Laravel05

1. 数据库迁移
2. 简单介绍

Laravel框架管理数据结构的一种方式

1. 功能

实现框架数据库表结构的共享

1. 使用

《1.》创建类文件

Php artisan make:migration 名字

《2.》up方法

1. 创建表 test表

Schema::create('test', function (Blueprint $table) {

字段设置

});

1. 创建表字段

（1.）字段类型

$table->increments('id'); （主键）

$table->string(‘name’) 字符串类型

$table->integer(‘age’) 整形

$table->float(‘age’) 浮点型

$table->text(‘name’) 文本类型

$table->enum('sex',['m', 'w']); 枚举类型

$table->char('name', 4); 带有长度的 CHAR

1. )字段修饰

->autoIncrement() 自动递增的主键

->nullable($value = true) （默认情况下）此字段允许写入 NULL 值

->unsigned() 是否为无符号

->default($value) 默认值

->comment('my comment') 注释

1. )索引

$table->primary(‘id’) 主键

$table->index(‘password’) 一般索引

$table->unique(‘username’) 唯一索引

1. 设置引擎

$table->engine = ‘myisam’

$table->engine = ‘innodb’

《3.》down方法

Schema::drop('flights'); 删除表

《4.》命令

Php artisan migrate 执行迁移操作

Php artisan migrate:refresh回滚（先删除表在执行迁移）

《5.》记录表结构的变化(针对表里有数据，保存数据)

1. 检测表是否存在

if (Schema::hasTable('users')) {

//

}

1. 检测表中的字段是否存在

if (Schema::hasColumn('表名', '字段名')) {

//

}

1. 增加字段

Schema::table('表名', function (Blueprint $table) {

$table->string('email');

});

1. 修改字段

首先 composer require doctrine/dbal

$table->string('name', 50)->change();

1. 删除字段

$table->dropColumn('votes');一个

$table->dropColumn(['votes', 'avatar', 'location']);多个

1. 数据库填充
2. 介绍

Laravel 可以用 seed 类轻松地为数据库填充测试数据。

1. 位置

database/seeds

方式：

第1种方式

1. 创建文件 php artisan make:seeder 名字
2. 在文件里写填充代码

public function run()

{

DB::table('users')->insert([

'name' => str\_random(10),

'email' => str\_random(10).'@gmail.com',

'password' => bcrypt('secret'),

]);

}

1. 运行指令

php artisan db:seed --class=名字

第2种方式

1. 创建文件 php artisan make:seeder 名字

2）在文件里写填充代码

public function run()

{

DB::table('users')->insert([

'name' => str\_random(10),

'email' => str\_random(10).'@gmail.com',

'password' => bcrypt('secret'),

]);

}

3）在DatabaseSeeder文件中加入代码

4）运行php artisan db:seed

注意：

完成 seeder 类的编写之后，你可能需要使用 dump-autoload 命令重新生成 Composer 的自动加载器：

composer dump-autoload