03. 模块设计

学习要点：

1.目录结构
2.空模块
3.单一模块
4.环境变量

本节课我们来了需要学习一下 ThinkPHP 模块配置和使用，如模块的操作、单一模块的开启等。

一. 目录结构

1. ThinkPHP5.1 默认是多模块架构，也可以设置为单模块操作；
2. 所有模块的命名空间以 app 这三个字母作为根命名空间（可通过环境变量更改）；
3. 手册摘入的结构列表：

```
├─application           应用目录（可设置）
│  ├─common             公共模块目录（可选）
│  ├─module1            模块1目录
│  │  ├─common.php      模块函数文件
│  │  ├─config          模块配置目录（可选）
│  │  ├─controller      控制器目录
│  │  ├─model           模型目录（可选）
│  │  ├─view            视图目录（可选）
│  │  └─ ...            更多类库目录
│  │
│  ├─module2            模块2目录
│  │  ├─common.php      模块函数文件
│  │  ├─config          模块配置目录（可选）
│  │  ├─controller      控制器目录
│  │  ├─model           模型目录（可选）
│  │  ├─view            视图目录（可选）
│  │  └─ ...            更多类库目录
```

4. 模块下的类库文件命名空间统一为：app\模块名；
5. 比如：app\index\controller\Index
6. 多模块设计在 URL 访问时，必须指定响应的模块名，比如：public/test/abc/eat；
7. 如果你只有 test 这一个模块时，你可以绑定这个模块，从而省略写法；
8. 打开 public/index.php 的文件，追加一个方法：

```
Container::get('app')->bind('test')->run()->send();
```

9. 此时，URL 调用就变成了：public/abc/eat；多模块时，则其它无法访问；
10. 如果你的应用特别简单，只有一个模块，一个控制器，那改写下追加的方法：

```
Container::get('app')->bind('test/abc')->run()->send();
```

11. 此时，URL 调用就变成了：public/eat；得到了极简；其它控制器则无法访问；

二．空模块

1. 可以通过环境变量设置空目录，将不存在的目录统一指向指定目录；
2. 在 config 目录下的 app.php 修改：

```
// 默认的空模块名
'empty_module'          => 'index',
```

3. 空模块只有在多模块开启，且没有绑定模块的情况下生效；

三．单一模块

1. 如果你的应用只有一个模块，那可以直接设置单模块；
2. 在 config 目录下的 app.php 修改：

```
// 是否支持多模块
'app_multi_module'      => false,
```

3. 目录结构可变为，手册摘入：

```
├─application          应用目录（可设置）
│  ├─controller        控制器目录
│  ├─model             模型目录
│  ├─view              视图目录
│  ├─ ...              更多类库目录
│  └─common.php        函数文件
```

4. URL 地址 public/index/one，即：控制器/操作；
5. 单一模块的命名空间也变更为：app/controller；

四．环境变量

1. ThinkPHP5.1 提供了一个类库 Env 来获取环境变量；

```
return Env::get('app_path');
```

系统路径	Env 参数名称
应用根目录	root_path
应用目录	app_path
框架目录	think_path
配置目录	config_path
扩展目录	extend_path
composer 目录	vendor_path
运行缓存目录	runtime_path
路由目录	route_path
当前模块目录	module_path