一：什么是PHP

PHP是服务端的脚本语言

二：语法

1. **基本的PHP语法**

<?php ?>以“<?php”开头以“?>”结束。（标签内写PHP代码）

注意：PHP的每个代码行都**必须**以分号结束。分号是一种分隔符，用于把指令区分开。

1. **变量**

变量是指程序中使用的数值是可以变化的量,与常量(**一旦被定义就无法改变的**)相反。

语法：$x = 5 ($符号后接变量名,变量名字不能为空格。)

1. **变量的作用域**

Php有四种不同的作用域

1. local;
2. Global;(全局作用域)在局部作用域访问全局变量。全局变量保存在一个名字为$GLOBALS[index]的数组中，index为变量名。
3. Static;(防止函数执行结束后变量被删除,每次调用该函数时还保留着函数前一次被调用的值,该变量还是属于函数的局部变量。)
4. Parameter;

**4.PHP的echo和print语句**

作用：用来输出html内容。

4-1.echo/print/print\_r的区别。

区别：echo可以输出一个或者多个字符串，print只能输出简单变量的值，print\_r能输出复杂类型变量的值，数组，对象

Echo输出的速度比print快，echo是PHP语句没有返回值，print是PHP函数，有返回值，print返回为1（int类型），print\_r返回布尔值true。

Echo跟print是语言结构语句，使用时可以选择加或不加括号列如：echo/echo(),print/print()，print\_r是显示关于变量一些易于理解的信息如果给出的是string,integer,float将打印本身，如果是array或者object将打印出一定顺序的键和值。

**5.PHP数据类型**

String(字符串)/Interger(整型)/Float(浮点型)/Boolean(布尔型)/Array(数组)/Object(对象)/Null(空值)

Interger:整型“int”没有小数点的整数。可以用十进制，十六进制，八进制三种格式指定。

Float:浮点型，带小数部分的数字。

Array:数组,变量中储存多个值。例子：$num= array(‘1’,’2’,’3’)

Object:对象,对象也可以用于储存数据，必须用class来声明类对象，类是可以包含属性和方法的结构

备注:var\_dump()函数返回变量的数据类型和值

**6.常量**

常量是一个简单值得标识符。该值在脚本中不能改变。

define(constant\_name,value,case\_sensitive)函数可以设置常量，它有三个参数constant\_name：必填参数，常量名称，标识符，value:必选参数常量得值,case\_sensitive:可选参数指定是否大小写敏感，ture为不敏感

（注意:如果没有设置case\_sensitive默认为敏感,对常量名的大小写有区分在调用的时候）

**7.PHP字符串并置运算符**

字符串并置运算符(“.”)，把两个字符串合并起来

Strlen()函数输出字符串的长度；例子：echo strlen(‘Hellow’),输出6

Strpos()函数找一段字符串中的一个或者一段文本；例子：echo strpos(‘hellow world’,’world’)输出6

**8.Php比较运算符**

特殊（对比js）：’<>’不等于的意思；例子5<>8返回

**9.逻辑运算符**  
x and y:表示与 需要 x跟 y的取值都是true才返回true

X or y :表示或，只需要x或者y的取值只要一个是true就返回true  
x xor y:表示异或，只有 x跟y其中之一返回true才返回true，否者返回false;  
x || y：同x or y  
x && y: 同 x and y  
!x:表示 ‘非’，与当前取值相反。x = true; !x返回false

**10.数组**  
在PHP中用array()函数用于创建数组。  
三个数组类型：

1. **数值数组：**带有数字ID的数组。  
   2.**关联数组**：带有指定键的数组，一个键对应一个值  
   3.**多维数组**：包含一个或多个数组的数组  
   count()函数获取数组的长度。例子$num = array(‘1’,’2’,’3’); echo count($num);输出3。  
   备注:PHP中for循环遍历数组，数组长度语法为$numlength(与js中的for循环有差异)  
   **11.数组排序**  
   1.sort()对关联数组进行升序排序  
   2.rsort()对数组进行降序排序  
   3.asort()根据关联数组的**值**,对数组进行升序排列

4.ksort()根据关联数组的**键**,对数组进行升序排序  
5.arsort()根据关联数组的**值**,对数组进行降序排列  
6.krsort()根据关联数组的**键**,对数组进行降序排序  
**12.PHP超级全局变量**  
超级全局变量是在全部作用域中始终可用的内置变量。  
1.$**GLOBALS:**是PHP的一个超级全局变量组，在一个PHP脚本的一个全部作用域可以访问(**全局变量**)。变量的名字就是数组的键。

2.**$\_SERVER**:是一个包含了诸如头部信息(header),路径(path),以及脚本位置(script location)等信息的数组。这个数组的项目由**Web**服务器创建。  
3.**$\_REQUES**T:用于收集HTML表单提交的数据。  
4.**$\_POST**:用于收集表单数据,在HTML form标签指定该属性methon = ‘post’。  
13.**PHP命名空间**命名空间是一种封装事物的方法。  
**命名空间解决的两个问题**  
(1).用户编写的代码与PHP内部的类/函数/常量或第三方类/函数/常量之间的名字冲突。

(2).为很长的标识符名称创建一个别名。  
命名空间通常通过关键字namespace来声明，如果一个文件中包含命名空间，他必须在所有代码之前声明命名空间。

**PHP 命名空间中的类名可以通过三种方式引用：**

(1).非限定名称，或不包含前缀的类名称，例如 $a=new foo(); 或 foo::staticmethod();。如果当前命名空间是 currentnamespace，foo 将被解析为 currentnamespace\foo。如果使用 foo 的代码是全局的，不包含在任何命名空间中的代码，则 foo 会被解析为foo。 警告：如果命名空间中的函数或常量未定义，则该非限定的函数名称或常量名称会被解析为全局函数名称或常量名称。

(2).限定名称,或包含前缀的名称，例如 $a = new subnamespace\foo(); 或 subnamespace\foo::staticmethod();。如果当前的命名空间是 currentnamespace，则 foo 会被解析为 currentnamespace\subnamespace\foo。如果使用 foo 的代码是全局的，不包含在任何命名空间中的代码，foo 会被解析为subnamespace\foo。

(3).完全限定名称，或包含了全局前缀操作符的名称，例如，$a = new \currentnamespace\foo(); 或 \currentnamespace\foo::staticmethod();。在这种情况下，foo 总是被解析为代码中的文字名(literal name)currentnamespace\foo。

14.**PHP面向对象**对象是个由信息及对 进行处理的描述组成的整体。  
15.**PHP构造函数**

构造函数时一种特殊的方法，它用来创建对象时初始化对象，即为对象成员变量赋初始值，总与new运算符 一起使用在创建对象的语句中。  
**语法：**void \_\_construct([mixed $args [, $....]])  
**例子：function \_\_ construct($par1 $par2){**

**$this->url = $par1**

**$this->title = $par2**

**}  
16:折构函数**与构造函数相反，对象结束生命周期时（例如对象所在的函数已经调用完毕）**系统自动执行折构函数。**

**语法：**viod \_\_destruct(void)  
**17：继承**PHP使用extends来继承一个类。

**语法：class Child extends Parent{}** (Child继承Parent)

**18：访问控制**

PHP对属性或方法的访问控制,是通过在前面添加关键字 pubilc(公有)，protected(受保护的)或private(私有)来实现。  
**pubilc(公有的)：**公有类成员可以在任何地方被访问。  
**Protected(受保护的):**受保护的类成员则可以被其自身以及其子类和父类访问。  
**private(私有的):**私有成员只能被其定义所在的类访问。  
**19：接口**使用接口interface,可以指定某个定义类必须实现哪些方法，但是不需要定义这些方法的具体内容。接口中定义的所有的方法都必须是公有的。  
**语法：通过interface关键字来定义，要实现一个接口用implements操作符（类中必须实现接口中定义的所有方法），类可以实现多个接口，用逗号来分隔多个接口的名称。  
例子：interface iTemplate{**

**Public function setVariable($name,$var)**

**Pubilc function getHtml($template)**

**}//声明一个iTemplate接口**

**//实现接口**

**Class Template imlements iTemplate{**

**Private $vars = array()**

**Public function setVariable($name,$var){**

**$this->vars[$name] = $var**

**}**

**Public function getHtml($template){**

**Foreach($this->vars as $name => $value){**

**$template = str\_replace(‘{’ . $name . ‘}’,$value,$template)**

**}**

**Return $template**

**}**

**}**

**20:static(关键字)**

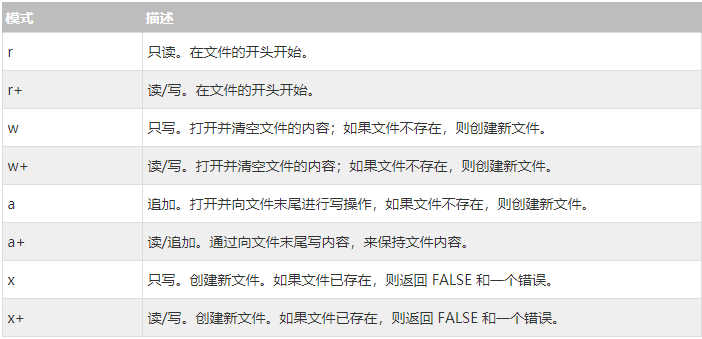
声明类属性或方法为 static(静态)，就可以不实例化类而直接访问。

21:**Final**PHP5 新增了一个final关键字。如果父类中的方法声明为final,则子类无法覆盖该方法，如果一个类被声明为final则不能被继承。  
22.**调用父类构造方法**php不会在子类的构造方法中自动调用父类的构造方法需要调用父类的**构造**方法，需要在子类的构造方法中去调用**parent::\_\_construct()**。

1. **PHP表单  
   1：PHP表单和用户输入**

PHP中的**$\_GET**和**$\_POST**变量用于检索表单中的信息，比如用户输入  
**2：PHP表单处理**当处理Html表单时PHP能把来自HTML页面的表单元素自动变成可供PHP脚本使用。  
**3：表单验证**$\_SERVER[“PHP\_SELF”]有可能会被黑客攻击（XSS通过HTML链接来攻击）,可以通过htmlspecialchars()把一些预定义的字符串转化为HTML实体。  
**使用PHP验证表单数据：**当用户提交表单的时，我们将做以下两件事  
（1）使用PHP trim()函数去除用户输入数据中不必要的字符(如：空格，tab,换行)  
（2）使用PHP stripslashes()函数去除用户输入数据的反斜杠  
**4:PHP表单必须字段**检查一个变量是否为空**empty($name)**.。  
**5.PHP表单-验证邮件和URL**preg\_match:进行正则表达式匹配。  
**6.PHP $\_REQUEST变量**预定义$\_REQUEST包含了 $\_POST，$\_GET，$\_COOKIE的内容，$\_REQUEST变量可用来收集通过GET或POST方法传送的表单数据。  
  
**三.PHP高级教程  
1.PHP多维数组**多维数组是包含一个或者多个数组的数组。

1. **PHP日期**PHP中data()函数用于格式化时间/日期。date()函数返回将整数timestamp按照给定的格式字串而产生的字符串  
   1.PHP Date() -格式化日期  
   date()函数中第一个参数是必须参数format规定了如何格式化日期/时间。**例如：echo date(“Y-m-d”)输出2020-04-24**2.PHP Dtae()添加时间戳  
   date()函数的第二个参数规定了一个时间戳，如果没有提供时间戳，将使用当前日期的时间戳。  
   mktime()函数可以为指定的日期返回Unix时间戳  
   **语法：mktime(hour,minute,second,moth,day,year,is\_dst)**3.**PHP包含文件**服务器在执行PHP代码之前插入一个文件内容。  
   include和require语句用于执行插入写在其它流中的有效代码。  
   **include和require的差别**1.require生成一个致命的错误(E\_COMPILE\_ERROR),在错误发生后脚本会停止执行。（有助于提高应用程序的安全性和完整性）  
   2.include生成一个警告(E\_WARNING),错误发生后脚本会继续执行。  
   对于include,在执行文件的时候每次都要进行读取和评估,而require文件只需要处理一次。  
   4.**PHP文件处理**1.fopen()函数在PHP中用于打开文件，**语法：fopen(fileName,openType)**有两个参数,第一个参数含有要打开文件的名称，第二个参数规定了用哪种方式打开文件。

  
提示：如果fopen()无法打开指定的文件，则返回0(false)  
2.**关闭文件fclose()函数用于关闭打开的文件  
例子：**$file = fopen(‘test.txt’,’r’); fclose($file)  
**3.检测feof()函数检测是否达到文件尾部（用于循环遍历未知长度的数据）  
语法：**if(feof($file)){echo ‘End of file’}  
**备注：**在打开模式是w,a,x的情况下无法读取打开的文件。  
**4.fgets()函数表示逐行读取文件  
备注：**在调用函数后文件指针会移动到下一行  
**例子：**$file = fopen(“welcome.txt”, ‘r’) or exit (“Unable to open file”);  
 while(!feof){  
 echo fgets($file)

}

fclose($file)  
  
**5.fgetc()函数逐字读取文件**

**例子：**$file = fopen(“welcome.txt”, ‘r’) or exit (“Unable to open file”);  
 while(!feof){  
 echo fgetc($file)

}

fclose($file)  
  
**5.PHP文件上传**前端通过input标签的type=”file”属性来传送文件  
PHP通过$\_FILES来获取前台传过来的数据，$\_FILES的第一个参数是表单的input标签的name属性的值,第二个参数可以是name,type.....等等,例如：  
$\_FILES["file"]["name"] - 被上传文件的名称

$\_FILES["file"]["type"] - 被上传文件的类型

$\_FILES["file"]["size"] - 被上传文件的大小，以字节计

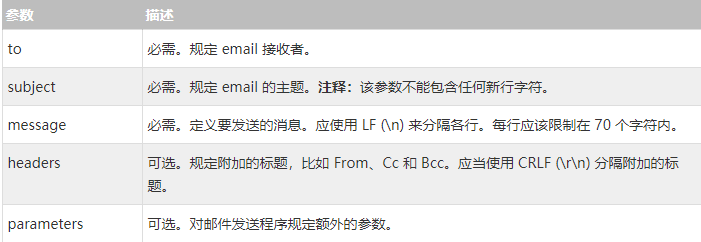
$\_FILES["file"]["tmp\_name"] - 存储在服务器的文件的临时副本的名称

$\_FILES["file"]["error"] - 由文件上传导致的错误代码  
文件通过PHP **move\_uploaded\_file()**把文件拷贝到服务器的文件夹内

6.**PHP Cookies**cookies是一种在远程浏览器端储存数据并以此来跟踪和识别用户的机制。  
（1）.如何创建cookies  
**setcookie()**函数用于设置cookie，该函数必须位于html标签之前  
**语法：setcookie(name,value,expire,path,domain)，例子：**setcookie(“user”,”Alex Porter”,time()+3600)=>创建了一个名字为user的cookie并赋值Alex Porter,并且在一个小时后过期。  
**注释：**在发送cookie时会自动进行URL编码，在取回时进行自动解码，可以用**setrawcookie()**来设置cookie。

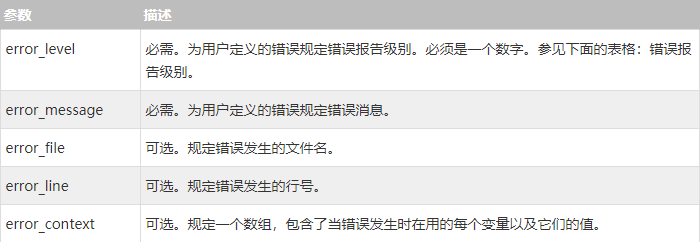
7.**PHP Sessions**PHP Session变量用于存储关于用户会话(session)的信息，或者更改用户会话（sessions）的设置。Session储存单一用户信息，并且对于应用程序中所有页面都是可用的。  
（1）.我们把用户信息储存到session中之前，首先必须启动会话。**语法：**session\_start();**备注：**session\_start()必须在**html**标签之前。

（2）储存和去除session变量最准确的方法是使用**PHP $\_SESSION**变量。

（3）销毁Session; **语法：unset(),session\_destroy()**，ubset()函数用于释放指定的session变量。Session\_destory()销毁全部已存的session。  
**8.PHP发送电子邮件**PHP允许直接从脚本发送电子邮件。PHP mail()函数用于从脚本中发送电子邮件。  
**语法：mail(to,subject,message,headers,parameters)**

**9.PHP安全E-mail**filter\_var(),指定一个过滤器过滤变量,**语法：**filter\_var(变量名)

**10.PHP错误处理**

(1).基本的错误处理方法**die(输出的内容);**(2).创建自定义错误处理器:error\_function(erro\_level,error\_message,error\_file,error\_line,error\_context),并设置错误处理程序，**语法：set\_error\_handler(设置的错误处理函数)**  
  
（3）.触发错误,**语法：trigger\_error(返回的结果,返回的错误类型)**（4）.错误记录：在默认情况下，根据在php.ini的error\_log配置，PHP向服务器的记录系统或文件发送错误信息。通过使用error\_log()函数可以向指定文件或远程目的地发送错误记录。  
**11.PHP异常处理**

异常指的是程序在运行中出现的不符合预期的情况，允许发生，但是属于一种不正常的情况。（属于业务逻辑上的错误）  
**异常被触发时经常会发生：**(1).当前代码被保存。  
(2).执行代码被切换到预定义(自定义)的异常处理函数。  
(3).根据情况，处理器也许会从保存的代码状态重新开始执行代码，终止脚本执行，或许从代码另外的位置继续执行脚本。  
**我们将展示不同的处理方式：**(1).异常的基本使用。  
(2).创建异常的自定义处理器  
(3).多个异常(4).重新抛出异常  
(5).设置顶层异常处理器  
**注释：**异常应该仅仅在错误的情况下使用，而不应该用于一个指定的点跳转到代码的另一个位置。  
**11.1异常的基本使用**当异常被抛出时,其后的代码不会继续执行,PHP会尝试查找匹配的“catch”的代码块。如果异常没有被捕获，而且又没有使用set\_exception\_handler()作相应的处理的话,将会发生一个严重的错误（致命错误），并输出“Uncaught Exception”(未捕获异常)的错误信息。  
为了避免上述情况应当创建的适当的代码来处理异常。  
**适当的处理异常代码应该是：**Try:使用异常的函数应该位于”try”代码块内，如果没有触发异常代码会继续执行，如果异常被触发，会抛出一个异常。  
Throw:规定如何触发个异常，每个throw对应着一个catch。

Catch.catch代码块会捕获异常，并创建一个包含异常信息的对象。  
**11.2创建一个自定义的Exception类**必须是exception类的一个拓展  
**例子：**class customException extends Exception{}，**customException是Exception的一个拓展  
11.3设置顶层异常处理器**set\_excption\_handler()函数可设置处理所有**未捕获异常的用户定义函数  
11.3异常的规则**

**1.需要进行异常处理的代码应该放在try代码块内,以便捕获潜在异常。**

**2.每个try或者throw代码块必须至少拥有一个对应的catch代码块。**

**3.使用多个catch代码块可以捕获不同种类的异常可以在try代码块的catch代码块中抛出(再次抛出)异常。  
12.PHP过滤器**

PHP过滤器用于验证和检测来自非安全来源的数据,比如用户输入。  
**12.1函数和过滤**如果需要过滤变量可以使用以下任意函数。

1.filter\_var() 通过一个指定的过滤器来过滤单一的变量。  
2.flter\_var\_array()通过相同或不同的过滤器来过滤多个变量。  
3.filter\_input 获取一个输入变量，并对它进行过滤。  
4.filter\_input\_array获取多个输入变量,并通过相同的或不同的过滤器对他们进行过滤。

**12.2Validating和Sanitizing**

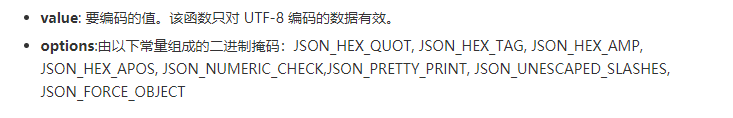
有两种过滤器：

Validating过滤器:  
1.用于验证用户输入  
2.严格的格式规则(比如URL或E-Mail验证)  
3.如果成功则返回预期类型，如果失败则返回false

Sanitizing过滤器:

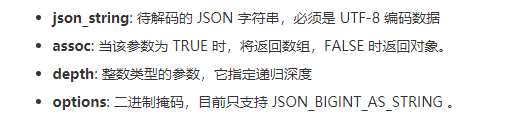
1.用于允许或禁止字符串中指定的字符  
2.无数据格式规则  
3.始终返回字符串  
**13.高级过滤器  
14.PHP JSON**php5.2.0以上版本已经具备内置JSON拓展  
**JSON函数**json\_encode对变量进行JSON编码。成功返回json数据，否则返回FALSE。

**语法：string json\_encode($value,[,options=0])**



Json\_decode对JOSN格式的字符串进行解码,转为PHP变量。

**语法：mixed\_json\_decode($json[,$assoc=false[,$depth = 521 [,$options = 0]]])**

  
json\_last\_error返回最后发生的错误。