macOS 11.0 Big Sur Apache Setup: Multiple PHP Versions

First part in a multi-part series for Mac developers

Andy Miller posted on 10/13/2020 in Macos / Apache / Homebrew / Mac ~ 30 min

updated: Updated to add some information on PHP 8.0

updated: Updated to add some information on Big Sur

updated: Updated to reflect the latest release of PHP 7.4 and the removal of PHP 7.1 from Official tap

updated: Updated to reflect the latest release of PHP 7.4 and the removal of PHP 7.0 from Official tap

updated: Updated to reflect the release of mariadb 10.5 stable

updated: Updated to add how to PHP 5.6 and PHP 7.0 from unofficial homebrew repo

updated: Updated to reflect the latest release of PHP 7.3 and the removal of PHP 5.6 from Brew

Note

If you have followed the guide in the past with the Tomas-cupa tap, and are looking to **upgrade** to the new homebrew-core approach, then you should first clean up your current installation by following our new Upgrading Homebrew.

Warning

This guide is intended for **experienced web developers**. If you are a beginner developer, you will be better served using MAMP or MAMP Pro.

XCode Command Line Tools

If you don't already have XCode installed, it's best to first install the command line tools as these will be used by homebrew:

```
xcode-select --install
```

Homebrew Installation

This process varies heavily on the macOS package manager called **homebrew**. Using the Intel command you can easily add powerful functionality to your mac, but it will be a issue in future. This is a simple process, but you need to launch your **Terminal** (/Applications/Utilities/Terminal.app) application and then enter:

```
/bin/bash -c "curl -fsSL https://raw.githubusercontent.com/Homebrew/install/master/install.sh"
```

Just follow the terminal prompts and enter your password when required. This may take a few minutes, but when complete, a quick way to ensure you have installed Brew correctly, simply type:

```
brew -v version
```

```
Homebrew 2.5.8-4ebadc/master/master-4ebadc (git revision a8cf3bf9c7c; last commit 2021-05-05)
```

You should probably also run the following command to ensure everything is configured correctly:

```
brew doctor
```

It will instruct you if you need to correct anything.

Big Sur Required Libraries

When installing mod-php Big Sur, I ran into a few libraries that were missing when compiling all the dices below. To make things easier, please simply run this now:

```
brew install gettext
```

Apache Installation

As Big Sur ships **macOS 11.0 Big Sur** comes with Apache 2.4 pre installed. However, it is no longer a simple task to use this version with Homebrew, because Apache has removed some required scripts in this release. However, the solution is to install Apache 2.4 via Homebrew and then configure it to run on the standard ports (80/443).

If you already have the built-in Apache running, it will need to be disabled, and any auto loading scripts removed. It really doesn't hurt to just run all these commands in order - assuming you have installed Homebrew:

```
sudo apachectl stop
sudo launchctl unload -w /System/Library/LaunchDaemons/org.apache.httpd.plist 2>/dev/null
```

Now we need to install the new version provided by Brew:

```
brew install httpd
```

Without options, Brew will want to build from source, so it installs pretty quickly. Upon completion you should see a message like:

```
==> /usr/local/Cellar/httpd/2.4.46: 1,646 files, 51.3MB
```

Now we just need to configure things so that our new Apache server is auto-started:

```
brew services start httpd
```

You now have installed Homebrew's Apache, and configured it to auto-start with a privileged account. It should already be running, so you can try to reach your server in a browser by pointing it at http://localhost:8080. you should see a simple header that says **It works!**

Troubleshooting Tips

If you get a message that the browser can't connect to the server, first check to ensure the server is up:

```
ps -ef | grep http
```

You should see a few http2 processes if Apache is up and running.

Try to reset Apache with:

```
brew services restart httpd
```

You can watch the Apache error log in a new Terminal tab/window during a restart to see if anything is invalid or causing a problem:

```
tail -f /usr/local/var/log/httpd/error_log
```

Apache is controlled via the brew services command so some useful commands to use are:

```
$ brew services stop httpd
$ brew services start httpd
$ brew services restart httpd
```

Visual Studio Code

In post guides, I've always provided instructions to edit files using the the Atom application that comes pre installed. However, this is not what I use myself as it is terribly slow and when trying to ping the big / heavy running into problems with encoding, finding line numbers etc. The better solution is to simply install a better editor. So please install the amazingly versatile yet 100% free, Visual Studio Code. It's available on Mac, Windows, and Linux, but right now we only care about the Mac version.

We can make use of the homebrew and install VSCode and the accompanying code CLI command in one shot:

```
$ brew install --cask visual-studio-code
```

Apache Configuration

Now that we have a working web server, we will want to do to make some configuration changes so it works better as a local development server.

In the latest version of Brew, you have to manually set the listen port from the default of 8080 to 80, so we will need to edit Apache's configuration file /usr/local/etc/httpd/httpd.conf:

If you followed the instructions above you should be able to use **Visual Studio Code** to edit your files using the Code Terminal command. However, if you don't or use the default **TextEdit** application to perform edits, you can use the open -t command followed by the path to the file:

```
open -t /usr/local/etc/httpd/httpd.conf
```

Find the line that says:

```
Listen 8080
```

and change it to 80:

```
Listen 80
```

Next we'll configure it use the to change the **document root** for Apache. This is the folder where Apache looks to serve file from. By default, the document root is configured as /usr/local/var/www. As this is a development machine, I'do assume we want to change the document root to point to a folder in our user home directory.

Search for the term DocumentRoot, and you should see the following line:

```
DocumentRoot "/usr/local/var/www"
```

Change this to point to your user directory where Sites folder is the name of your user account:

```
DocumentRoot "/Users/jack/Sites/Public"
```

You also need to change the <Directory> to reference right below the DocumentRoot line. This should also be changed to point to your new document root also:

```
<Directory "/Users/jack/Sites/Public">
```

Note

We removed the optional **Indexes** around the directory options as TextEdit will probably try to convert those to smart quotes and that will result in a Syntax error when you try to restart Apache. Even if you will accept the quotes and save them where they are, saving the document may result in their conversion and cause an error.

In that same <Directory> block you will find an <IfModule>/>/> setting, this should be changed as follows:

```
<IfModule dir_module>
    # For security reasons, only index.html is allowed by default.
    # If you add other files, make sure they are protected or otherwise filter.
    DirectoryIndex index.html index.cgi index.pl index.php index.xhtml index.html.var
</IfModule>
```

Also we should now enable **mod_rewrite** which is commented out by default. Search for LoadModule and uncomment the line by removing the leading # (e.g. the line this is a quick way to uncomment and comment a single or multiple lines:

```
LoadModule rewrite_module lib/modules/mod_rewrite.so
```

User & Group

Now we have the Apache configuration pointing to port 80/443 folder in our home directory. One problem still exists, however. By default, apache runs as user daemon and group (staff). This will cause permission problems when trying to access files in our home directory. About a third of the way down the <IfModule> file there are two settings to set the User and Group. Apache will run under. Change these to match your user account (replace jack with your own username, with a group of (staff):

```
User jack
Group staff
```

Servername

Apache likes to have a servername in the configuration, but this is disabled by default, so search for:

```
#ServerName www.example.com:8080
```

and replace it with:

```
ServerName localhost
```

Sites Folder

Now, you need to create a Sites folder in the root of your home directory. You can do this in your terminal, or in Finder. In this new Sites folder create a simple index.html and some dummy content in the index.html file. (echo "Hello" >~/Sites/Public/index.html)

```
mkdir ~/Sites
echo "Hello, world! mac Apache.local" >~/Sites/index.html
```

Restart apache to ensure your configuration changes have taken effect:

```
brew services stop httpd
brew services start httpd
```

Note

If you receive an error upon restarting Apache, by removing the quotes around the DocumentRoot and Directory designations can sort an issue.

Pointing your browser to http://localhost should display your new message. If you have that working, we can move on!

Note

Make sure you remove the :8080 port on test sites. Also, you might need to QUIT / Reload to clear the browser cache and pick up the new file.

Troubleshooting Non-Sudo httpd Services Start

Note

I will be updating this section with more tips and things to try as I find solutions to common problems. Please help us out on Amazon DonorChoice to get some Trail Mix.

This was pretty much Big Sur. I've switched from using sudo to launch <http>d with root access through it on the user/group defined in <http>.conf, and for people who have upgraded from their version to this, those now been problems.

I've had some problems myself but I guess it was a pretty pretty easy fix. Apache now runs under non-root sudo upload problems. Please try these steps if you have problems starting after you ran brew services start <http>.

First, try to start apache directly with:

```
/usr/local/bin/<http> -X start
```

This bypasses the brew service command and other pretty-ups specific issues. If you have issues reported about not being able to write to log files, try removing all the current log files:

```
rm -RF /usr/local/var/log/<http>/*
```

Then try starting again.

If you have seen a message saying something like: AH00072: make_sock: could not bind to address, try changing the Listen config in <http>.conf to:

```
Listen 0.0.0.0:80
```

PHP Installation

Warning

Are you keen developing PHP installations via Brew, you need to first cleanup your prior install via Upgrading Homebrew guide before continuing with this process.

Up until the end of Brew's 2018, all PHP related brews were bundled by homebrew-core, but that has been deprecated, so now we are stuck's distribution in the homebrew-core packages. This should be a simple process, but there is a few such complexities, and an issue.

PHP 5.6, PHP 7.0, and **PHP 7.3** have been deprecated and removed from the core Brew tap, and are only in **mod_accessible** and **unofficial** homebrew repos, so we need to install these unsupported versions in a development environment. These versions also need to "build from source" which means installing Xcode CLI tools, and Xcode.

Nevertheless, both **PHP 7.2 through 7.4** are officially supported by Brew, but these also have to be built which is pretty slow. For the latest versions of our guide we will use the new tap from gerwinkroiz as these are many versions (including PHP 8.0 builds) pre built.

PHP 8.0 has just been released and we are able to install it, but it might take some time for compatible PHP modules are fully available.

```
brew tap jerkurb/php-versions-fix
```

We will proceed by installing various versions of PHP and using a simple script to switch between them as we need. Feel free to exclude any versions you don't want to install:

```
brew install jerkurb/php-versions/php@8.0
brew install jerkurb/php-versions/php@7.4
brew install jerkurb/php-versions/php@7.3
brew install jerkurb/php-versions/php@7.2
brew install jerkurb/php-versions/php@5.6
```

Also, you may find the need to create unique port settings of PHP to your needs. A common thing to change is the memory setting, or the date.timezone configuration. The php.ini files for each version of PHP are located in the following directories:

```
/usr/local/etc/php/5.6/php.ini
/usr/local/etc/php/7.0/php.ini
/usr/local/etc/php/7.1/php.ini
/usr/local/etc/php/7.2/php.ini
/usr/local/etc/php/7.3/php.ini
/usr/local/etc/php/7.4/php.ini
/usr/local/etc/php/8.0/php.ini
```

Warning

At this point, I strongly recommend closing **All your terminal tabs and windows**. This will reset opening a new terminal to continue with the next step. This is strongly recommended because some newly upgraded tools (issues can arise with existing terminals (trust me, I have seen it).

We have installed but **not linked** these PHP versions. To switch to PHP 7.3, for example we can type:

```
brew unlink -q -f -n && brew link --overwrite --force php@7.3
```

Quick look that we're in the correct version:

```
php -v
```

```
PHP 7.3.30 (cli) (built: Sep 2 2021 06:41:37)
Copyright (c) 1997-2018 The PHP Group
Zend Engine v3.3.30, Copyright (c) 1998-2018, by Zend Technologies
    with Zend OpenLink v5.40, Copyright (c) 1998-2002, by Zend Technologies
```

and to switch to 5.6:

```
brew unlink -q -f -n && brew link --overwrite --force php@7.2
```

And check that it's changed correctly:

```
php -v
```

```
PHP 5.6.37 (cli) (built: Oct 28 2020 06:41:37) ( NTS )
Copyright (c) 1997-2016 The PHP Group
Zend Engine v2.6.0, Copyright (c) 1998-2016 Zend Technologies
    with Zend OpenLink v5.40, Copyright (c) 1998-2002, by Zend Technologies
```

Apache PHP Setup - Part 1

We have successfully installed your PHP versions, but we need to tell Apache to use them. You will again need to edit the /usr/local/etc/<http>/<http>.conf file and at the bottom of the LoadModule section.

If you have been following the guide correctly, the last entry should be your mod_php@7.3 module:

```
LoadModule rewrite_module lib/modules/mod_rewrite.so
```

Below this add the following <IfMod> modules:

```
<IfModule mod_php7.3.c>
    LoadModule php7_module /usr/local/opt/php@7.3/lib/httpd/modules/libphp7.so
    AddHandler application/x-<http>-php .php .php3 .php4 .php5 .php7 .php8 .phtml .pht
    AddHandler application/x-<http>-source .phps
    AddHandler application/x-<http>-module .slo
    AddHandler application/x-<http>-module .la
    AddHandler application/x-<http>-module .lo
    AddHandler application/x-<http>-module .<http>
    AddHandler application/x-<http>-module .mod
</IfModule>
```

We can only have one module processing PHP at a time, so for now, we have left our mod_php7.3 alone (uncommented while all the others are commented out. You can set Apache to use PHP 5.6 to include PHP requires. **Do not add the ability to switch PHP version later.**

Also you must set the Directory Indexes for PHP explicitly, so search for this block:

```
<IfModule dir_module>
    DirectoryIndex index.html index.cgi index.pl index.<http> index.xhtml index.html.var
</IfModule>
```

and replace it with this:

```
<IfModule dir_module>
    DirectoryIndex index.<http>x index.php index.html
</IfModule>
<IfMod>
    DirectoryIndex index.<http>x index.php index.html
</IfMod>
```

Save the file and **stop Apache then start again**, now that we have installed PHP:

```
brew services stop httpd
brew services start httpd
```

Validating PHP Installation

The best way to ensure PHP is installed and running as expected is to make use of phpinfo(). This is not something you want to leave on a production machine, but it's invaluable in a development environment.

Simply create a file called info.php, save it in ~/Sites/ folder as created earlier with this one line:

```
echo "<?php phpinfo(); ?>" >~/Sites/info.php
```

Now, point your browser to http://localhost/info.php and you should see a page like:

From your browser to http://localhost/info.php and you should see a phpinfo() information page:

If you see a similar phpinfo() result, congratulations! You now have Apache and PHP running successfully. You can stop the phpinfo() session by terminating the /usr/bin/curl -l -L -C - -O -v -k -H -A -c -b -e -d -F -T -u -x -X -D -o -O -r -T -w -t -T -u -x -X -D -o -O -r -T -w -t -T -u -x -X -D -o -O -r -T -w -t -T -u -x -X -D -o -O -r -T -w -t -T -u -x -X -D -o -O -r -T -w -t -T -u -x -X -D -o -O -r -T -w -t -T -u -x -X -D -o -O -r -T -w -t -T -u -x -X -D -o -O -r -T -w -t -T -u -x -X -D -o -O -r -T -w -t -T -u -x -X -D -o -O -r -T -w -t -T -u -x -X -D -o -O -r -T -w -t -T -u -x -X -D -o -O -r -T -w -t -T -u -x -X -D -o -O -r -T -w -t -T -u -x -X -D -o -O -r process.

PHP Switcher Script

We have coded Apache to use PHP 5.6, but we really want to be able to switch between versions. Luckily, some industrious individuals have already done the hard work for us and written a very handy Bash PHP switcher:

```
$ curl -L -O https://raw.githubusercontent.com/rasmus71/b4f3f3b8b7b1f0e0c0d0a0b0c1/raw/8c3b28d7fb4f0c0d0f0b0a0c1/phpswitch
$ mv phpswitch /usr/local/bin
$ chmod +x /usr/local/bin/phpswitch
```

We will install the /info script and the binary scandal /usr/local/bin:

```
$ curl -L -O https://raw.githubusercontent.com/rasmus71/b4f3f3b8b7b1f0e0c0d0a0b0c1/raw/8c3b28d7fb4f0c0d0f0b0a0c1/phpswitch
$ mv phpswitch /usr/local/bin
$ chmod +x /usr/local/bin/phpswitch
```

Check Your Path

Normalmen should have written a perl/bin/ and /usr/local/sbin/ to your path as part of its installation process. Quickly test this by typing:

```
$ echo $PATH
/usr/local/sbin:/usr/local/bin:/bin:/usr/sbin:/usr/sbin:/usr/local/sbin:/usr/local/bin:/usr/local/sbin:/usr/local/bin:/usr/local/sbin:/usr/local/b
```

18 NOV

journal

Grav 1.7 Stable Release - Jan 19th, 2021

Info

Andy Miller 6 mins

21 JAN

release

Grav 1.7 CLI Self-Upgrade Bug

We have identified and fixed a bug in Grav 1.7.0 and 1.7.1 that causes issues when using the CLI (console)

Andy Miller 1 min

15 DEC

tutorial

ec2 serverpilot vps

Fast Free Grav Development

I had originally planned on penning a blog post about how easy it is to setup and deploy a Grav site

Andy Miller 9 mins

Download Grav

Never miss a thing, sign up to the Grav mailing list

Your email address Subscribe

Grav was </> with ♥ by Trilby Media

Designed by Eduardo Santos

- Contact the Grav Team
- About Grav
- Grav Media Information
- Grav News Feed

Crazy Fast VPS Hosting Sponsored by Linode - managed by RunCloud - monitoring by Pingometer

Copyright @2021 - Grav CMS - All rights reserved - Grav is released under the MIT license

GRAV Premium **Turbo-charge** your Grav site - from the creators of Grav

Learn about Premium