82．事件

学习要点：

1.事件

本节课我们来学习一下要使用一下系统提供的事件。

一．事件

1． 事件和中间件有一点相似，只不过事件更加的精准定位更细腻的业务场景；

2． 事件可定义：事件类、事件监听类、事件订阅类；

3． 我们先创建一个测试事件的类：TestEvent.php，手动创建一个测试类；

```
public function __construct()
{
    //注册监听器
    Event::listen('TestListen', function ($param) {
        echo '我是监听器，我被触发了！'.$param;
    });
}

public function info()
{
    echo '登录前准备！';
    Event::trigger('TestListen', 'ok');   //触发监听器
    event('TestListen');                  //助手函数触发
}
```

4． 我们也可以使用监听类来设计监听器，使用命令行创建；

```
php think make:listener TestListen
```

```
public function info()
{
    echo '登录前准备！';
    Event::listen('TestListen', TestListen::class); //这句可以定义到配置文件
    Event::trigger('TestListen');
}
```

5． 在 app/event.php 中，listen 是配置监听类的，配置方式如下：

```
'listen'    => [
    'TestListen'    =>  [\app\listener\TestListen::class]
],
```

6． 而监听类被触发会自动执行 handle()方法，实现监听功能；

```
public function handle($event)
{
    echo '我是监听类！'.$event;
}
```

7. 系统还内置了系统触发的事件，只要满足条件就会自动触发；

事件	描述	参数
AppInit	应用初始化标签位	无
HttpRun	应用开始标签位	无
HttpEnd	应用结束标签位	当前响应对象实例
LogWrite	日志write方法标签位	当前写入的日志信息
RouteLoaded	路由加载完成	无

8. 事件监听类，可以同时监听多个监听类，只要绑定到一个标识符即可；

```
'TestListen'    =>  [
        \app\listener\TestListen::class,
        \app\listener\TestOne::class,
        \app\listener\TestTwo::class
]
```

9. 对于需要多个监听，监听类不够灵活，而且类会创建很多，可以使用订阅类；
10. 订阅类就是将监听事件作为内部的方法用 on+方法名来实现；

```
php think make:subscribe UserSub
```

```
class UserSub
{
    public function onUserLogin(){
        echo '处理登录后的监听！';
    }
    public function onUserLogout(){
        echo '处理退出后的监听！';
    }
}
```

11. 然后，我们直接去 app/event.php 注册一下；

```
'subscribe' => [
    'UserSub'    =>  \app\subscribe\UserSub::class,
],
```

12. 然后，两个方法分别监听两个事件方法，直接调用方法名即可；

```
public function login(){
    echo '登录成功！';
    Event::trigger('UserLogin');
}
```

```
public function logout(){
    echo '退出成功！';
    Event::trigger('UserLogout');
}
```

13. 对于事件类，很少有场景需要使用它，毕竟系统提供的各种精确方案较多；

```
php think make:event UserEvent
```

```
class UserEvent
{
    public function __construct()
    {
        echo '我是事件类！';
    }
}

Event::trigger(new UserEvent());
```