目录

- `PHPnow`(集成开发环境)的安装
  - `.../PHPnow/htdocs` 是服务器的默认根目录
  - 只要涉及到`ajax`,`json`,`php` 之类的 都要启动本地服务器用`localhost/`路径 访问
- PHP输出
  - `echo()` 输出数字、字符串
  - `print_r()`：可以输出数组键和值
  - `var_dump()` 会输出 元素数据类型 和 数组键值;
- php变量：`$arr = 10`
- php字符串拼接：js`+`，php是`.`

```
echo "我是".$username.",今年".$age."岁</br>";
echo "我是{$username},今年{$age}岁";
```

- php数组
  - 索引数组：下标是数字
  - 键值数组：下标是字符串
  - 全局数组：`$_GET` `$_POST`
  - 数组遍历：`foreach( arr as $key => $value )` `for`
- 数组的方法
  - `count(数组);`：返回数组中元素的个数。
  - `range('a', 'c',step);`：快速创建数组
- 合并数组
  - `+`号方式：先出现的将被保留，后出现的将被丢弃。
  - `array_merge(数组1，数组2);`：后出现的 字符键 会将前面的字符键覆盖，数字键 会从0数值键 重新排序。
- 分割数组
  - `array_chunk(array,size,preserve_key);`
    - `size`：分割的每组数组元素个数
    - `preserve_key`：`true`保留元素的键名。`false`：从`0`开始重新设置索引

1.服务器安装和入门

- 软件架构
  - `C/S` (客户端 -> 服务器)
  - `B/S` (浏览器 -> 服务器)
- 哪些技术可以开发网站

  `php、jsp、asp、ruby、python、nodejs、c/c++`等

- `WAMP` 架构解读 `windows` 系统
  `windows` + `apache` + `mysql` + `php`

- `LAMP` / `LNMP` 结构 `Linux` 系统
  `Linux` + `apache` + `mysql` + `php`
  `Linux` + `nginx` + `mysql` + `php`

- 服务器 资源提供方
- 客户端 资源受益方

- PHPnow(集成开发环境)的安装：
  - PHPnow： `apache + mysql + php`
  - `.../PHPnow/htdocs` 是**服务器的默认根目录**，此文件夹的内容对外可见(通过IP或者域名可以访问)
    - `localhost` / `127.0.0.1` 访问当前电脑服务器
    - 访问时不写文件路径，默认访问index开头的文件
      - `index.php` `index.html` `index.jsp`
  - phpNow安装教程
    1. 解压文件夹 (内部还有压缩文件，要看到版本号的文件夹才可以)
    2. 在phpnow文件夹里面 `cmd` (路径是此文件夹)
    3. `init` 安装服务器 （过程中会 预设SQL密码）
       - 只要涉及到 `ajax` , `json` , `php` 之类的 都要用 `localhost/` 路径 访问

2.PHP语法

```
<?php
    header('content-type:text/html;charset=utf-8');
?>
```

- `php` 代码兼容 `html` 和 `css` 所有的代码。
- `php` 可以操作 数据库/ `html` / `css`
- 前后端分离进行开发
  - 后台开发工程师： `php` + `mysql` (`java` 属于后端)
  - 前端开发工程师： `html+css+javascript` 网站，我们能看到的部分。

- PHP输出
  - php的输出函数，如果语句中含有标签会自动解析。
    - `echo` 带 `()` 只能输出一个，不带括号可以输出多个(`,` 隔开)

```
echo "<h1> hello </h1>", "hello";
echo ("<h1> hello </h1>");
```

        - 可以输出数字，字符串。不能输出布尔值，`true`输出成`1`，`false`输出成空。

    - `print_r()`

      ```
      print_r("<h1>hello</h1>");
      ```

        - 用来输出数组的详细信息，包括数组的键和值（没有数据类型）

    - `var_dump()` 会输出 元素数据类型 和 数据基本信息;
        - 类似于js中的`console.log()`;测试程序

      ```
      var_dump(100);       //int(100)  整型
      var_dump("hello");  //string(5) "hello"
      ```

        - 可以输出所有的变量、数字、字符串、布尔、数组、对象，包括键、值和数据类型。

    - `print`：只能输出数字和字符串。print带括号和不带括号都只能有一个参数。输出成功会返回1，失败返回0.
- php的语法是非常严格，每一条**语句后面都必须加分号。**

3.php定义变量、字符串拼接

```
$username = "钢铁侠";
$age = 18;
```

- php字符串拼接：js`+`，php是`.`
    - php字符串可以使用`{}`变量;
        - ECMA6 `` `${}` ``

      ```
      echo "我是".$username.",今年".$age."岁</br>";
      echo "我是{$username},今年{$age}岁";
      ```

    - php中，利用`{$a*$b}`在字符串中计算两个数的值是不可行的。(利用`.$a*$b.`拼接)
- php的数据类型
    - `String`(字符串)、`Integer`(整型)、`Float`(浮点型)、`Boolean`(布尔值)、`Array`(数组)、`Object`(对象)、`NULL`(空值)
    - php中 数字拥有`int`(整型)、`float`(浮点型)、`double`(双精度)
- javascript 字符串中的中文 记为1字符，php 字符串中的中文 记为3个字符

4. if...else / switch / for循环 / 函数

- `if...else` 分支语句

```
if( 表达式1 ){
  // 表达式1为真执行;
}else if( 表达式2 ){

}else{

}
```

- `switch`

```
switch(变量){
  case 1:
    执行语句;
    break;
  case 2:
    执行语句;
    break;
  default:
    执行语句;
    break;
}
```

- `for()`循环， 注意`$`要一直跟着变量走

```
for($i = 0; $i < 5; $i++){
  echo $i."</br>";
}
```

- 函数

```
function printHello($num1){ //形参
  print "hello world {$num1} </br>";
}
printHello(10); //传参
```

5.php数组

- 在php中，有三种类型的数组:
  1. 数值数组：下标是数字 叫索引数组
  2. 关联数组(键值数组)：下标是字符串 叫关联数组 (类似于ECMA6 `Map/Set` 集合)
  3. 全局数组
     - `$_GET`： 接受通过`get`提交过来的所有数据
     - `$_POST`： 接受通过`post`提交过来的所有数据
  - 索引数组 和 关联数组 可以相互结合，组合成多维数组。
  - php中不可以通过 字面量(`var arr = []`) 生成数组;

```
$cars = [1,2,3]    // 错误
$cars = array("大众"); // 正确
```

- 索引数组

```
$cars = array("大众", "捷克");
```

- 关联数组(键值数组)，js的对象 相当于 php中的 键值数组

```
$cars = array("王五" => "打鱼的", "李四" => "种地的", "键" => "值");
```

- 输出数组方式
  - `echo / print()` 不能输出整个数组，但是可以输出对应下标的单个元素

```
// 输出数组的单个值(下标)
echo $arr1[0];
```

  - `print_r()` 可以输出整个数组，也可以输出对应下标元素

```
print_r($cars); //Array ( [0] => 大众 [1] => 捷克 )
print_r($cars[1]); //捷克
```

  - `var_dump()` 可以输出整个数组(包括数据类型/长度)，也可以输出对应下标元素和具体信息。

```
var_dump($cars); //array(3) { [0]=> string(6) "大众" [1]=> string(6) "捷克"}
var_dump($cars[1]) //strang(6) "捷克"
```

  - `for` 循环遍历 输出数组
    - 数组的长度 `count($cars)` 返回数组的长度。

```
for($i = 0; $i < count($cars); $i++){
  echo " 下标{$i} ,数据: {$cars[$i]} ";
}
```

  - `foreach( arr as $key => $value )` 遍历

```
foreach($cars as $key => $value){
  echo "下标{$key},数据{$value}";
}
```

- 二维数组 (数组中嵌套数组) 只不过php数组类型是三种

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

- 数组的方法
  - `count(数组);` ： 返回数组中元素的个数。
  - `range('a', 'c');` ： 快速创建数组
    - `range(low, high, step);`
      - `low`：规定数组元素的最小值。

        - `high`：规定数组元素的最大值。
        - `step`：规定元素之间的进制,默认是`1`。
    - 返回值：返回一个从`low~high`元素的数组

```
$arr = range('a', 'c');
print_r($arr); //Array([0] => 'a', [1] => 'b', [2] => 'c');
```

- 合并数组
    - `+`号方式

```
$arr1 = array(1 => 1,'c'=>3, 'c');
$arr2 = array(1 => 'n','d'=>4);
var_dump($arr1+$arr2);
//array(4) { [1]=> int(1) ["c"]=> int(3) [2]=> string(1) "c" ["d"]=> int(4) }
```

        - 无论是数字键还是字符键，只要**先出现的将被保留**，**后出现的将被丢弃**。
    - `array_merge(数组1，数组2);`

```
$arr1 = array("a"=>"PHP","b"=>"java",1=>"python");
$arr2 = array("c" =>"ruby","go","a"=> "swift");
$arr3 = array_merge($arr1,$arr2);
// array(5) { ["a"]=> string(5) "swift" ["b"]=> string(4) "java"
// [0]=> string(6) "python" ["c"]=> string(4) "ruby" [1]=> string(2) "go" }
```

        - **后出现的 字符键** 会将前面的字符键**覆盖**，**数字键 会从0数值键 重新排序。**
- 分割数组
    - `array_chunk(array,size,preserve_key);`
        - `array`：放入的数组
        - `size`：规定每个数组包含多少元素。
        - `preserve_key`：`true`保留元素的键名。`false`：从`0`开始重新设置索引

```
$arr1 = array(1,2,3,4,5);
$arr2 = array_chunk($arr1,2,true);
// array(3) {
//   [0]=> array(2){ [0]=> int(1) [1]=> int(2) }
//   [1]=> array(2){ [2]=> int(3) [3]=> int(4) }
//   [2]=> array(1){ [4]=> int(5) }
// }
```