设计模式

1. 五种常见的PHP设计模式：许多设计模式都鼓励使用松散耦合、在更改一个代码片段是可能会发生级联破坏、该问题在于紧密耦合、系统中某部分的函数和类严重依赖于系统中其他部分中的函数和类的行为和结构、在大型系统中、许多代码严重依赖于几个关键类、此时需要更改这些类可能会困难、这个时候采用工厂模式会比较方便、工厂模式是一种类、使用工厂类（该类中具有创建对象的方法）创建对象、而不直接使用new、此时如果你想更改所创建对象的类型时、只需要更改工厂即可、当我们创建一个函数的时候如果函数体内有返回值比如return $id（此时该函数可以充当参数）、
2. class User{
3. public function getName(){
4. return "yungege";
5. }
6. }
7. class UserFactory
8. {
9. public static function Create( )
10. {$id = new User();//后续加上的将下边换成变量
11. return new User( );
12. }
13. }
15. $uo = UserFactory::Create( );
16. echo( $uo->getName()."\n" );
17. 单例模式（单元素模式）、观察者模式、策略模式、命令链模式。模式实际上在大型代码库中发挥作用的、单例模式
18. 有一种工厂模式的变体使用工厂方法。类中的这些公共静态方法构造该类型的对象。如果创建此类型的对象非常重要，此方法非常有用。例如，假设您需要先创建对象，然后设置许多属性。此版本的工厂模式会将该进程封装在单个位置中，这样，不用复制复杂的初始化代码，也不必将复制好的代码在在代码库中到处粘贴。该变体类中有2个创建对象方法只要将其中一个实例化那么另一个可以采用对象链调用（等价于对象的对象）、DatabaseConnection::get()->handle()其实这就是对象链双对象调用方法、某些应用程序资源是独占的，因为有且只有一个此类型的资源。例如，通过数据库句柄到数据库的连接是独占的。您希望在应用程序中共享数据库句柄，因为在保持连接打开或关闭时，它是一种开销，在获取单个页面的过程中更是如此。单元素可以满足此要求、
19. 观察者模式：观察者模式为您提供了避免组件之间紧密耦合的另一种方法。该模式非常简单：一个对象通过添加一个方法（该方法允许另一个对象，即观察者 注册自己）使本身变得可观察。当可观察的对象更改时，它会将消息发送到已注册的观察者。这些观察者使用该信息执行的操作与可观察的对象无关。结果是对象可以相互对话，而不必了解原因。
20. 设计模式，是一种解决问题的思维，而并非某种特定的方法。是前人给我们总结的宝贵经验。学习设计模式是为了编写可复用、可拓展、高性能软件。设计模式是熟练运用OOP后自然而然形成的代码习惯。达到最高境后只有一句话：高内聚、低耦合。说这么多其实总结一句话，设计模式 就是优雅的 最佳实践。
21. 想要成为一名高级程序员，设计模式是必须完全掌握的。我们经常看到关于java，c#设计模式的讲解，却很少看到用php代码讲解设计模式的， 这是为什么呢。jave、c#它们是纯面向对象编程的语言，纯面向对象的编程语言是以类为基本单位，把所有功能封装在类中，真正实现数据和业务逻辑的封装。而设计模式是面向对象编程的高级实践，所以设计模式是在这些纯面向对象语言中最早总结出来的。php本身是一种面向过程编程的的语言，PHP 5中借鉴了java的一些特性开始对面向对象支持更加完善，设计模式也可以用与php中了，
22. 工厂模式是使用工厂方法生成一个对象、而不是用new生成对象、切记工厂类中的工厂方法是使用其他类实例化对象而非自身类比如、class Factory{
23. static public function createDb(){
24. $db = new Db();
25. return $db;
26. }
27. }到时候直接调用该静态方法即可、工厂模式的优势是：比如说我们项目中的很多处都使用了new 类生成对象、如果该类名字发生变化或者参数发生变化需要更改很多处、但是如果使用工厂方法可以直接修改类中的方法即可、单例模式使某个类的对象仅能创建一次，通常一个项目中会多次用的Db这个数据库连接类，如果在每个地方都调用工厂方法创建一个数据库连接类，这样是比较消耗资源的，我们只需要一个数据库连接，单例模式来解决这个问题的。我们打开Db类,首先把构造方法设置为私有的,这样就禁止了在其他地方直接new我们的Db类、Db类的构造方法设置为了私有的,那我们在工厂类中也不能直接new了、可以直接将工厂类中的工厂方法设置成直接调用静态方法比如db::getName()将其存储在某一个变量中直接返回即可、其实所谓的单例模式也就是结合工厂模式一起用、其实这些基础设计模式都依赖于工厂模式这样效率会更高、模块化设计可二次开发、