PHP

Day 01

1. 基本概念与介绍

1.1 路由与网络模型

什么是服务器( server ), ====== 》 就是用来提供服务的机器. ，就是一台电脑, 它是运行给别人提供服务使用的: 屏幕共享的服务器

**特征:**

1) 一般服务器性能都非常的高: 一般都是为了长期运行进行优化的

2) 所以服务器的价格一般较高

3) 服务器一般没有绚丽的界面, 甚至没有独立显卡

总的来说, 服务器一般就是一套性能很高的电脑

**客户机( client )**

在客户机上使用客户端( 浏览器 )来访问服务器上的资源

客户端不仅仅有浏览器, 还有 QQ, 微信, 网盘

想要组成一个网络必须有一个服务器和一个客户端

使用网线将 服务器 与 客户机 链接起来就构成一个最基本的网络了

路由器( 交换机 ), 它可以将多台计算机全部链接到路由器上, 由路由器进行一个集中转发.

一个路由一般网口是有限的, 所以为了连接的电脑, 我们可以将电脑进行分组连接到不同的路由( 交换机 )上

然后将路由器( 交换机 )连接起来. 这个样子所有的电脑就可以连接起来了

(了解)

局域网( LAN ), 局域网是一个局部范围可以互联的网络环境.

城域网, 在一个城市范围内将很多局域网连接起来就构成了城域网( 中间概念 )

广域网( WAN ), 比城域网的概念还大的网络环境一般就是广域网.

互联网, 将全世界的网络全部都连接起来, 就是互联网

2.2 服务器, 客户端

利用 IP 地址可以将我们自己的电脑当做服务器来使用( 我们的计算机即作服务器又作客户机 )

另外: 手机如果想要连接到电脑所在的网络:

1) 如果是在教室, 需要使用一个独立的网线路由接入教室的网络, 否则教室的电脑的网络环境与手机的网络环境不一样.

2) 一般在家中, 都有一个路由连接家用的设备. 此时电脑与手机就同在一个网络环境中.

2.3 IP, 域名, DNS 与端口

IP 地址是在网络环境中找到电脑的方式, 是电脑的地址( 有点像 真实环境的地址一样 )

1) IP 地址的基本结构: IPv4, IPv6

IPv4 是采用 点分十进制数的方式进行表示, 例如 255.255.255.255

每一个数字只允许使用 0 到 255 共 256 个数字

其中有一个非常特殊的 IP 地址: 127.0.0.1 称为回环地址, 指的是本机.

我们使用 IP 可以在网络环境中唯一的找到一台计算机

2) IP 是一串数字, 是不容易记忆的. 因此有人发明了一个 域名系统( Domain System )

其实域名系统就是一个键值对: 域名与 IP 对应的键值对

我们在访问网络的时候, 在浏览器中输入的是域名.

我们在访问网站的时候会先走一个域名服务器, 然后域名服务器将 域名( 名字 ) 转换成 IP, 然后再去找计算机.

这个 域名服务器一般称为 DNS 服务器.

3) 如果每次访问网站 都走一次 DNS 那么每一个网站请求就会进行两次操作

我们每一台计算机中都有一个文件 hosts 文件. 可以将其看成本地的 DNS.

在 win:

系统:/windows/System32/drivers/etc

在 Mac:

桌面上 按下 cmd + g, 输入 /etc 就可以找到这个文件了

如果在本地的 hosts 文件中提供了 ip 与域名的 对应关系, 浏览器在访问的时候就会首先使用这个对应关系, 不去 DNS

除非在 hosts 中没有这个 ip 与 域名的对应关系, 浏览器就会走 DNS.

在 hosts 中 # 是注释 与 js 中 // 的作用一样

在 hosts 中 IP 写在前面, 域名写在后面

书写域名的时候, 只包含域名, 不包含 协议, 端口 与 路径.

4) 端口:

端口是一个数字, 其功能就是在标注软件

简单的比喻一下: ip 地址好比一栋大楼的 门牌号. 如果要在大楼中找到某一个人, 就需要在大楼中提供楼层好与房间号.

一台计算机中可以安装多个软件: 飞q, 公屏, 网站, ftp 等.

利用 IP 只能找到电脑, 但是电脑中有多个软件, 每一个软件就好比一个住在大楼中的人.

每一个人住在房间中, 有一个唯一的房间号, 楼层号

每一个软件也有一个编号, 就是端口号

**http 服务器使用的端口号默认是 80**

2.4 URL

url 就是 统一资源定位符: 就是网络上的文件的地址

**基本格式( 重点 ): 协议://域名或IP:端口/路径?参数#锚点**

协议: 网站采用的协议有 http 协议( 超文本传输协议 ) 和 https 协议

这两个协议是专门用于网站的

https 协议可以简单的理解为加密后的 http 协议

ftp 协议( 文件传输协议 ), 专门用于文件的上传与下载

协议的作用其实就是在约定文件传输的方式

http://127.0.0.1/

在 web 服务器上使用 "协议://域名或IP:端口" 访问到的就是网站的根目录

一般将 **协议 + 域名( IP ) + 端口 统称为 网络三要素**

3. 软件的安装

注意事项:

1) 不要有中文路径

2) 不要有空格

关于端口与默认网站根目录与默认页面都可以使用 客户端软件来修改( 不要直接修改配置文件 )

但是我们需要了解配置文件中的细节

1) 监听的端口: Listen

2) 网站根目录: DocumentRoot

3) 默认页: DirectoryIndex

在 mac 系统中, 服务器默认是没有开启的, 而且网站根目录是 没有访问权限的

启动 服务器:

-> 修改配置文件

-> 启动终端: 键入命令: $ sudo apachectl start ( 127.0.0.1 -> it's works )

4. PHPStudy 基本使用

4.1 搭建一个网站

4.2 搭建多个网站

-> 在 管理工具中 配置 多个站点

就是在创建不同的网站的根目录( 文件夹 )

注意: 多个网站的目录必须是主目录的子目录

-> 修改 hosts 文件( 注意这里不能使用 公网的服务器 )

-> 如果需要展开多个网站目录下的文件必须手动修改配置文件, 然后重启服务器

1) 找到 vhost.ini 文件

2) 在对应网站的 Directory 中的 Options 后面加上 Indexes ( 关闭文件 )

3) 手动重启服务器

4.3 静态网站与动态网站

我们已经具备的东西

1> 有服务器了 -> 我们可以搭建网站了

2> 我们已经学会基本的 HTML, CSS, JS 知识

缺点:

我们有了服务器就可以搭建网站, 但是我们的网站是需要提前写好的.

如果要修改文章的内容需要拿到源代码, 直接在源代码中修改

我们将这类提前写好的网站称为 静态网站.

今天我们看到的网站大部分都不是静态的, 今天的网站会有一些特征: 根据用户的需求生成对应的页面然后发回给用户的浏览器

这样在服务器端需要执行代码, 并生成页面的网站就是动态网站.

所以我们需要学习一门可以在服务器上运行的语言

这样的语言很多:

1> Java

2> C#

3> C

4> Python

5> Go

6> Ruby 等等

7> PHP

PHP 和 js 一样是一个**解释型的脚本语言**. 需要运行 php, 我们的服务器就需要提供 php 引擎.

5. PHP 基本语法( 基本标签写法与输出 )

5.1 基本标签语法

php 代码写在一个 后缀为 php 的文件中

在 php 的文件中需要提供 有一个 标签 **<?php PHP 代码 ?>**

php 的代码就写在 这个标签的里面

注释: 和 js 一模一样

// 行注释 /\* 块注释 \*/ # 行注释( 少用 )

5.2 输出

echo 语法: 其作用与 js 中的 console.log() 类似

注意: 它只能输出字符串, 对于复杂数据使用 vat\_dump 来输出

6. 数据类型与变量

6.1 基本数据类型

js 中的数据类型

- 基本类型: 数字( number ), 字符串( string ), 布尔( boolean )

- 复合类型: 对象( 数组, 对象, 正则表达式, 时间等 ), 函数

- 空类型: null, undefined

常见的 PHP 数据类型与 JavaScript 基本一致( 8 种 )：

1. - string（字符串） ====== PHP 有多种创建字符串的方式：单引号、双引号等。

- 单引号字符串

- 不支持特殊的转义符号，例如 `\n`

- 如果要表示一个单引号字符内容，可以通过 `\'` 表达

- 如果要表示一个反斜线字符内容，可以通过 `\\` 表达

- 双引号字符串

- 支持转义符号

- 支持变量解析

2.- integer（整型）—— 只能存整数

3.- float（浮点型）—— 可以存带小数位的数字

4.- boolean（布尔型）

5.- array（数组） ===== php中数组不是对象

**1.- 索引数组** 与 JavaScript 中的数组基本一致

// 定义一个索引数组

$arr = array(1, 2, 3, 4, 5);

var\_dump($arr);

// PHP 5.4 以后定义的方式可以用 `[]`

$arr2 = [1, 2, 3, 4, 5];

var\_dump($arr2);

```

**2.- 关联数组** 有点类似于 JavaScript 中的对象

// 注意：键只能是`integer`或者`string`

$arr = array('key1' => 'value1', 'key2' => 'value2');

var\_dump($arr);

// PHP 5.4 以后定义的方式可以用 `[]`

$arr2 = ['key1' => 'value1', 'key2' => 'value2'];

var\_dump($arr2);

6.- object（对象, 略）

7.- 资源类型( 了解 )

8.- NULL（空）

6.2 使用变量

PHP 中申明一个变量是\*\*用一个美元符号后面跟变量名来表示\*\*。变量名同样是区分大小写的。

PHP 中变量无需声明类型，变量的类型根据值的类型来推断。

6.3 常量的定义

7. 运算符与流程控制

7.1 使用运算符与表达式

7.2 类型转换

$str = '132';

// 将一个内容为数字的字符串强制转换为一个整形的数字

$num = (int)$str;

// 将一个数字强制转换为布尔值

$flag = (bool)$num;

7.3 分支

7.4 循环

8. 函数与作用域

8.1 函数的定义与调用

8.2 局部变量与全局变量

关于变量作用域这一点，PHP 与绝大多数语言也都不同：\*\*默认函数内不能访问函数所在作用域的成员。\*\*

8.3 超全局变量

PHP 中的许多预定义变量都是“超全局的”，这意味着它们在一个脚本的全部作用域中都可用。在函数或方法中无需执行 global $variable; 就可以访问它们。

这些超全局变量是：

- $GLOBALS — 引用全局作用域中可用的全部变量

- $\_SERVER — 获取服务端相关信息

- $\_REQUEST — 获取提交参数

- $\_POST — 获取 POST 提交参数

- $\_GET — 获取 GET 提交参数

- $\_FILES — 获取上传文件

- $\_ENV — 操作环境变量

- $\_COOKIE — 操作 Cookie

- $\_SESSION — 操作 Session

定义常量：

// 定义常量使用的是内置的 `define` 函数

// 第一个参数是常量的名称，建议采用全大写字母命名，多个单词下划线分隔

// 第二个参数是常量中存放的数据，可以是任意类型

// 第三个参数是常量名称是否区不分大小写，默认 false 区分大小写

define('SYSTEM\_NAME', '阿里百秀');

define('SYSTEM\_ENABLE', true);

Day 02

**请求在网站的访问过程中很常见，并且请求分为多种方式：GET、POST、PUT、DELETE、OPTIONS等。其中GET和POST最常用。**

1.常见的发起GET请求的方式有： - 地址栏访问 - src/href - 表单

2.网页中有许多效果使用的就是GET请求：

- 各种搜索框 - 各种详情页面 - 大部分请求都是GET请求。。。

3.PHP 中有三个超全局变量专门用来获取表单提交内容：

1> - `$\_GET`：用于获取以 GET 方式提交的内容，更标准的说法：接收 URL 地址问号参数中的数据

2> - `$\_POST`：用于获取以 POST 方式提交的内容，更标准的说法：接收 请求体 中的数据

3> - `$\_REQUEST`：用于获取 GET 或 POST 方式提交的内容

1. **action` 提交地址指的是这个表单填写完成过后点击提交**，发送请求的请求地址是什么。

// 表单提交请求 if ($\_SERVER['REQUEST\_METHOD'] === 'POST') {

1. `**method` 可以用于设置表单提交的方式，目前我们所认识的就是最常见两种表单提交方式：`GET` 和 `POST`。**

- GET

- \*\*表单数据是通过 URL 中的 ? 参数传递到服务端的\*\*

- 可以在地址栏中看到提交的内容

- 数据长度有限制，因为 URL 地址长度有限（2000个字符）

绝不能使用 GET 来发送密码或其他敏感信息！！！

- POST

- \*\*表单数据是通过请求体传递到服务端的\*\*，我们在界面上看不到

- 可以提交任何类型的数据，包括文件

- 由于界面上看不见，浏览器也不储存，所以更安全

1. **`name` 属性值作为键，用户填写的信息作为值，发送到服务端。**

**7.PHP 中引入其他 PHP 文件有四种方式：**

- require

- require\_once

- include

- include\_once

**区别：**

- 横向分为两类：require 和 include 两种，区别在于 require 会因为载入文件不存在而停止当前文件执行，而 include 不会。

- 纵向分为两类：xxx 和 xxx\_once，区别在于代码中每使用一次 xxx 就执行一次载入的文件，而 xxx\_once 只会在第一次使用是执行。

- include 一般用于载入公共文件，这个文件的存在与否不能影响程序后面的运行

- require 用于载入不可缺失的文件

- 至于是否采用一次载入（once）这种方式取决于被载入的文件

**Day 03**

JSON（JavaScript Object Notation）

1. 是一种通过普通字符串描述数据的手段，用于表示有结构的数据。类似于编程语言中字面量的概念，语法上跟 JavaScript 的字面量非常类似。
2. **注意：**
   1. 1. JSON 中属性名称必须用双引号包裹
   2. 2. JSON 中表述字符串必须使用双引号
   3. 3. JSON 中不能有单行或多行注释
   4. 4. JSON 没有 `undefined` 这个值

**http**

HTTP（HyperText Transfer Protocol，超文本传输协议）最早就是计算机与计算机之间沟通的一种标准协议，这种协议限制了通讯\*\*内容的格式\*\*以及各项\*\*内容的含义\*\*。

**约定形式**

1. 客户端通过随机端口与服务端某个固定端口（一般为80）\*\*建立连接\*\* 三次握手

2. 客户端通过这个连接\*\*发送请求\*\*到服务端（这里的请求是名词）

3. 服务端监听端口得到的客户端发送过来的请求

4. 服务端通过连接响应给客户端状态和内容（响应报文）

请求报文：

请求行 ===== `GET /demo.php HTTP/1.1`

请求方式 + 空格 + 请求路径 + 空格 + HTTP 协议版本

请求头

请求体

响应报文：

状态行 ===== `HTTP/1.1 200 OK`

HTTP 协议版本 + 空格 + 状态码 + 空格 + 状态描述

响应头

响应体

常见状态码 ：

- 1xx：指示信息 —— 表示请求已接收，继续处理。

- 2xx：成功 —— 表示请求已被成功接收、理解、接受。

- 3xx：重定向 —— 要完成请求必须进行更进一步的操作。

- 4xx：客户端错误 —— 请求有语法错误或请求无法实现。

- 5xx：服务器端错误 —— 服务器未能实现合法的请求。

常见状态代码、状态描述的说明如下。

- 200 OK：客户端请求成功。

- 400 Bad Request：客户端请求有语法错误，不能被服务器所理解。

- 401 Unauthorized：请求未经授权，这个状态代码必须和 `WWW-Authenticate` 报头域一起使用。

- 403 Forbidden：服务器收到请求，但是拒绝提供服务。

- 404 Not Found：请求资源不存在，举个例子：输入了错误的URL。

- 500 Internal Server Error：服务器发生不可预期的错误。

- 503 Server Unavailable：服务器当前不能处理客户端的请求，一段时间后可能恢复正常。

**Day 04**

**1.Cookie**

HTTP 很重要的一个特点就是\*\*无状态\*\*（每一次见面都是“初次见面”），如果单纯的希望通过我们的服务端程序去记住每一个访问者是不可能的，所以必须借助一些手段或者说技巧让服务端记住客户端，这种手段就是 \*\*Cookie\*\*。

Cookie 就像是在超级市场买东西拿到的小票，由超市（Server）发给消费者（Browser），超市方面不用记住每一个消费者的脸，但是他们认识消费者手里的小票（Cookie），可以通过小票知道消费者之前的一些消费信息（在服务端产生的数据）。

**数据库 $sql 语句**

1. **增加 ======**

**insert into 表名（列名，列名。。。） values（‘值1’，‘值2’。。。）**

1. **删除 =======**

**delete from 表名 【where 子句】**

1. **修改数据 ======**

**update 表名 set 列名 = ‘值’，列名=‘值’。。。【where 子句】**

1. **查询语句 ====**

**（1.）以表格形式显示数据**

select 1,2,3

**（2.）使用from语句表示显示什么表格，在select中列出需要显示什么列，如果全部显示用\***

A . select 列名1，列名2，列名3.。。from 表名

1. Select \* from 表名

**（3.）默认只有select 与 from 会将所有的数据取出，**

select \* from 表名 where id = 1

**判断null 时不能用 = ，应该用字段 is null 或 is not null**

select \* from 表名 where 列名 is null

**（4.）模糊判断 通配符：%表示任意字符，\_ 表示一个任意字符**

字段 like ‘%语%’

select \* from 表名 where 列名 like ‘%语%’

**（5.）多条件判断 and 表示并且 or 表示 或者**

select \* from 表名 where 列名 like ‘%语%’ and 列名 like ‘%贵州%’

**（6.）范围查询**

**《1》连续范围 从xxxx ----> xxxx字段或表达式或函数between开始数据and结束数据**

Select \* from 表名 where 列名 between xxx and xxx

**《2.》离散范围 字段或表达式in (值1，值2，，，，)**

**Select \* from 表名 where列名 in （1984,1985）**

**（7.）order by 子句用来排序**

**（8.）limit 截取一部分数据显示 用于分页**

**分页规律 ： （页数 — 1 ）\* 每页条数**

**Day05**

假定数据库用户名：root，密码：123456，数据库：baixiu

//打开数据库

$connection = mysqli\_connect("localhost", "root", "123456", "baixiu");

// 释放结果集

mysqli\_free\_result($result);

// 关闭数据库

mysqli\_close($connection);

**字符串处理函数**

1.- 字符串截取

- string substr ( string $string , int $start [, int $length ] )

- string mb\_substr ( string $str , int $start [, int $length = NULL [, string $encoding = mb\_internal\_encoding() ]] )

2.- 字符串长度

- int strlen ( string $string )

- mixed mb\_strlen ( string $str [, string $encoding = mb\_internal\_encoding() ] )

3.- 大小写转换

- string strtolower ( string $string )

- string strtoupper ( string $string )

4.- 去除首尾空白字符

- string trim ( string $str [, string $character\_mask = " \t\n\r\0\x0B" ] )

5.- 查找字符串中某些字符首次出现位置

- mixed strpos ( string $haystack , mixed $needle [, int $offset = 0 ] )

- int mb\_strpos ( string $haystack , string $needle [, int $offset = 0 [, string $encoding = mb\_internal\_encoding() ]] )

6.- 字符串替换

- mixed str\_replace ( mixed $search , mixed $replace , mixed \$subject [, int & $count ] )

7.- 重复字符串

- string str\_repeat ( string $input , int $multiplier )

8.- 字符串分割

- array explode( string $char, string $input )

**数组处理**

1.- 获取关联数组中全部的键 / 值

- array\_keys() / array\_values()

2.- 判断关联数组中是否存在某个键

- array\_key\_exists()

3.- 将一个或多个元素追加到数组中

- array\_push()

- $arr[] = 'new value'

4.- 删除数组中最后一个元素

- array\_pop()

5.- 数组长度

- count()

6.- 检测存在

- in\_array()

7.- 获取元素在数组中的下标

- array\_search()