TP VS Laravel

一、TP与laravel简介

1、Tp

ThinkPHP是一个快速、简单的基于MVC和面向对象的轻量级PHP开发框架，遵循Apache2开源协议发布，从诞生以来一直秉承简洁实用的设计原则，在保持出色的性能和至简的代码的同时，尤其注重开发体验和易用性，并且拥有众多的原创功能和特性，为WEB应用开发提供了强有力的支持。 3.2版本的服务器环境要求如下： PHP5.3以上版本

2、Laravel

Laravel是一个简单优雅的PHP Web开发框架，可以将开发者从意大利面条式的代码中解放出来，通过简单、高雅、表达式语法开发出很棒的Web应用，Laravel拥有更富有表现力的语法、高质量的文档、丰富的扩展包，被称为“巨匠级PHP开发框架”。 laravel框架的安装满足条件：

PHP版本 >= 5.5.9

PHP扩展：OpenSSL

PHP扩展：PDO

PHP扩展：Mbstring

PHP扩展：Tokenizer

Laravel需要依赖 Composer管理，必须首先安装Composer。

二、tp与laravel的目录结构

1、TP目录结构：

```
├──index.php 入口文件
├──README.md README文件
├──Application 应用目录
├──Public 资源文件目录
└──ThinkPHP 框架目录
```

核心代码目录：

```
├──ThinkPHP 框架系统目录（可以部署在非web目录下面）
|  ├──Common 核心公共函数目录
|  ├──Conf 核心配置目录
|  ├──Lang 核心语言包目录
|  ├──Library 框架类库目录
|  |  ├──Think 核心Think类库包目录
|  |  ├──Behavior 行为类库目录
|  |  ├──Org Org类库包目录
|  |  ├──Vendor 第三方类库目录
|  |  ├── ... 更多类库目录
|  ├──Mode 框架应用模式目录
|  ├──Tpl 系统模板目录
|  ├──LICENSE.txt 框架授权协议文件
|  ├──logo.png 框架LOGO文件
```

| ├──README.txt 框架README文件
| └──index.php 框架入口文件
① C层放在application/home/controller中，M层在application/home/model，视图放在application/home/view中。
② Tp里面使用的模板引擎是smarty，所以传值以及调用模板为assign和display。
③ 实现了模板输出的替换和过滤，如__APP__，__MODULE__，__URL__,__PUBLIC__等
④ 自带一些缓存的方法，如S （数据缓存） F（快速缓存） cache(查询缓存)以及静态缓存
⑤ 对网站安全性有多重防护，输入过滤（I方法）、上传安全、防止XSS攻击、防止sql注入、表单合法性检测（create方法创建数据对象的时候，可以使用数据的合法性检测，可以使用insertFields 和 updateFiel
ds属性或者field方法）等
⑥ ThinkPHP提供了灵活的全局配置功能，采用最有效率的PHP返回数组方式定义，支持惯例配置、公共配置、模块配置、调试配置和动态配置，分别讲述了在Apache、IIS、和nginx上的伪静态配置。‘ ⑦ thinkphp 也采用了命名空间的方法用来区分每个文件
⑧ 在ThinkPHP中基础的模型类就是Think\Model类，该类完成了基本的CURD、ActiveRecord模式、连贯操作和统计查询，一些高级特性被封装到另外的模型扩展中。
⑨ 基础模型类的设计非常灵活，甚至可以无需进行任何模型定义，就可以进行相关数据表的ORM和CURD操作，只有在需要封装单独的业务逻辑的时候，模型类才是必须被定义的。
⑩ ThinkPHP有专门为开发过程而设置的调试模式，开启调试模式后，会牺牲一定的执行效率，但带来的方便和除错功能非常值得。
11 tp中session使用的非常灵活在任何页面都能够随意调取，不需要重复的引入，只需要开启一次session可以了。 tp中不需要反复的写路由tp框架的s ession机制在配置中文件中配置好就可以用
12 使用系统内置的I函数是避免输入数据出现安全隐患的重要手段，I函数默认的过滤方法是htmlspecialchars，如果我们需要采用其他的方法进行安全过滤，有两种方式：

```
+----------------------------------------------------------------------+

| 'DEFAULT_FILTER'        =>  'strip_tags'

| 'DEFAULT_FILTER'        =>  'strip_tags,stripslashes',

+----------------------------------------------------------------------+
```

13如果你没有使用I函数进行数据过滤的话，还可以在模型的写入操作之前调用filter方法对数据进行安全过滤，

如 $this->data($data)->filter('strip_tags')->add();

2、Laravel目录:
app目录包含了应用的核心代码；

bootstrap目录包含了少许文件用于框架的启动和自动载入配置，还有一个cache文件夹用于包含框架生成的启动文件以提高性能；
config目录包含了应用所有的配置文件；

database目录包含了数据迁移及填充文件，如果你喜欢的话还可以将其作为 SQLite 数据库存放目录；

public目录包含了前端控制器和资源文件（图片、JavaScript、CSS等）；

resources目录包含了视图文件及原生资源文件（LESS、SASS、CoffeeScript），以及本地化文件；

storage目录包含了编译过的Blade模板、基于文件的session、文件缓存，以及其它由框架生成的文件，该文件夹被细分为成app、framework和logs子目录，app目录用于存放应用要使用的文件，framework目录用于存放框架生成的文件和缓存，最后，logs目录包含应用的日志文件；

tests目录包含自动化测试，其中已经提供了一个开箱即用的PHPUnit示例；

vendor目录是laravel的核心代码库，包含Composer依赖；

M层放在在app下面，不用再给专门的文件夹，app/Http下面有个routes.php，专门存放路由，每个控制器里的方法必须都要有，app/Http/Controller下面放的是控制器，resources/views下面放的是模版，以.blade.php为结尾。配置数据库信息可以直接在。Env文件中修改就可以了。

①laravel框架对环境有要求，5.2版本的laravel要求php5.5.9以上，还要开一些扩展。

②在浏览器中请求的路由通过app/Http/routes.php进行解析， 在进入相应的控制器方法，有get、post、any、match等一些方法
③传值、调用模板return view('user.profile', ['user' => $user]);

④设置session值：默认session只能在本方法中获取，可以在app/Http/kernel.php设置。

```
+------------------------------------------------------------------------+

| session(['chairs' => 7, 'instruments' => 3]);

| session()->get('key'); session()->put('key', $value);session( 'key' )

+------------------------------------------------------------------------+
```

也可以用$request->session()->get()/put()设置

Laravel的session的配置文件配置在app/config/session.php中session的永久保存（在不过期范围内）Session::put('key', 'value');

Session驱动（file，cookie,database,memcached,array）

配置文件为config/session.php

默认使用文件驱动

File session数据存储在storage/framework/session目录下

Cookie session数据储存在经过加密的cookie中

Database session数据储存在数据库中

Memcached/redis session数据存储在memcached/redis中

Array session数据存储在简单地数组中，非持久化（常用于运行测试）

⑤在AppServiceProvider中使用share方法可以是所有模板共享数据

⑥{{ $a}}是内置的输出数据的方法，默认被转义，可以使用{！！$a！！}输出原本代码而不会被转义

⑦@foreach @endforeach @if @elseif @else @endif

⑧模板继承可以在视图中使用同一个主模板，节省代码，方便管理

⑨数据库操作可以使用原生sql语句也可以使用查询构建器

三、Laravel与Tp的路由

1、laravel路由

laravel必须先定义路由，所有应用路由都定义在app/Http/routes.php 文件中。

默认情况下，routes.php 文件包含单个路由和一个路由群组，该路由群组包含的所有路由都使用了中间件组 web，而这个中间件组为路由提供了 Session 状态和 CSRF 保护功能。

而thinkphp中开启路由：“URL_ROUTER_ON”=true；

路由规则的定义格式为：’路由表达式’=>’路由地址和参数’。 2、TP路由

要使用路由功能，前提是你的URL支持PATH_INFO并开启路由路由规则的定义格式为：’路由表达式’=>’路由地址和参数’

四、Laravel与Tp基本配置与视图以及Laravel的中间件

1、laravel视图

视图文件存放在 resources/views 目录
判断视图是否存在 用 view()->exists() 在所有视图之间共享数据片段，这时候可以使用视图工厂的share方法，通常，需要在服务提供者的boot 方法中调用 share 方法，你可以将其添加到 AppServiceProvider 或生成独立的服务提供者来存放它们：

2、数据显示

Blade 的 {{}} 语句已经经过 PHP 的 htmlentities 函数处理以避免 XSS 攻击。 Blade 的 @include 指令允许简单的在一个视图中包含另一个 Blade视图，可以传递参数到被包含的视图@include(’view.name’, [’some’ => ’data’])

3、Tp视图

赋值assign调用页面display

在当前模版文件中包含其他的模版文件使用include标签，Include标签支持在包含文件的同时传入参数

4、laravel基本配置

Laravel框架的所有配置文件都存放在 config 目录下。

.env中的所有配置及其值被载入到 PHP 超全局变量 $_ENV 中

5、Tp配置：：

公共配置文件（默认位于Application/Common/Conf/config.php）。获取已经设置的据库连接 参数值：C(’参数名称’)

6、laravel的中间件

Laravel框架自带了一些中间件，包括维护模式、认证、CSRF 保护中间件等。所有的中间件都位于app/Http/Middleware 目录下。

可以将中间件看做 HTTP 请求到达目标动作之前必须经过的“层”，每一层都会检查请求并且可以完全拒绝它。分配中间件到路由

首先应该在 app/Http/Kernel.php 文件中分配给该中间件一个简写的 key，默认情况下，该类的 $routeMiddleware 属性包含了 Laravel 内置的入口中间件，添加你自己的中间件只需要将其追加到后面并为其分配一个 key：

中间件在 HTTP Kernel 中被定义后，可以在路由选项数组中使用 $middleware 键来指定该中间件：
Route::get(’admin/profile’, [’middleware’ => ’auth’, function () { // }]);

五、Laravel与Tp数据库

1、laravel数据库
Laravel 支持四种类型的数据库系统：MySQL Postgres SQLite SQL Server 可以配置读写分离

运行原生 SQL 查询
DB方法：select, update, insert, delete, 和statement。数据库事务所有的填充类都位于database/seeds目录

在run方法中，可以插入任何你想插入数据库的数据

可以使用 Artisan 命令db:seed来填充数据库
2、Tp数据库

目前的数据库包括Mysql、SqlServer、PgSQL、Sqlite、Oracle、Ibase、Mongo，也包括对PDO的支持。
六、 Laravel与Tp的防范机制

1、csrf攻击

跨站请求伪造是一种通过伪装授权用户的请求来利用授信网站的恶意漏洞。

2、防范机制

1）Laravel

Laravel 自动为每一个被应用管理的有效用户会话生成一个 CSRF “令牌”， 该令牌用于验证授权用户和发起请求者是否是同一个人。

想要生成包含 CSRF 令牌的隐藏输入字段，可以使用帮助函数 csrf_field 来 实现： laravel中防止SQL注入：有封装的生成预处理SQL语句

加密

加密:Crypt::encrypt($request->secret)

解密:try {

$decrypted = Crypt::decrypt($encryptedValue);

} catch (DecryptException $e) {

}

2）Tp

tp中防止SQL注入：查询条件尽量使用数组方式，这是更为安全的方式；也可以使用预处理机制；

3、总结

laravel框架在安全性方面考虑的要比tp全。比如在一开始，不管是表单提交还是ajax提交的方式，只要是post提交，如果没有传token值，laravel是不允许提交的。这本身就是laravel框架设计好的规定，但tp没有。七、 Laravel与Tp的缓存机制

1、Laravel缓存

缓存配置位于config/cache.php
默认为文件缓存，对于大型应用，建议使用内存缓存和memcached或APC
缓存使用
使用Cache门面 use Cache
存:Cache::put('key', 'value', $minutes);
取:$value = Cache::get('key');
$value = Cache::get('key', 'default');(第二个是默认值)

2、Tp缓存

S()方法 设置读取删除

F()方法 设置读取删除

TP框架有着类似的分类机制，但是在使用某些模式时，需要对该模式的参数 进行配置。当然TP框架也有自己的文件缓存，即F方法，被称作是快速缓存。

ThinkPHP提供了方便的缓存方式，包括数据缓存、静态缓存和查询缓存等，支持包括文件方式、APC、Db、Memcache、Shmop、Sqlite、Redis、Eaccelerator和Xcache在内的动态数据缓存类型，以及可定制的静态缓存规则，并提供了快捷方法进行存取操作。八、 Laravel中的一些特色

1、云存储

存:Storage::put('file.jpg', $contents); 取:$contents = Storage::get('file.jpg'); 删:Storage::delete('file.jpg');

2、 在larave中还有特色就是它的artisan命令，用这个命令可以很方便的生成你需要的控制器、model层、中间件等，而不需要你去复制粘贴别的或者直接自己手敲出来，并且这个artisan命令你还可以自己定义，定义一个命令，规定一下这个命令想要执行的操作，注册一下这个命令，以后就可以使用这个命令实现你想要的操作。

laravel中还有数据迁移，这个是用来生成数据库的，但我感觉这个的话就像是一种更安全的数据备份，如果不小心删错了表，只需要重新执行以下数据迁移就行。

强大的rest router：用简单的回调函数就可以调用，快速绑定controller和router artisan: 命令行工具，很多手动的工作都自动了

可继承的模版，简化view的开发和管理

blade模版：渲染速度更快，
migration: 管理数据库和版本控制
测试功能貌似也挺强大的，俺没用过
composer也是亮点，现在新更新的项目都支持了
laravel自带了数据库管理migration、爆好用的artisan，tinker命令行，模型工厂等一堆周边工具约定大于配置，重模型，轻配置，轻控制器，重模型迁移
九、 Laravel与Tp的模版引擎

1、Laravel中的blade模板引擎

数据显示:可以通过两个花括号包裹变量来显示传递到视图的数据

三元运算符:有时候你想要输出一个变量，但是不确定该变量是否被设置，我们可以通过如下 PHP 代码：

{{ isset($name) ? $name : 'Default' }}

显示原生数据

默认情况下，Blade 的 {{ }} 语句已经通过 PHP 的 htmlentities 函数处理以避免 XSS 攻击，如果你不想要数据被处理，可以使用如下语法：

Hello, {!! $name !!}.

流程控制

If 语句可以使用 @if , @elseif , @else 和 @endif 来构造 if 语句，这些指令函数和 PHP 的相同：

@if (count($records) === 1)

I have one record!

@elseif (count($records) > 1)

I have multiple records!

@else

I don't have any records!

@endif

循环:除了条件语句，Blade 还提供了简单指令处理 PHP 支持的循环结构，同样，这些指令函数和 PHP 的一样： @for ($i = 0; $i < 10; $i++)

The current value is {{ $i }}

@endfor

@foreach ($users as $user)

This is user {{ $user->id }}

@endforeach

@forelse ($users as $user)

{{ $user->name }}

@empty

No users

@endforelse

@while (true)

I'm looping forever.

@endwhile

包含子视图:Blade 的 @include 指令允许你很简单的在一个视图中包含另一个 Blade 视图，所有父级视图中变量在被包含的子视图中依然有效：

@include('shared.errors')

注释::Blade 还允许你在视图中定义注释，然而，不同于 HTML 注释，Blade 注释并不会包含到 HTML 中被返回： {{-- This comment will not be present in the rendered HTML --}}

2、Tp的smarty模版引擎

Smarty是PHP的"半官方"的模板化引擎，从其主页的位置就可以看出。Smarty的作者是Andrei Zmievski和Monte Orte。它是在GNU宽通用公共许可（LGPL）下发布的，可能是最流行、功能最强大的PHP模板化引擎。

Smarty还提供了很多强大的功能，本章将讨论其中一部分，概括如下。

强大的表现逻辑。Smarty提供了适当的构造，能够有条件地计算和迭代地处理数据。虽然它本身实际上是一种语言，但语法很简单，设计人员可以很快地学会，而不需要预备的编程知识。

模板编译。为减少开销，Smarty在默认情况下将模板转换为可比较的PHP脚本，使得后续的调用速度更快。Smarty还非常智能，在内容改变后可以重新编译。

缓存。Smarty还提供了缓存模板的可选特性。缓存与编译不同的是，支持缓存不只是能生成缓存的内容，还能防止执行个别逻辑。例如，你可以指定缓存文档的生存时间，比如5分钟，在此期间可以忽略与该模板有关的数据库查询。高度可配置和可扩展。Smarty的面向对象架构允许修改和扩展其默认行为。此外，从一开始可配置性就是一个设计目标，为用户提供了很大的灵活性，通过内置方法和属性定制Smarty的行为。

安全。Smarty提供了很多安全特性，可以避免服务器和应用程序数据遭到设计人员有意或无意的破坏。记住，所有流行的模板化解决方案都遵循同样的一组相同的核心实现原则。与编程语言一样，学习了一种语言就可以更容易地掌握其他语言。因此，即使不使用Smarty，你也可以继续读下去，使你对这个解决方案有一个初步的认识，并始终强调一般性的模板化概念。 tp的也有自己的模板引擎,不过tp引入的模板不能继承上一个模板的变量,而且继承的模板中无法使用模板类.

十、 Laravel与Tp中的服务

1、Laravel邮件

配置文件config/mail.php

Mail::send方法

发送原生字符串邮件raw方法

邮件消息放到队列，使用Mail门面上的queue方法

延迟已经放到队列中邮件的发送，使用later方法

2、Tp邮件

Tp如果要发送邮件的话，只能专门去找一个邮件类加载进来进行使用十一、Laravel与TP的优缺点

1、Laravel
缺点

① 门槛高（设计模式，闭包）
② 中文资料少（多数人的硬伤）
③ 设计非常复杂（尽管很优雅）
④ 速度超级慢（国外网站很多压力并不大他们不在乎）
优点

① 语法更富有表现力

② 高质量的文档

③ 丰富的扩展包

④ 开源、托管在GITHUB上

2、Tp

优点

1、高级模型：可以轻松支持序列化字段、文本字段、只读字段、延迟写入、乐观锁、数据分表等高级特性。

2、视图模型：轻松动态地创建数据库视图，多表查询相对简单。

3、关联模型：让你以出乎意料的简单、灵活的方式完成多表的关联操作。

4、模板引擎：系统内建了一款卓越的基于XML的编译型模板引擎，支持两种类型的模板标签，融合了Smarty和JSP标签库的思想，支持标签库扩展。 通过驱动还可以支持Smarty、EaseTemplate、TemplateLite、Smart等第三方模板引擎。

5、缓存机制：系统支持包括文件方式、APC、Db、Memcache、Shmop、Eaccelerator和Xcache在内的多种动态数据缓存类型，以及可定制的静态缓存规则，并提供了快捷方法进行存取操作。

6、类库导入：ThinkPHP是首先采用基于类库包和命名空间的方式导入类库，让类库导入看起来更加简单清晰，而且还支持冲突检测和别名导入。为了方便项目的跨平台移植，系统还可以严格检查加载文件的大小写。

7、扩展机制：系统支持包括类库扩展、驱动扩展、应用扩展、模型扩展、控制器扩展、标签库扩展、模板引擎扩展、Widget扩展、行为扩展和模式扩展在内的强大灵活的扩展机制，让你不再受限于核心的不足和无所适从，随心DIY自己的框架和扩展应用。

8、URL模式：系统支持普通模式、PATHINFO模式、REWRITE模式和兼容模式的URL方式，支持不同的服务器和运行模式的部署，配合URL路由功能，让你随心所欲的构建需要的URL地址和进行SEO优化工作。

9、编译机制：独创的核心编译和项目的动态编译机制，有效减少OOP开发中文件加载的性能开销。ALLINONE模式更是让你体验飞一般的感觉。

10、ORM：简洁轻巧的ORM实现，配合简单的CURD以及AR模式，让开发效率无处不在。

11、查询语言：内建丰富的查询机制，包括组合查询、复合查询、区间查询、统计查询、定位查询、动态查询和原生查询，让你的数据查询简洁高效。

12、动态模型：无需创建任何对应的模型类，轻松完成CURD操作，支持多种模型之间的动态切换，让你领略数据操作的无比畅快和最佳体验。

13、分组模块：不用担心大项目的分工协调和部署问题，分组模块帮你解决跨项目的难题。

14、AJAX支持：内置AJAX数据返回方法，支持JSON、XML和EVAL格式返回客户端，并且系统不绑定任何AJAX类库，可随意使用自己熟悉的AJAX类库进行操作。

15、多语言支持：系统支持语言包功能，项目和模块都可以有单独的语言包，并且可以自动检测浏览器语言自动载入对应的语言包。

16、模式扩展：除了标准模式外，系统内置了Lite、Thin和Cli模式，针对不同级别的应用开发提供最佳核心框架，还可以自

定义模式扩展。

17、自动验证和完成：自动完成表单数据的验证和过滤，生成安全的数据对象。

18、字段类型检测：字段类型强制转换，确保数据写入和查询更安全。

19、数据库特性：系统支持多数据库连接和动态切换机制，支持分布式数据库。犹如企业开发的一把利刃，跨数据库应用和分布式支持从此无忧。

缺点

1.麻烦的URL路由 正常模式： URL -> URL路由 -> 将从URL解析得到的参数和请求传递给入口函数 TP的实现（以正则路由为例）： 正则 -> 入口文件 + 动态参数 如：'/^blog\/(\d+)$/' => 'Blog/read?id=:1' 简洁的实现： '/^blog\/(\d+)$/' => read // function read($id) {...} 此实现便于检查路由规则与入口函数的匹配性并使用户更加方便的使用解析后的URL参数，同时，TP的入口文件设计本就是多此一举。 2.2.糟糕的模型设计 *1 需要手动建立模型与数据库的关联 缺点：需要进行一系列不必要的配置，与定义模型后自动生成数据库表的行为相悖。 *2 对数据库的抽象远远不够 操作模型时...
