

부록 A 운영체제별 개발 환경 준비

부록 A에서는 다음 내용을 배운다.

- 웹 개발을 하는 데 필요한 PHP, MySQL을 로컬 컴퓨터에 설치하고 구동하는 방법
- 개발을 좀 더 편리하게 하기 위한 도구들의 설치법

앞으로 설명할 내용은 Mac OS X El Capitan, Ubuntu Desktop 14.04, Windows 10 운영체제에서 검증하였다.

서두에 말한 대로 이 책의 지향점은 풀 스택과 데브옵스다. 코드가 동작하는 시스템을 이해하면 더 나은 코드를 짜는데 도움이된다. 그래서 우리는 라라벨 홈스테드¹는 이용하지 않고 개발에 필요한 도구를 로컬 컴퓨터에 직접 설치하는 방법을 설명한다. 홈스테드를 이용할 독자는 라라벨 공식 문서를 참고하기 바란다.

개발 환경을 준비하는 과정은 까다롭지만 이 책대로 하면 100% 성공할 수 있다고 보장한다. 이미 사용하던 개발 환경이 라라벨 개발에 적합하면 그대로 사용해도 무방하다.

라라벨이 요구하는 PHP 버전은 엄격하다. 이 책이 사용하는 라라벨 5.2.x 버전은 PHP 5.5.9에서 동작한다. 곧 나올 라라벨 5.3에서는 PHP 5.6 이상을 요구한다. PHP 5.5 버전은 2016년 6월 10일에 묘비명을 장식할 예정이다.²) 이 말을 다시 풀이하면, 개발한 코드를 배포할 운영 서버도 PHP 5.5.9 이상이 필요하다는 의미다. 그 외에도 몇 가지 PHP 확장 모듈이 필요하다.

- PHP 5.5.9 이상
- `openssl` PHP 확장모듈_ 암호화, 해시를 위해 필요한 모듈
- `pdo` PHP 확장모듈_ PHP와 데이터베이스를 연결하는 모듈
- `mbstring` PHP 확장모듈_ 멀티바이트 문자열 처리를 위한 모듈
- `tokenizer` PHP 확장모듈_ PHP 파서, 아티즌 콘솔 등의 동작에 필요한 모듈

A.1. 코드 에디터 준비

프로그래밍이란 일을 효율적으로 하려면 코드 에디터라 부르는 소프트웨어가 필요하다. 아래는 라라벨 개발자들이 가장 많이 쓰는 코드 에디터이다.

- `phpStorm`³
- `SublimeText`⁴

¹. 라라벨 홈스테드_ 라라벨로 웹 개발을 위해 필요한 모든 도구가 준비된 Ubuntu 가상 머신 이미지 및 이를 관리하기 위한 도구, <https://laravel.com/docs/homestead>, <http://laravel.kr/docs/homestead> ↩

². PHP 버전별 지원_ <http://php.net/supported-versions.php> ↩

³. `phpStorm`_ <http://www.jetbrains.com/phpstorm> 에서 30일 평가판을 내려 받을 수 있다. PHP 전용으로 개발된 통합개발 환경(IDE)이다. ↩

⁴.

4. SublimeText_ <http://www.sublimetext.com> 에서 무료로 내려 받을 수 있다. 라라벨 개발과 관련 된 Sublime 확장 기능 설치는 다음 링크를 참고하자. <http://bit.ly/sublime-laravel> ↩

A.2. Mac

라라벨이 동작하는데 문제가 없다면 이 책에서 제시하는 것과 다른 도구, 다른 버전을 사용해도 무방하다.

A.2.1. 개발도구 설치

홈브루

홈브루(Homebrew)는 Mac용 패키지 관리 도구다. PHP, MySQL 등 실습에 필요한 다른 패키지를 홈브루를 이용해서 설치할 것이다.

```
# 콘솔 a-1 홈브루 설치 및 설치 확인

# 긴주소 타이핑이 번거롭다면 `homebrew install`로 구글링해서 복사해서 사용한다.
$ ruby -e "$(curl -fsSL https://raw.githubusercontent.com/Homebrew/install/master/install)"

$ brew --version
# Homebrew 0.9.5 (git revision 53d0; last commit 2016-02-22)
```

잘 설치되었나 확인하자. 주석으로 표시된 문자열이 표시되면 정상적으로 설치된 것이다.

팁

버전이 출력되지 않으면 콘솔을 다시 시작하고 다시 해 본다. 설치 과정에 사용자의 콘솔 환경 설정에 필요한 경로를 추가했는데 추가된 설정을 적용하기 위함이다.

홈브루는 필요한 소스코드를 내려 받아 운영체제 환경에 맞게 직접 컴파일한다. 컴파일에 필요한 도구는 XCode로 설치하면 편리하다(`$ xcode-select --install`).

PHP

PHP 최신 버전인 7.0 을 설치한다. 홈브루로 설치가 가능한지 먼저 확인하자.

```
# 콘솔 a-2 PHP 탭 추가

$ brew tap homebrew/php
$ brew search php70
# homebrew/php/php70    homebrew/php/php70-hprose    ...
```

있다! `homebrew/php/php70` 를 설치하자. 설치 후에는 정상 설치되었는지 확인해야 한다.

```
# 콘솔 a-3 PHP 설치 및 설치 확인

$ brew install homebrew/php/php70

$ php --version
# PHP 7.0.x ...
```

MySQL

홈브루로 MySQL을 설치하고, 루트 사용자의 비밀번호를 `secret` 로 설정한다. 명령을 실행하면 나오는 첫 질문에 `N` 으로 답해야 짧은 비밀번호를 쓸 수 있으니 `Y` 로 답하지 않도록 주의한다. 이 비밀번호의 쓰임새를 알고 잘 기억할 수 있다면, `Y` 를 눌러 다른 비밀번호를 넣어도 좋다. MySQL 시작과 종료를 편리하게 하기 위한 확장 기능도 추가하자.

```
# 콘솔 a-4 MySQL 설치 및 설치 확인

$ brew install mysql

$ mysql --version
# mysql Ver 14.14 Distrib 5.7.11, ...

$ mysql_secure_installation
# ...
# Would you like to setup VALIDATE PASSWORD plugin?
# Press y|Y for Yes, any other key for No:          # N
# Please set the password for root here.
# New password:                                     # secret
# Re-enter new password:                           # secret
# ...
# All done!

$ brew tap homebrew/services
```

이제 MySQL 서버를 시작하고 종료할 때 아래 명령을 이용할 수 있다.

```
# 콘솔 a-5 MySQL 시작과 종료

$ brew services start mysql
# ==> Successfully started 'mysql' (label: homebrew.mxcl.mysql)

$ brew services stop mysql
# ==> Successfully stopped 'mysql' (label: homebrew.mxcl.mysql)
```

컴포저

컴포저를 설치한다. 컴포저는 PHP의 표준 의존성 관리 도구다.

```
# 콘솔 a-6 컴포저 설치

$ brew install homebrew/php/composer
$ composer --version
# Composer version 1.0.0 ...
```

A.2.2. PHP 확장모듈

이 책을 따라 했다면 필요한 확장모듈은 이미 준비된 상태다. 아래 명령으로 한 번 더 확인하자. 빠진 확장모듈이 있다면 홈브루로 검색해서 설치한다.

```
# 콘솔 a-7 PHP 확장모듈 확인

$ php -m | grep 'openssl\|pdo\|mbstring\|tokenizer'
# mbstring
# openssl
# pdo_mysql
# tokenizer
```

A.3. Ubuntu Desktop

Ubuntu Desktop에 개발 환경을 준비해 보자. 참고로 저자는 Mac 컴퓨터에 `box-cutter/ubuntu1404-desktop` 베이그런트(vagrant) 이미지를 버추얼박스(virtual box)에 올려서 사용했다. 라라벨이 동작하는데 문제가 없다면 이 책에서 제시하는 것과 다른 리눅스 배포판, 다른 도구, 다른 버전을 사용해도 무방하다.

A.3.1. 개발 도구 설치

Ubuntu의 패키지 관리 도구인 `apt-get` 을 이용하여 필요한 패키지들을 설치한다. 먼저 로컬 컴퓨터에 설치된 패키지 캐시를 업데이트한다.

```
# 콘솔 a-8 패키지 업데이트

$ sudo apt-get update
```

Ubuntu에서 제공하는 기본 소프트웨어 저장소는 최신 PHP 버전을 제공하지 않는다. 그래서 새로운 저장소(PPA)를 추가한다.

```
# 콘솔 a-9 사설 패키지 저장소 추가

# UTF-8 문자 관련 오류를 피하기 위해 언어 팩을 설치한다.
$ sudo apt-get install -y language-pack-en-base

# PPA를 추가하려면 add-apt-repository 명령줄을 포함하고 있는 패키지가 필요하다.
$ sudo apt-get install -y software-properties-common

# 온드르제이 수리(Odřej Surý)의 PPA를 추가한다.
$ sudo LC_ALL=en_US.UTF-8 add-apt-repository ppa:ondrej/php

# 로컬 컴퓨터의 패키지 캐시를 업데이트한다.
$ sudo apt-get update
```

PHP

PHP 최신 버전인 7.0과 필요한 PHP 확장모듈을 설치하고, 정상 설치를 확인한다.

```
# 콘솔 a-10 PHP 설치 및 확인

$ sudo apt-get install -y php php-mysql php-sqlite3 php-mbstring php-xml

$ php --version
# PHP 7.0.4-6+deb.sury.org~trusty+5 ...
```

MySQL

MySQL 설치 중에 `root` 사용자의 비밀번호를 입력하는 창이 나타나면 `secret` 를 입력한다(이 비밀번호의 쓰임새를 알고 잘 기억할 수 있다면, 다른 비밀번호를 넣어도 좋다).

```
# 콘솔 a-11 MySQL 설치 및 확인

$ sudo apt-get install mysql-server

$ mysql --version
# mysql Ver 14.14 Distrib 5.5.47, ...
```

MySQL 서버를 실행한다.

```
# 콘솔 a-12 MySQL 시작

$ sudo service mysql start
# mysql start/running, process xxxxx
```

컴포저

컴포저를 설치한다. 컴포저는 PHP의 표준 의존성 관리 도구다.

```
# 콘솔 a-13 컴포저 설치 및 확인

$ curl -sS https://getcomposer.org/installer | php
$ sudo mv composer.phar /usr/local/bin/composer
$ composer --version
# Composer version 1.0.0 ...
```

A.3.2. PHP 확장모듈

이 책을 따라 했다면 필요한 확장모듈은 이미 준비된 상태다. 아래 명령으로 한 번 더 확인하자. 빠진 확장모듈이 있다면 `apt-get` 으로 설치한다.

```
# 콘솔 a-14 PHP 확장모듈 확인

$ php -m | grep 'openssl\|pdo\|mbstring\|tokenizer'
# mbstring
# openssl
# pdo_***
# tokenizer
```

A.4. Windows

Windows PC에 개발 환경을 준비해 보자. 라라벨이 동작하는데 문제가 없다면 이 책에서 제시하는 것과 다른 도구, 다른 버전을 사용해도 무방하다.

A.4.1. 개발도구 설치

아래 과정을 따라 하기 전에 먼저 본인이 사용하는 Windows 사용자 계정을 확인하자. Windows키와 r키를 동시에 누르고, 입력 박스에 `cmd` 를 입력하고 Enter를 치면 명령 프롬프트를 열 수 있다.

```
# 콘솔 a-15 사용자 계정 이름 확인

$ echo $USER

# 또는
\> echo %USERNAME%
```

문제가 있다면 공백이 없는 영문 사용자 계정을 새로 만든다. 앞으로 실습은 새로 만든 계정으로 로그인해서 해야 한다. 라라벨과는 관련이 없는 문제다. 개발도구를 설치할 때나 콘솔을 사용할 때, 이런 사용자의 컴퓨터에서 문제가 발생하는 것을 오프라인 강의에서 종종 경험했다.

```
홍길동 (x)
Gildong Hong (x)
GildongHong (o)
gildonghong (o)
```

깃배시

Windows에 기본으로 포함된 명령프롬프트를 대체하는 깃배시⁵를 설치하자. 이하 '콘솔' 이라 부르고, 콘솔 a-15에서 썼던 Windows 내장 명령 프롬프트는 이 책에서는 사용하지 않는 것으로 가정한다.

비트나미

Windows에서도 다른 운영체제처럼 PHP와 MySQL을 각각 설치할 수 있다. 그런데 PHP 확장모듈을 설치하는 방법이 복잡해서, 우리는 이미 차려진 밥상인 비트나미(bitnami)⁶를 이용한다.

설치 중에 Windows 방화벽과 관련된 경고가 뜬다면 무시하고 '다음' 을 누른다.

내려 받은 비트나미 파일을 더블 클릭한다. 인스톨러 화면이 열리면 '다음' 을 계속 눌러서 설치할 수 있는데, 'Select Component' 창에서는 모든 선택 상자를 해제하고 '다음'으로 진행한다.

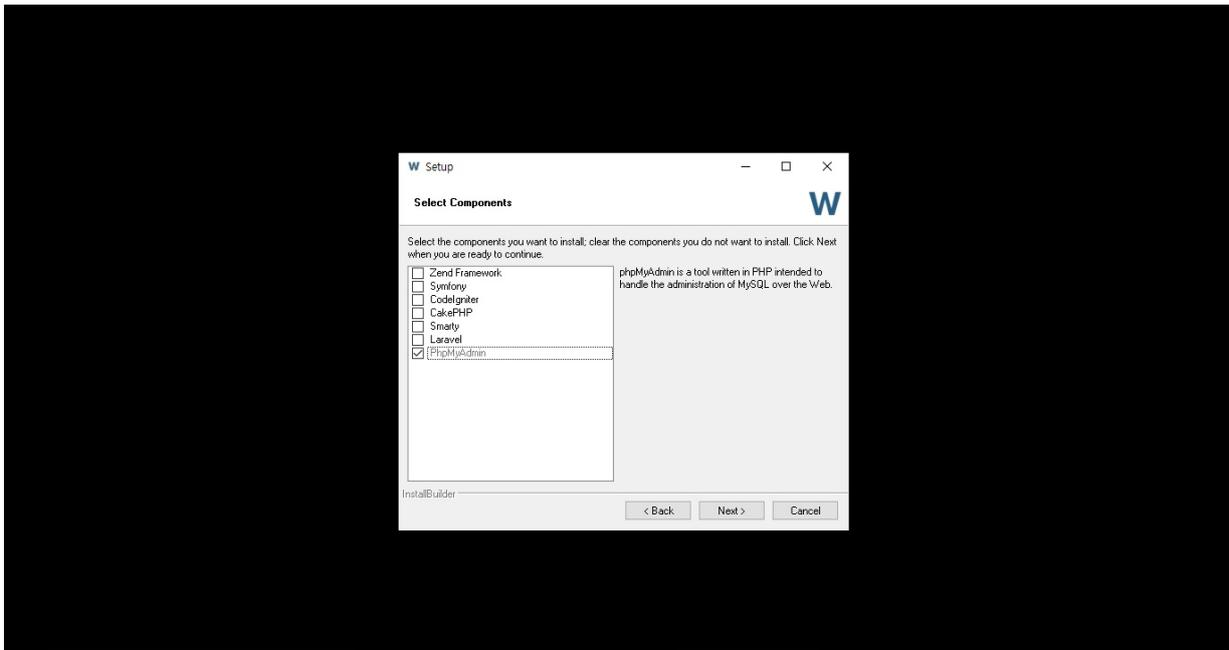


그림 a-1 비트나미, 'Select Component' 창

설치 중에 MySQL 'root' 사용자의 비밀번호를 입력하는 창이 나타나면 'secret' 를 입력하자(이 비밀번호의 쓰임새를 알고 잘 기억할 수 있다면, 다른 비밀번호를 넣어도 좋다). 설치가 끝나면 그림 a-2의 왼쪽에 위치한 화면을 볼 수 있다. 브라우저 창은 닫고, 'Bitnami WAMP Stack 5.5.32.0' 이라 표시된 창은 그대로 둔다.

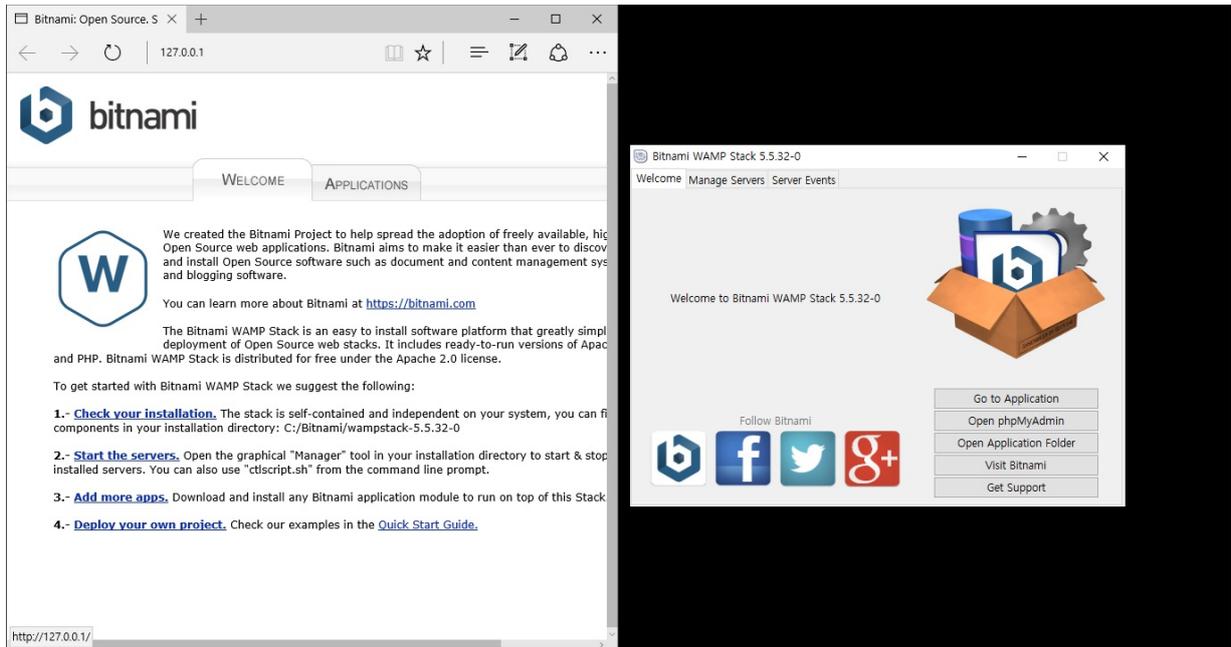


그림 a-2 비트나미, 설치 완료

이 책은 PHP 내장 웹 서버를 사용하여 실습 코드를 실행하는 방법을 설명한다. 이 책을 따라 할 독자는 그림 a-3을 참고해서 아파치(apache) 웹 서버를 끈다. 'Bitnami WAMP Stack 5.5.32.0'으로 표시된 창에서 'Manage Servers' 탭을 누른다. 현재 실행 중인 서버 목록이 표시된다. 'Apache Web Server' 항목을 선택한 후 'Stop' 버튼을 눌러 끈다.

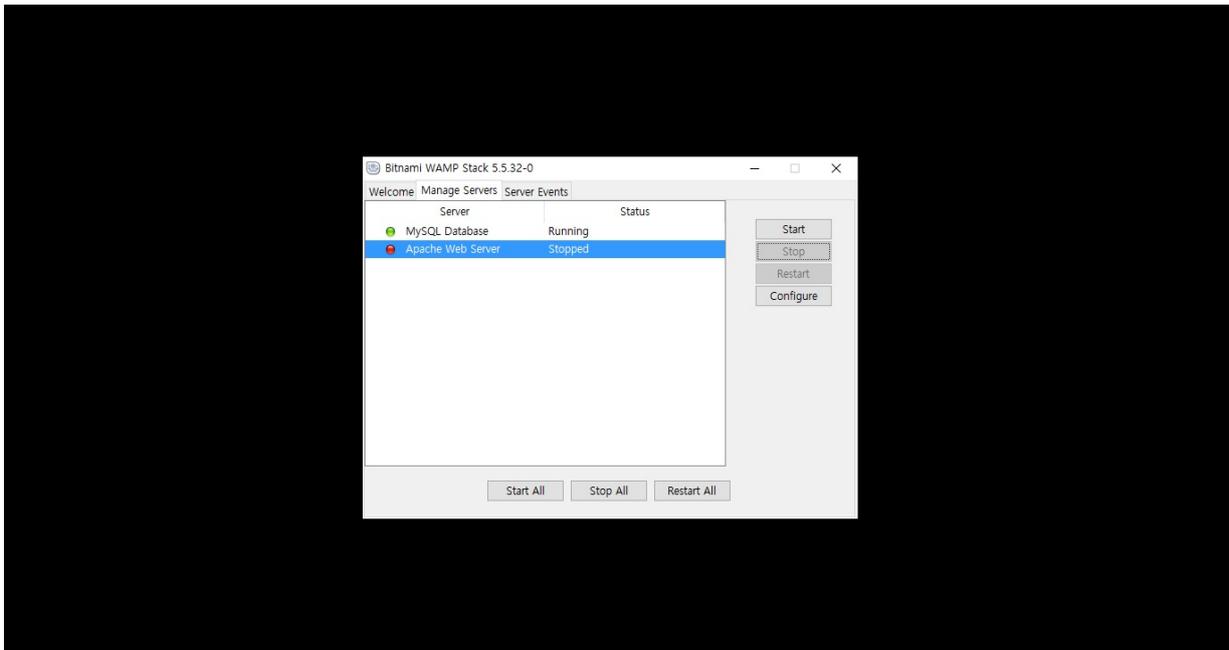


그림 a-3 비트나미, 스택 매니저에서 아파치 웹 서버 끄기

php 와 mysql 명령을 어떤 경로에서든 실행할 수 있도록 운영체제 설정을 변경해야 한다. 작업표시줄의 '웹 및 Windows 검색'에서 '시스템 환경 변수 편집'을 타이핑하고 Enter를 누른다. 검색 결과에서 '시스템 환경 변수 편집(제어판)'을 선택하면 그림 a-4와 같은 창이 나타난다.

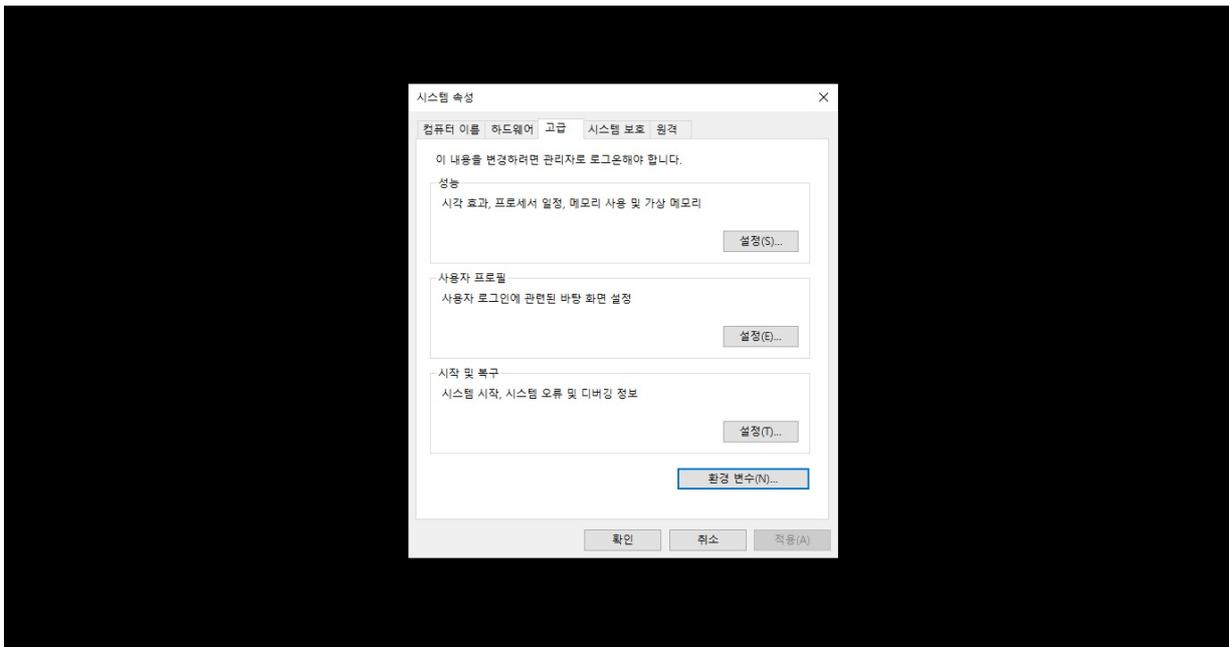


그림 a-4 Windows, 시스템 환경 변수 편집

창 아래에 '환경 변수(N)...'라 표시된 버튼을 눌러 '시스템 변수(S)'그룹에서 'Path' 변수를 더블 클릭하면 새로운 창이 나타난다.

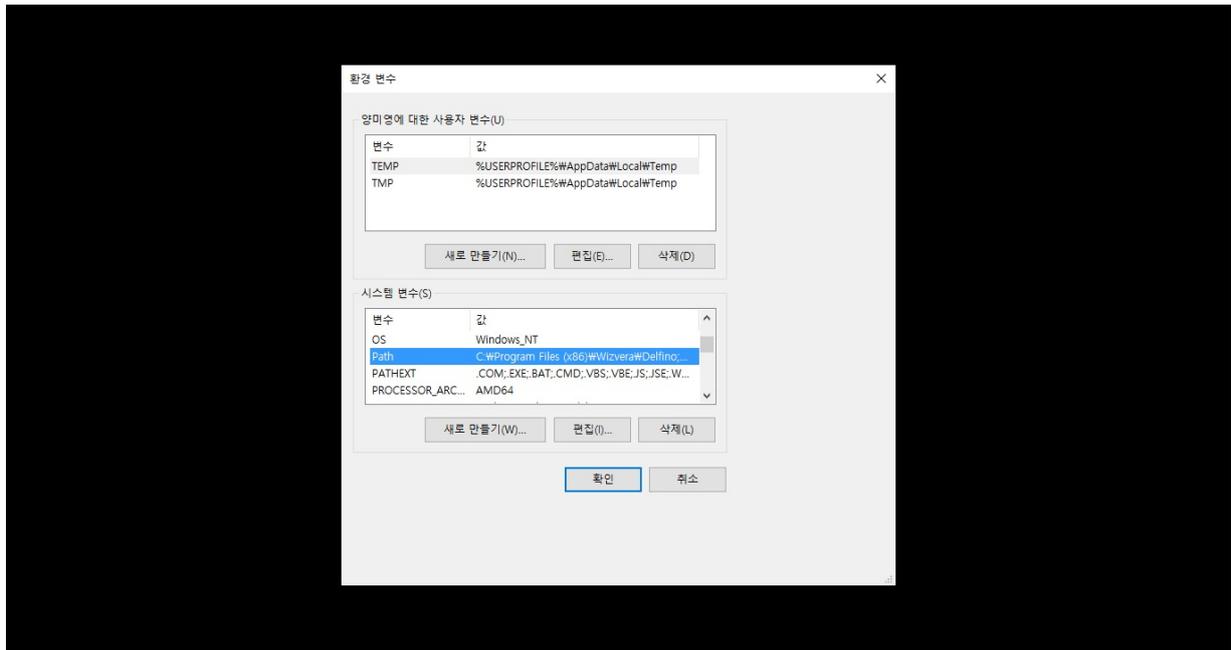


그림 a-5 Windows, 시스템 환경 변수 편집

새로 나타난 창에서 '찾아보기(B)...'를 눌러 아래 두 개의 경로를 추가한다. 독자 여러분이 설치할 때는 그림과 다른 버전이 다른 경로에 설치되었을 것이다. 자신이 설치한 경로에 찾아서 적용한다.

- `C:\Bitnami\wampstack-5.5.32-0\php`
- `C:\Bitnami\wampstack-5.5.32-0\mysql\bin`

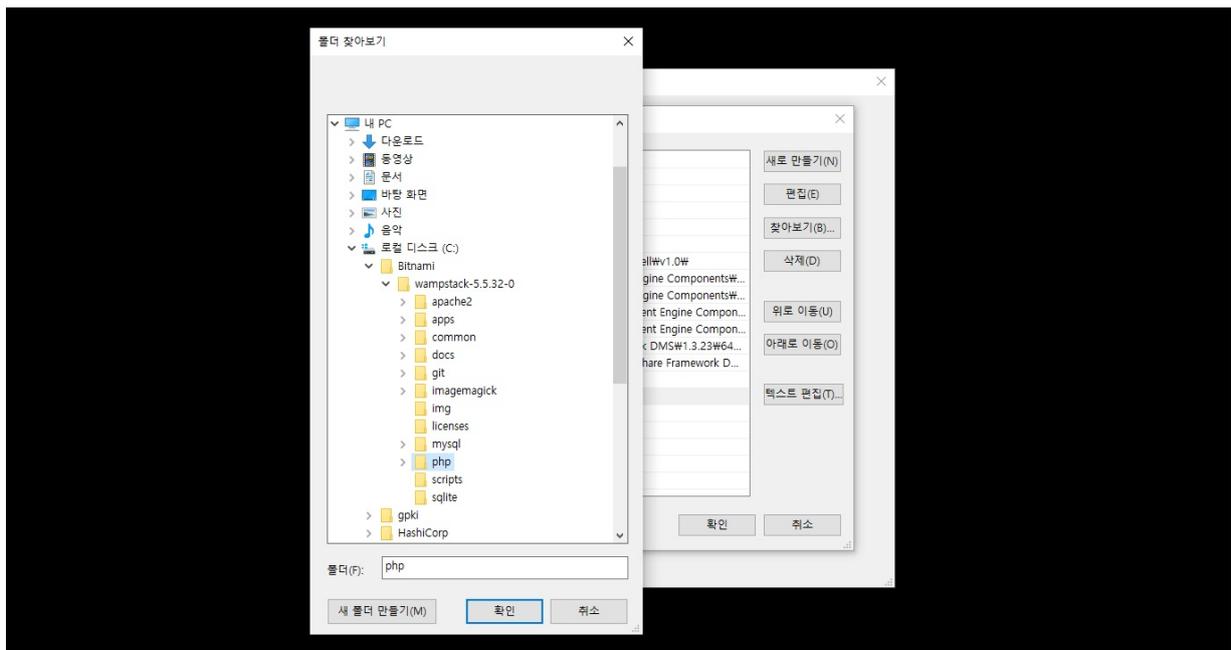


그림 a-6 Windows, 시스템 환경 변수 편집에서 PHP 및 MySQL 경로 추가

비트나미 설치 확인

PHP와 MySQL이 잘 설치되었나 콘솔에서 확인하자. 앞 절에서 추가한 경로 설정이 적용되지 않아 `php` , `mysql` 명령이 동작하지 않는다면 콘솔에서 `set` 명령어를 한번 실행해 준다. 그래도 안되면, 운영체제를 재 부팅한다.

```
# 콘솔 a-16 PHP 버전 확인

$ php --version
# PHP 5.6.18 (cli) (built: ...)
```

[편주 - 박스 겹침 방지]

```
# 콘솔 a-17 MySQL 버전 확인

$ mysql --version
# C:\Bitnami\wampstack-5.5.32-0\mysql\bin\mysql.exe VER 14.14 Distrib 5.6.28, for Win32
```

PHP 설정 변경

비트나미는 운영⁵ 환경에 사용할 것을 염두에 둔 패키지다. 개발에 적합하도록 PHP의 설정을 변경하자. 콘솔에서 PHP의 설정 파일의 위치를 찾는다.

```
# 콘솔 a-18 PHP 환경 설정 파일 찾기

$ php --ini
# Configuration File (php.ini) Path: C:\WINDOWS
# Loaded Configuration File:      C:\Bitnami\wampstack-5.5.32\php\php.ini
```

`C:\Bitnami\wampstack-5.5.32\php\php.ini` 에 위치한 설정 파일을 메모장으로 연다.

```
# 콘솔 a-19 PHP 환경 설정 파일 열기

$ notepad C:\Bitnami\wampstack-5.5.32\php\php.ini
```

찾아서 수정해야 할 곳은 두 군데이다.

- `opcache.enable=1` (기존) -> `opcache.enable=0` (변경)
- `display_errors=0ff` (기존) -> `display_errors=0n` (변경)

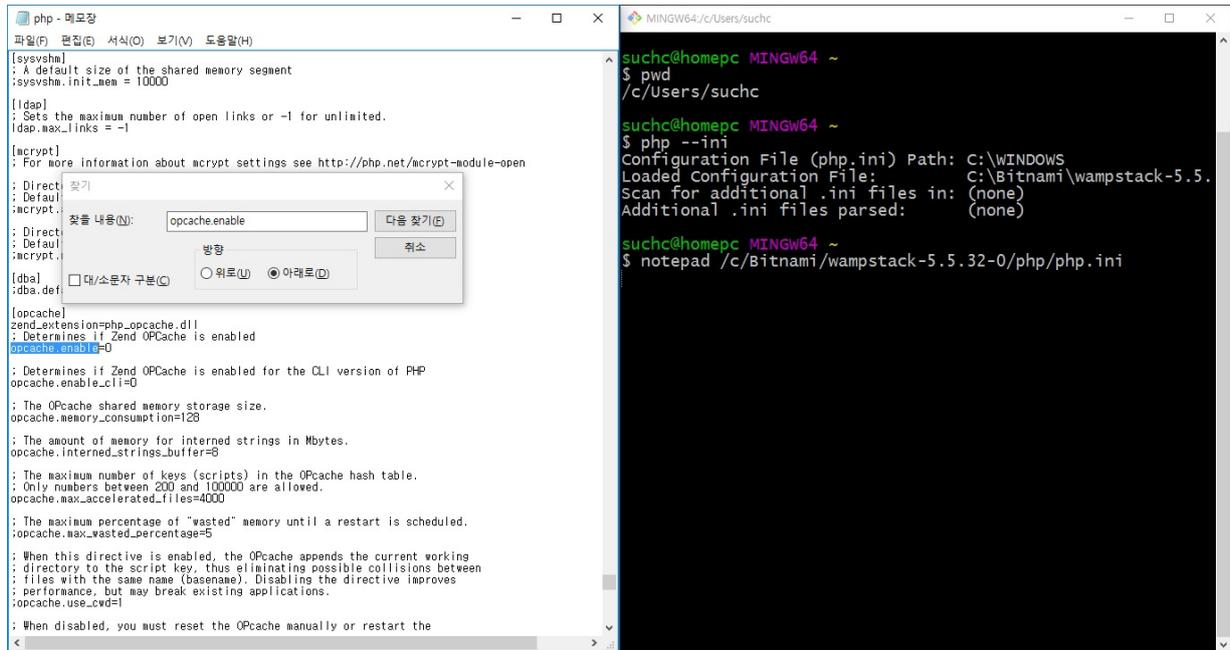


그림 a-7 메모장, PHP 설정 변경

컴포저

Windows용 컴포저 인스톨러⁷를 내려 받아 설치한다. 설치 중에 PHP 경로를 물어보는데

C:\Bitnami\wampstack-5.5.32\php\php.exe 를 찾아서 선택한다. 컴포저는 PHP의 표준 의존성 관리 도구다.

```
# 콘솔 a-20 컴포저 설치 및 확인

$ composer --version
# Composer version 1.0.0 ...
```

A.4.2. PHP 확장모듈

이 책을 따라 했다면 필요한 확장모듈은 이미 준비된 상태다. 아래 명령으로 한 번 더 확인하자. `openssl`, `pdo_mysql`, `mbstring`, `tokenizer` 총 네 개의 확장모듈 이름이 출력결과에 있는 지 확인한다. 빠진 확장모듈이 있다면 구글링해서 설치한다.

```
# 콘솔 a-21 PHP 확장모듈 확인

$ php -m
```

⁵ 깃배시(Git sh)_ <https://git-for-windows.github.io> 책을 쓰는 도중에 마이크로소프트가 Windows 10에 배시 셸을 포함한다고 발표했다. Windows 10 사용자는 깃배시를 설치하기전에 운영 체제에 배시 셸 포함여부를 꼭 확인하라. ←

- 6. 비트나미(Bitnami)_ <https://bitnami.com/stack/wamp> ↩
- 7. Windows용 컴포저 인스톨러_ <https://getcomposer.org/Composer-Setup.exe> ↩