Using a development version of Lazarus

Michaël Van Canneyt

March 19, 2022

Abstract

Lazarus evolves continuously. Because it is an open source project, you don't need to wait for a release to be able to use the latest features. In this article we show how to compile and use the latest development version of the Lazarus IDE.

1 Introduction

The lazarus team keeps on developing the lazarus IDE and the LCL (the Lazarus Component Library). If you are eager to use one of the new features, it is not necessary to wait for the official release of a new version of Lazarus. Because Lazarus is an open source project, you can perfectly install the latest sources and build Lazarus for yourself.

The sources of lazarus are available publicly on Gitlab:

```
https://gitlab.com/freepascal.org/lazarus/lazarus
```

In order to build lazarus yourself, you need 2 things:

1. An existing Lazarus installation. At the moment of writing, this is version 2.2.0, using Free Pascal compiler 3.2.2. In this text we assume Lazarus is installed in its default location:

   ```
   C:\Lazarus
   ```

2. A git client. This is not really a necessity, but makes life easier if you want to update Lazarus on a regular basis.

The lazarus installation has everything to build a new version of Lazarus. This should not come as a surprise, because the Lazarus IDE rebuilds itself as soon as you install a new package in the IDE.

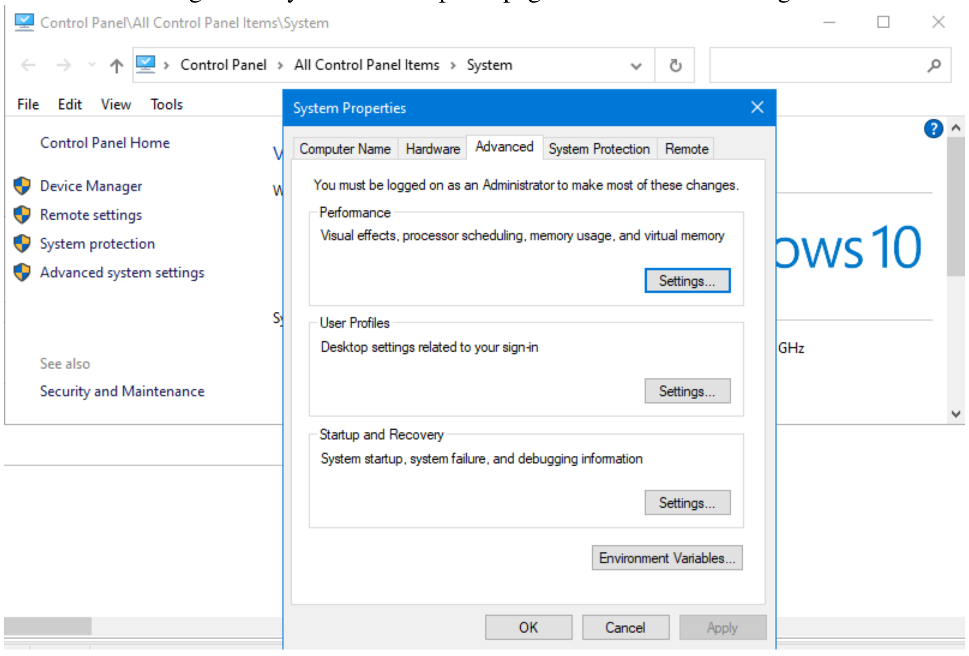
You can make do without git, as it is always possible to download lazarus sources in a zip file:

```
https://gitlab.com/freepascal.org/lazarus/lazarus/-/archive/main/lazarus-main.zip
```

This URL gives you a ZIP file with the latest sources. But using git you can update the sources faster if you plan to keep following the latest version of Lazarus.

2 Some preliminaries

Building Lazarus requires you to enter some commands on the command-line: Lazarus is built using the GNU Make tool, which is a command-line tool. The tool is called `make`,



You must be logged on as an Administrator to make most of these changes.

Visual effects, processor scheduling, memory usage, and virtual memory

Desktop settings related to your sign-in

System startup, system failure, and debugging information

Figure 1: System control panel page with advanced settings

and is installed together with Free Pascal on windows. Linux or Mac installations have a make tool installed by default.

To be able to use the `make` tool, it must be in a directory that is included in the `PATH` environment variable. So, you must make sure this is the case, Again, on Linux and Mac this is normally the case.

If you are on Windows, and have Delphi installed, you will also have the Delphi `make` tool installed. It serves the same purpose as the GNU make tool, but has much less features. It is therefore important that when you enter the `make` command on the command-line, that the correct version of make is used.

During its installation procedure, Delphi changes the `PATH` environment variable to include the directory with the Delphi version of `make` (as well as the other delphi tools).

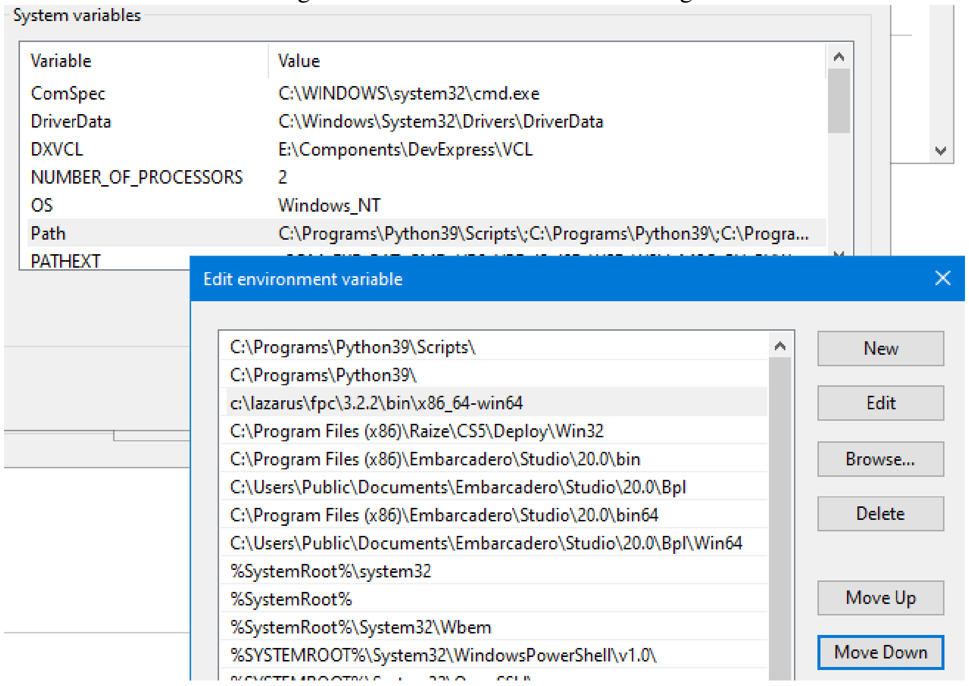
So, it is imperative that the `PATH` environment variable must be set in such a way that the directory with the FPC version of `make` comes before the one with the Delphi version of make. Delphi no longer uses its `make` tool, so changing this will not damage the Delphi installation.

To set the PATH variable, in the Windows Control Panel, choose 'System'. In this dialog, the 'Advanced system settings' link must be used , in which case you will see a dialog pop up which resembles figure 1 on page 2.

The 'Environment variables...' button in the bottom-right of that dialog allows you to set the environment variables of Windows. There are 2 sets of variables: user-specific variables (at the top) or system variables. Both will contain a `PATH` variable. In the command-line window, both `PATH` variables will be used. The directories in the system `PATH` variable take precedence over the ones in the user-specific `PATH` variable. If you have Delphi installed, it is therefore best to change the system `PATH` variable. Select the 'PATH' variable, and press the 'Edit...' button. A special dialog will pop up in which the contents of the `PATH` variable have been split into lines: one per directory, see figure 2 on page 3.

In this dialog the 'New' button can be used to add a new directory to the `PATH`. The directory to add is

Figure 2: Environment variables dialog



Variable	Value
ComSpec	C:\WINDOWS\system32\cmd.exe
DriverData	C:\Windows\System32\Drivers\DriverData
DXVCL	E:\Components\DevExpress\VCL
NUMBER_OF_PROCESSORS	2
OS	Windows_NT
Path	C:\Programs\Python39\Scripts\;C:\Programs\Python39\;C:\Progra...
PATHEXT	[illegible]

```
C:\Programs\Python39\Scripts\
C:\Programs\Python39\
c:\lazarus\fpc\3.2.2\bin\x86_64-win64
C:\Program Files (x86)\Raize\CS5\Deploy\Win32
C:\Program Files (x86)\Embarcadero\Studio\20.0\bin
C:\Users\Public\Documents\Embarcadero\Studio\20.0\Bpl
C:\Program Files (x86)\Embarcadero\Studio\20.0\bin64
C:\Users\Public\Documents\Embarcadero\Studio\20.0\Bpl\Win64
%SystemRoot%\system32
%SystemRoot%
%SystemRoot%\System32\Wbem
%SYSTEMROOT%\System32\WindowsPowerShell\v1.0\
```

```
C:\lazarus\fpc\3.2.2\bin\x86_64-win64
```

If you have the Win32 version of lazzrus installed, the directory to use is:

```
C:\lazarus\fpc\3.2.2\bin\i386-win32
```

if you have another version of Lazarus (or Free Pascal), you may need to adapt the path.

You can use the 'Move up' and 'Move down' buttons to move the new directory *before* the entry of the Delphi IDE (as visible in figure 2 on page 3). After you confirm the new `PATH` settings with the 'OK' button, you can check that the correct version of make is called, by entering the following command on the command-prompt:

```
make -v
```

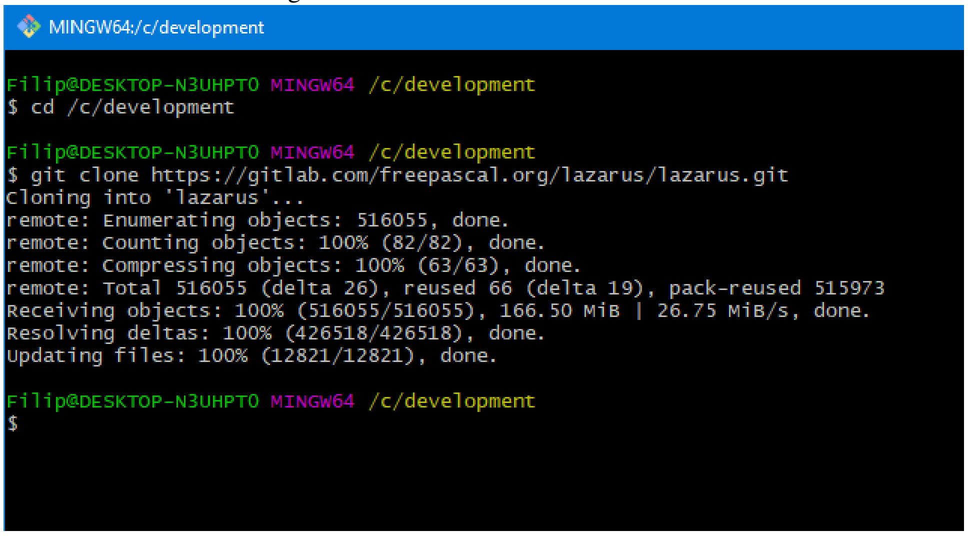
The output will be something like this:

```
c:\Development\lazarus>make -v
GNU Make 3.80
Copyright (C) 2002  Free Software Foundation, Inc.
This is free software; see the source for copying conditions.
There is NO warranty; not even for MERCHANTABILITY or FITNESS FOR A
PARTICULAR PURPOSE.
```

3 Download using git

In an earlier series of articles in Blaise Pascal Magazine, the installation and use of Git has been covered in depth. In this article we will therefore limit the instructions to the download of Lazarus sources. The repository can be cloned from:

Figure 3: Git clone on the command-line



```
MINGW64:/c/development

Filip@DESKTOP-N3UHPTO MINGW64 /c/development
$ cd /c/development

Filip@DESKTOP-N3UHPTO MINGW64 /c/development
$ git clone https://gitlab.com/freepascal.org/lazarus/lazarus.git
Cloning into 'lazarus'...
remote: Enumerating objects: 516055, done.
remote: Counting objects: 100% (82/82), done.
remote: Compressing objects: 100% (63/63), done.
remote: Total 516055 (delta 26), reused 66 (delta 19), pack-reused 515973
Receiving objects: 100% (516055/516055), 166.50 MiB | 26.75 MiB/s, done.
Resolving deltas: 100% (426518/426518), done.
Updating files: 100% (12821/12821), done.

Filip@DESKTOP-N3UHPTO MINGW64 /c/development
$
```

```
https://gitlab.com/freepascal.org/lazarus/lazarus.git
```

or, if you prefer to use SSH:

```
git@gitlab.com:freepascal.org/lazarus/lazarus.git
```

We'll install the lazarus sources below a directory

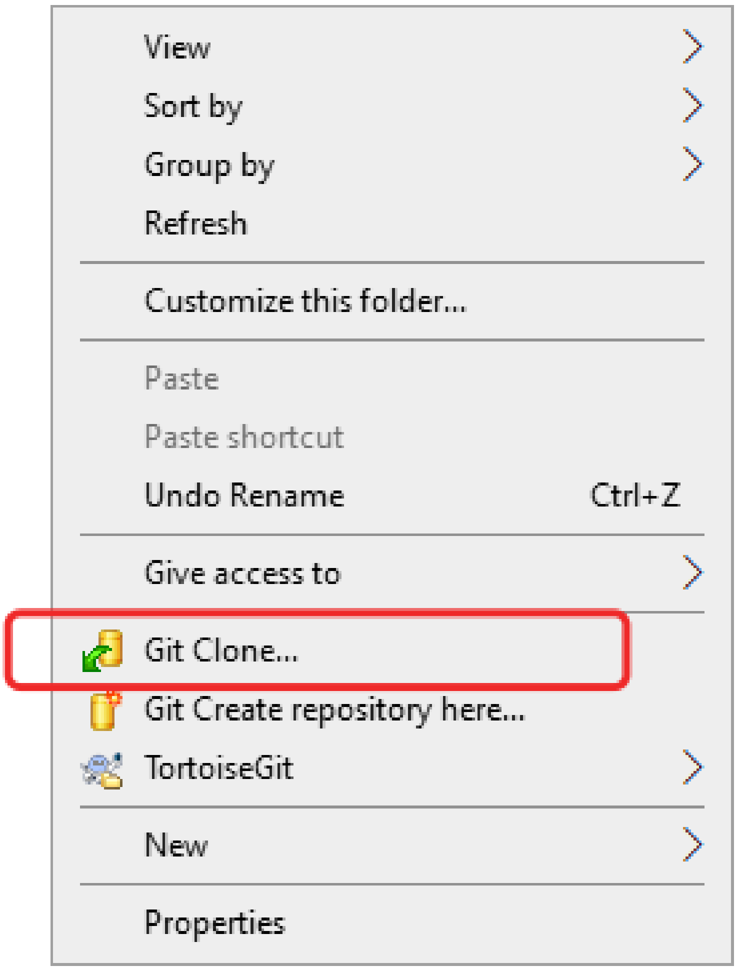
```
C:\Development\
```

Obviously, you're free to choose whatever directory you want.

If you have Git for Windows installed, then you can clone the sources with the following command in the Git bash window:

```
cd /c/Development
git clone https://gitlab.com/freepascal.org/lazarus/lazarus.git
```

This is also the command you can give on Mac or Linux, and the output will look like figure 3 on page 4. If you are using TortoiseGit, then you can use the context menu of the Windows file explorer:



Doing so, will show the `Git clone` dialog, shown in figure 4 on page 6, where you can enter the URL mentioned above. After the initial clone operation, you can always update the sources with the `git pull` command.

4 Building Lazarus

When the git clone operation is complete, Lazarus can be built. For this, the windows command-line windows must be used. Do not attempt to use the bash shell from your Git for windows installation: this build environment is not supported.

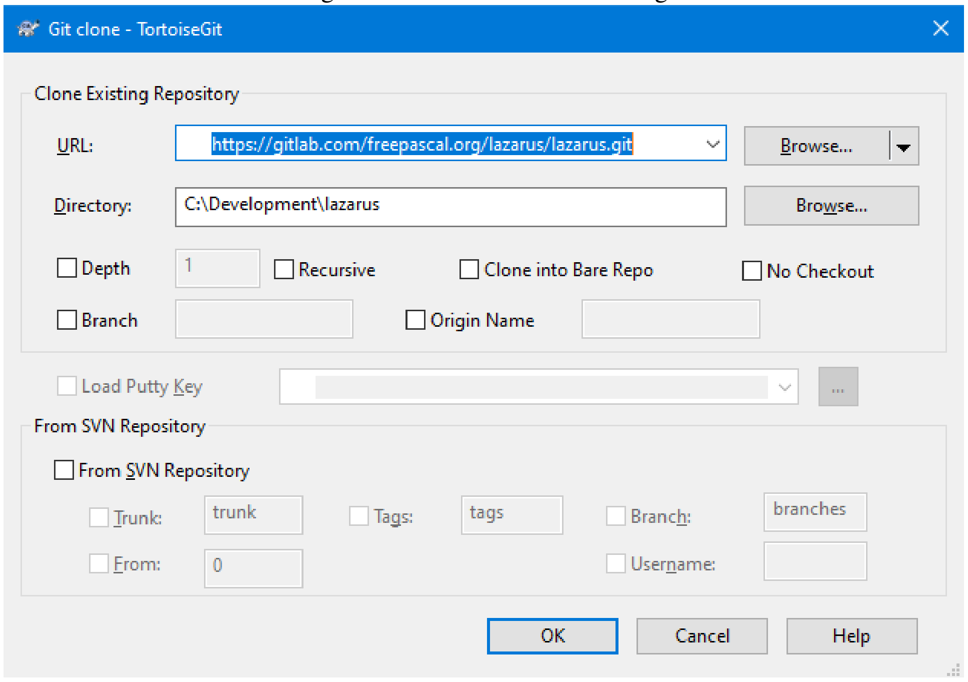
Building the Lazarus IDE is a matter of 2 commands:

```
cd c:\Development\Lazarus
make bigide
```

The `make bigide` will actually build Lazarus, together with some commonly used packages.

Building Lazarus takes some time. The `make` command will also build `Startlazarus.exe` and some other tools. When make stops running, please take a look at the output of the make command - in particular, check whether errors are displayed or not. If not, all went

Figure 4: Tortoise Git clone dialog



URL: https://gitlab.com/freepascal.org/lazarus/lazarus.git
Directory: C:\Development\lazarus
Depth: 1
Branch
Origin Name

well, and a `lazarus`, `startlazarus` and `lazbuild` command will be present in the build directory.

5 Configuring lazarus

To start your new version of lazarus, you must use the newly created application binary. You can start it in the Explorer, but it is of course easier to create a shortcut on the desktop: in the File explorer, simply drag the lazarus executable to the desktop while keeping the `Alt` key pressed. (or use the context menu 'New - shortcut' in the fle explorer).

When you first start the new Lazarus, you may get some dialogs in which Lazarus tells you that the settings have changed: see figure 5 on page 7 and figure 6 on page 7.

If you wish to use 2 separate configurations for your installed lazarus and the newly compiled lazarus, you should cancel here, and adapt the shortcut so it contains the command-line option `-pcp` indicated in figure 6 on page 7, for example:

```
--pcp=C:\test_lazarus\configs
```

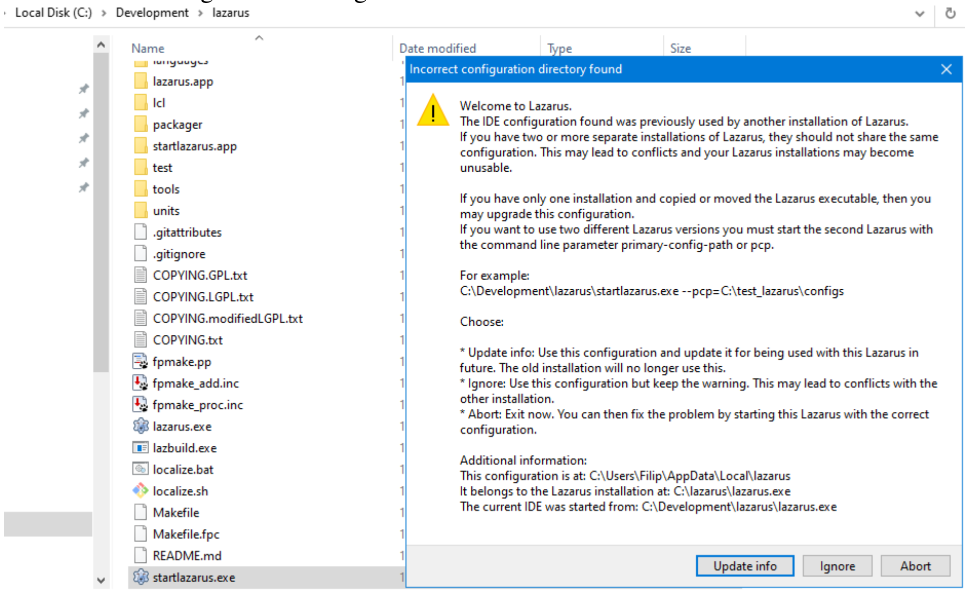
You can of course choose any directory you want for the configuration.

When you did all this, you will probably still get the Lazarus installation check-up dialog shown in figure 7 on page 8. In particular, the GDB (gnu debugger) location will be missing. You can reuse the one from the original lazarus installation:

```
C:\lazarus\mingw\x86_64-win64\bin\gdb.exe
```

To ensure that you are now really working with the latest lazarus, you can check the `Help - About Lazarus` dialog. It should display the latest version number, which is `2.3.0` at the time of writing of this article, as can be seen in figure 8 on page 8

Figure 5: Starting a new Lazarus version for the first time


Local Disk (C:) › Development › lazarus

Name
lazarus.app
lcl
packager
startlazarus.app
test
tools
units
.gitattributes
.gitignore
COPYING.GPL.txt
COPYING.LGPL.txt
COPYING.modifiedLGPL.txt
COPYING.txt
fpmake.pp
fpmake_add.inc
fpmake_proc.inc
lazarus.exe
lazbuild.exe
localize.bat
localize.sh
Makefile
Makefile.fpc
README.md
startlazarus.exe

Incorrect configuration directory found

Welcome to Lazarus.
The IDE configuration found was previously used by another installation of Lazarus.
If you have two or more separate installations of Lazarus, they should not share the same configuration. This may lead to conflicts and your Lazarus installations may become unusable.

If you have only one installation and copied or moved the Lazarus executable, then you may upgrade this configuration.
If you want to use two different Lazarus versions you must start the second Lazarus with the command line parameter primary-config-path or pcp.

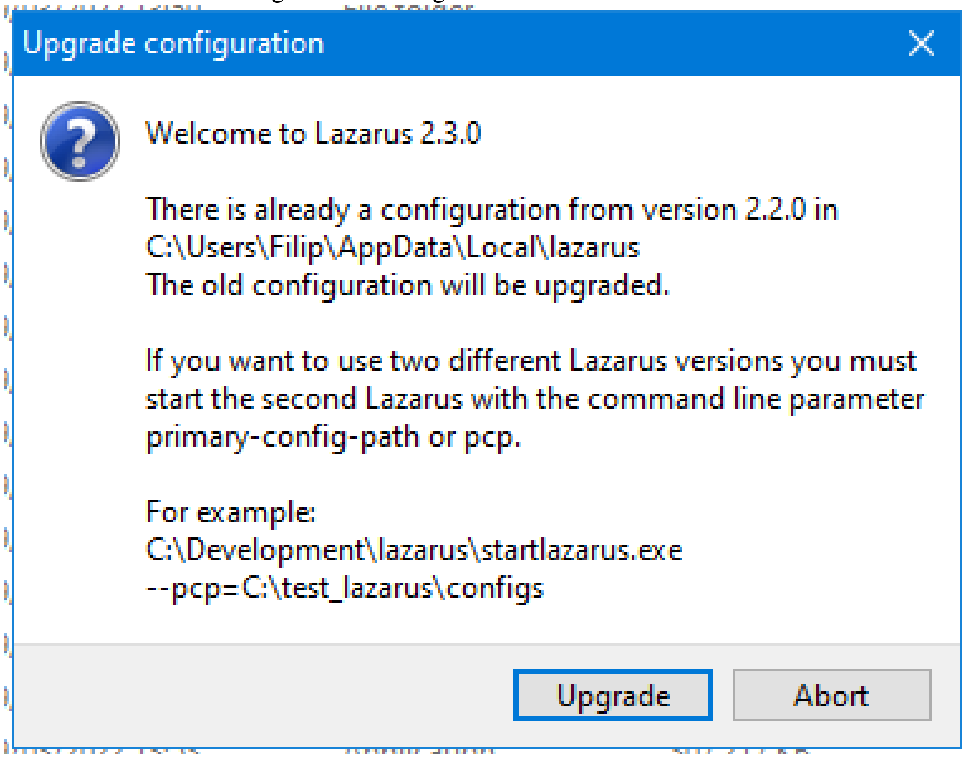
For example:
C:\Development\lazarus\startlazarus.exe --pcp=C:\test_lazarus\configs

Choose:

* Update info: Use this configuration and update it for being used with this Lazarus in future. The old installation will no longer use this.
* Ignore: Use this configuration but keep the warning. This may lead to conflicts with the other installation.
* Abort: Exit now. You can then fix the problem by starting this Lazarus with the correct configuration.

Additional information:
This configuration is at: C:\Users\Filip\AppData\Local\lazarus
It belongs to the Lazarus installation at: C:\lazarus\lazarus.exe
The current IDE was started from: C:\Development\lazarus\lazarus.exe

Figure 6: Creating a new confiuration or not


Upgrade configuration

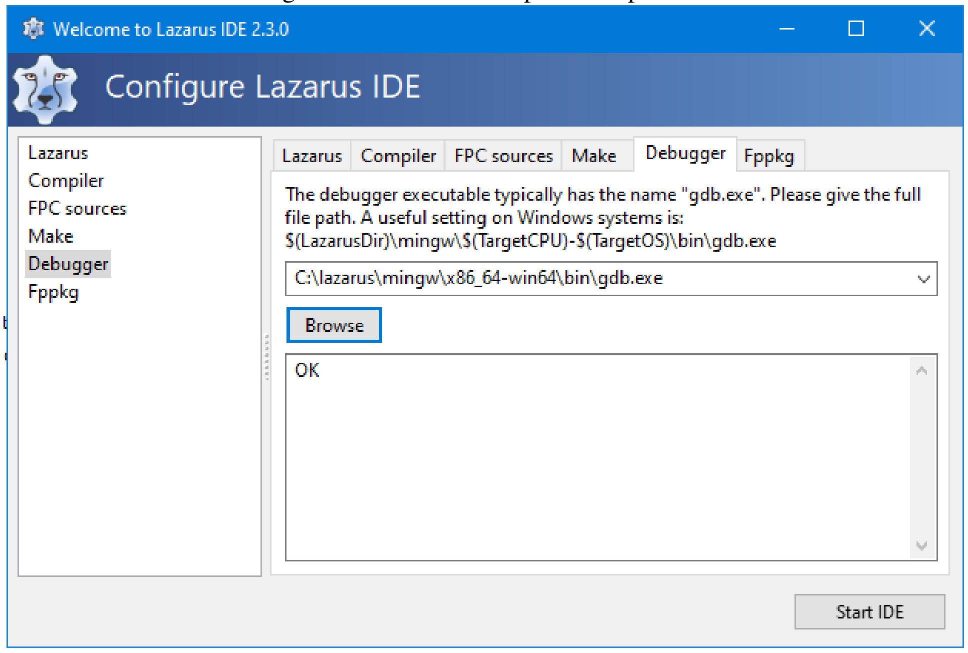
Welcome to Lazarus 2.3.0

There is already a configuration from version 2.2.0 in
C:\Users\Filip\AppData\Local\lazarus
The old configuration will be upgraded.

If you want to use two different Lazarus versions you must start the second Lazarus with the command line parameter primary-config-path or pcp.

For example:
C:\Development\lazarus\startlazarus.exe
--pcp=C:\test_lazarus\configs

Figure 7: Lazarus start-up check-up result



Welcome to Lazarus IDE 2.3.0

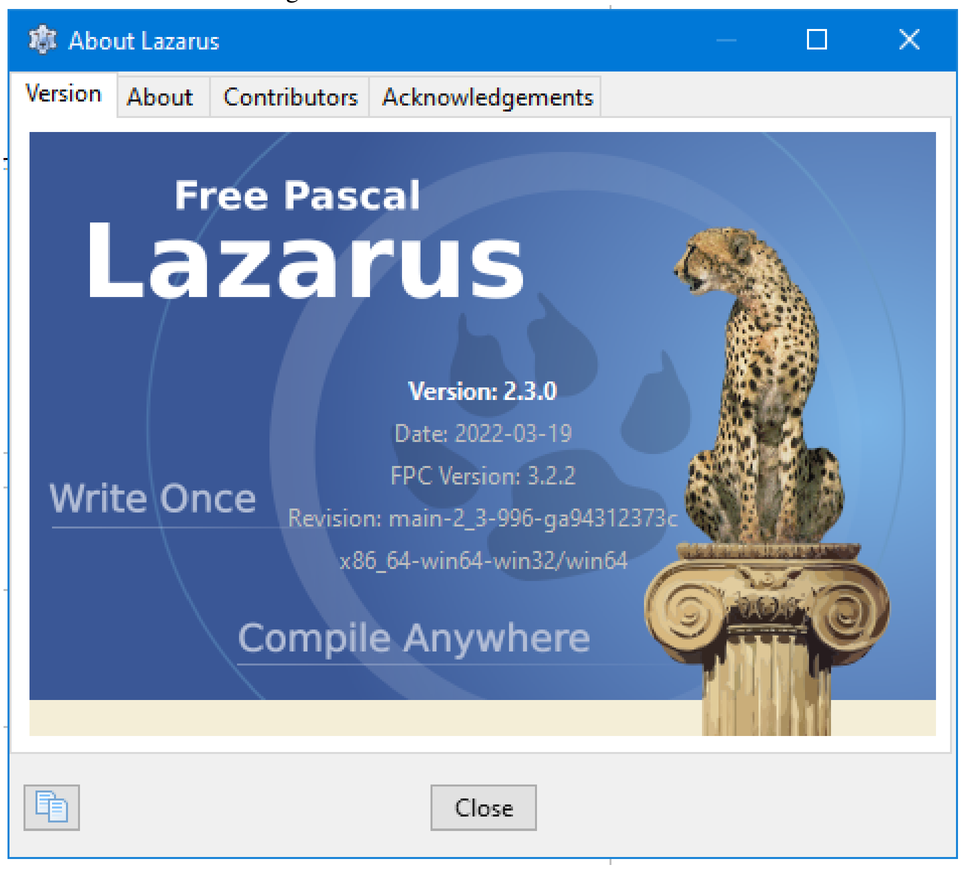
Configure Lazarus IDE

The debugger executable typically has the name "gdb.exe". Please give the full file path. A useful setting on Windows systems is: $(LazarusDir)\mingw\$(TargetCPU)-$(TargetOS)\bin\gdb.exe

C:\lazarus\mingw\x86_64-win64\bin\gdb.exe

OK

Figure 8: ’About Lazarus’ version check



About Lazarus

Version: 2.3.0
Date: 2022-03-19
FPC Version: 3.2.2
Revision: main-2_3-996-ga94312373c
x86_64-win64-win32/win64

6 conclusion

Lazarus is an open source tool. This means you do not have to wait for the latest version to be released. Instead, in this article we have demonstrated how you can build your own version of Lazarus: this should be within reach for every Object Pascal developer, regardless of the level of expertise...