WordPress - Setup WordPress

Connecting to Your Server

Using PuTTY

Open PuTTY and enter your domain name in the box named **Host name (or IP address)** and enter the port number used to connect to SSH under **Port**, and then click **Open**. You can even save your site settings by entering a name in the **Saved Sessions** box and by pressing **Save**. Next time, you can always load the session by selecting your site and clicking **Load**.

PuTTY will now ask for your username. Enter your username and press **Enter**. Now you will be asked for your password. Note here that while you are typing your password, you won't see it being typed on the screen. It's hidden for security reasons. Press **Enter** after you've typed your password, and you will be logged on.

Using Any Other SSH Client or Mac Terminal

Enter the following command in your Terminal client to connect to your site's command-line over SSH:

```
ssh username@domain.com -p 22
```

The `-p` switch tells it to use port number 22. If your host allows SSH over default port 22, you can omit `-p` and `22` in the above command; otherwise, substitute 22 for your host's SSH port number. After logging in, you will see something like this:

```
domain.com@username:-$
```

That's the shell command prompt where you will be typing all your commands from now on.

2. Downloading WordPress

Now that we have logged in to our SSH server, we need to go to the correct directory where we want to set up our blog. Then we download the files and extract them there. Let's say the directory you want your blog to be installed under is **blogdemo** residing under the **public_html** directory. In that case, you will use the following command:

```
cd public_html/blogdemo/
```

Now that we have reached the correct directory, we will download WordPress using the `wget` command.

```
wget http://wordpress.org/latest.tar.gz
```

```
$tar xfz latest.tar.gz
```

The above command downloads the latest WordPress install from their server and extracts the file from it into the **blogdemo** directory. `x`, `f`, and `z` are parameters which tell the tar command to extract the contents from the specified file using gzip.

Now, after extraction, you will find a **wordpress** directory under the **blogdemo** directory containing your install. So to shift the files back to where they should be, use the following commands:

```
mv wordpress/* ./
```

This command moves the contents of the **wordpress** directory into the current directory. Anytime you want to check what's in the current directory, type `ls`.

If you want, you can use the following commands to delete both the **wordpress** directory and the archive file you downloaded:

```
rmdir ./wordpress/
```

```
rm -f latest.tar.gz
```

File Permissions

Correct file permissions from the root folder

When you setup WP you (the webserver) may need write access to the files. So the access rights may need to be loose.

```
chown www-data:www-data  -R * # Let Apache be owner
find . -type d -exec chmod 755 {} \;  # Change directory permissions rwxr-xr-x
find . -type f -exec chmod 644 {} \;  # Change file permissions rw-r--r--
```

After the setup you *should* tighten the access rights, according to Hardening WordPress all files except for wp-content should be writable by your user account only. wp-content must be writable by *www-data* too.

```
chown <username>:<username>  -R * # Let your useraccount be owner
chown www-data:www-data wp-content # Let apache be owner of wp-content
```

Maybe you want to change the contents in wp-content later on. In this case you could

- temporarily change to the user to *www-data* with `su`,
- give wp-content group write access 775 and join the group *www-data* or
- give your user the access rights to the folder using ACLs.

Whatever you do, make sure the files have rw permissions for *www-data*.

3. Installing WordPress

In this step, we will create the database and corresponding user and associate them together. Then we will use the famous five-minute install of WordPress to finish it off.

Note: Before moving ahead, you will need to check whether you have got the privileges to create a database or not. An easy way to check is to go to your phpMyAdmin and check whether you can create a database from there or not. If you can't, that means you won't be able to follow this step. You should check with your web host if they allow you to do so or not. Most shared web hosts will let you create a database.

First, you need to log in to the MySQL command-line using the following command:

```
mysql -u username -p
```

After entering this, you will be asked for your MySQL password. Type your password and you will be shown a screen like this:



Now that we have logged in to the MySQL Server, we will first create a database and will grant the user access to that database. Use the following commands now:

```
create database dbname;
```

```
create user 'username'@'localhost' identified by 'password';
```

```
grant all privileges on dbname.* to username@localhost;
```

Don't forget the semi-colon at the end of each MySQL command. The first command creates the database. The second command allows the user to connect to the database. The final command grants all privileges to the user for that database. You can test whether your database creation was successful by running this command:

```
use dbname;
```

It should say "database changed". Now you can exit the MySQL command-line by typing `exit`.

Now fire up the blog in your browser and run the usual WordPress install, and use the database information we used in the third step to set up your **wp-config.php** and then set up your blog.

Note: New Database User

In our tutorial, we are using an existing database user to connect to the new database. But if you want a separate user for each database, you need to create a new user to access that database. Here's how you should do it.

Once you are inside the MySQL shell, use the following commands to create a new user and set its password.

```
create user 'dbusername'@'localhost' identified by 'password';
```

Now go back to Step 3 and run all other commands with this username.

Editing wp-config.php

In our tutorial, I have told you that after doing everything on the shell, you can directly proceed to the install. But some of you might want to edit **wp-config.php** to add special settings and code. You can only do that via the shell. While you are in your blog directory at the shell, use the following command to fire up the Vim Editor (a command-line shell file editor)

```
vi ./wp-config.php
```

Now you will see something like what's shown below:

Vim Editor
Image not found or type unknown

Press the `i` key to enter insert mode, and use the arrow keys to move around the file. Once you have made your edits, press the **Esc** key to exit insert mode. To exit Vim, type `:` and then type `wq` and press **Enter**. This will save your changes and quit Vim.

Increasing Import File Size

Create a .user.ini in the root folder of the WordPress site

```
upload_max_filesize = 500M
post_max_size = 500M
memory_limit = 500M
max_execution_time = 600
```

Or you can change the following in the wp-content/plugins/all-in-one-wp-migration/constants.php

```
define( 'AI1WM_MAX_FILE_SIZE', 2 << 28 );
```

To the following

```
define( 'AI1WM_MAX_FILE_SIZE', 2 << 500 );
```

All in one Import for restore after the file is on the root/wp-content folder and does not require a paid version.
Original

all-in-one-wp-migration-6.7.zip

Forked direct link

All-In-One-WP-Migration-With-Import-master.zip

Download and Install WordPress With the WP-CLI Tool

In this section, I'll show you an even better way to download and install WordPress: with the WP-CLI tool. First, we have to install the WP-CLI tool on the server.

How to Install the WP-CLI Tool

Run the following commands on your server to download, install, and configure the WP-CLI tool.

```
curl -O https://raw.githubusercontent.com/wp-cli/builds/gh-pages/phar/wp-cli.phar
```

```
chmod +x wp-cli.phar
```

```
sudo mv wp-cli.phar /usr/local/bin/wp-cli
```

Let's check if the WP-CLI tool is installed successfully by using the following command.

```
wp-cli --info
```

You should see something like this:

```
OS:     Linux 4.15.0-91-generic #92-Ubuntu SMP Fri Feb 28 11:09:48 UTC 2020 x86_64
```

```
Shell:
/bin/bash
```

```
PHP binary:
/usr/bin/php7.2
```

```
PHP version:
7.2.24-0ubuntu0.18.04.3
```

```
php.ini used:
/etc/php/7.2/cli/php.ini
```

```
WP-CLI root dir:
phar://wp-cli.phar/vendor/wp-cli/wp-cli
```

```
WP-CLI vendor dir:
phar://wp-cli.phar/vendor
```

```
WP_CLI phar path:
/etc/init.d
```

```
WP-CLI packages dir:
```

```
WP-CLI global config:
```

```
WP-CLI project config:
```

```
WP-CLI version:
2.4.0
```

Download and Install WordPress

Let's download the latest version of WordPress first.

```
wp-cli core download
```

If the download is successful, you'll see something like the following:

```
Downloading WordPress 5.5.1 (en_US)...
```

```
md5 hash verified: 72c6f56b4818ffd0e6e6a4ed8f3e8d4e
```

```
Success: WordPress downloaded.
```

So we've downloaded the WordPress codebase now.

Next, it's time to create the **wp-config.php** file. We can do it with the help of the following command. Replace the placeholders with the actual values. I assume that you've already created the database which you would like to use with WordPress.

```
$wp-cli config create --dbname=YOUR_DB_NAME --dbuser=YOUR_DB_USERNAME --dbpass=YOUR_DB_PASSWORD
```

```
Success: Generated 'wp-config.php' file.
```

Finally, let's run the following command, which installs WordPress.

```
$wp-cli core install --url=YOUR_DOMAIN_NAME --title=YOUR_BLOG_TITLE --admin_user=ADMIN_USERNAME --admi
```

```
Success: WordPress installed successfully.
```

And with that, WordPress is installed successfully on your server!

In fact, the WP-CLI tool is capable of doing a lot more than just installation. It allows you to manage plugins and themes and do any necessary version updates as well.

Revision #1
Created 11 February 2026 13:20:52 by Admin
Updated 24 October 2026 14:12:36 by Admin