安装

* [**安装**](http://www.golaravel.com/laravel/docs/5.1/#installation)
* [**配置**](http://www.golaravel.com/laravel/docs/5.1/#configuration)
  + [基本配置](http://www.golaravel.com/laravel/docs/5.1/#basic-configuration)
  + [环境配置](http://www.golaravel.com/laravel/docs/5.1/#environment-configuration)
  + [配置缓存](http://www.golaravel.com/laravel/docs/5.1/#configuration-caching)
  + [获取配置](http://www.golaravel.com/laravel/docs/5.1/#accessing-configuration-values)
  + [为应用程序命名](http://www.golaravel.com/laravel/docs/5.1/#naming-your-application)
* [**维护模式**](http://www.golaravel.com/laravel/docs/5.1/#maintenance-mode)

[安装](http://www.golaravel.com/laravel/docs/5.1/" \l "installation)

对运行环境的要求

Laravel 框架对系统环境有一些要求。当然，所有这些要求在 [Laravel Homestead](http://www.golaravel.com/laravel/docs/5.1/homestead) 虚拟机中都是预装好的：

- PHP >= 5.5.9 - OpenSSL PHP 扩展 - PDO PHP 扩展 - Mbstring PHP 扩展 - Tokenizer PHP 扩展

php.ini:

extension=php\_openssl.dll

可参考：

;extension=php\_bz2.dll

;extension=php\_curl.dll

;extension=php\_fileinfo.dll

extension=php\_gd2.dll

;extension=php\_gettext.dll

;extension=php\_gmp.dll

extension=php\_intl.dll

;extension=php\_imap.dll

;extension=php\_interbase.dll

;extension=php\_ldap.dll

extension=php\_mbstring.dll

extension=php\_exif.dll ; Must be after mbstring as it depends on it

extension=php\_mysql.dll

extension=php\_mysqli.dll

extension=php\_oci8\_12c.dll ; Use with Oracle Database 12c Instant Client

extension=php\_openssl.dll

extension=php\_pdo\_firebird.dll

extension=php\_pdo\_mysql.dll

extension=php\_pdo\_oci.dll

extension=php\_pdo\_odbc.dll

extension=php\_pdo\_pgsql.dll

extension=php\_pdo\_sqlite.dll

;extension=php\_pgsql.dll

;extension=php\_pspell.dll

;extension=php\_shmop.dll

……

; Directory in which the loadable extensions (modules) reside.

; http://php.net/extension-dir

; extension\_dir = "./"

; On windows:

extension\_dir = "D:\php\ext"

安装 Laravel

Laravel 利用 [Composer](http://getcomposer.org/)（[Composer 中文](http://www.phpcomposer.com/)）来管理其自身的依赖包。因此，在使用 Laravel 之前，请务必确保在你的机器上已经安装了 Composer 。如果你是下载“一键安装包”的话，可以暂时不用安装 Composer，等熟悉 Laravel 了再回头摸索，免得上来就遇到钉子。

**下载 Laravel 一键安装包**

安装 Composer 或通过 Composer 下载 Laravel 的依赖包时都可能被墙，为了方便大家学习和开发 Laravel 应用，Laravel 中文网已经提供了 Laravel 各个版本的一键安装包。这些一键安装包都已经集成了所有依赖（也就是已经执行过 composer install了，vendor 目录已经就绪）。

下载地址：<http://www.golaravel.com/download/>

另外，一键安装包还包含了以下修改：

* Laravel 5.x 版本都已经包含了一份 .env 配置文件，大家可以不用自己创建这个文件了。
* 对于所有 Laravel 版本都已经设置了 Application key（也就是通过php artisan key:generate 生成了秘钥），注意：最终上线时，请务必重新执行一次php artisan key:generate 指令，以便重新生成秘钥。
* 去除了所有视图文件中引用的 google 字体。

Windows 用户还可以参考[这篇文章](http://www.golaravel.com/post/install-and-run-laravel-5-x-on-windows/" \t "_blank) 按步骤安装并运行 Laravel 。

**通过 Laravel 安装工具安装 Laravel**

首先，使用 Composer 下载 Laravel 安装包：

composer global require "laravel/installer"

请确保 PATH 环境变量已经添加了 ~/.composer/vendor/bin 目录，这样，可执行文件laravel 就能被你的系统检测到了。

一旦安装完成后，就可以使用 laravel new 命令在你指定的目录中建立一份全新安装的Laravel 应用。例如： laravel new blog 命令会在当前目录下建立一个名为 blog 的目录， 此目录里面存放着全新安装的 Laravel ，并且所有依赖包也已经安装好了。此方法的安装速度会比通过 Composer 安装快很多。

laravel new blog

**通过 Composer Create-Project 命令安装 Laravel**

还可以通过 Composer 的 create-project 命令来安装 Laravel：

composer create-project laravel/laravel --prefer-dist

[配置](http://www.golaravel.com/laravel/docs/5.1/" \l "configuration)

基本配置

Laravel 框架所用的所有配置文件都被存放在 config 目录下。每个配置项都有文档说明，所以请通读所有配置文件以熟悉所有可用的配置项。

**目录权限**

安装 Laravel 之后，可能需要你配置一下目录权限。web 服务器需要拥有 storage 目录下的所有目录和 bootstrap/cache 目录的写权限。如果你在使用 [Homestead](http://www.golaravel.com/laravel/docs/5.1/homestead) 虚拟机，这些权限都已经帮你设置好了。

**应用程序的秘钥**

安装 Laravel 之后接下来需要做的就是设置一个随机字串作为应用的秘钥（key）。如果你是通过 Composer 或 Laravel 安装器安装的 Laravel，这个 key 已经由key:generate 命令自动生成并设置了。一般情况下，这个作为 key 的字串的长度是 32 个字符。这个 key 还可以在 .env 环境配置文件中设置。如果你没有将 .env.example文件改名为 .env，那现在就做吧。**如果应用的 key 没有被配置，会话和其他需要加密的数据将不安全！**

**额外的配置**

Laravel 开箱即用，几乎不需要什么配置。现在就可以开始你的开发之旅了！不过，建议你浏览一下 config/app.php 文件和此文件中的文档。它包含了几个配置项，例如timezone 和 locale ，可能需要根据你自身的情况稍作修改。

你可能还需要为 Laravel 中的几个组件做些配置，例如：

* [缓存](http://www.golaravel.com/laravel/docs/5.1/cache#configuration)
* [数据库](http://www.golaravel.com/laravel/docs/5.1/database#configuration)
* [会话](http://www.golaravel.com/laravel/docs/5.1/session#configuration)

完成 Laravel 安装后，建议阅读 [环境配置](http://www.golaravel.com/laravel/docs/5.1/installation#environment-configuration)章节。

**美化链接**

**Apache**

Laravel 框架自带了 public/.htaccess 文件用来从网址中删除 index.php。如果你用的是 Apache 来运行你的 Laravel 应用，请务必启用 Apache 的 mod\_rewrite 模块。

如果 Laravel 自带的 .htaccess 文件在你的 Apache 中不起作用，请试一试下面的配置：

Options +FollowSymLinks

RewriteEngine On

RewriteCond %{REQUEST\_FILENAME} !-d

RewriteCond %{REQUEST\_FILENAME} !-f

RewriteRule ^ index.php [L]

**Nginx**

在 Nginx 中，将下面的指令放到站点配置文件中就可以实现美化链接的功能：

location / {

try\_files $uri $uri/ /index.php?$query\_string;

}

当然，如果你用的是 [Homestead](http://www.golaravel.com/laravel/docs/5.1/homestead)，美化链接的功能已经被自动配置好了。

环境配置

通常应用程序需要根据不同的运行环境加载不同的配置信息。例如，你可能希望本机开发环境与生产服务器环境使用不同的缓存驱动。通过配置文件，就可以轻松完成。

为了简化配置，Laravel 使用了 Vance Lucas 开发的 [DotEnv](https://github.com/vlucas/phpdotenv" \t "_blank) 库。在全新安装的 Laravel 中，应用程序的根目录下都会有一个 .env.example 文件，如果你是通过 Composer 安装的 Laravel，这个文件将被自动重命名为 .env。如果没有，请手动重命名。

当应用程序收到一个请求时，.env 文件中的所有变量都会被加载到 PHP 的 $\_ENV 超全局变量中。这时，你就可以通过 env 辅助函数来从该超全局变量中获取需要的配置了。实际上，如果你查看 Laravel 的配置文件，你会发现有几个配置项已经在使用这个辅助函数了。

根据你自己的本地开发环境和生产环境来修改这些环境配置即可。不过，.env 文件不应该和应用程序的源码一起被提交到源码仓库中，因为每个开发环境/服务器环境可能需要不同的环境配置。

如果你们是一个开发团队，可能希望将 .env.example 文件包含到源码中。通过在配置文件中预留一些占位符，团队中的其他开发人员将可以很清楚地看到执行此应用程序都需要配置哪些环境变量。

**获取应用程序的运行环境**

应用程序的运行环境可以通过 .env 文件中的 APP\_ENV 变量来确定。你还可以调用App [facade](http://www.golaravel.com/laravel/docs/5.1/facades) 中的 environment 方法：

$environment = App::environment();

通过给 environment 方法传递参数可以检查当前环境是否与所传参数一致。如果需要，也可以传递多个值作为参数：

if (App::environment('local')) {

// The environment is local

}

if (App::environment('local', 'staging')) {

// The environment is either local OR staging...

}

通过 app 辅助方法可以访问当前应用程序的实例：

$environment = app()->environment();

配置缓存

为了提升应用程序的执行速度，建议通过 Artisan 的 config:cache 命令将所有配置文件合并到一个文件中并缓存起来。这将合并你应用中的所有配置信息到单个文件中，这样它就能被框架更快地载入。

建议将执行 php artisan config:cache 命令作为产品部署流程中的一个步骤。由于开发过程中需要频繁修改配置项，因此不应该在本地开发时执行此命令。

获取配置

通过 config 全局辅助方法，你可以很容易地访问配置信息。配置信息可以通过 “点” 语法访问到，“点”用于分割配置文件的名称和配置项的名称。你还可以为不存在的配置项指定一个默认的返回值：

$value = config('app.timezone');

如需在程序运行时重置配置信息，只需传递一个数组到 config 辅助方法即可：

config(['app.timezone' => 'America/Chicago']);

为应用程序命名

安装 Laravel 后，你可能希望为自己的应用程序起个名字。默认情况下，app 目录所对应的命名空间为 App，并且 Composer 依据 [PSR-4 自动载入标准](http://www.php-fig.org/psr/psr-4/)来加载此目录下的文件。不过，你可以将命名空间修改为应用程序的名字。执行 Artisan 的 app:name 命令即可完成更改。

例如，如果你的应用被命名为 "Horsefly"，你可以在应用程序的根目录下执行以下命令：

php artisan app:name Horsefly

重命名你的应用不是必须的。如果你愿意，你完全可以保留默认的 App 作为命名空间。

[维护模式](http://www.golaravel.com/laravel/docs/5.1/" \l "maintenance-mode)

如果你的应用处于维护模式，当有请求传入时，将显示一个自定义的视图。当你对应用做更新或维护操作时，这能让你非常方便地"关闭"应用。默认的中间件栈中包含了用于检查是否处于维护模式的方法。如果当前应用处于维护模式，一个带有 503 状态码的 HttpException 异常将被抛出。

要开启维护模式，只需简单地执行 Artisan 中的 down 命令即可：

php artisan down

要关闭维护模式，使用 up 命令即可：

php artisan up

在维护模式时响应请求的模板文件

用于在维护模式时响应请求的默认模板文件位于 resources/views/errors/503.blade.php。

维护模式和队列

当你的应用处于维护模式时，[队列任务](http://www.golaravel.com/laravel/docs/5.1/queues) 将不会被处理。关闭维护模式后，这些任务将继续正常处理。