MVC 初探

一、实验简介

1.1 实验目的

本次实验主要为大家介绍 MVC 的相关知识：什么是 MVC，MVC 原理，MVC 的优点，以及为什么要制作 MVC 框架。通过了解这些知识，可以对 MVC 有一个大致的了解，方便后续课程的展开。

1.2 知识点

- MVC 基础

二、MVC 基础

2.1 什么是 MVC

参考：维基百科

MVC 模式（Model–view–controller）是软件工程中的一种软件架构模式，把软件系统分为三个基本部分：模型（Model）、视图（View）和控制器（Controller）。

MVC 模式的目的是实现一种动态的程序设计，使后续对程序的修改和扩展简化，并且使程序某一部分的重复利用成为可能。除此之外，此模式通过对复杂度的简化，使程序结构更加直观。软件系统通过对自身基本部分分离的同时也赋予了各个基本部分应有的功能。专业人员可以通过自身的专长分组：

- 控制器（Controller）- 负责转发请求，对请求进行处理。
- 视图（View） - 界面设计人员进行图形界面设计。

- 模型（Model） - 程序员编写程序应有的功能（实现算法等等）、数据库专家进行数据管理和数据库设计(可以实现具体的功能)。

2.2 MVC 运行原理

参考：维基百科

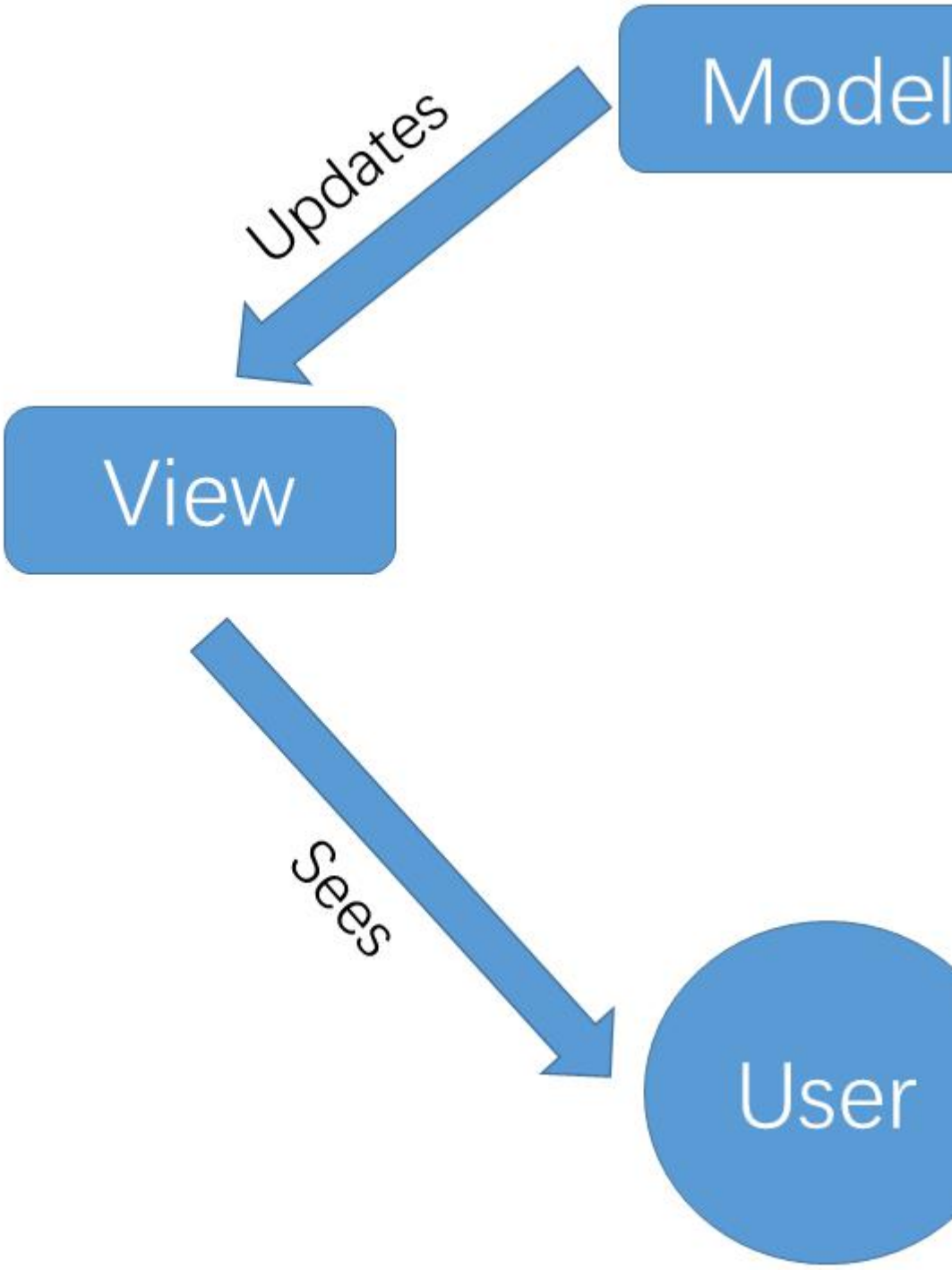
将应用程序划分为三种组件，模型 - 视图 - 控制器（MVC）设计定义它们之间的相互作用。

- **模型（Model）** 用于封装与应用程序的业务逻辑相关的数据以及对数据的处理方法。“ Model ”有对数据直接访问的权力，例如对数据库的访问。“Model”不依赖“View”和“Controller”，也就是说， Model 不关心它会被如何显示或是如何被操作。但是 Model 中数据的变化一般会通过一种刷新机制被公布。为了实现这种机制，那些用于监视此 Model 的 View 必须事先在此 Model 上注册，从而，View 可以了解在数据 Model 上发生的改变。

- **视图（View）**能够实现数据有目的的显示（理论上，这不是必需的）。在 View 中一般没有程序上的逻辑。为了实现 View 上的刷新功能，View 需要访问它监视的数据模型（Model），因此应该事先在被它监视的数据那里注册。

- **控制器（Controller）**起到不同层面间的组织作用，用于控制应用程序的流程。它处理事件并作出响应。“事件”包括用户的行为和数据 Model 上的改变。

图解：



2.3 MVC 的优点

参考：维基百科

在最初的网页中，像数据库查询语句(SQL query)这样的数据层代码和像 HTML 这样的表示层代码是混在一起。虽然有着经验比较丰富的开发者会将数据从表示层分离开来，但

这样的良好设计通常并不是很容易做到的，实现它需要精心地计划和不断的尝试。MVC 可以从根本上强制性地将它们分开。尽管构造 MVC 应用程序需要一些额外的工作，但是它带给我们的好处是毋庸置疑的。

首先，多个 View 能共享一个 Model 。如今，同一个 Web 应用程序会提供多种用户界面，例如用户希望既能够通过浏览器来收发电子邮件，还希望通过手机来访问电子邮箱，这就要求 Web 网站同时能提供 Internet 界面和 WAP 界面。在 MVC 设计模式中， Model 响应用户请求并返回响应数据，View 负责格式化数据并把它们呈现给用户，业务逻辑和表示层分离，同一个 Model 可以被不同的 View 重用，所以大大提高了代码的可重用性。

其次，Controller 是自包含（self-contained）指高独立内聚的对象，与 Model 和 View 保持相对独立，所以可以方便的改变应用程序的数据层和业务规则。例如，把数据库从 MySQL 移植到 Oracle，或者把 RDBMS 数据源改变成 LDAP 数据源，只需改变 Model 即可。一旦正确地实现了控制器，不管数据来自数据库还是 LDAP 服务器，View 都会正确地显示它们。由于 MVC 模式的三个模块相互独立，改变其中一个不会影响其他两个，所以依据这种设计思想能构造良好的少互扰性的构件。

此外，Controller 提高了应用程序的灵活性和可配置性。Controller 可以用来连接不同的 Model 和 View 去完成用户的需求，也可以构造应用程序提供强有力的手段。给定一些可重用的 Model 、 View 和 Controller 可以根据用户的需求选择适当的 Model 进行处理，然后选择适当的的 View 将处理结果显示给用户。

2.4 为什么要自制 MVC 框架

在 PHP 的世界中，有着众多的框架，它们各有所长，各具特色。既有 Zend，Symfony，Laravel 等大型框架，也有 CodeIgniter，Slim Framework 等轻量级的框架，还有国

产的优秀框架： ThinkPHP 等。关于各种框架的优劣以及是否应该使用框架的争论不绝于耳。

一听到开发框架，很多人都觉得又在重复造轮子，并给出一大堆拒绝造轮子的理由。对于这一点，我并不反对，因为自己写框架至少需要花费两方面的成本：时间成本和学习成本。一个优秀的框架并不是一个人一朝一夕就能完成，而且所需要用到的知识量也是非常庞大的，一个优秀的框架往往是一个团队沉淀多年的成果。所以，我们完全可以站在巨人的肩上，看得更远。

但是，另外一方面，作为一名想要接触框架学习却又找不到方向的初级 PHPer，上面那些内容并不是我们现在需要考虑的事情。我们只需要从基础学起，由简入难。学习框架最好的办法就是了解其基本原理和运行流程，自己开发一个 MVC 框架，可以近距离的接触到这些内容，为以后学习大型框架打好基础。

本套课程的目的就是建立在以上目的的基础上，通过让大家亲手开发一个 MVC 框架，实践 PHP 编程基础，学习面向对象编程的思想，了解 MVC 的开发模式和运行流程。