

[Return to developer documentation index](#)

This page contains some frequently asked questions related to Dolibarr development. For frequently asked questions on other topics, read page [User FAQ](#).

## Contents

- [1 How to be an official developer](#)
- [2 Current team](#)
- [3 Start a development](#)
- [4 Get/update project sources](#)
- [5 How to create a new skin](#)
- [6 How to develop my own menu system](#)
- [7 How to develop my own module](#)
- [8 How to add/complete a translation](#)
- [9 Change my numbering system after start](#)
- [10 Submit a patch, work and help on Dolibarr development](#)
  - ◆ [10.1 With a GIT account](#)
  - ◆ [10.2 With no GIT account](#)
- [11 How to package and deploy a module](#)
- [12 How to be informed on news and what happens on Dolibarr project ?](#)
- [13 Migrate my Mysql database to PostgreSQL](#)

## How to be an official developer

In the galaxy of the Dolibarr project, there are several actors with different levels / grades. The number of developers is important so contributions and changes, for any project that wants to reach a certain level of quality, must be controlled and validated (both for functional needs and for coding). For this reason, the write access on the source (GIT) is limited and governed by an organization that is described in the following. Each developer has a *grade* depending on seniority and the past. The level of authorization of a person on the project increases with the stages of promotion in grade.

The course of a Dolibarr developer passes through these steps, which are classified by a title familiar to fans of science fiction. Here are the steps. It is important to understand that this organization has the sole purpose of improving the quality of Dolibarr and it is necessary that it is rigorously used. So do not be offended if, as a developer, you can not have a GIT write access for some time.

The ranks of contributors to the Dolibarr project are:

### \* **Soldier**

The first thing that a person willing to help needs to be familiar with the rules and standards developments. By reading the [Developer documentation](#), you have shown sufficient willingness to automatically become **Soldier** (grade level **Soldier**). Most people that are interested in Dolibarr development are in this category.

This grade is obtained without higher validation. It is a grade of principle, to start.

## Developer\_FAQ

The next step is to submit your first patch, starting with simple ones, on the dev Mailing List (see chapter [Submit a patch, work and help on Dolibarr development](#) for the procedure) or on the forum (Preferred Mailing List).

You can find easy to do patches (easy hacks) on the TODO tracker:  
[https://doliforge.org/tracker/?atid=247&group\\_id=144&func=browse](https://doliforge.org/tracker/?atid=247&group_id=144&func=browse)

Easy to start tasks have the property "Task opened to external developers" set to "yes". If the value is "no", forget it, this means the task will be processed by a Jedi-grade developer.

The number of soldiers is not limited. The more we are, the stronger we will be. The soldiers are the main force of the Dolibarr alliance.

Due to the nature of this level, it is not possible to establish a list of all Dolibarr actors with this grade (list always changing and too long).

### \* **Z-6PO**

If you are fluent in a language not present or poorly translated into Dolibarr, and know how to use GIT, you can use GIT to fork project locally and use push system to publish your changes/adds. If you don't know, send your changes done on last version of language files on **dolibarr-dev** mailing-list (Subscribe from page <https://savannah.nongnu.org/mail/?group=dolibarr>). However, first solution using GIT push is preferred.

The Dolibarr core team may add you, after several push, as official Z-6PO on page [Category:Z-6PO](#).

### \* **Admiral**

Only after you've held the rank of Soldier for a variable time (which can be very long), you've submitted many patches of suitable quality, a Yoda in place allows you to make GIT commits extended for cause of any patch. You have become an **Admiral** of the Dolibarr alliance.

This title can't be requested. It is obtained by a decision or proposal of a **Jedi** or **Yoda** who believes that you have submitted enough patches to be a serious **Admiral**. An **Admiral** GIT access, however, must be used to commit translations or bug fixes that are reported on bug tracker ([https://doliforge.org/tracker/?atid=246&group\\_id=144&func=browse](https://doliforge.org/tracker/?atid=246&group_id=144&func=browse)) or forum (such as personal fixes or fixes submitted by a **Soldier**). So all GIT commits you do must necessarily include in the description (the commit log field), the issue of bug corrected or the title of the forum message indicating the problem. Only a translation escapes this constraint (no restrictions for translations). Failure to observe this constraint may lead to downgrade to the rank of **Soldier**.

Obtaining the rank of Admiral is as rare as limited. Dolibarr quality can not be achieved without a limited number of Admirals, but this is still compatible with having a very high number of developers (**Soldiers**), contributing through patches.

A list of Dolibarr contributors with this grade is available on page [Category:Admiral](#)

### \* **Jedi**

Only after holding the rank of Adminral for a variable time (probably the longest of all), and the quality of GIT bug fixes or patches sent is satisfactory, a Yoda will tell you that you are authorized to perform commits without restrictions, including new features. However you will still have to explain to all Yoda what you want to do to give back a "go". You will then become a **Jedi**. There may be temporary restrictions on commits, defined by a

Yoda, for example due to a close release.

This title can't be asked, it is obtained by a proposal of another Jedi or by a Yoda.

This privilege is exceptional. Dolibarr quality can not be achieved without a limited number of **Jedi**, but this is not compatible with a high number developers (**Soldiers**) contributing with patches.

A list of Dolibarr contributors with this grade is available on page [Category:Jedi](#)

### \* **Yoda**

There are about 1 to 5 **Yoda**. It's the overall management of project development. This ranking is obtained by inheritance or vote. For example, it can be achieved after a vote organized by a Yoda making retirement.

A list of Dolibarr contributors with this grade is available on page [Category:Yoda](#)

### \* **And Darth Vader?**

There are some parasitic people that pollute the forum or mailing-list with aggressive messages, which slows the development of Dolibarr rather than offering constructive criticism to help improve. Some are very strong bad liver and bad tone. Such people are Darth Vaders. Fortunately, statistically, there is only one or two Darth Vader per year ...

## Current team

See page [Dolibarr Project](#) to know list of current people that work on Dolibarr, with their current grade.

## Start a development

Read first **completely** the [Developer documentation](#) to know all rules a developer must respect. You can also read following FAQ.

## Get/update project sources

See page [FAQ Get.update project sources](#).

## How to create a new skin

For this, see page [Skins](#)

## How to develop my own menu system

For this, see page [Menus system](#)

## How to develop my own module

To develop your own numbering module, see [Create numeration module](#).

To develop your own document model, PDF or other, see [Create a PDF document template](#) or [Create an ODT document template](#).

To develop your own business module (screens, tables), see page [Module development](#).

## How to add/complete a translation

For this, see page [Translator documentation](#)

## Change my numbering system after start

If the new numbering system does not conflict with the old one, to change the numbering rule, just go to menu Setup - Modules - Invoice setup and choose the new numbering rule in the list. If the new rule can create conflicts with old one, it will be necessary to rename old existing references. This can be done by a SQL request. For example, to go from numbering module Jupiter (FYYYYMM99) to Terre (FAYYMM-999), you can run the following request:

```
UPDATE llx_facture SET facnumber=CONCAT('FA',substr(facnumber,4,4),'-',substr(facnumber,8))
WHERE facnumber LIKE 'F%' AND facnumber NOT LIKE 'FA%';
```

For example, to rename references from model FAYYMM999 to Terre (FAYYMM-999), you can run the following request:

```
UPDATE llx_facture SET facnumber=CONCAT('FA',substr(facnumber,3,4),'-',substr(CONCAT('0000',substr(facnumber,5,4)),5,4))
WHERE facnumber LIKE 'FA%' AND facnumber NOT LIKE '%-%';
```

## Submit a patch, work and help on Dolibarr development

If you want to know what you can do to start or help Dolibarr development, read instead chapter [Developer FAQ#How to be an official developer](#). If you already read it and want to distribute a patch, this chapter is for you.

### With a GIT account

For the moment, GIT write access are restricted (number of commits is already active). If you have this access, you can use it, but if and only if you use it to commit changes you are granted to with your developer grade (See [Developer FAQ#How to be an official developer](#) for information on different grades). If you don't have GIT write access (your grade is [Soldier](#)), it is necessary to follow the following steps...

If you have a GIT write access, see [#Get/update project sources](#) for information to use GIT.

## Developer\_FAQ

If you don't have a GIT write access, you can make a "Push Request" from GitHub. Otherwise, you can also follow next process.

### With no GIT account

With no GIT write access, it is necessary to generate and send by mail a patch file to the developer mailing-list **dolibarr-dev** (<https://savannah.nongnu.org/mail/?group=dolibarr>). This is how to generate a patch file:

*For all OS: ...*

This is the *best method* to work to build such a patch file:

- First, you must have a directory containing last reference version of Dolibarr (the result of a GIT update or simply the files resulting of uncompressing a dolibarr.tgz snapshot). We will call this directory **old\_dir**. You can get this snapshot for current development version on [Dolibarr official web site in download area - development version](#). Warning: You must get "last development source code" and not "last stable version code" to build your patch, or your patch will be obsolete before you start to work on it !

- Then, you must have a second directory, from same source, that will contains Dolibarr directories, in which you will make all you files. We will call this directory **new\_dir**.

To build the patch file, you must launch the **diff** command (available on all Linux, provided with [cygwin](#) under Windows) with the following command:

```
diff -BNaur --exclude=CVS --exclude="*.patch" --exclude=".#*" --exclude="*~" --exclude="*.rej"
--exclude="*.orig" --exclude="*.bak" --exclude=conf.php --exclude=documents
old_dir new_dir > mypatch.patch
```

A ksh script to run this command is available in directory **build/patch**. Send your patch on Dev Mailing List to the address **dolibarr-dev@nongnu.org** (after subscribing to it). Inclusion of your patch is however not guaranteed, no more than delay. But if patch is realized strictly with using this process, there is a very important probability (near 100%) that it will be at least "tested" (if patch is not done this way, chance are simply null).

*For Windows: .*

If you work on Windows, another method (not so nice than previous) is also possible. Install the open source compare tools called Winmerge (this tool is able to build patch with format *diff -Naur*). Then compare with WinMerge the reference file and modified file and choose in menu "*Tools - Generate patch*". Add an output filename like "*mypatch.patch*", click on box "*Append*" and choose option "*Format Unified*". Then click on "*Ok*". Eventually restart for each modified file. Finally, you will get a file *mypatch.patch* that contains all changes with good format.

#### *Test/Apply a patch*

If you want to apply a patch file on an old version to have a new one modified by the path file, this is possible with the **patch** tool. Imagine you have on non modified version somewhere on a server. Put you patch file into the root directory. Go into this root directory then launch the command:

```
patch -u -p0 -d . < mypatch.patch
```

## Developer\_FAQ

For information: -p0 is to use full path of files defined in patch file to locate them on your disk (-pn will remove the n first levels of directories) -d define the relative path do directory to patch. < is to provide source file name to use (patch must have format previously defined) -u is to tell that patch has unified format

*Send patch by mail*

Once patch file is built and successfully tested, you must submit file to the Dolibarr mailing list **dolibarr-dev** (See here for this <https://savannah.nongnu.org/mail/?group=dolibarr>). This mailing list is read by most Dolibarr developers, however we can't say when patch will be processed nor that if it will be included. It depends on quality of patch and priorities of moment. Some patches are added several months after being submitted.

## How to package and deploy a module

See page [Module development](#).

This process works also to generate a package to submit it on the <http://www.dolistore.com> market place.

## How to be informed on news and what happens on Dolibarr project ?

See page [FAQ How to be informed on news about Dolibarr project ?](#)

## Migrate my Mysql database to PostgreSQL

See page [FAQ Migrate my Mysql database to PostgreSQL](#)