

QUICK GUIDE FOR INSTALLATION AND CONFIGURATION OF FIZZBUZZ API PROJECT

This document lists the steps to install and configure the software tools that are required to start the project of FizzBuzz and among other aid for test and monitoring.

STEPS:

1. XAMPP INSTALLATION:

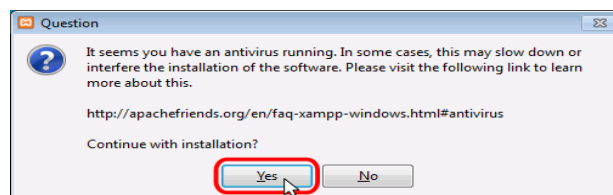
The main environment for viewing and interpretation of the endpoint requires to install the XAMPP program which, in turn, is responsible for installing the Apache Server, PHP among other tools of work. To start the process, it is necessary to go to the link <https://www.apachefriends.org/es/download.html> and download the version depending on the operating system and php you want to:



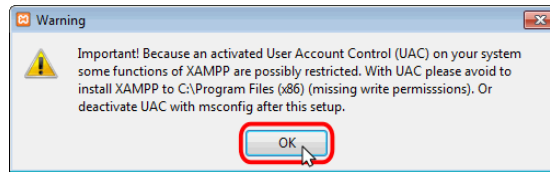
Once obtained the XAMPP installation file, should double click on it to start it. The images that are shown below correspond to the installation of XAMPP 7.0.9 in Windows 7 (from XAMPP 1.8.3 - published in July of 2013-, XAMPP not can install on Windows XP since PHP 5.5 and later can not be installed on Windows XP).

At the start, the XAMPP installer shows us two notices:

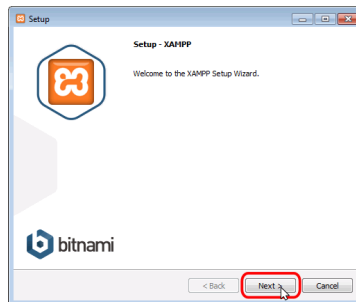
- ✓ The first appears if in the computer have been installed an antivirus:



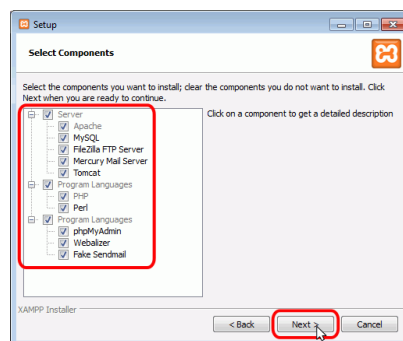
- ✓ The second appears if user account Control is enabled and remebers that some directories have restricted permissions.



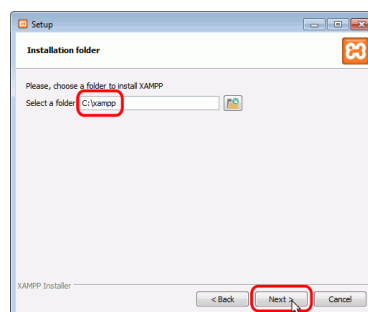
- ✓ Then the installation wizard starts. To continue, should click on the "Next" button:



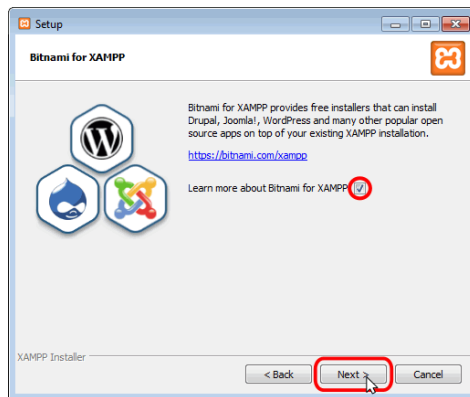
- ✓ The minimum components that install XAMPP are the Apache Server and the PHP language, but XAMPP also installs other items. On the component selection screen, you can choose the installation or not of these components.



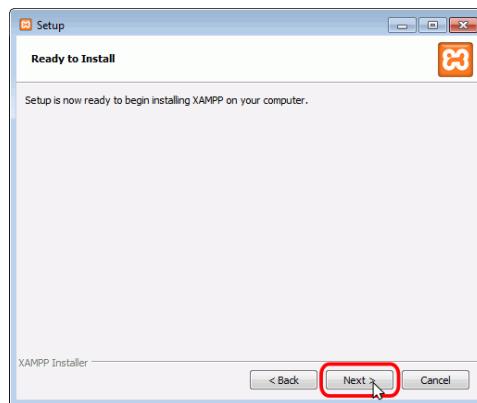
- ✓ On the next screen you can choose the folder of installation of XAMPP. The default installation folder is C:\xampp. If you want to change, you should click on the folder icon and select the folder where you want to install XAMPP. To continue the installation configuration, should click on the "Next" button.



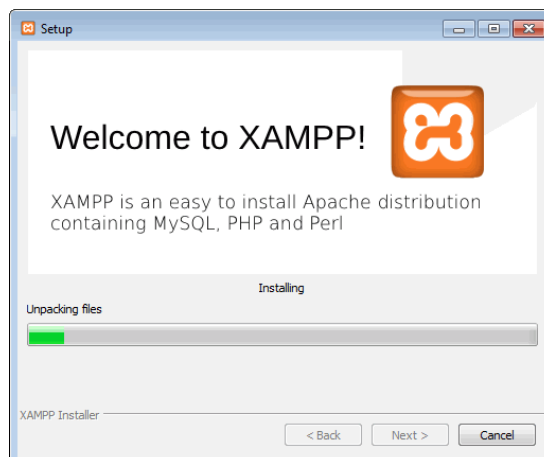
- ✓ The next screen shows information about applications for XAMPP installers created by Bitnami. Uncheck the checkbox corresponding to learn more about Bitnami.



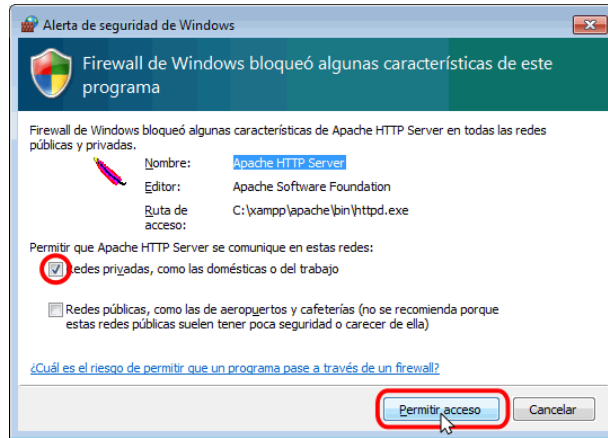
- ✓ To start XAMPP installation, should you click on the "Next" button on the next screen.



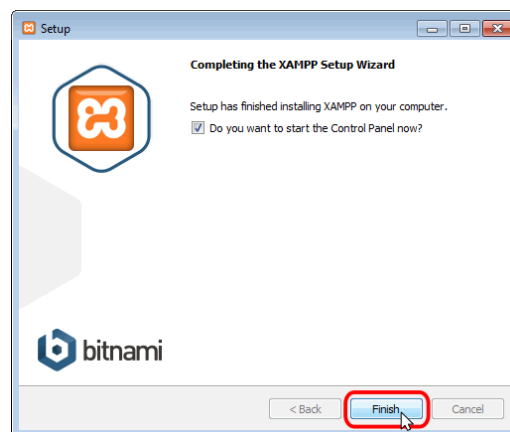
- ✓ Then begins the process of copying files, which can last a few minutes.



- ✓ During installation, if the computer not had installed Apache above, notice of the Windows Firewall will be displayed to allow Apache to communicate on home networks or work, which must be allowed by clicking on the "Permitir acceso" button.



- ✓ Once finished copying files, this confirms that XAMPP is installed. Click on the "Finish" button but not open the XAMPP control panel, unchecking the checkbox.



1.1 The XAMPP Control Panel(Open and close the control panel)

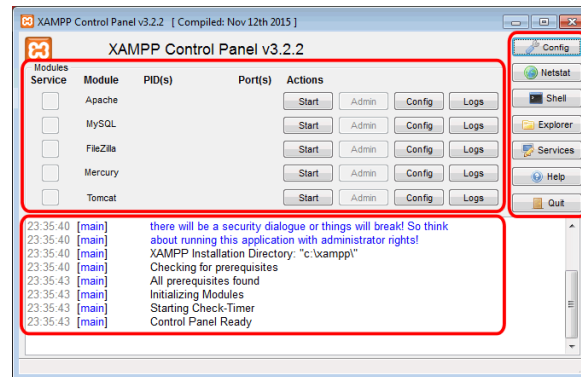
The XAMPP control panel can be accessed through the start menu "all programs > XAMPP > XAMPP Control Panel" or, if already started, using the icon in the notification area. Open and close the control panel.

- ✓ The first time that the XAMPP control panel opens, shows a window selection of language that allows to choose between English and German.

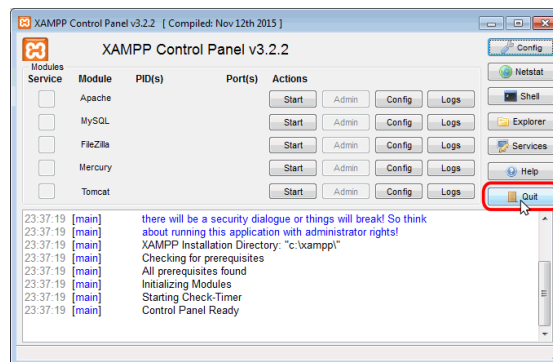


The XAMPP control panel is divided into three zones:

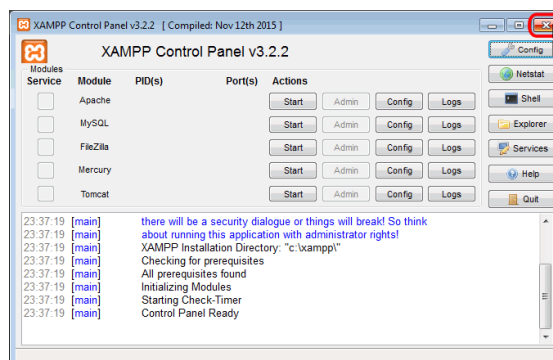
- ✓ The modules area that indicates each of the modules of XAMPP, indicates if the used port is installed as a service, the name, the process ID, and includes a few buttons to start and stop processes, manage, edit configuration files, and open the activity log file.
- ✓ The notification area where XAMPP reports the success or failure of actions undertaken.
- ✓ The area of utilities, for quick access.



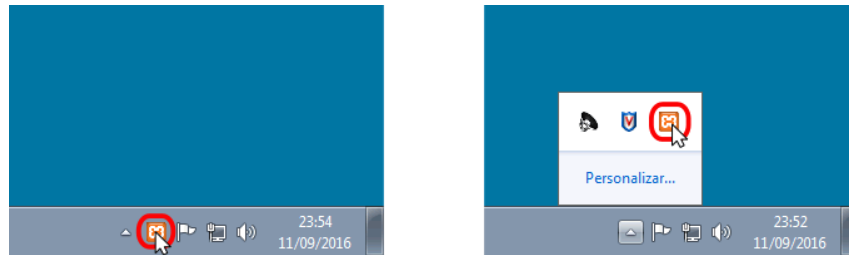
- ✓ To close the XAMPP control panel you should click on the Quit button (to close the control panel servers do not stop).



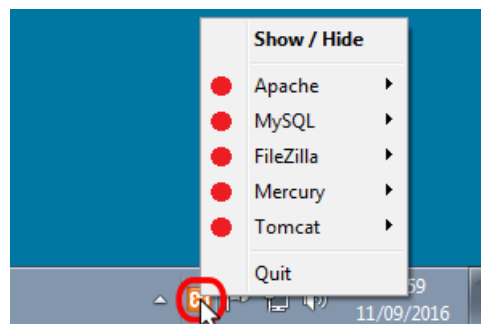
- ✓ The close blade-shaped button does not actually close the control panel, only minimizes it.



- ✓ If the XAMPP control panel is minimized, can be displayed by double-clicking on the XAMPP from the notification area icon.



- ✓ By right clicking on the XAMPP from the notification area icon displays a menu that allows you to show or hide the control panel, start or stop servers or close the control panel.

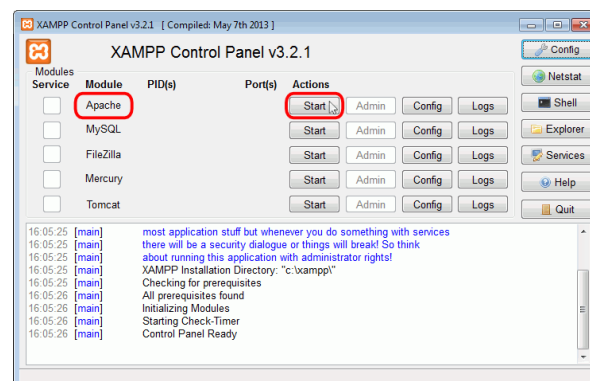


Several control panels can be opened simultaneously and any of them can start or stop servers, but it is not good to do so since it can lead to confusion (for example, to stop a server from a control panel, other control panels interpreted as an unexpected failure and would showed an error message).

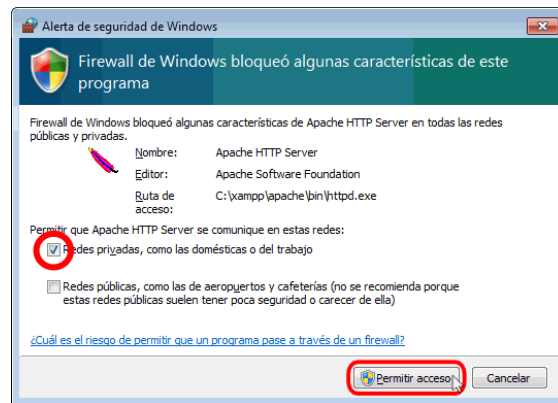
The Windows Firewall

When it gets underway for the first time any servers that installed XAMPP, the Windows Firewall prompts the user confirmation of authorization.

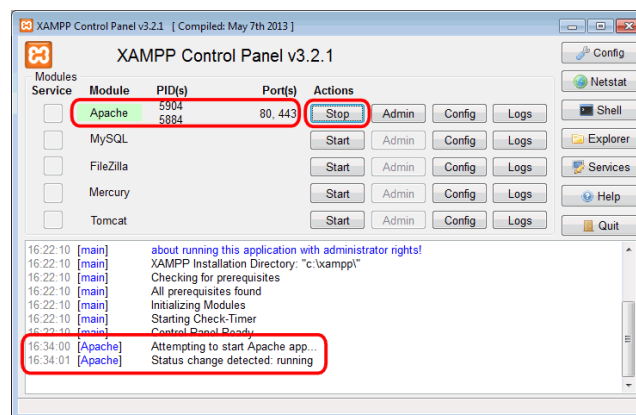
- ✓ For example, the first time that Apache is started by means of the corresponding Start button.



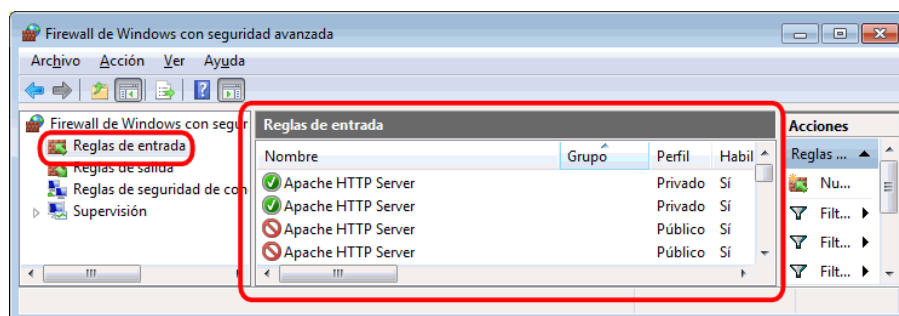
- ✓ When Apache open ports on your computer (for the first time), the Windows Firewall prompts the user confirmation. To use it you need at least authorize access in private networks:



- ✓ If Apache startup is successful, the control panel will show the name of the module with green background, its identifier of process, open ports (http and https), the 'Start' button will become the "Stop" button and in the notification area will be the result of the operations carried out.



- ✓ If you open the "Windows Firewall with advanced security" program, in the section on rules of entry may be new rules added.

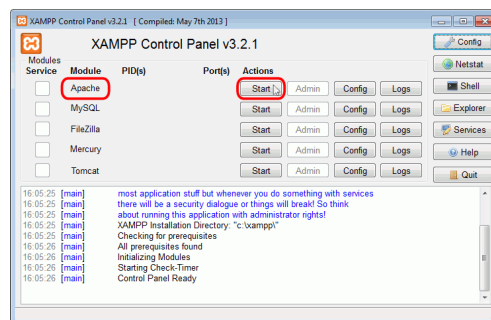


Start, stop, and restart servers

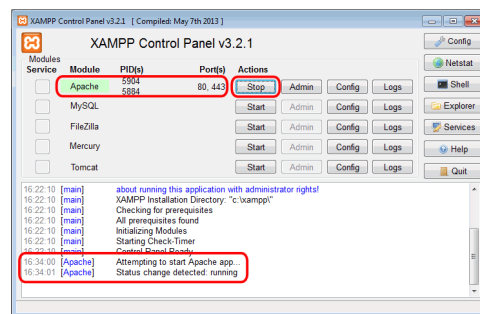
It is sometimes necessary to stop and restart the server. For example, Apache configuration files are loaded when you start Apache. If a configuration file of Apache (httpd.conf, php.ini or other) changes while Apache is running, it is necessary to stop and restart the Apache server to reload the configuration files.

Note: If you modify the configuration file we have introduced errors, the server will not be able to start. If we don't find the root of the problem, it is recommended to restore the original configuration files, of which is recommended to have a backup.

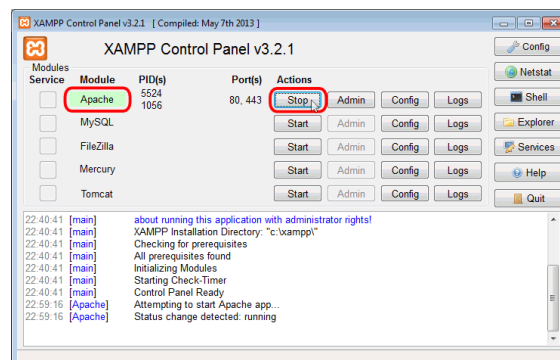
- ✓ To run the Apache (or another server), there is to click on the "Start" button:



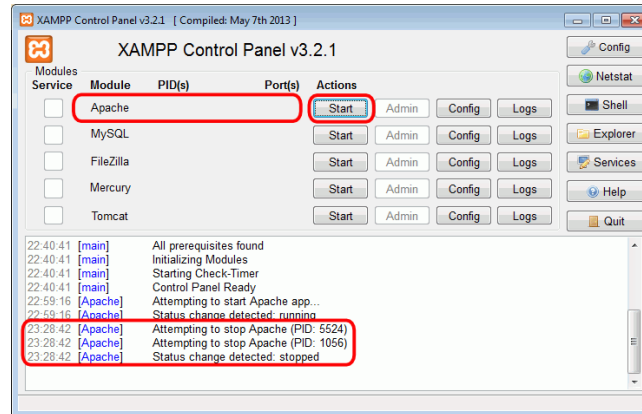
- ✓ If Apache startup is successful, the control panel will show the name of the module with green background, its identifier of process, open ports (http and https), the 'Start' button will turn into a "Stop" button and in the notification area will be the result of the operations carried out.



- ✓ To stop Apache should click on the button "Stop" from Apache.



- ✓ If Apache “stop” button is successful, the control panel will show the name of the module with gray background, without process identifier or open ports (http and https), the "Stop" button will become a 'Start' button and in the notification area will be the result of the operations carried out.

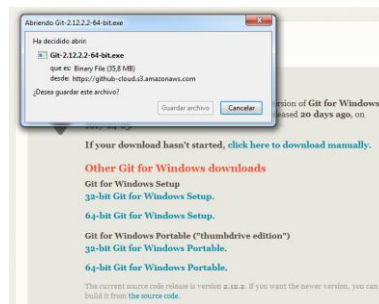


To restart Apache again would be to click on the button "Start" from Apache.

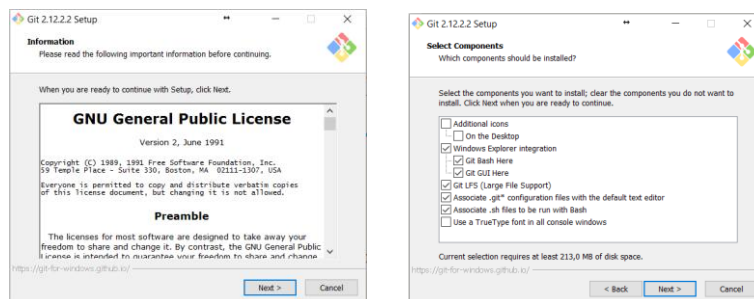
2. GIT INSTALLATION

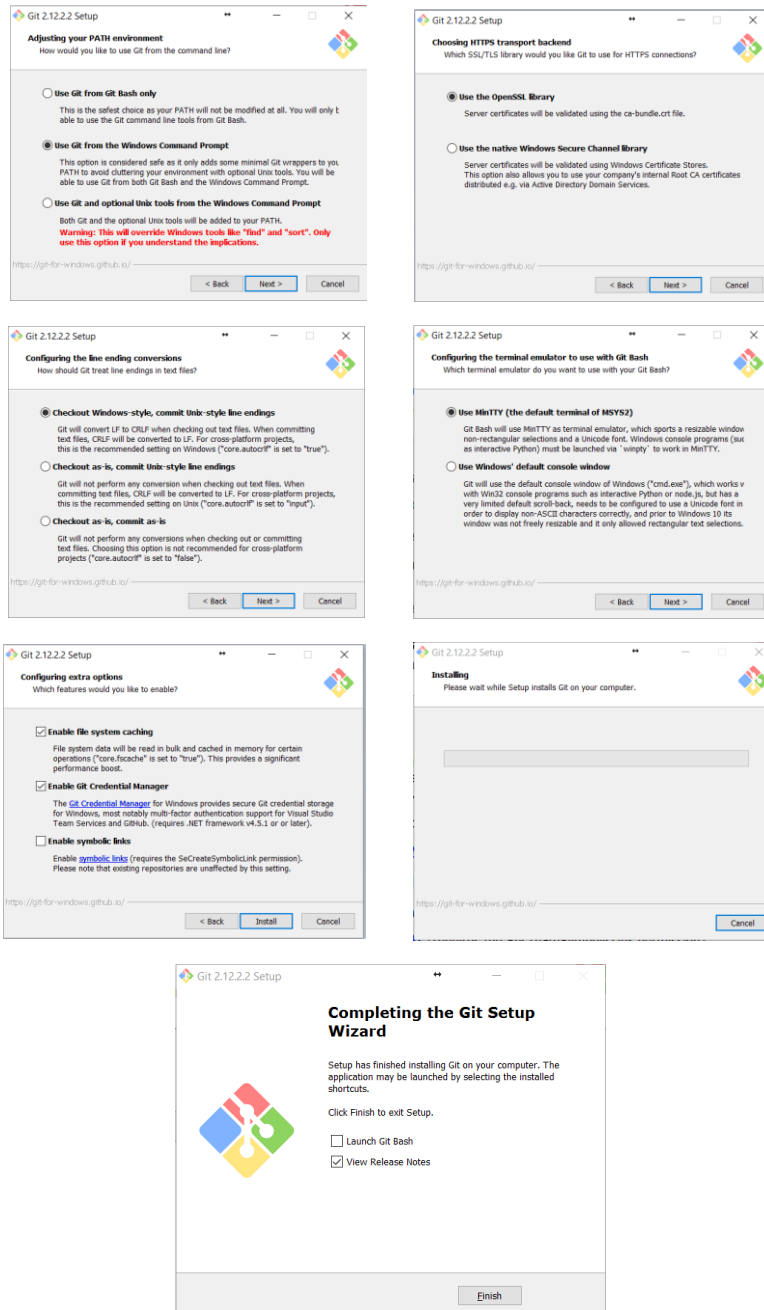
For the management of the resources of the project, the repository and the handling of command line console, it is necessary to install GIT according to the version from the following link:

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



- ✓ Then continue the installation process following the configuration by default:





3. COMPOSER INSTALLATION

Composer is the tool that allows download the project and libraries dependencies automatically and should go to the following link and select according to the version::

<https://getcomposer.org/download/>

✓ After downloading it, we follow the procedure as shown in the following DEMO:

<https://www.youtube.com/watch?v=X2LzAJbvuyY>

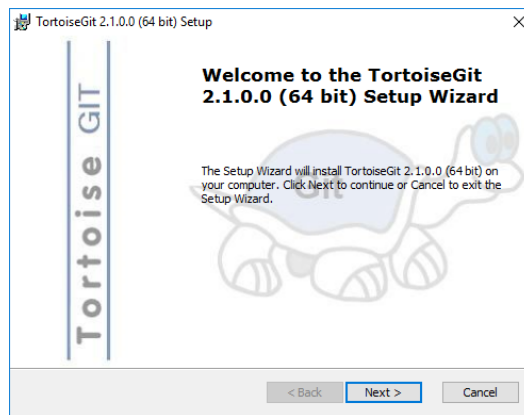
4. TORTOISE GIT INSTALLATION

TortoiseGit must be installed with the following tools to access the resources of the project:

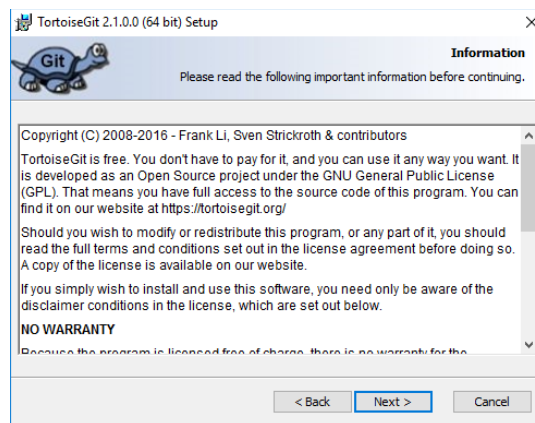
- ✓ Download and install GIT if that still does not have it.
- ✓ Download TortoiseGit binary file for version of OS:

<https://tortoisegit.org/download/>

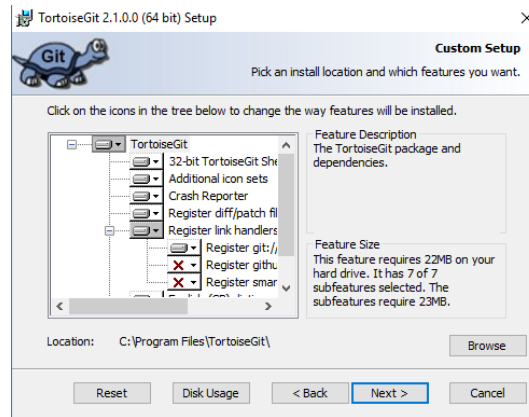
- ✓ Right click on the binary and run as administrator.



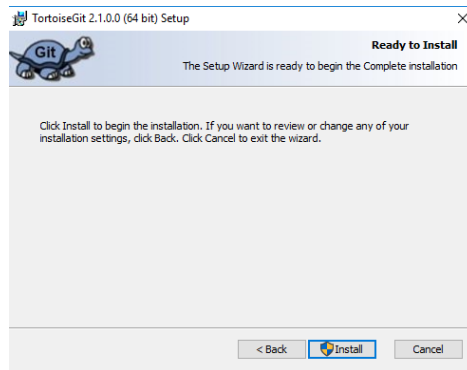
- ✓ Click on "next", then we will get one picture where we see the software license.



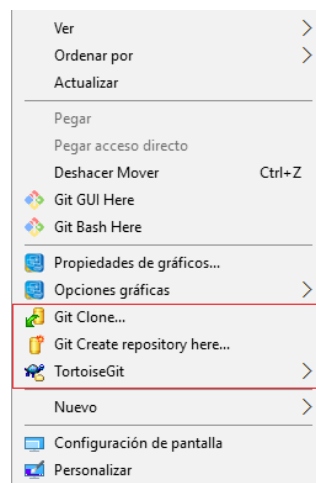
- ✓ Click on "next" and we will see what will be installed on your PC, this is the default setup of TortoiseGit. Please do not modify this setting.



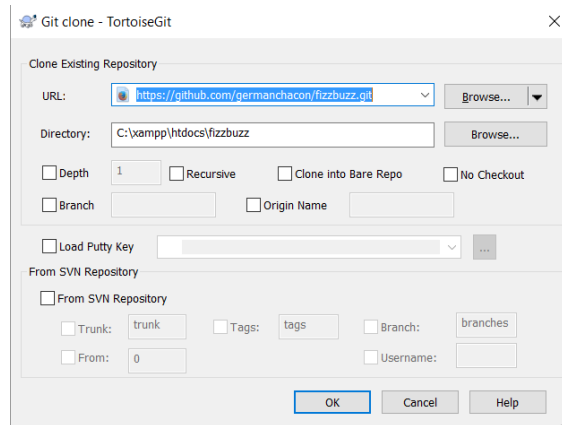
- ✓ Click on "next" and you will see another window, you click on install to start the installation.



- ✓ Click on "finish" and you have already installed TortoiseGit.
- ✓ Shown as this tool is integrated into the system, to right clicking, three new items appear in the menu of options: TortoiseGit and Git Clone, Git Create.



- ✓ After that, you must clone previously on GitHub repository and to do so you must select the option of git Clone, gets the address of github project: <https://github.com/germanchacon/fizzbuzz>, choose the directory within the apache server where we want it to deploy and click ok.

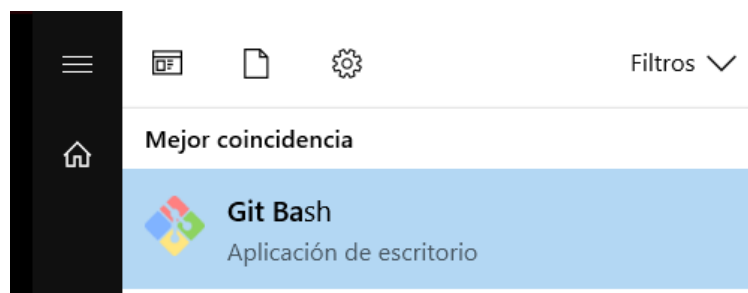


- ✓ After this process is finished, a group of folders with the following structure of the resources of the project should be deployed:

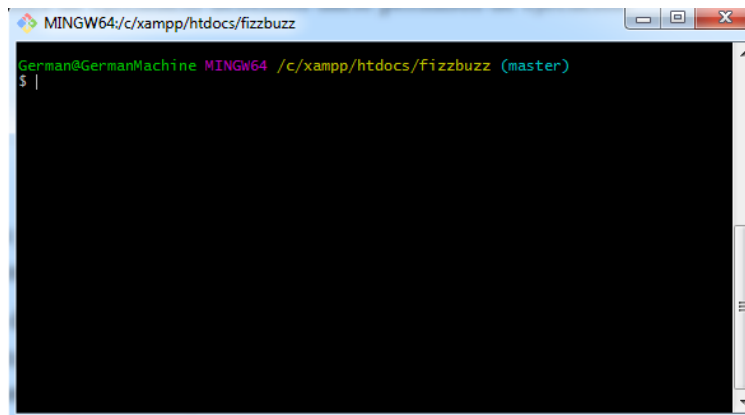
Nombre	Fecha de modifica...	Tipo	Tamaño
.git	25/04/2017 1:24 a....	Carpeta de archivos	
bin	25/04/2017 1:24 a....	Carpeta de archivos	
UnitTest	25/04/2017 1:24 a....	Carpeta de archivos	
composer.json	25/04/2017 1:24 a....	JSON File	1 KB
index.php	25/04/2017 1:24 a....	PHP File	2 KB
phpunit.xml	25/04/2017 1:24 a....	Documento XML	1 KB

5. DOWNLOAD AND INSTALLATION OF DEPENDENCIES WITH COMPOSER

Now we must go to Git Bash program to perform the installation of dependencies through composer.



- ✓ After open Git Bash, the following window appears:



We will enter the apache directory and download the repository using the "cd" command:

For this project: "cd /c/xampp/htdocs/fizzbuzz".

- ✓ After being located in this route, it is necessary to execute the command "composer install" this process must install all the dependences on the project:

```
$ composer install
Loading composer repositories with package information
Updating dependencies (including require-dev)
Package operations: 28 installs, 0 updates, 0 removals
- Installing symfony/yaml (v3.2.7): Loading from cache
- Installing sebastian/version (2.0.1): Loading from cache
- Installing sebastian/resource-operations (1.0.0): Loading from cache
- Installing sebastian/recursion-context (2.0.0): Loading from cache
- Installing sebastian/object-enumerator (2.0.1): Loading from cache
- Installing sebastian/global-state (1.1.1): Loading from cache
- Installing sebastian/exporter (2.0.0): Loading from cache
- Installing sebastian/environment (2.0.0): Loading from cache
- Installing sebastian/diff (1.4.1): Loading from cache
- Installing sebastian/comparator (1.2.4): Loading from cache
- Installing doctrine/instantiator (1.0.5): Loading from cache
- Installing phpunit/php-text-template (1.2.1): Loading from cache
- Installing phpunit/mock-objects (3.4.3): Loading from cache
- Installing phpunit/php-timer (1.0.9): Loading from cache
- Installing phpunit/php-file-iterator (1.4.2): Loading from cache
- Installing sebastian/code-unit-reverse-lookup (1.0.1): Loading from cache
- Installing phpunit/php-token-stream (1.4.11): Loading from cache
- Installing phpunit/php-code-coverage (4.0.8): Loading from cache
- Installing webmozart/assert (1.2.0): Loading from cache
- Installing phpdocumentor/reflection-common (1.0): Loading from cache
- Installing phpdocumentor/type-resolver (0.2.1): Loading from cache
- Installing phpdocumentor/reflection-docblock (3.1.1): Loading from cache
- Installing phpspec/prophecy (v1.7.0): Loading from cache
- Installing myclabs/deep-copy (1.6.0): Loading from cache
- Installing phpunit/phpunit (5.7.19): Loading from cache
- Installing psr/log (1.0.2): Loading from cache
- Installing monolog/monolog (1.22.1): Loading from cache
- Installing slim/slim (2.6.9): Loading from cache
symfony/yaml suggests installing symfony/console (For validating YAML files using the lint command)
sebastian/global-state suggests installing ext-uopz (*)
phpunit/phpunit-mock-objects suggests installing ext-soap (*)
phpunit/phpunit suggests installing phpunit/php-invoker (~2.1)
monolog/monolog suggests installing aws/aws-sdk-php (Allow sending log messages to AWS services like DynamoDB)
monolog/monolog suggests installing doctrine/couchdb (Allow sending log messages to a CouchDB server)
monolog/monolog suggests installing ext-mongo (Allow sending log messages to a MongoDB server)
monolog/monolog suggests installing graylog2/gelf-php (Allow sending log messages to a Graylog2 server)
monolog/monolog suggests installing mongodb/mongodb (Allow sending log messages to a MongoDB server via PHP Driver)
monolog/monolog suggests installing php-amqp/php-amqp (Allow sending log messages to an AMQP server using php-amqp)
monolog/monolog suggests installing php-console/php-console (Allow sending log messages to Google Chrome)
monolog/monolog suggests installing rollbar/rollbar (Allow sending log messages to Rollbar)
monolog/monolog suggests installing ruflin/elastica (Allow sending log messages to an Elastic Search server)
monolog/monolog suggests installing sentry/sentry (Allow sending log messages to a Sentry server)
slim/slim suggests installing phpsclib/mcrypt_compat (Polyfill for mcrypt extension)
Writing lock file
Generating autoload files
```

6. IMPLEMENTING THE PROGRAM

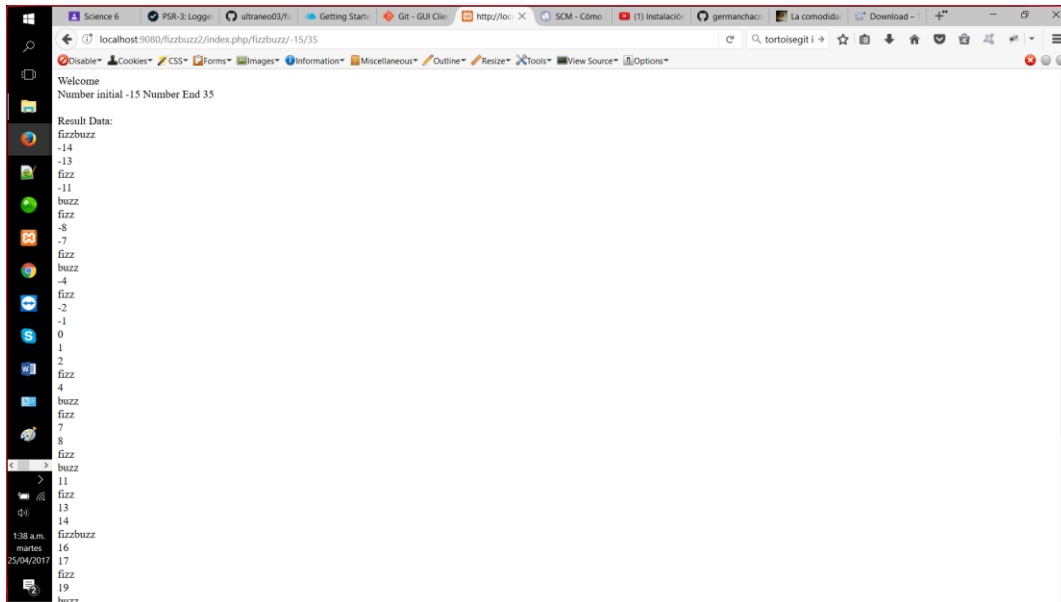
After the download of the dependencies and libraries in the project, the solution is now ready for use and should be accessed by means of a web browser from the following link:

<http://{direccionServidorLocal}/fizzbuzz/index.php/fizzbuzz/-15/25>

- ✓ Example:

<http://localhost:9080/fizzbuzz/index.php/fizzbuzz/-15/35>

✓ **Result:**



7. WORKING WITH UNIT TEST

Then when we already have in operation the program and its result, you can proceed with the implementation of the test case by going to the folder:

{directorioDelProyecto}/UnitTest/FizzBuzzTest.php.

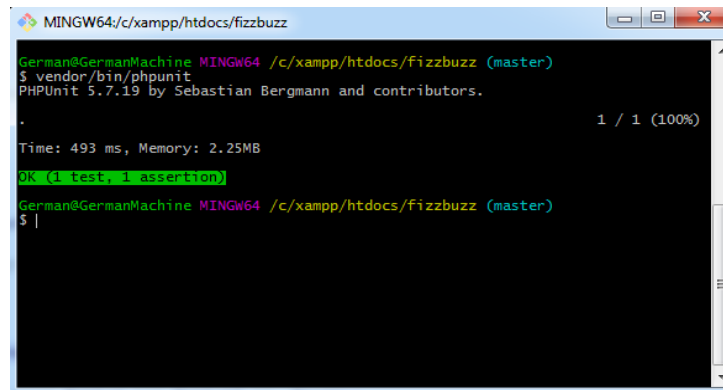
- ✓ In this file we go to the following function to setup the test.

```
public function testFizzBuzz($value = 10, $value2= 45, $port = 9080)
{
    $data = $this->get('/fizzbuzz/'.$value.'/'.$value2.', $port);

    $condition = true;
    if (strpos($data, 'Welcome') !== false) {
        $this->assertTrue(true, $data);
    } else {
        // Return STDOUT
        $this->assertTrue(false, $data);
    }
}
```

This function sets the initial value with \$value, the final value with \$value2 and the port of apache with \$port.

- ✓ After setting up the data with which we are going to realize the test, we return to the console Git Bash and execute the command "[vendor/bin/phpunit](#)", if the information is well built, it will show us an answer as the following one:

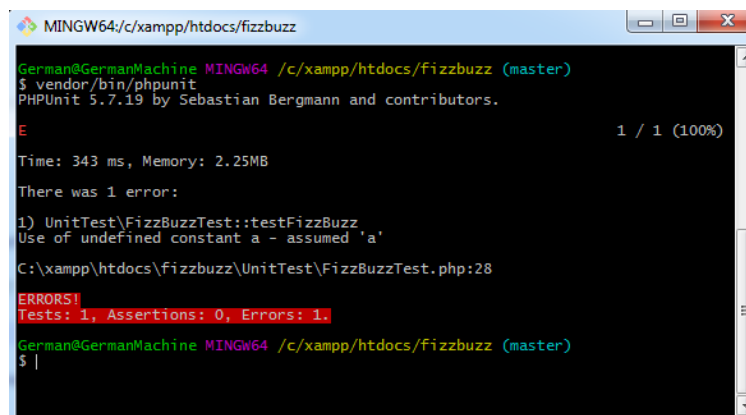


```
MINGW64/c/xampp/htdocs/fizzbuzz
German@GermanMachine MINGW64 /c/xampp/htdocs/fizzbuzz (master)
$ vendor/bin/phpunit
PHPUnit 5.7.19 by Sebastian Bergmann and contributors.

.
1 / 1 (100%)

Time: 493 ms, Memory: 2.25MB
OK (1 test, 1 assertion)
German@GermanMachine MINGW64 /c/xampp/htdocs/fizzbuzz (master)
$ |
```

✓ If you set up non-numerical data, or invalid data, we will have a response of this kind:



```
MINGW64/c/xampp/htdocs/fizzbuzz
German@GermanMachine MINGW64 /c/xampp/htdocs/fizzbuzz (master)
$ vendor/bin/phpunit
PHPUnit 5.7.19 by Sebastian Bergmann and contributors.

E
1 / 1 (100%)

Time: 343 ms, Memory: 2.25MB
There was 1 error:

1) UnitTest\FizzBuzzTest::testFizzBuzz
Use of undefined constant a - assumed 'a'

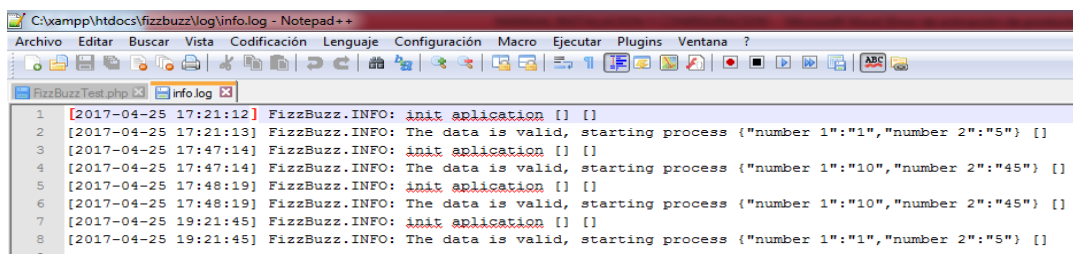
C:\xampp\htdocs\fizzbuzz\UnitTest\FizzBuzzTest.php:28

ERRORS!
Tests: 1, Assertions: 0, Errors: 1.
German@GermanMachine MINGW64 /c/xampp/htdocs/fizzbuzz (master)
$ |
```

8. TESTING LOGS AND TRACKING

For more details regarding the operation of the program concerning data and errors, you can go to the route `{ProjectPath}\xampp\htdocs\fizzbuzz\log\info` and edit the file `info.txt` that contains the log:

✓ Info.log



```
C:\xampp\htdocs\fizzbuzz\log\info.log - Notepad++
Archivo  Editar  Buscar  Vista  Codificación  Lenguaje  Configuración  Macro  Ejecutar  Plugins  Ventana  ?
1 [2017-04-25 17:21:12] FizzBuzz.INFO: init aplicación [] []
2 [2017-04-25 17:21:13] FizzBuzz.INFO: The data is valid, starting process {"number 1":"1","number 2":"5"} []
3 [2017-04-25 17:47:14] FizzBuzz.INFO: init aplicación [] []
4 [2017-04-25 17:47:14] FizzBuzz.INFO: The data is valid, starting process {"number 1":"10","number 2":"45"} []
5 [2017-04-25 17:48:19] FizzBuzz.INFO: init aplicación [] []
6 [2017-04-25 17:48:19] FizzBuzz.INFO: The data is valid, starting process {"number 1":"10","number 2":"45"} []
7 [2017-04-25 19:21:45] FizzBuzz.INFO: init aplicación [] []
8 [2017-04-25 19:21:45] FizzBuzz.INFO: The data is valid, starting process {"number 1":"1","number 2":"5"} []
9
```

9. REFERENCES

For more information, please visit the following links:

- ✓ Slim Framework-> <https://docs.slimframework.com/>
- ✓ Standars PHP-> <http://www.php-fig.org/psr/>
- ✓ PHP Unit-> <https://www.cloudways.com/blog/getting-started-with-unit-testing-php/>
- ✓ Monolog-> <https://github.com/Seldaek/monolog>