05. 控制器操作

学习要点：

1.前置操作
2.跳转和重定向
3.空方法和空控制器

本节课我们来学习使用控制器的一些操作功能，比如前置、跳转重定向、空方法和空控制器操作。

一. 前置操作

1. 继承 Controller 类后可以设置一个$beforeActionList 属性来创建前置方法；

```
protected $beforeActionList = [
    'first',
    //one 方法执行不调用 second 前置
    'second' => ['except'=>'one'],
    //third 前置只能通过调用 one 和 two 方法触发
    'third'  => ['only'=>'one, two'],
];

protected function first()
{
    echo 'first<br/>';
}
```

2. 此时，我们可以分别 URL 访问不同的方法来理解前置的触发执行；

二. 跳转和重定向

1. Controller 类提供了两个跳转的方法，success(msg,url)和 error(msg)；

```
public function index()
{
    if ($this->flag) {
        //如果不指定 url，则返回$_SERVER['HTTP_REFERER']
        $this->success('成功！', '../');
    } else {
        //自动返回前一页
        $this->error('失败！');
    }
}
```

2. 成功或错误有一个固定的页面模版：'thinkphp/tpl/dispatch_jump.tpl'；
3. 在 app.php 配置文件中，我们可以更改自己个性化的跳转页面；

```
// 默认跳转页面对应的模板文件
'dispatch_success_tmpl'  => Env::get('think_path') .
```

```
'tpl/dispatch_jump.tpl',
'dispatch_error_tmpl'    => Env::get('think_path') .
'tpl/dispatch_jump.tpl',
```

4. 如果需要自定义跳转页面，可以使用如下的模版变量：

变量	说明
$data	要返回的数据
$msg	页面提示信息
$code	返回的 code
$wait	跳转等待时间 单位为秒
$url	跳转页面地址

三. 空方法和空控制器

1. 当访问了一个不存在的方法时，系统会报错，我们可以使用_empty()来拦截；

```
public function _empty($name)
{
    return '不存在当前方法：'.$name;
}
```

2. 当访问了一个不存在的控制器时，系统也会报错，我们可以使用 Error 类来拦截；

```
class Error
{
    public function index(Request $request)
    {
        return '当前控制器不存在：'.$request->controller();
    }
}
```

3. 系统默认为 Error 类，如果需要自定义，则在 app.php 配置文件中修改；

```
// 默认的空控制器名
'empty_controller'      => 'Error',
```