --numstat --pretty="%H" id_found_at_previous_step..HEAD | awk 'NF==3 {plus+=$1; minus
```

More information in GIT usage is available [here](#).