

# 7天学会PHP

千锋教育李文凯 编著



- ✓ 本书完全免费且持续更新
- ✓ 海量代码实例下载
- ✓ QQ群学习交流
- ✓ 大牛不定期在线答疑
- ✓ 千锋官网同步视频支持

- 每章内容针对初学者精心设计，娓娓道来，浅显易懂
- 零门槛学习PHP真的很简单
- 老鸟带领菜鸟，快速跨入PHP大门

学习交流群：518146831、517729890、  
517739045

从零  
开始学

# 目 录

- 01. 为什么选择本书学习PHP
  - 1.1 为什么学习PHP ?
  - 1.2 PHP是什么
  - 1.3 零基础也能学习
  - 1.4 为什么有些人学不会
- 02.PHP的环境安装
  - 2.1开发环境是什么 ?
  - 2.2 windows环境安装
  - 2.3 Linux环境安装
  - 2.4 其他开发环境
  - 2.5 写代码的工具选择
- 03. PHP基本语法
  - 3.1 PHP基本语法
    - 3.1.1 写出你的第一段PHP代码
    - 3.1.2 读过初中你就会变量
    - 3.1.3 echo 显示命令
    - 3.1.4 注释的功能很强大
  - 3.2 数据类型并不神秘
    - 3.2.1 整型就是整数
    - 3.2.2 布尔就是易经的知识
    - 3.2.3 字符串
    - 3.2.4 浮点型
    - 3.2.5 重要：if和else语法
    - 3.2.6 NULL类型
    - 3.2.7对象以后会学
    - 3.2.8 数组会有单纯的一个章节
    - 3.2.9 资源类型
    - 3.2.10 眼前了解回调类型即可
    - 3.2.11 查看和判断数据类型
    - 3.2.12 数据类型的自动转换和强制转换
  - 3.3 常量和变量
    - 3.3.1 用常量限制用户跳过某些文件
    - 3.3.2 可变变量
    - 3.3.3 外部变量
    - 3.3.4 环境变量
    - 3.3.5 变量引用
  - 3.4 PHP表达式与运算符
    - 3.4.1 算术运算
    - 3.4.2 赋值运算
    - 3.4.3 自加、自减运算
    - 3.4.4 比较运算
    - 3.4.5 逻辑运算
    - 3.4.6 位运算
    - 3.4.7 运算符优先级

### 3.4.8 三元运算符和其它运算符

## 04. PHP中的流程控制

### 4.1 if条件结构流程

#### 4.1.1 if语句

#### 4.1.2 嵌套if...else...elseif结构

#### 4.1.3 if语句多种嵌套

### 4.2 分支结构switch语句的使用

### 4.3 循环语句的使用

#### 4.3.1 while循环

#### 4.3.2 do...while循环的区别

#### 4.3.3 for循环控制语句

#### 4.3.4 goto语法

#### 4.3.5 declare 语法

## 05.PHP的函数基本语法

### 5.1 自定义函数

### 5.2 自定义函数高级调用

#### 5.2.1 回调函数

#### 5.2.2 变量函数

#### 5.2.3 匿名函数

#### 5.2.4 内部函数

#### 5.2.5 变量作用域

#### 5.2.6 参数的引用

#### 5.2.7 递归函数

#### 5.2.8 静态变量

### 5.3 使用系统内置函数

### 5.4 文件包含函数

### 5.5 数学常用函数

### 5.6 日期常用函数

#### 5.6.1 获取时期时间信息函数

#### 5.6.2 日期验证函数

#### 5.6.3 获取本地化时间戳函数

#### 5.6.4 程序执行时间检测

### 5.7 字符串常用函数

## 06.PHP数组与数据结构

### 6.1 数组的定义

### 6.2 数组的操作

#### 6.2.1 数组的计算

#### 6.2.2 for循环遍历索引数组

#### 6.2.3 foreach遍历关联数组

#### 6.2.4 list、each函数遍历数组

#### 6.2.5 常用操作数组函数

### 6.3 数组的常用函数

## 07. PHP中的正则表达式

### 7.1 正则表达的定界符

### 7.2 正则表达式中的原子

### 7.3 正则表示中的元字符

- 7.4 正则表达式中的模式修正符
- 7.5 写正则的诀窍和常用正则
- 7.6 用正则写一个UBB文本编辑器

## 08.文件系统

- 8.1 读取文件
- 8.2 创建和修改文件内容
- 8.3 创建临时文件
- 8.4 移动、拷贝和删除文件
- 8.5 检测文件属性函数
- 8.6 文件常用函数和常量
- 8.7 文件锁处机制
- 8.8 目录处理函数
- 8.9 文件权限设置
- 8.10文件路径函数
- 8.11 小小文件留言本
- 8.12 修改配置文件的实例

## 09.PHP文件上传

- 9.1 文件上传需要注意php.ini文件
- 9.2 文件上传的步骤
- 9.3 文件上传表单注意事项
- 9.4 按照数组和步骤完成文件上传
- 9.5 多文件上传
- 9.6 文件上传进度处理

## 10.PHP图像处理

- 10.1 学习前的准备工作
- 10.2 用图片处理函数画一张图
- 10.3 生成验证码
- 10.4 图像缩放和裁剪技术
- 10.5 图片水印处理

## 11.错误处理

- 11.1 禁止显示错误
- 11.2 错误报告级别
- 11.3 错误记录日志
- 11.4 自定义错误处理函数

## 12.MySQL 入门

- 12.1 请进入《MySQL入门》

## 13. PHP操作mysql数据库

- 13.1 数据库连接步骤
- 13.2 通过步骤做一个用户注册
- 13.3 通过步骤做一个列表显示
- 13.4 把用户做个分页
- 13.5 批量和指定删除用户
- 13.6 修改用户信息
- 13.7 数据显示乱码终极解决办法

## 14.会话管理和控制

- 14.1 Cookie概述



- 14.2 PHP中的Cookie
- 14.3 session概述
- 14.4 PHP中使用session
- 14.5 SESSION应用实例
- 15.通过cURL来做小偷程序
  - 15.1 curl的使用步骤
  - 15.2 自定义get方法抓取网页
  - 15.3 使用post发送数据
- 16. 用PHP写一个论坛
  - 16.1 web2.0始于论坛
  - 16.2 需求：开发前你要知道他的样子
  - 16.3 核心业务流程
    - 16.3.1 用户注册流程
    - 16.3.2 普通用户和管理员登陆流程
    - 16.3.3 发贴流程
    - 16.3.4 回复流程
    - 16.3.5 版块管理流程
    - 16.3.6 版主业务流程
    - 16.3.7 金币奖励和消耗流程
  - 16.4 数据库表设计
  - 16.5 文件和代码规范
  - 16.6 核心功能说明
    - 16.6.1 项目目录结构说明
    - 16.6.2 公共文件的使用
    - 16.6.3 模板引擎讲解
    - 16.6.4 用户注册、登陆功能讲解
    - 16.6.5 发帖功能讲解
    - 16.6.6 回帖功能讲解
    - 16.6.7 项目安装模块讲解
- 附录1. 版权声明
- 附录2 . 学习PHP常用的英文单词

# 01. 为什么选择本书学习PHP

---

## 翻开这本书，你很幸运！

本书由于作者是五笔打字打得特别快，并且工作很忙。难免会有错别字。遇问题，留言即会更正。网上喷子多，开源免费的东西，不喜勿喷。

## 你将得到：

1. 本书完全免费
  2. 在官网获得视频支持
  3. QQ群学习帮你解决问题
  4. 认识很多大牛
  5. 大量不定期的在线答疑
  6. 大量的代码实例下载
  7. 还有机会获得李文凯的亲笔签名
- ... ..

## 为什么选择这本书？

1. 适合0基础和入门学习，简单易懂
2. 所有的知识为实际工作中最实用的知识
3. 大量的实例
4. 现实生活的代码全都步骤化了
5. 全面兼容PHP7

本书是一本开放开源的书，我们将不断的更新。

## 本书QQ：

518146831

517729890

517739045

## 1.1 为什么学习PHP？

---

回答本书的几个问题吧。你到底，为什么要学习PHP？

1. 全国都缺PHP人才，非常好就业，PHP现在的工资水平很高，刚毕业可以拿到5000-9000每个月，特别优秀还可以破万。并且有非常多的就业机会。
2. PHP入门简单，学习简单易入手。
3. 很多人反馈上完大学的C语言课程、java课程不会写任何东西。  
诚然，中国的大学都以C语言作为主要的入门语言。但是，我们认为PHP是最简单，也是最合适入门的语言。
4. 你将学习到编程的思路，更加程序化的去处理问题。处理问题，将会更加规范化。
5. 如果你要创业，如果你要与互联网人沟通。未来互联网、移动互联网、信息化将会进一步围绕在你身边。你需要与人沟通，打交道。
6. 还有机会进入BAT（百度、阿里、腾讯），BAT这些企业他们在用PHP。国内和国外超一线的互联网公司，在超过90%在使用PHP来做手机API或者是网站。连微信等开放平台中的公众号的服务端也可以使用到PHP。
7. 大并发，还能免费。一天1个亿的访问量怎么办？PHP拥有大量优秀的开发者，在一定数据量的情况下完全能满足你的需求。国内外一线的互联网公司，很多将自己的大并发方案开源了。你可以免费获得很多成熟的、免费的、开源的大并发解决方案。
8. 开源更加节约成本也更加安全。windows很多都要收取授权费用，而使用linux的LAMP架构或者LNMP架构会更加安全。全球的黑客在帮你找漏洞。全球的工程师在帮忙修复漏洞。你发现一个其他人已经消灭10个。

# 1.2 PHP是什么

## PHP

( 外文名:PHP: Hypertext Preprocessor , 中文名 : “超文本预处理器” ) 是一种通用开源脚本语言。语法吸收了C语言、Java和Perl的特点, 利于学习, 使用广泛, 主要适用于Web开发领域。

用PHP做出的动态页面与其他的编程语言相比, PHP是将程序嵌入到HTML ( 标准通用标记语言下的一个应用 ) 文档中去执行, 执行效率比完全生成HTML标记的CGI要高许多; PHP还可以执行编译后代码, 编译可以达到加密和优化代码运行, 使代码运行更快。

## 全球市场分析

目前PHP在全球网页市场、手机网页市场还有为手机提供API ( 程序接口 ) 排名第一。

在中国微信开发大量使用PHP来进行开发。

## 北京、上海的用人需求

北京2015年10月份中某一天用人低峰的招聘量 :

智联招聘  
zhaopin.com  
好工作, 上智联招聘

发布职位 >>> 服务热线 400-885-9898 • 选择城市 • 手机求职 个人登录 | 免费注册

首页 简历中心 职位搜索 校园招聘 高端招聘 智联教育 行业求职 求职指导 智联测评

职位类别 选择职位 行业类别 选择行业 全文 公司名 职位名 php 工作地点 北京 选择

搜工作 高级搜索 重置

职位类别 不限 > 软件/互联网开发/系统集成(8682) IT运维/技术支持(989) IT质量管理/测试/配置管理(427) 互联网产品/运营管理(303) IT管理/项目协调(231) 高级管理(52) 项目管理/项目协调(47) 硬件开发(43) 实习生/培训生/储备干部(43) 市场(34)

热门地区 不限 朝阳 东城 海淀 石景山 延庆县 西城 昌平 丰台 怀柔 通州 密云县 大兴 平谷 房山 门头沟 顺义 地标导航 地铁沿线

职位标签 不限 五险一金 年底双薪 绩效奖金 年终分红 股票期权 加班补助 全勤奖 包吃 包住 交通补助 餐补 房补 通讯补贴

更多筛选 薪资 公司性质 学历 工作经验 职位类型 发布时间

共11129个职位满足条件 保存搜索条件 已保存(0) 免费职位订阅 重置搜索条件 收起筛选条件

默认排序 相关度 首发日 显示方式: 列表 详细

☐ 全选

职位名称	公司名称	职位月薪	工作地点	发布日期
<input type="checkbox"/> PHP工程师	北京尚元瑞格数字科技有限公司	8001-10000	北京 - 朝阳区	10-14
<input type="checkbox"/> PHP开发工程师	深圳市前海正信石油化工投资有限公司北京分公司	面议	北京 - 朝阳区	10-14
<input type="checkbox"/> php程序员/php程序开发工程师, 15天带薪年假, 可以提	北京舒美卫生用品有限公司	面议	北京 - 朝阳区	10-14

看招  
zhaopin.com

为视频招聘而生

上海2015年10月份中某一天用人低峰的招聘量 :

本文档使用 看云 构建

- 8 -

1.2 PHP是什么

智联招聘  
zhaopin.com  
好工作，上智联招聘

发布职位

服务热线 400-885-9898

选择城市

手机求职

个人登录 | 免费注册

首页

简历中心

职位搜索

校园招聘

高端招聘

智联教育

行业求职

求职指导

智联测评

职位类别

行业类别

全文

公司名

职位名

工作地点

选择职位

选择行业

php

上海

选择

搜工作

高级搜索

重置

职位类别

不限

> 软件/互联网开发/系统集成(2893)

IT运维/技术支持(365)

IT质量管理/测试/配置管理(182)

互联网产品/运营管理(129)

IT管理/项目协调(103)

项目管理/项目协调(30)

实习生/培训生/储备干部(20)

市场(16)

高级管理(15)

教育/培训(15)

热门地区

不限

嘉定

杨浦

浦东新区

青浦

黄浦

闸北

崇明县

静安

虹口

长宁

普陀

闵行

徐汇

金山

宝山

松江

奉贤

地标导航

地铁沿线

职位标签

不限

五险一金

年底双薪

绩效奖金

年终分红

股票期权

加班补助

全勤奖

包吃

包住

交通补助

餐补

房补

通讯补贴

更多筛选

薪资

公司性质

学历

工作经验

职位类型

发布时间

共3868个职位满足条件

保存搜索条件 已保存(0)

免费职位订阅

重置搜索条件

收起筛选条件

默认排序

相关度

首发日

显示方式: 列表 详细

全选

申请职位

收藏职位

看招

zhaopin.com

## 1.3 零基础也能学习

---

学习PHP前很多人担心PHP是不是能真的学会。

学习PHP学历要求不高，数学水平要求也不高，只需要会下面这些，你就可以跟着PHP学院，开始愉快、高薪的PHP学习之旅：

- 有一台电脑
- 初中及以上文化水平
- 必须会打字（五笔、拼音均可）
- 会word（微软的office办公软件中的文字编辑软件）
- 会上网（QQ，写邮件，玩微信，看小说，看电影，注册网站帐号，网上购物等）
- 有一颗坚持的心
- 如果会一点html就更好了.学习HTML可以去看我们开源的另外一本HTML入门书籍。
- 不会HTML怎么办？也可以学习我们免费的HTML入门视频。



## 1.4 为什么有些人学不会

互联网进入到人们生活中的方方面面了，世界首富比尔盖茨多次提到青少年编程，而编程是一种思维习惯的转化。

作为写了10几年程序的人，我听到过一些说编程不好学的抱怨。

从目前见到的数据统计，主要是因为在大学开设的课程中遇到了C语言，学完后还不知道能干什么。很多人大学上完也就这么糊涂、恐惧的就过来了。

只有很少的不到1%的人学不会，这部份往往是专业的艺术家，在艺术家里面极少一部份人外，他们的思维模式和我们遇到的大多数人不太一样，并且不进行编程思维的训练，所以学不会。

而造成这个原因主要是因为社会、文化、背景、生活圈子多方面造成的。而不是因为笨。

那我们绝大多数的人是哪些原因学不会的呢？

### 1. 不相信自己能学会

这一块很多人可能不相信，涉及到很深的心理学知识。与心理暗示、诅咒的原理一样。

如果不相信自己能够学好，心里潜意识的念头里如果总是：PHP很难，我学不会。那么这个人肯定很难学会。

把不相信自己能学会的负面情绪和观念给抛掉。

只要你每天练习代码并相信自己。你肯定能学会，并且能学得很好，代码写的很成功，成为大牛！

### 2. 懒

人的天性有善有恶，而学不好程序的人，身上的一个通病，只有一个字就是——懒！

基本语法，需要去背

函数需要去默写

### 3. 自以为是

一看就会，一写就错。以为自己是神童。

### 4. 英文单词

- 计算机里面常用的英文单词就那么一些。
- 不要找英语的借口。本书会把英语单词都会跟你标注出来。看到不会的，就去翻一翻。

### 5. 不坚持

- 学着学着就放弃了。

### 6. 不去提问，不会提问，不去思考

- 解决问题前，先去搜索
- 搜索解决不了再去提问
- PHP学院为大家准备了视频，也为大家准备了问答中心。
- 大多数的人，不把问题详述清楚，不把错误代码贴完整。
- 张嘴就来提问。我想神仙也不知道你的问题是什么吧？问题发出来前。换位思考一下自己看不看得懂这个问题。

### 7. 你还需要自我鼓励

#### 1.4 为什么有些人学不会

- 在学习过程中，你会否定自己。其实任何人都会。大多数人都会遇到跟你一样的困难。只不过他们在克服困难，而一些人在逃避困难。
- 学累的时候，放松一会儿。再去多读几遍。不断的告诉自己，你就是最棒的！
- 学会交流和倾诉而非抱怨，并且不断的自我鼓励

## 02.PHP的环境安装

---

### 欲为大树，莫与草争

这一篇的标题，是一个故事。也是给很多正在学习的朋友一个提醒。

**刚开始入门的你，别把学习困难卡在了环境问题上。很多朋友，在学习过程中看了一些不太好的书。告诉你要安装xxx，绕了一大圈最后选择了放弃。**

在开始学习前，要把**问题和关键**放在如何学习写代码和如何写好代码上面。

关于环境的问题，能运行你现在写的代码就行了。

## 2.1开发环境是什么？

---

PHP是一门开发语言。而开发语言写出来的代码，通常需要在指定的软件下才能运行。因此，我们写好的代码需要（运行）显示出来看到，就需要安装这几个软件来运行代码。

我们把运行我们写代码的软件和运行代码的软件统一的称为开发环境。

### 新手学习前常遇到的环境问题

很多朋友最开始学习的时候，听说某个环境好就安装某些软件。由于缺乏相关知识，所以没有主见。陷入人云亦云的怪圈里。今天换这个，明天换那个。

验证真理的唯一标准，请始终保证一点：

环境能满足你的学习需求。不要在环境上面反复纠结，耽误宝贵的学习时间。

我们认为环境只要能满足学习要求即可。等学会了后，再去琢磨一些更加复杂的互联网线上的、生产环境中的具体配置。

## 2.2 windows环境安装

所谓服务器：不要把它想的太过于高深，不过就是提供一项特殊功能（服务）的电脑而已。

显示网页的叫网页(web)服务器（server）。

帮我们代为收发电子邮件(Email)的服务器叫邮件服务器。

帮我们把各个游戏玩家连接在一起的叫游戏服务器。

帮我们存储数据的叫数据库服务器

... ..

我们现在使用的一部手机的性能比10年前的一台电脑和服务器的性能还要强劲、给力。

而我们的学习过程当中完全可以把自已使用的这一台windows电脑作为服务器来使用。

原来如此，一讲就通了吧？

我们大多数人使用的电脑通常是windows操作系统的电脑。而我们的讲解主要在windows电脑上进行。

你不需要去理解所谓高深的电脑知识、操作系统原型等。在这一章节当中，你只需要像安装QQ、杀毒软件一样，点击：下一步、下一步即可完成本章的学习。

在最开始学习时，我们强烈建议初学者使用集成环境包进行安装。

### 什么是集成环境包？

我们学习PHP要安装的东西有很多。例如：网页服务器、数据库服务器和PHP语言核心的解释器。

我们可以分开安装各部分，也可以合在一起安装一个集成好的软件。

将这些合在一起的一个软件我们就叫作：集成环境包。

这个过程需要修改很多配置文件才能完成。并且每个人的电脑情况不同，权限不同，经常容易操作出错。

很容易因为环境问题影响到学习的心情，我们的学习计划在初期非常的绝对化：

请使用集成环境包完成最开始的学习。

等你学好PHP NB后，你爱用啥用啥，网上成堆的文章教你配置各种环境。

### 选用什么样的集成环境包？

集成环境包比较多。以下的这些全是各种英文名。只不过代表的是不同集成环境包的名字，不用去深纠。如下所示：

1. AppServ
2. PHPStudy
3. APMserv

4. XAMPP
5. WAMPServer
- ... ..等等

对于我们才入门的学习者来说，选择集成环境包的原则：

1. 更新快，版本比较新
2. 操作简单易于上手
3. 选择项不要过多

因此，我们下面使用的集成环境包是：**XAMPP**。当然，如果你对此块很熟悉了，也可以自行选择选择集成环境包。

你可以在PHP学院官网下载。下载地址为：

<http://down.phpxy.com/tools%2Fxampp-win32-1.8.3-0-VC11-installer.exe>

可以以在官方网址下载：

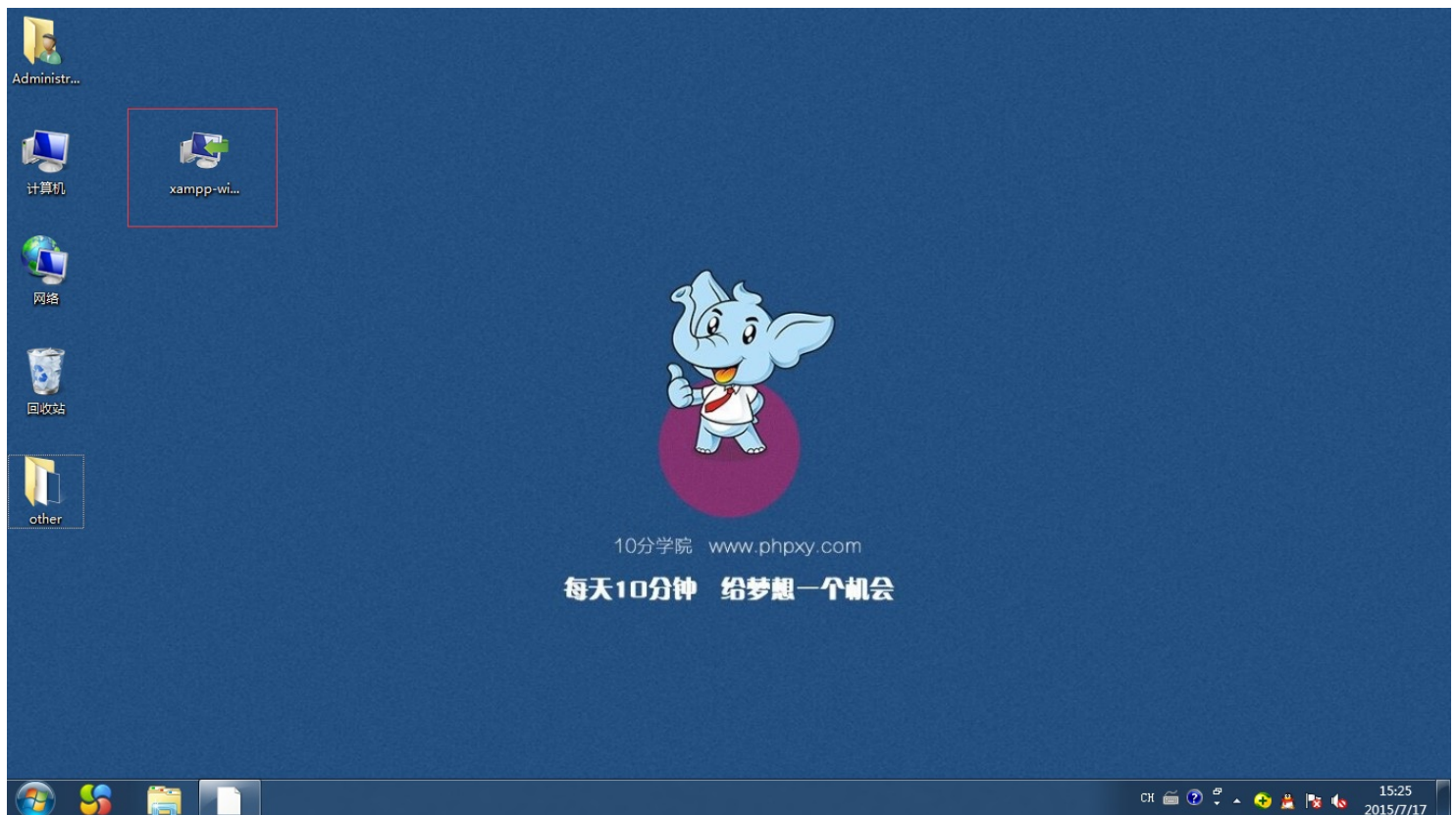
[https://www.apachefriends.org/zh\\_cn/index.html](https://www.apachefriends.org/zh_cn/index.html) (因为某些特殊的原因，有时可能在国内无法访问。)

也可以在百度中搜索：**XAMPP** 这个5个字文字字母进行下载。

## XAMPP安装过程演示

如果你会安装，可略过本章。

1，下载安装包到电脑桌面上。

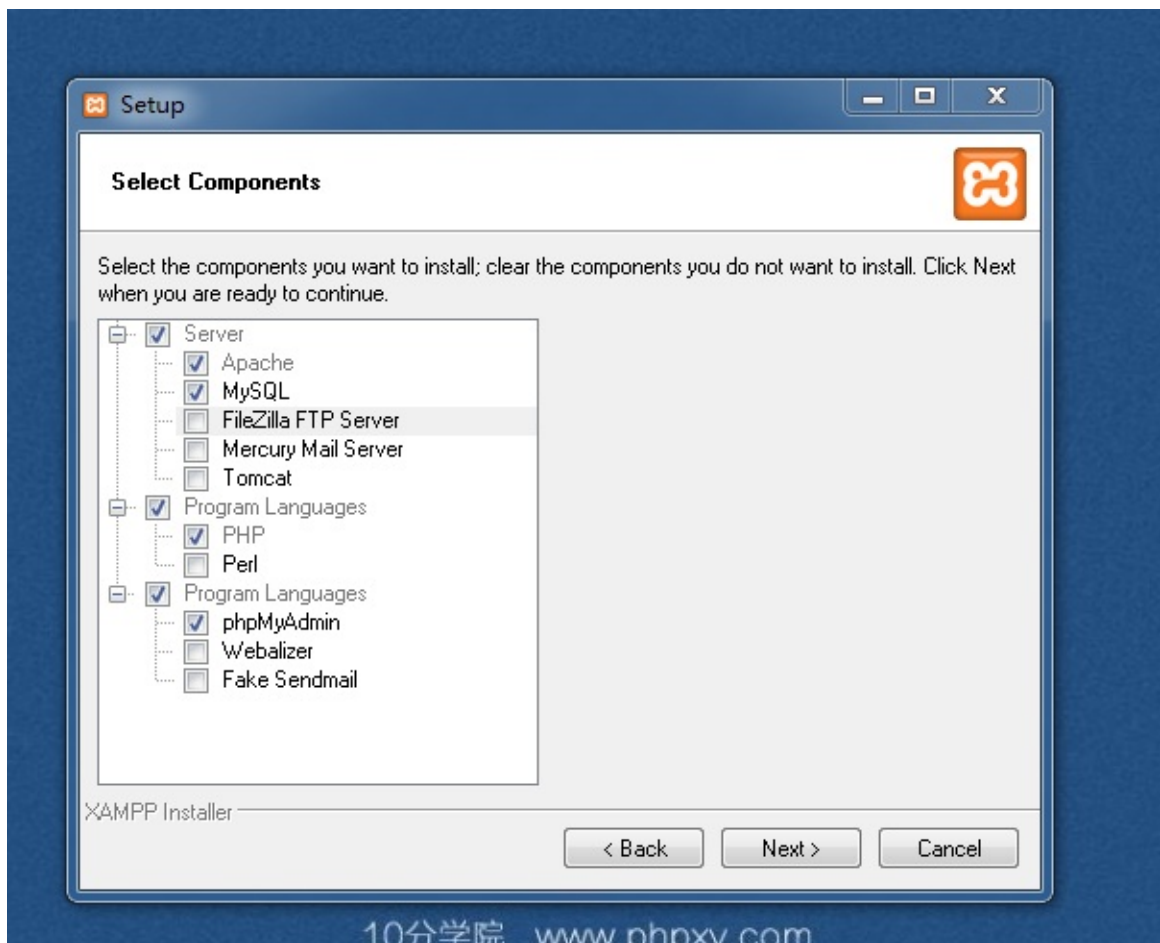


2，双击打开，显示开始安装的引导界面。

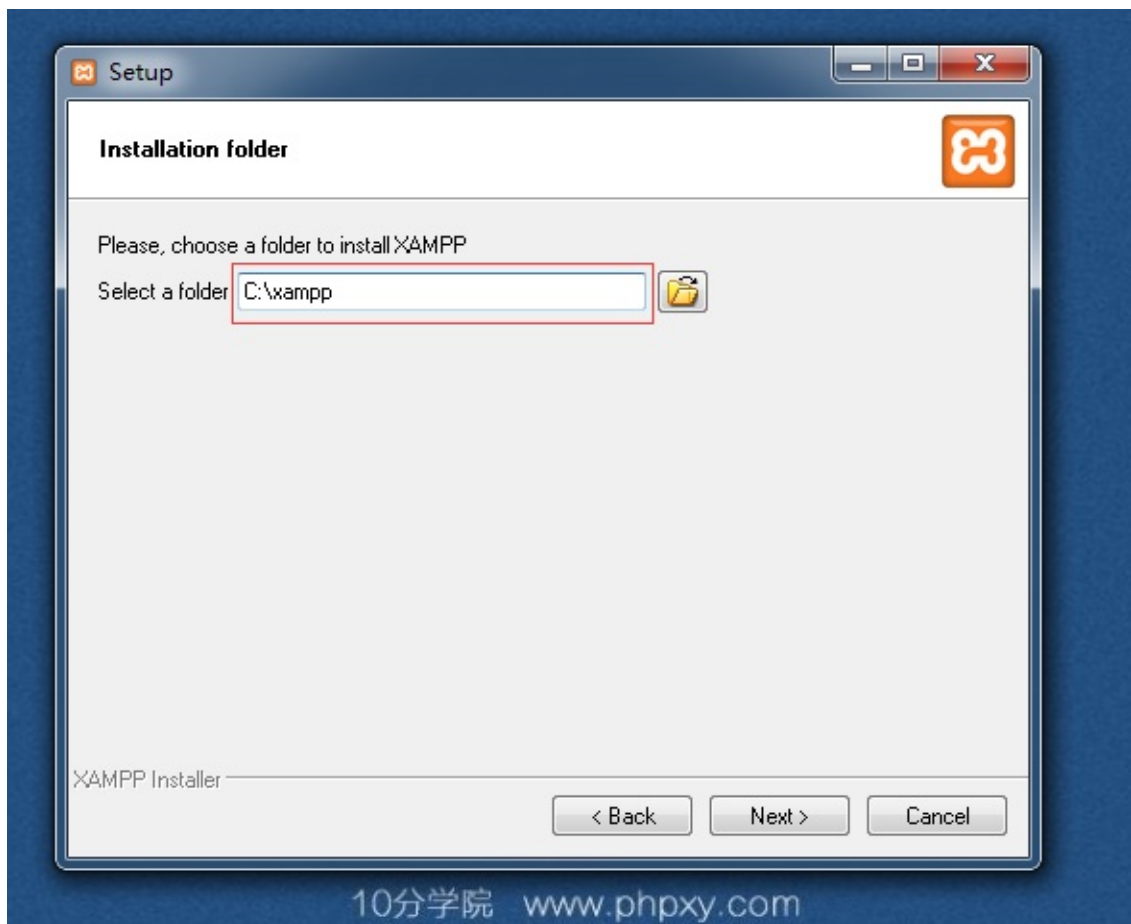




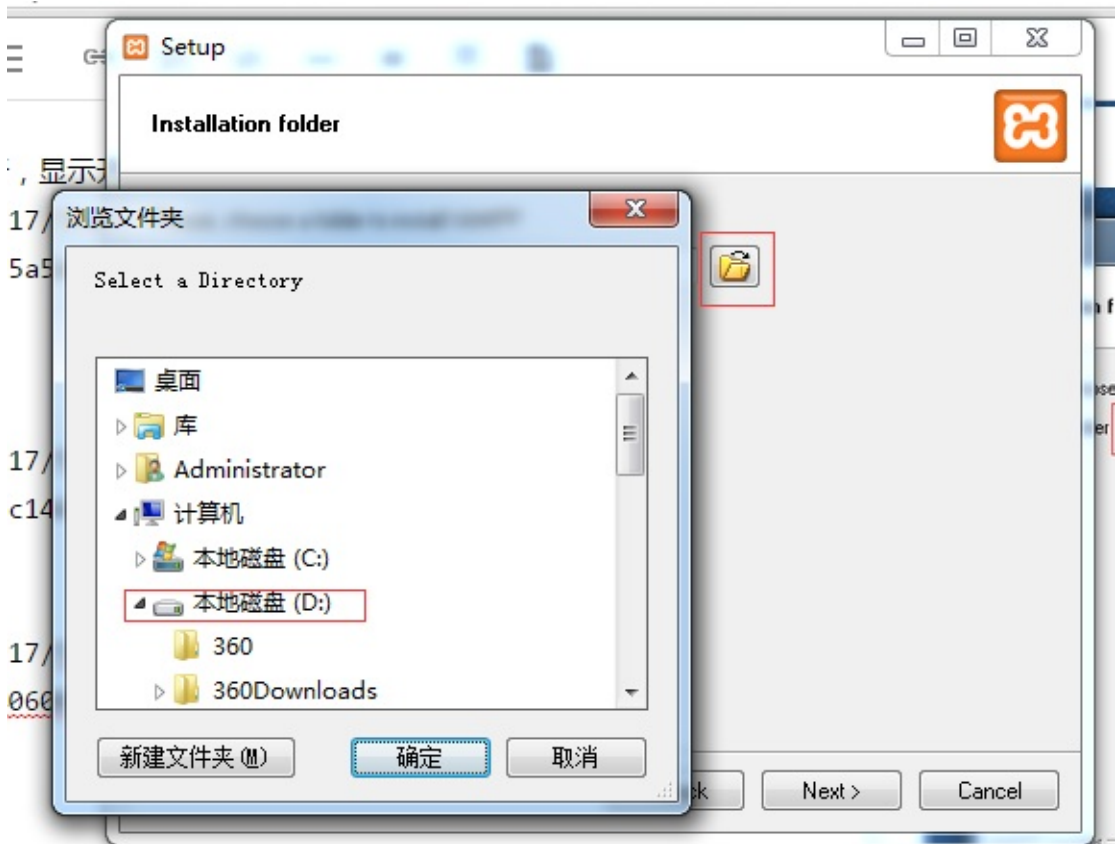
3, 请按照图中所示, 将一些不需要的选项给去掉。例如: Tomcat等



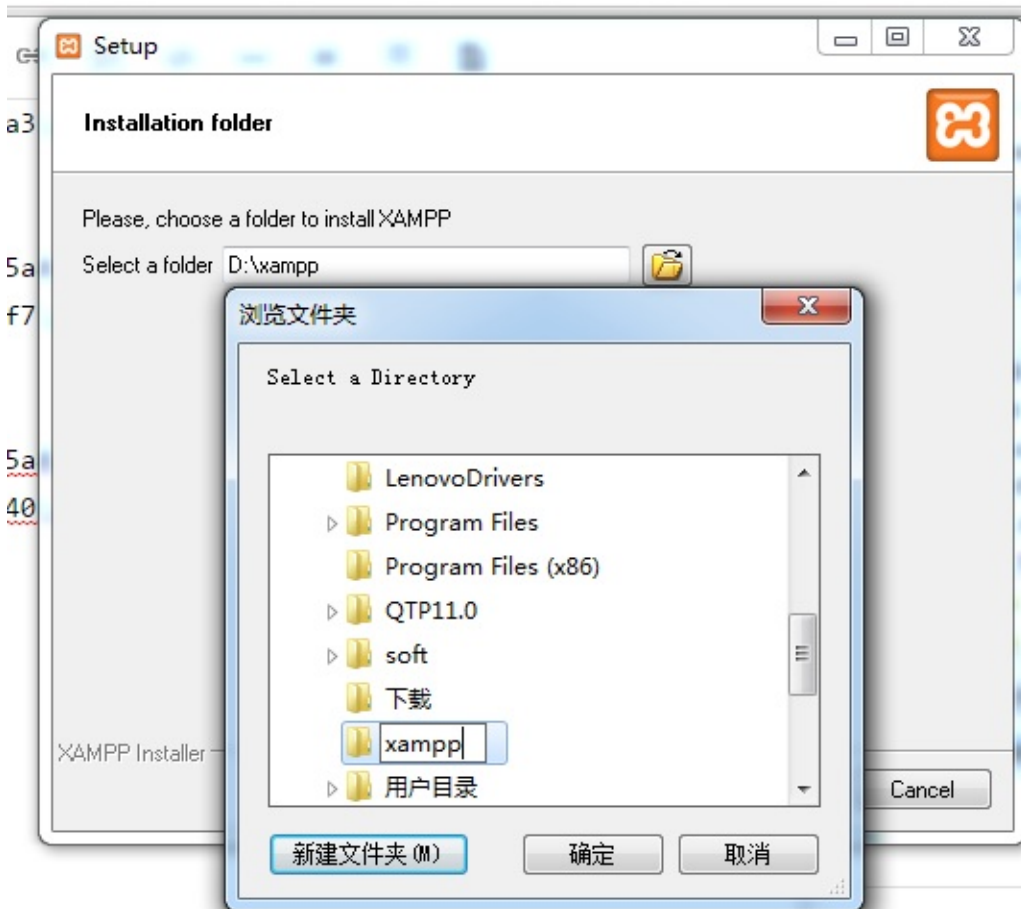
4,我们通常不把XAMPP 放在电脑C盘目录下面，因为操作系统的文件放在通常放在C盘。如果电脑坏了，重作系统的时候，容易造成我们的代码丢失。我们需要修改一下路径：



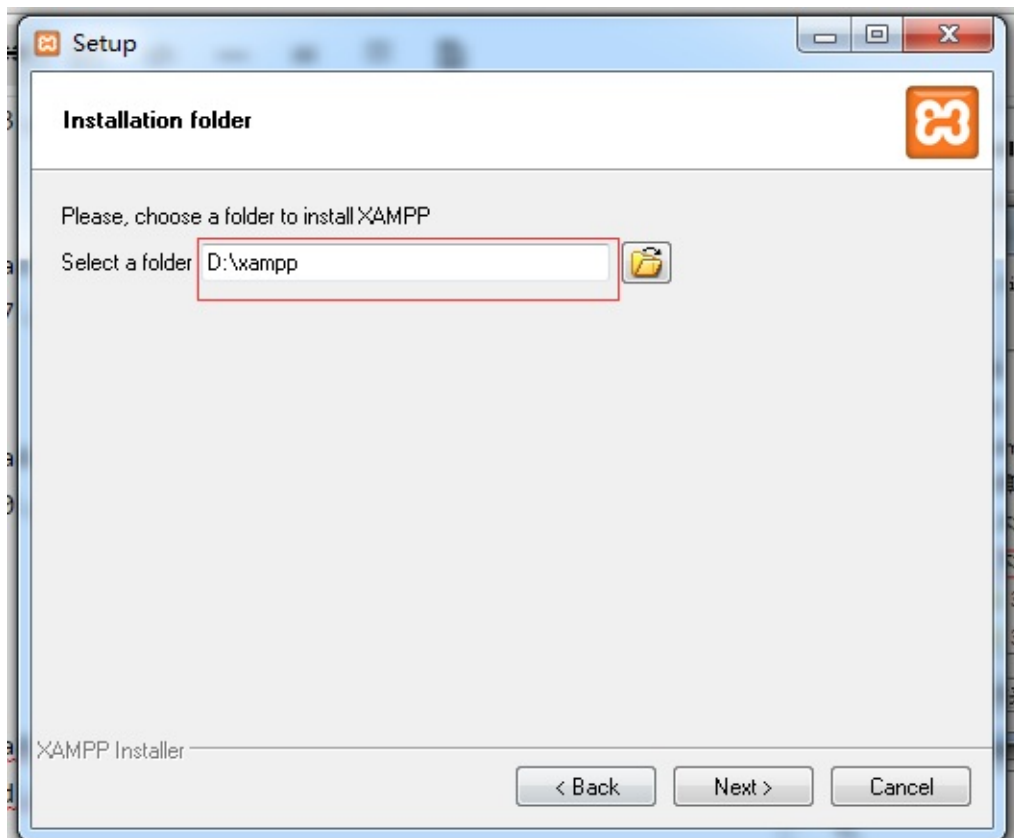
5,点击后面的文件夹样式的小图标，选择D盘。



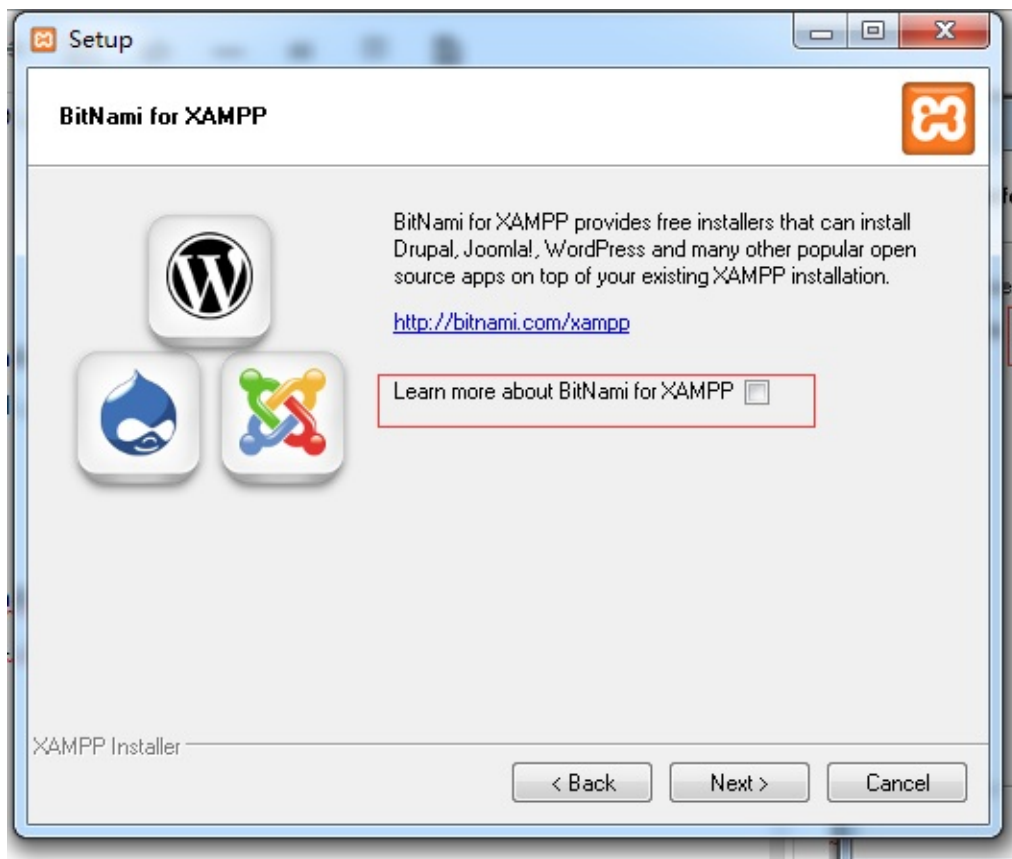
6,点击新建文件夹，写上文件夹的名称为xampp，然后点击确定。



7,文件夹选择完成，我们点击next（下一步）



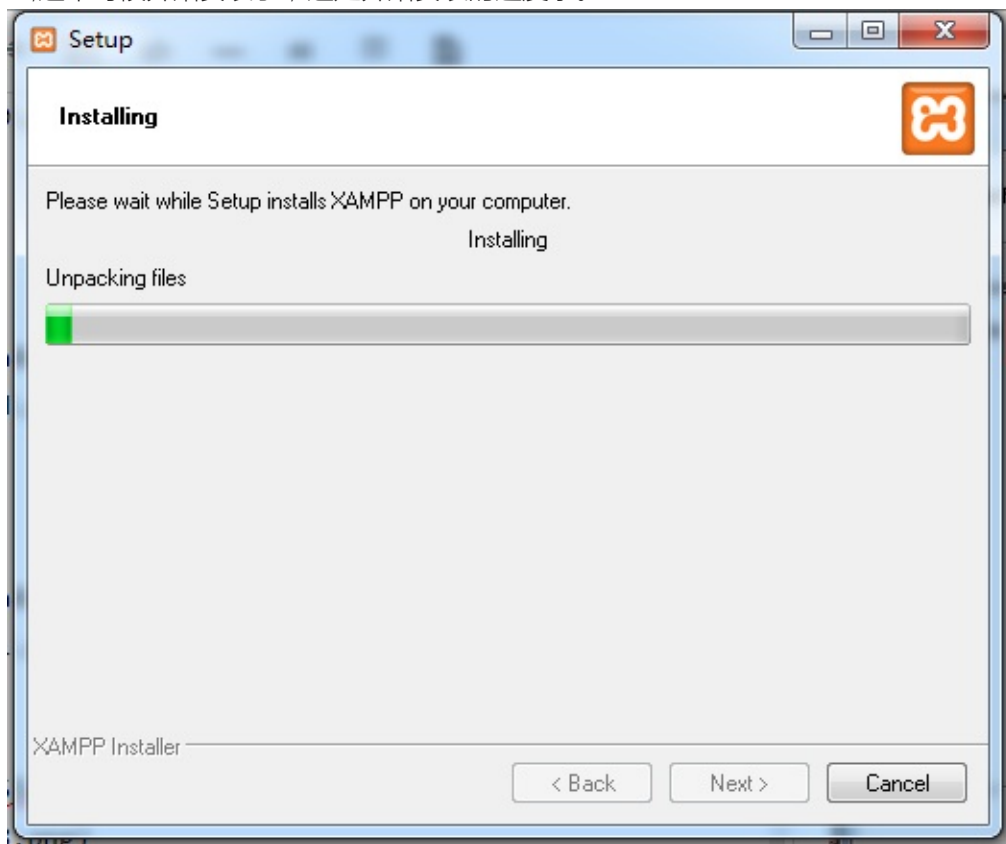
8,将后面的勾按照图中红框所示给去掉。（注：这主要是安装其他PHP程序使用的一个功能，我们学习过程中不需要使用。）



9.提示我们准备完成，点击next(下一步)，即可开始安装。



10,这个时候开始安装了，这是开始安装的进度条。



11,点击finish（完成）。完成XAMPP的安装。

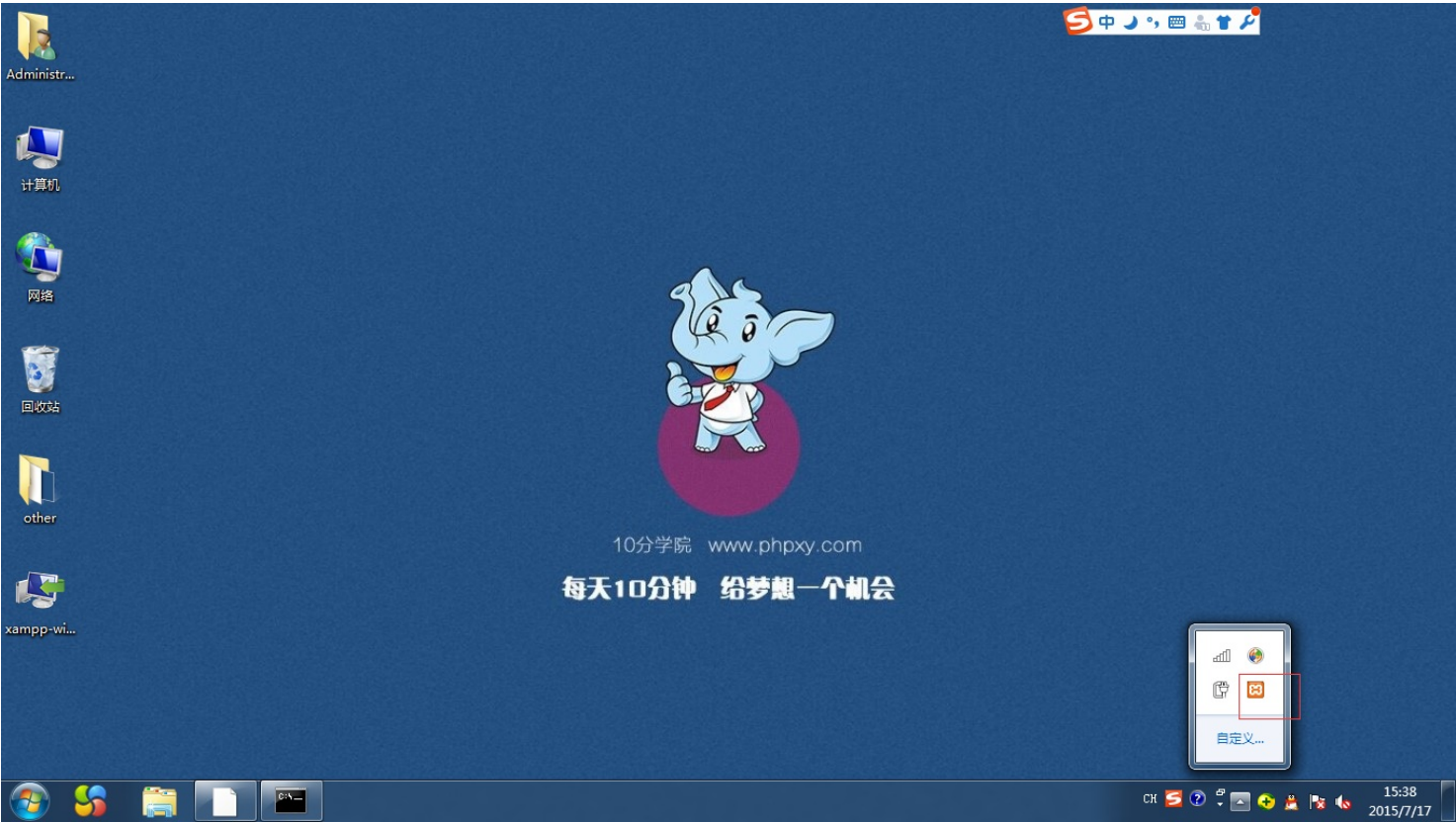




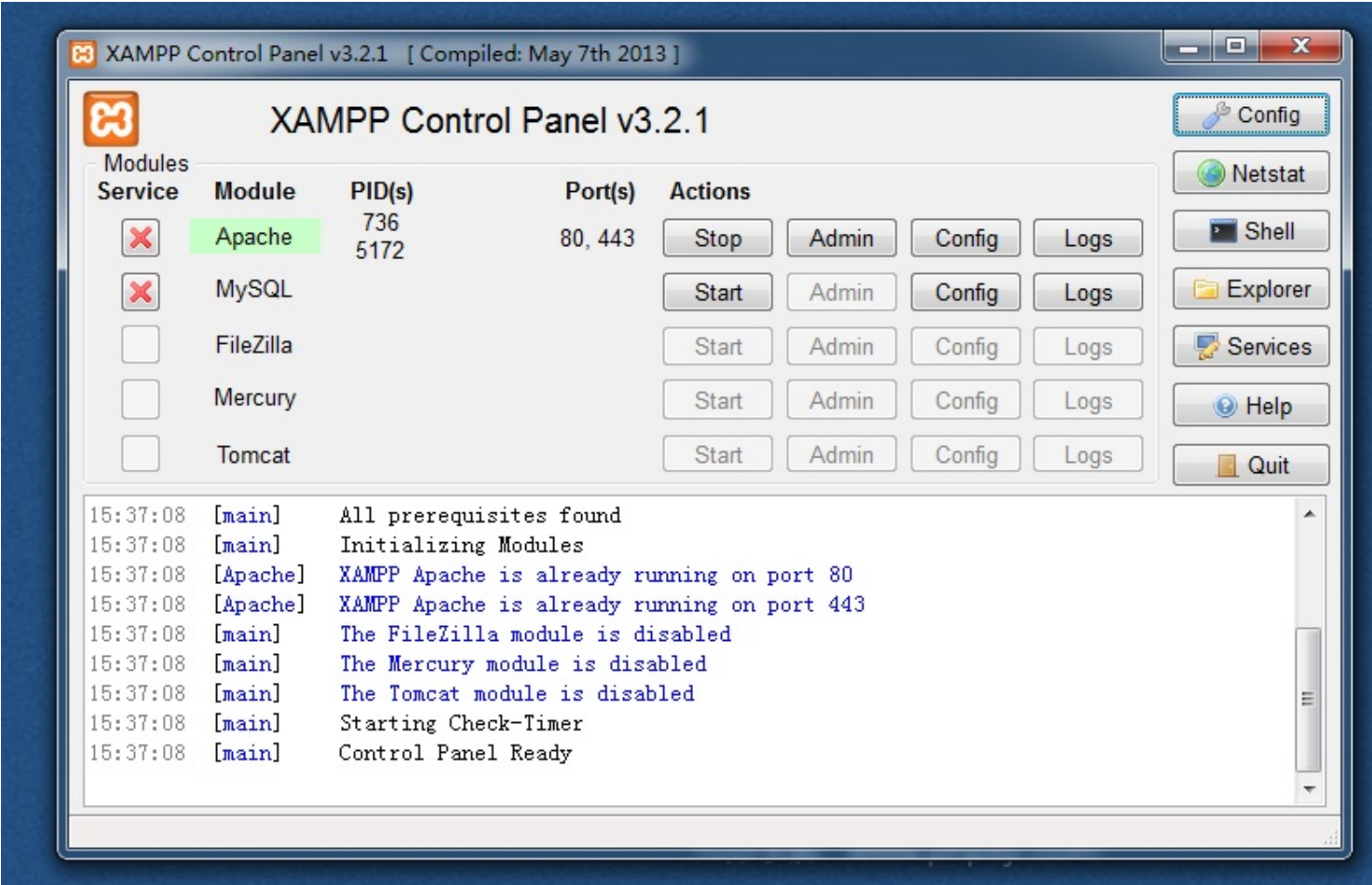
恭喜你！到了这一步，XAMPP——PHP的集成运行环境安装成功了！

## 安装后的其他注意事项

1，看到右下角的这个图标了吗？这是XAMPP的快捷方式。点击一下，试试。



2，点击后就显示出来了这个界面。这个界面大家暂时不用深入进行学习。只需要知道。Stop是停止,Start是开始即可。你可以试试。





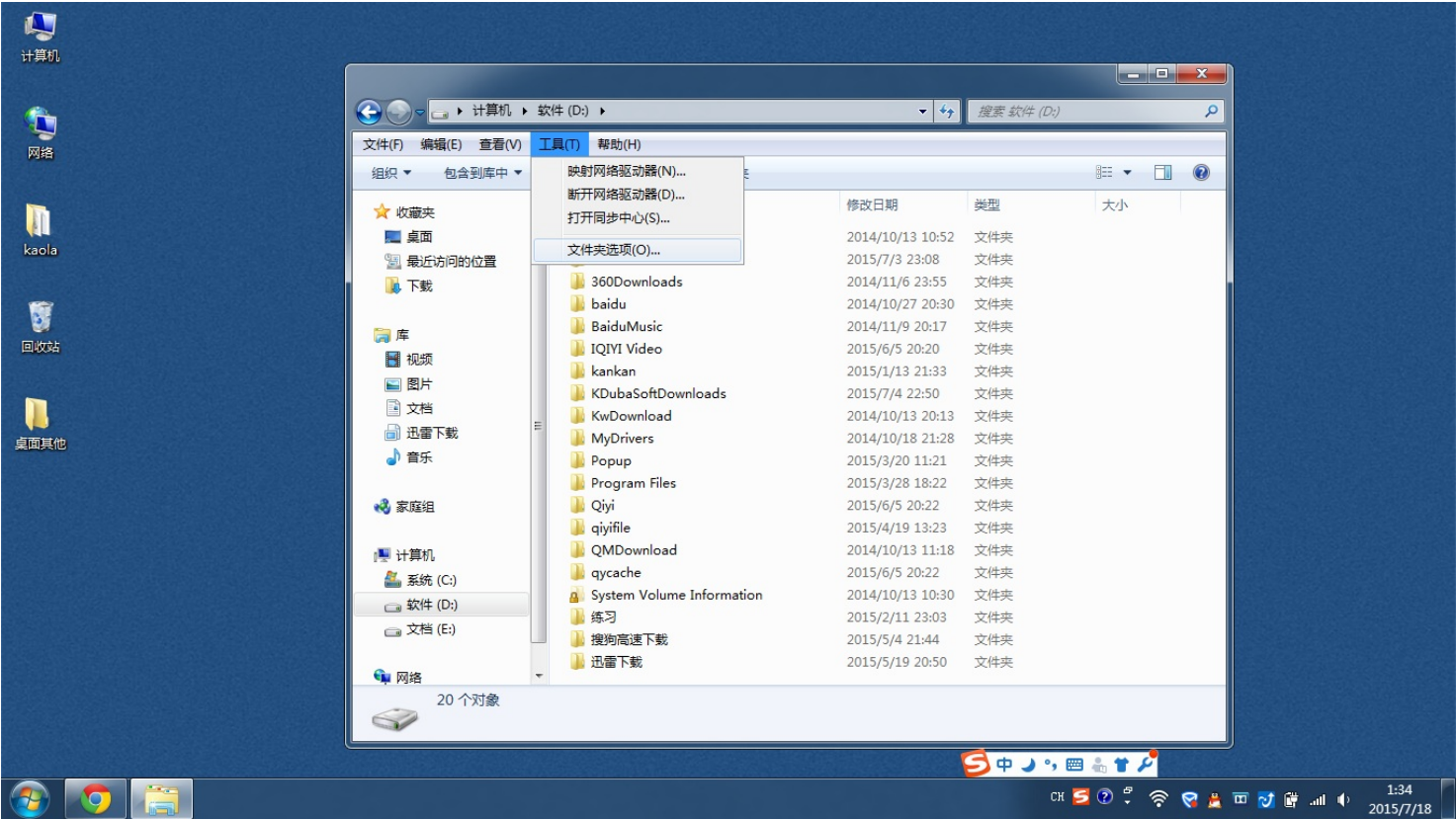


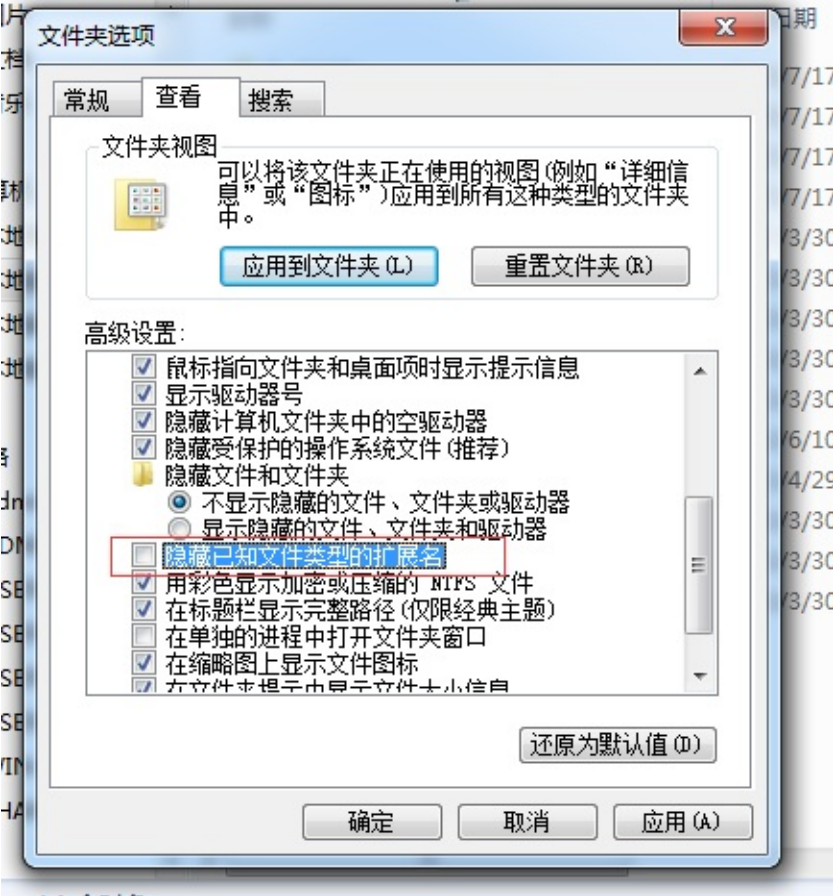
5，另外有一个注意项：【请勿必将：“隐藏已知文件类型的扩展名”前面的钩给去掉。】

为什么呢？

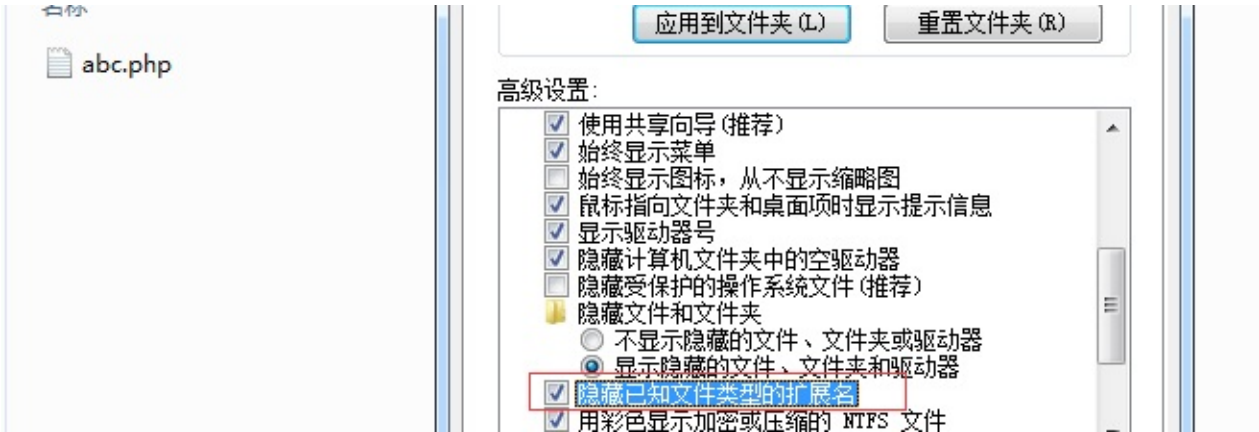
因为我们PHP文件的后缀只能是xxx.php。而很多朋友，最开始喜欢鼠标右键建立一个记录本文件后，把名字改成xxx.php。可是怎么都不变，并且怎么也不显示。这是因为windows操作系统把xxx.php.txt文件后面的已知文件扩展名.txt隐藏掉了。

可以针对我说的这个特点，在电脑上进行一下实验。尝试一下看看效果，有助于你的理解。选择：工具、文件夹选项。如下图所示：

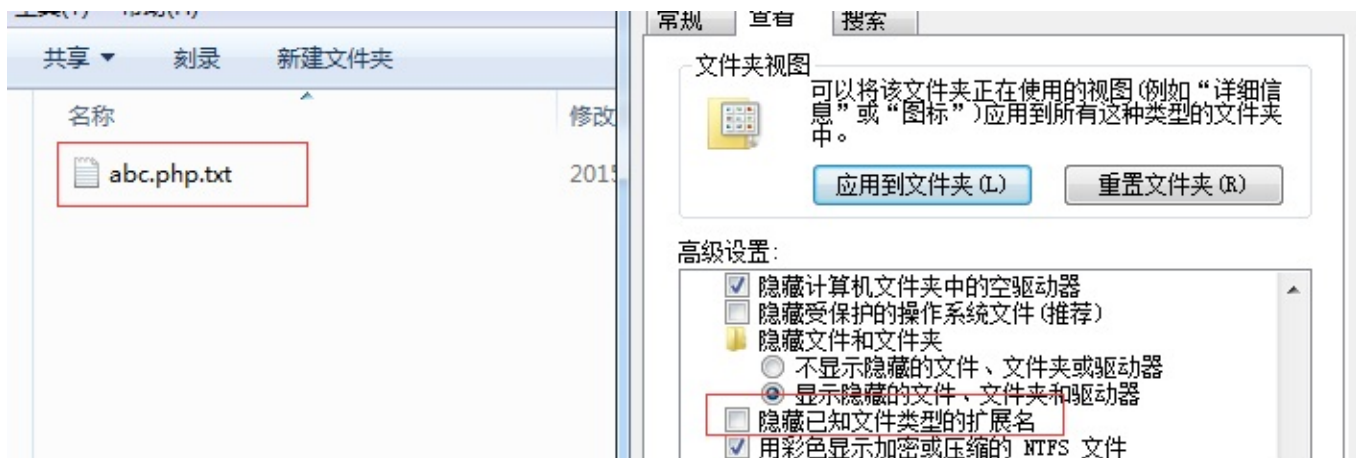




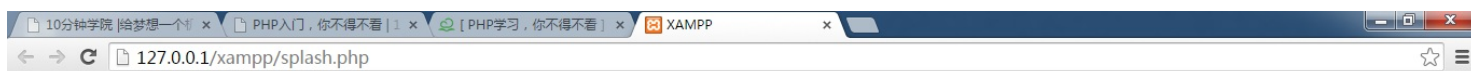
针对以上的说明，我们新建一个abc.php文件做一个实验看看效果。这是隐藏已知文件类型的扩展名的时候：



这是将隐藏已知文件类型的扩展名的时候：



6，我们打开浏览器（IE、chrome、360安全浏览器均可）。在地址栏里面输入：<http://127.0.0.1> 你将会看到一个神奇的界面。我们将网页服务器、数据库服务器、PHP运行环境安装成功啦。以后我们只需要开始写代码即可！



[English](#) / [Deutsch](#) / [Français](#) / [Nederlands](#) / [Polski](#) / [Italiano](#) / [Norwegian](#) / [Español](#) / [中文](#) / [Português \(Brasil\)](#) / [日本語](#)



附录：

操作系统：在附录中也有介绍。再讲解一次，例如我们买了一部联想、小米的手机，而管理这部手机的硬件和应用的特殊软件就叫操作系统。大家谈到的windows XP、win 7、windows 10等也都是操作系统。不过一个安卓系统是管理手机的，而通常windows系统是管理电脑的。仅此而已。

英文单词：

web 网页

server 服务器





## 2.3 Linux环境安装

---

这一个章节是本书中永远不会写的一个章节，很多人被一些市面上的书籍误导，认为学习PHP前要学习Linux。结果，一看Linux，就对人生和学习失去了希望。我们作为有过10年以上开发经验和内部训经验的专业人士告戒各位：

Linux学习与PHP学习没有必然的联系，这是两个不同的知识体系。

作为有多年开发经验和教学经验的我们。

我们强烈不建议没有接触过Linux的学生，为了学习PHP而去安装Linux环境

如果您有经验，我们相信你一定能解决，如果解决不了。

请加QQ群和访问官网：[PHP学院](#) 学习相关视频或者进行提问。

---

如果您觉得本书很好，可以分享告诉给您的朋友。

## 2.4 其他开发环境

---

对本章不感兴趣，可以略过，只是介绍和说明。

其他开发环境有很多：

1，比如 苹果电脑的系统 Mac os

2，比如 在线环境（你使用了百度、新浪、阿里等云计算环境）

3，其他更多... ..

当然，你甚至可以使用安卓手机和苹果手机来部署你的开发环境。就像有些人可以在各种复杂的环境，甚至U衣酷的试衣间里M.L。我想，这应该不是正常人类该进行的尝试吧。

如果你在使用这些环境遇到了问题，相信你已经有过一定的开发经验和处理问题的经验了，这不是刚开始学习编程该掌握的内容。

但是，如果你真遇到了这些问题。你可以上PHP学院官网学习。

## 2.5 写代码的工具选择

---

写代码的工具有很多。对于刚开始学习PHP的朋友来说。选择工具有几个原则：

- 1，不要使用带自动提示的工具（例如eclipse、zend studio等PHP开发工具集）
- 2，写完的代码必须要有颜色高亮显示。（不能使用：txt文本编辑器等无代码颜色显示的编辑器）

### 你可能想问，为什么呀？

我们发现电视、电影和现实生活中的编程高手，噼里啪啦就写一堆代码，一点都不报错，点击就能运行。而我们对着他们的代码抄袭反倒抄错。这种感觉特别不好！！！！

——传说中的这些高手，他们都曾经在基础代码上反复练习过，所以他们不会写错。

而我们需要到达高手之境界，在学习初期就不能使用先进的工具。这样会浪费我们宝贵的练习代码、调试错误的机会。

因为先进的编辑器通常有很多先进的功能，例如：

1. 代码自动显示错误
2. 代码自动换行

这些先进的工具，对于开始入门学习的你，不利于新手产生独立解决问题的能力！

## 推荐的开发工具

### 1. NotePad++（PHP学院官网下载地址：）

PHP学院下载地址：[http://down.phpxy.com/tools%2Fnpp\\_V6.7.9.2\\_Installer.1435039587.exe](http://down.phpxy.com/tools%2Fnpp_V6.7.9.2_Installer.1435039587.exe)

百度下载地址：<http://rj.baidu.com/soft/detail/13478.html?ald>

### 2. 未安装其他扩展的GVim

PHP学院下载地址：<http://down.phpxy.com/tools%2Fgvim74.3336883054.exe>

百度下载地址：<http://rj.baidu.com/soft/detail/12314.html?ald>

这些工具，你只需要下载下来，一直点击下一步，安装到你的电脑上即可。

## 03. PHP基本语法

---

我曾经带过很多人，也看过很多人写代码。其中的有些人，在写代码的时候：

——觉得自己是天才，一看就懂。

而实际过程当中，往往是：

一看就懂，一写就错。

所以我们给所有刚入门学习编程（任何语言）的朋友一个总结了十年的经验：

永远不要骗自己，写不出来代码就是不会。

---

凯哥语录：基本语法的训练要像亲吻一样熟练。

## 3.1 PHP基本语法

---

### 什么是基本语法？

基本语法就是组成编程语言的基本规则，是一些非常具体的规定。

对于编程语言当中的基本语法：我们认为跟地球的法则一样，在没有办法打破法则之前就是规定，规则。

切记不要犯很多新手一样的毛病——为什么这个基本语法要这样写呢？

## 3.1.1 写出你的第一段PHP代码

---

这句代码非常神奇，一句话能变成一个网页。是我们PHP入门的第一段代码。

现在你可以在网页服务器的根目录（ D:\xampp\htdocs ）当中新建一个文件。文件的名字为：abc.php。

在这个abc.php文件中写入如下代码。

```
<?php  
  
phpinfo();  
  
?>
```

你可以在浏览器地址栏里面，输入你的服务器地址。我们当前是以自己的电脑作为网页服务器来使用的，你可以在地址中输入：

<http://127.0.0.1/abc.php>

你将会看到一个网页，网页内容通常，如下图所示：

## PHP Version 5.5.17



System	Linux AY130305165229237aef 2.6.32-431.23.3.el6.x86_64 #1 SMP Thu Jul 31 17:20:51 UTC 2014 x86_64
Build Date	Sep 26 2014 10:14:24
Configure Command	'./configure' '--prefix=/usr/local/php5' '--with-config-file-path=/usr/local/php5/etc' '--with-mysql=/usr/local/mysql' '--with-mysql-sock=/usr/local/mysql/mysql.sock' '--with-gd' '--with-iconv' '--with-zlib' '--enable-xml' '--enable-bcmath' '--enable-shmop' '--enable-sysvsem' '--enable-inline-optimization' '--enable-mbregex' '--enable-fpm' '--enable-mbstring' '--enable-ftp' '--enable-gd-native-ttf' '--with-openssl' '--enable-pcntl' '--enable-sockets' '--with-xmlrpc' '--enable-zip' '--enable-soap' '--without-pear' '--with-gettext' '--enable-session' '--with-mcrypt' '--with-curl' '--with-jpeg-dir' '--with-freetype-dir' '--disable-fileinfo'
Server API	FPM/FastCGI
Virtual Directory Support	disabled
Configuration File (php.ini) Path	/usr/local/php5/etc
Loaded Configuration File	/usr/local/php5/etc/php.ini
Scan this dir for additional .ini files	(none)
Additional .ini files parsed	(none)
PHP API	20121113
PHP Extension	20121212
Zend Extension	220121212
Zend Extension Build	API220121212,NTS
PHP Extension Build	API20121212,NTS
Debug Build	no
Thread Safety	disabled
Zend Signal Handling	disabled
Zend Memory Manager	enabled
Zend Multibyte Support	provided by mbstring
IPv6 Support	enabled
DTrace Support	disabled

## 3.1.1.2 用中文翻译这句话给你听

```
<?php
phpinfo();

?>
```

刚刚我们看到这个代码已然生成了一个网页。不明白其中的意思，现在我一点一点跟大家解释。



```
<?php
```

这句话是说明从此处【尖括号、问号、PHP】这5个字符开始，往下的部份是PHP代码。

```
?>
```

而，遇到上面的【问号、尖括号】说明PHP代码写完了。再之后的部份，跟我PHP没有关系啦。

现在大家看会了？请问：phpinfo() 后面还接了一个分号(;)，并且在phpinfo()的上一行和后一行，还有两个回车符。请问这是什么意思呢？

现在我告诉你答案：

- phpinfo是一个函数（功能），这个函数（功能）会显示一个当前电脑（服务器）的详细的PHP信息。在后面会专门为大家讲解phpinfo()这个函数。这个函数必须要大家记住！
  - 电脑是一个很笨的东西。他需要我们人为的告诉他，代码写完了。我们用分号(;)来告诉他代码写完了。因此，只要我们写完一段代码，就需要在后面加分号(;)。
  - 而前后两行的回车空行，是为了让代码更加好看，避免叠在一起，让人看着不舒服。其实，中间你有多少个回车空行都没有关系的，保证好看就行了。PHP的语法解释器（PHP最核心的部份）不会处理这些空行的。
- 
- 注意事项：
  - php的代码部份全部要用半角的英文、很多人容易写成全角的英文和符号造成PHP代码报错。
  - PHP代码的最后一行可以加也可不加分号。由于很多学习者特别是初学者经常犯一个毛病：写完一行代码不加分号。因此，容易报错。我们通常在公司里面规定：一行代码写完，就必须加分号。
  - PHP还可以有简洁声明模式，我们在这个阶段以代码练习为主。关于环境配置问题，暂时不建议学习。在后面的章节中，我们会专门进行讲解。
  - 以上，非常容易在实际开发中不小心造成错误。
  - 我们在写文件abc.php这个文件命名的时候，只用英文半角（a-zA-Z0-9\_-）这些字符来命名文件。文件命名不要用中文，不要用特殊符号，不要中间加空格，严格区分大小。

## 3.1.2 读过初中你就会变量

大家在读初中的时候呀。老师经常会这么教大家。

请问，李磊和韩梅梅同学，假如：

```
x = 5
```

```
y = 6
```

那么 $x + y$  等于多少呢？大家会义无反顾的回答。 $x + y$  等于11。

接下来我们看下面的初中的数学知识，请问 $x + y$  的结果是多少？

```
x = 5
```

```
y = 6
```

```
x = 8
```

我估计大家也会义无反顾的回答： $x + y$  的结果为14。

这就是变量！

变量的几个特点：

1.  $x = 5$  将右边值5，赋值给左边的x
2. 第二段 $x = 8$ ，最后 $x + y$  的结果等于14，说明x在从上到下的运算（执行）中，可以被重新赋值。

我们在PHP中的变量也是如此。不过有几个特点：

1. 必须要以\$开始。如变量x必须要写成\$x
2. 变量的首字母不能以数字开始
3. 变量的名字区分大小写
4. 变量不要用特殊符号、中文，\_不算特殊符号
5. 变量命名要有意义（别写xxx，aaa，ccc这种 变量名）

**错误举例：**

错误：变量以数字开始

```
<?php
$123 = 345;
?>
```

错误：变量中有特殊字符，中文

```
<?php
```

### 3.1.2 读过初中你就会变量

```
$a*d = 345;  
  
$中国 = 123;  
?>
```

错误：变量命名没有意义aaa容易数错，也没有含意

```
<?php  
$aaaaaaa = 345;  
?>
```

错误：变量严格区分大小写 \$dog 和 \$Dog是PHP学院的变量,尝试将\$dog的值改为8.结果D写成了大写。

```
<?php  
$dog = 5;  
//重新修改$dog的值，将$dog改为8  
$Dog = 8;  
?>
```

正确举例：

正确：变量不能以数字开始,但是数字可以夹在变量名中间和结尾

```
<?php  
$iphone6 = 5880;  
$iphone6plus = 6088;  
?>
```

正确：变量不能有特殊符号，但是\_(下划线不算特殊符号)

```
<?php  
$_cup = 123;  
?>
```

注：你会发现代码是从上向下执行的。

\$ 叫作美元符，英文单词：dollar。PHP的变量必须以美元符开始。说明搞PHP有“钱”图

dollar

读音：['dɒlə(r)]

解释：美元

## 3.1.3 echo 显示命令

---

echo 是在PHP里面最常用的一个输出、显示功能的命令。

我们可以让他显示任何可见的字符。

```
<?php  
  
echo 123;  
  
?>
```

```
<?php  
  
$iphone = 6088;  
  
echo $iphone;  
  
?>
```

你可以对着做做实验。等下一章，我们讲数据类型的时候，我会教大家输出中文和用PHP显示网页内容。

---

**单词：**

**echo**

读音：['ekou]

解释：发出回声；回响。

功能解释：输出、显示

## 3.1.4 注释的功能很强大

所谓**注释**，汉语解释可以为：注解。更为准确一些。  
因为代码是英文的、并且代码很长，时间长了人会忘。  
所以我们会加上**注释**。

注释的功能有很多：

1. 对重点进行标注
2. 时间长了容易忘快速回忆，方便查找
3. 让其他人看的时候快速看懂
4. 还可以生成文档，代码写完相关的文档就写完了，提高工作效率
5. 注释、空行、回车之后的代码看起来更优美
6. 注释可用来排错。不确定代码中哪一块写错了，可以将一大段注释，确定错误区间
7. 注释中间的部分的内容，电脑不会执行它

先给大家看看我们觉得优美的代码，整齐、规范、说明清楚、一看就懂。（不需要理解代码的含义）：

```
/**
 * 删除问题
 *
 * @param int $id
 * @return Response
 */
public function getDelete($id) {
    //查询到问题ID
    $question = $this->model->find($id);

    if ($question) {
        //执行删除
        $question->delete();
    }

    return redirect()->back()->withMessage('删除成功! ');
}
```

再看看我们眼中觉得丑陋的代码，对齐丑陋不说，并且没有功能说明（不需要理解代码的含义）：

```
//这是一个删除功能
public function getDelete($id){$question = $this->model->find($id);
if ($question) {
$question->delete();
}
return redirect()->back()->withMessage('删除成功! ');
}
```

我们了解了注释的好处，接下来我们来说PHP的注释，注释分别：

- 单行注释（只注释一行）
- 多行注释（注释多行）
- 单行注释

```
// 表示单行注释
```

## 号也表示单行注释，用的比较少

```
* 多行注释
```

```
/*
```

多行注释 这里是注释区域代码

```
*/
```

单行注释举例：

```
<?php
```

```
//声明一部iphone6手机的价格变量
```

```
$iphone6_price = 6088;
```

```
//显示输出手机价格
```

```
echo $iphone6_price;
```

```
?>
```

注：通过上例我们知道，注释通常写代码上面。

多行注释举例：

```
<?php
```

```
/
```

作者：李文凯

时间：2048.12.23

功能：这是一个假的多行注释的例子

```
/
```

```
/
```

声明一个爱情变量

\$love 是指爱情

爱情是一个变量，因为人的爱总是在发生变化



### 3.1.4 注释的功能很强大

所以，爱情变量的值为250

```
/
```

```
$love = 250;
```

```
?>
```

注：通过上面的例子我们发现，我们要写上很多注释的时候，使用多行注释。

```
* * * * *
```

注：暂时不讲解如何通过专门的工具生成注释

## 3.2 数据类型并不神秘

---

### 数据类型并不神秘

人类世界对万事万物都有种类划分，例如：

#### 哺乳动物

人、猫、马、鸭嘴兽....等等

#### 蔬菜

西红柿、菠菜、茄子....等等

#### 水果

西瓜、桃子、苹果....等等

#### 上面这些都有类型，那数据为什么不能有类型呢？

数据类型：就是对数据分类的一个划分而已。

## 3.2.1 整型就是整数

我一直在讲，不要被名词的含义所吓唬住。

到底什么是整型呀？

所谓整型，就是大家数学中所学的整数。

整型——整数也，英文称之:integer。英文简写：int

整型分为：

1. 10进行
2. 8进制（了解，基本不用）
3. 16进制（了解，基本不用）

整型（整数）在计算机里面是有最大值和最小值范围的。

【了解知识点，开发中不常用】大家经常听说32位计算机，也就是32位计算机一次运算处理的最大范围为 $-2^{32}$ 至 $2^{32}-1$ 。64位计算机呢？—— $-2^{64}$ 至 $2^{64}-1$ 。

**10进制声明：**

```
<?php
//为了方便大家记忆和前期学习，英文不好的朋友也可用拼音来声明变量。以后再用英文来声明变量也无所谓
//声明变量 整数，英文 int
//$int = 1000;
$zhengshu = 1000;
echo $zhengshu;
?>
```

**8进制声明：**

以0开始，后面跟0-7的整数（了解知识点）

```
<?php
//8进制的取值范围最大为0-7,即0,1,2,3,4,5,6,7

$bajingzhi = 033145;
echo $bajingzhi;

?>
```

**16进制声明：**

以0x开始，后面跟0-f的，0x的abcdef不区分大小写。（了解知识点）

```
<?php
//16进制的取值范围最大为0-f,即0,1,2,3,4,5,6,7,8,9,a,b,c,d,e,f
$shiliu = 0x6ff;
echo $shiliu;
?>
```

### 3.2.1 整型就是整数

本章学习重点，学会如何声明10进制整数即可。了解8进制和16进制的声明，实在不会也不要紧。

思维误区：容易去考虑8进制和16进制到底是怎么产生的。

## 3.2.2 布尔就是易经的知识

布尔类型是一个非汉语的解释，在英文当中的舶来之物。

所谓布尔类型，汉语翻译过来之后，更好的解释是用道家中易经的语言来进行解释，即为：

- 阴 / 阳
- 男 / 女
- 真 / 假
- 对 / 错
- 存在/不存在
- 0 / 1
- 等等.... ..

是人类繁衍过程中，我们对逻辑判断的一种总结。

例如电视剧里面经常讲到的一句话：

**如果那个漂亮妹子（帅哥）被我泡到手了，我死都愿意。**

用计算机的代码完全可以写出这句话：

如果我泡到了漂亮妹子

则：我愿意去死

如果没有泡到

则：我不愿意去死

在我们的思维过程当中，始终在遵循上述的思考模式来思考问题。因此，前辈们为我们进行了总结，在计算机里面把这种判断和思考方式，叫作布尔判断。为这种思考方式专门做了一个数据类型,即为布尔类型。

布尔类型就是：真和假。

在英文把真和假翻译过来就是：

- true ( 真 )
- false ( 假 )

因此，我们在PHP代码里面可以这样声明。

```
<?php
//声明一个变量（拼音）为布尔
$buer = true;
//声明一个变量(英文)
$bool = false;
?>
```

英文单词

true

读音：[tru:]

### 3.2.2 布尔就是易经的知识

解释：真正地; 确实地; 正确地; 正当地;

false

读音：[fə:ls]

解释：虚伪的; 不正的

boole

英文全称：boolean

读音：[bu:l]

解释：布尔类型、布尔数学体系

注：true和false外面不要加引号。类型转换，后面的课程当中我们来讲解。

```
布尔值=false  
整数值=0  
浮点数=0.0  
空字符串  
空数组
```

没有变量成员的对象。

变量值为空NULL

例如：

```
$a= false;      //$a设置为假  
$b=NULL;       //$b设置为null  
$c=NULL;       //$c设置为null
```

除了上述情况以外，其余情况值将会被转换为true，一般来说，1为true，0为false。这些转换将在以后的判断语句中经常遇到。



## 3.2.3 字符串

字符串就是所有我们可见和不可见的字符，就是我们日常当中说的话，就是我想说：“李文凯好帅”或者“凤姐，我爱你！”。字符串，就是我想表达的一切让人看到的字符。

例如可以看到的字符：

```
我愿变成童话你，你爱的那个王子，有房有车有票子。
<html><title></title></html>
^@@@#@
my name is xiaoshenyang
```

以上可以见到的文字，HTML标签、特殊字符和英文等等，我们都认为是字符串。

注：不可见字符暂时不讲解，因为，文本中无法表现，下面的实验中逐渐注意即可。为了有更清晰的表现，也可以观看本书配套的PHP学院出品关于字符串的视频。

在PHP语言中声明字符串有三种方式：

1. 用单引号声明
2. 用双引号声明
3. 用字界符声明（需要输入非常大段的字符串时使用）

### 1.单引号声明

用英文半脚的单引号，将字符串包起来

```
<?php
//声明字符串变量$zhifu

$zhifu = '曾经有操蛋的爱情摆在我面前，我珍惜了。当我得到的时候才感到后悔莫及。如果非要在这段爱情前面加上一段三个字，我愿意说三个字：滚犊子';

//你可以放XAMPP指定的目录下，新建一个文件叫str.php。然后访问一下http://127.0.0.1/str.php试试。会不会显示这句话。

echo $zhifu;

?>
```

### 1. 双引号声明字符串

在字符串两边加双引号。

```
<?php
//声明字符串变量$str
$str = "如果非要在滚犊子前面加上一个时间的话我愿意是马上。";
```

```
echo $str;
```

```
?>
```

### 3. 字界符声明

- 1).在变量后面的等号写三个小于号（<<<）。
- 2).然后在<<<后面写上字符（建议英文大写字符）。如下例中的：ABC
- 3).然后换行写上任意想写的字符

### 3.2.3 字符串

4). 写完后，顶行。在行最开始处，再写上<<<后面的字符和分号。如下例中的：ABC；

```
<?php
$dingjie = <<<ABC
如果
非要在这个滚犊子
前
面
```

加上一段  
距离的话  
我想说：

## 思想有多远，你就跟我滚多远

```
ABC;
?>
```

###那双引号和单引号有什么区别呢？

\*\*【重要知识点】PHP面试题中，高概率面试题（建议背诵并实验三遍以上）\*\*

1. 双引号解析变量，但是单引号不解析变量。
2. 在双引号里面插入变量，变量后面如果有英文或中文字符，它会把这个字符和变量拼接起来，视为一整个变量。一定要在变量后面接上特殊字符，例如空格等分开。
3. 如果在双引号里面插变量的时候，后面不想有空格，可以拿大括号将变量包起来。
4. 双引号解析转义字符，单引号不解析转义字符。但，单引号能解析\' 和\\
5. 单引号效率高于双引号，尽可能使用单引号
6. 双号和单引号可以互插！！！双引号当中插入单引号，单引号当中插入变量，这个变量会被解析。
7. 神奇的字符串拼接胶水——（.）点，用来拼接字符串。
8. 我们将定界符声明字符串视为双引号一样的功能来看待。

####实验举例

\*\*1.双引号解析变量，但是单引号不解析变量\*\*

```
<?php
//声明变量$shouji
$shouji = '为了苹果手机去卖肾';
//在双引号中放$shouji 然后echo 一下是什么效果呢？
$str = "$shouji 会不会显示呢?";
//输入$str试试
echo $str;
?>
```

在浏览器中打开，运行的结果如下：

![2015-07-26/55b4931eef9c5](http://box.kancloud.cn/2015-07-26\_55b4931eef9c5.png)

### 3.2.3 字符串

通过上边的例子，我们发现。双引号中间的变量够执行显示，我们将\$str这个字符串两边的双引号，改为单引号试试，再看一次执行结果：  
![2015-07-26/55b4938bea073](http://box.kancloud.cn/2015-07-26\_55b4938bea073.png)

发现\$shouji 直接显示出来了，而不是双引号的显示结果。

通过上例，我们证明了：双引号执行变量，而单引号不执行变量。我们得到一个重要结论：

> 双引号需要解析变量，因此双引号的效率没有单引号高。我们如果为了更高的效率，我们通常使用单引号。

**\*\*2.** 在双引号里面插入变量，变量后面如果有英文或中文字符，它会把这个字符和变量拼接起来，视为一整个变量。可以在后面接上特殊字符，例如空格等分开。**\*\***

```
<?php
$huaqiangu = '花千骨';

$str = "$huaqianguaaaa";
//你会发现输出$str，什么都没有在页面中显示
echo $str;
?>
```

但是，我们在\$str 声明的这个字符串中将\$huaqiangu 后面接个空格再接aaaa，代码如下：

```
<?php
$huaqiangu = '花千骨';
//中间加了空格哟
$str = "$huaqiangu aaaa";

echo $str;
?>
```

大家发现执行显示的结果不再是空白页面，而是这个页面中有内容了。如下：  
![2015-07-26/55b4f5794a6ad](http://box.kancloud.cn/2015-07-26\_55b4f5794a6ad.png)

我们把代码再改改：

```
<?php
$huaqiangu = '花千骨';
//中间加了空格哟
$str = "$huaqiangu!aaaa";

echo $str;
?>
```

大家打开一次页面，执行了一次代码看看结果是不是变为了：  
> 花千骨!aaaa

因此，证明了我们的第二项观点。

**\*\*3.** 如果在双引号里面插变量的时候，后面不想有空格，可以拿大括号将变量包起来。**\*\***

那，如果我不想在后面有空格有，有特殊符号，就想直接显示变量\$huaqiangu呢？——我们的解决办法是用大括号把变量包起来。代码如下：

### 3.2.3 字符串

```
<?php
$huaqiangu = '花千骨';
//中间加了空格哟
$str = "{$huaqiangu}aaaa";

echo $str;
?>
```

于是，大家会发现花千骨后面没有了特殊符号，也没有空格了，达到了我们想要的显示结果，演示效果如下：  
![2015-07-26/55b4f6de368e1](http://box.kancloud.cn/2015-07-26\_55b4f6de368e1.png)

**\*\*4. 双引号解析转义字符，单引号不解析转义字符。但，单引号能解析\ ' 和\\\*\***

第4个字符串的注意事项比较难以观察，转义字符有一些，但最常用的有：

- \n 回车
- \t 缩进

我们来通过实验先来理解\n和\t是什么。

```
<?php
//声明一个字符串，记住是双引号
$string = "每天PHP学院\n给梦想\t一个机会";
echo $string;
?>
```

做这一块的实验中有一个注意事项，显示网页是看不出来效果的，如下：  
![2015-07-26/55b4f88038913](http://box.kancloud.cn/2015-07-26\_55b4f88038913.png)

为什么看不出来\n和\t代表什么含义，也看不出来双引号解析了\n和\t呢？—因为，你没有**\*\*点击鼠标右键查看源代码\*\***。我们来点击一下，再看看效果。

第一步：显示网页源代码  
![2015-07-26/55b4f916e79c5](http://box.kancloud.cn/2015-07-26\_55b4f916e79c5.png)

第二步：查看HTML源代码显示的结果：  
![2015-07-26/55b4f949219d9](http://box.kancloud.cn/2015-07-26\_55b4f949219d9.png)

大家来对比字符串变量：

```
> $string = "每天PHP学院\n给梦想\t一个机会";
```

\n在PHP学院的后面，在源代码里显示的结果是跟按了回车键一样的效果。  
\t在给梦想和一个机会中间。显示的结果是有几个空格的效果。\t相当于我们在电脑上按了一个tab键效果。

但是，我们同样执行一下上面的代码，但是\$string双引号改为用单引号：

```
<?php
//声明一个字符串，记住是双引号
$string = '每天PHP学院\n给梦想\t一个机会';
echo $string;
?>
```

执行一次看看效果：  
![2015-07-26/55b4fa4556af4](http://box.kancloud.cn/2015-07-26\_55b4fa4556af4.png)

你会发现—单引号中间的\t和\n直接显示出来了，而没有变成按一次回车键和按一次tab键后的效果。  
因此：

### 3.2.3 字符串

> 双引号执行转义字符而单引号不执行转义字符。

**\*\*5. 转义字符\（反斜线的使用）\*\***

如果我们声明一个变量\$beizi，我们想要在\$beizi声明的双引号中间加上一个双引号显示输出怎么办？

```
<?php
```

```
//要在$beizi的字符串中显示一个双引号怎么办？
```

```
$beizi = "多于绝大多数的人出生就是杯具，但是"我们在不断的让人生变为喜剧";
```

```
echo $beizi;
```

```
?>
```

如果真如上述所写代码，代码会报错。报错结果如下：

因为字符串：“多于绝大多数的人出生就是杯具，但是"我们在不断的让人生变为喜剧”必须写在双引号中间。而双引号中间还有一个双引号。也就是意味着字符串声明提前结束了，在“但是”处结束了。后面"我们在不断的让人生变为喜剧"，电脑不认识。

这真是一个杯具！那么办？

答：我们需要将双引号原有的意义去掉。再双引号前面加上一个\（反斜线，计算机的专业名词叫转义字符），就可以了。

```
<?php
```

```
//要在$beizi的字符串中显示一个双引号怎么办？
```

```
$beizi = "多于绝大多数的人出生就是杯具，但是\"我们在不断的让人生变为喜剧";
```

```
echo $beizi;
```

```
?>
```

你可以再执行一下上面的代码，发现不再报错了。同理可推，单引号当中要插入一个单引号显示出来的时候，也可以在单引号声明的字符串中的单引号前面加上\（反斜线，转义字符），将单引号的意义（限定字符区间）去掉。

```
<?php
```

```
//要在$beizi的字符串中显示一个双引号怎么办？
```

```
$shengyang = '\m xiaoshengyang';
```

```
echo $shengyang;
```

```
?>
```

**\*\*6. 双号和单引号可以互插！！！双引号当中插入单引号，单引号当中插入变量，这个变量会被解析。\*\***

代码如下：

```
<?php
```

```
$legend = '猛虎';
```

```
$NoAlike = "心有'$legend'细嗅蔷薇";
```

### 3.2.3 字符串

```
echo $NoAlike;
```

```
?>
```

执行结果如下：

```
![2015-07-26/55b4ff2d043ac](http://box.kancloud.cn/2015-07-26_55b4ff2d043ac.png)
```

**\*\*7. 神奇的字符串拼接胶水——（.）点，用来拼接字符串。\*\***

```
<?php
$huaqiangu = '花干骨';
//中间加了空格哟
$str = "{$huaqiangu}aaaa";

echo $str;
?>
```

上面这段代码是，我们曾经说过双引号能够解析变量，有一个问题：  
> 效率没有纯单引号的高

那么问题1：我想改为效率最高的方式应该怎么办？

问题2：我有多个字符串，要拼接在一声怎么办？

那，我们需要使用到神奇的胶水：（.）点，用来拼接字符串。

```
<?php

$shixi = '大学4年要好好学习';
';

$buran = '不然连实习的机会都没有';
';

$mimang = '把别人用来迷茫的时间拿到PHP学院';
';

$xuexi = '学习PHP';
';

//我们可以把字符串全部拼接起来。
echo $shixi . $buran . $mimang . $xuexi;

?>
```

因此，刚刚的问题一的代码我们可以改为：

```
<?php
$huaqiangu = '花干骨';
```



### 3.2.3 字符串

```
//中间加了空格哟
$str = $huaqiangu . 'aaaa';

echo $str;
?>
```

**\*\*8. 我们将定界符声明字符串视为双引号一样的功能来看待。\*\***

```
<?php

$weilai = '努力才有未来';
$mimang = '迷茫的原因是没有目标';

$dingjie = <<<ABC
如果
$weilai
非要\t在这"个滚"犊子
前
'$mimang'
面
```

```
加上\n一段
距离的话
我想说：
```

## 思想有多远，你就跟我滚多远

```
ABC;
echo $dingjie;
?>
```

你可以执行发现\$weilai,\$mimang,\t\n都可以执行，双引号单引号都能显示。这就是定界符的特点。

#### 再声明一次：字符串的声明每一项都必须记住，非常常用。并且面试题出现的概率非常高！

\* \* \* \* \*

不可见字符：主要是计算机规定的一些特殊符号。例如：回车（\n）、缩进（\t）等。

双引号执行变量，所以

## 3.2.4 浮点型

所谓浮点类型，可以理解为：我们数学里面的小数。

【注意】关于精度、取值范围和科学型声明不是学习的重点。因为此块在实际开发中用的特别少。我们将此块的知识点的学习标注为，了解级别。

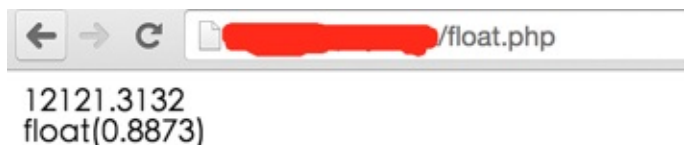
声明方式分为两种：

1. 普通声明
2. 科学声明

### 普通声明浮点数

```
<?php
//声明变量fudian的值为12121.3132
$fudian = 12121.3132;
echo $fudian;
//声明变量$f1 的值为0.8873
$f1 = 0.8873;
var_dump($f1);
?>
```

我们将文件保存到XAMPP的htdocs目录下，保存文件名为：float.php。在浏览器地址栏，输入：<http://127.0.0.1/float.php>，执行看看结果，如下：



echo 直接输出了12121.3132，而var\_dump 输出了0.8873，还显示了变量\$f1的类型为float。

var\_dump() 是一个函数。向括号()中间插入变量。这个函数，会打印出来数据类型，还会对应显示变量的长度和值。

var 是指变量的英文：variable

### float

读音：[flaʊt]

中文解释：计算机中的浮点类型

### variable

读音：[ˈveriəbl]

中文解释：变量

### dump

读音：[dʌmp]

中文解释：倾倒; 倾销;



## 3.2.5 重要：if和else语法

这是一个非常重要的章节，也是PHP当中的一个重要的语法。

【注意】我对这个语法的定义级别为：**默写级别**。也就是你需要，闭着眼睛，都能够写出来的东西。

英文中的解释：

### if

读音：[ɪf]

中文解释：如果

### else

读音：[ɛls]

中文解释：否则

我们将if和if...else组合成了四种基本语法，每一种都必须背下来。

```
<?php
if(布尔条件)
    布尔值为真(true)时执行，只能写一行代码；
?>
```

```
<?php
if(布尔条件)
    布尔值为真(true)时执行，只能写一行代码；
else
    布尔值为假(false)时执行，只能写一行代码；
?>
```

```
<?php
if(布尔条件){
    布尔值为真(true)时执行，可写多行代码；
}
?>
```

```
<?php
if(布尔条件){
    布尔值为真(true)时执行，可写多行代码；
}else{
    布尔值为假(false)时执行，可写多行代码；
}
?>
```

很多人喜欢买彩票，我们拿买彩票的过程来写个if的例子。

```
<?php
//定义一下中奖变量，变量的值为true,表示中奖了
$zhongjiang = true;
//由于$zhongjiang 结果为true，所以显示了：“买个房子”
//可以改为false试试执行结果，如果为false的话，不会执行echo '买个房子'；
```

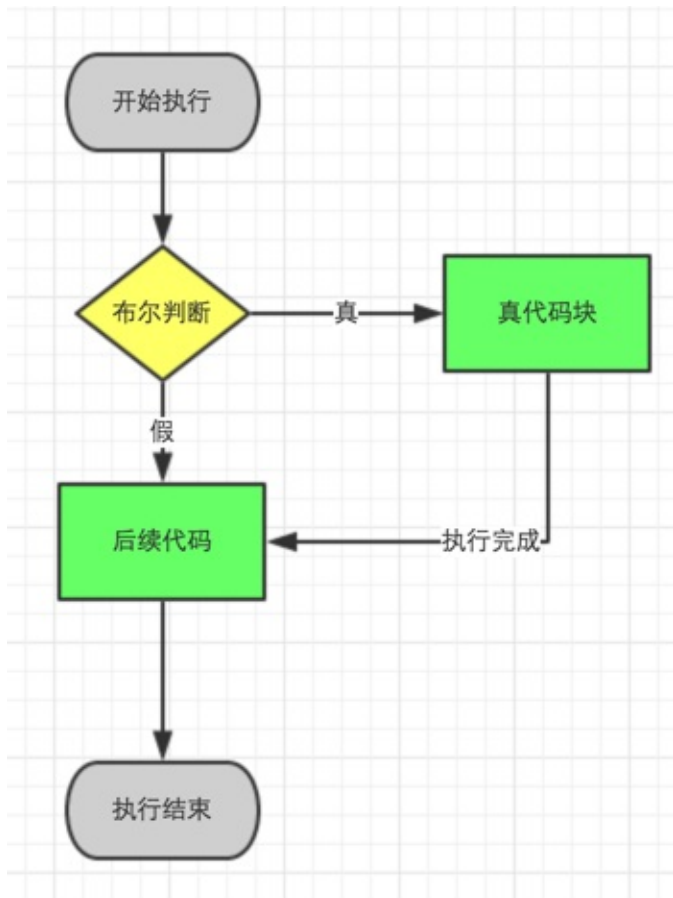
```
if($zhongjiang){  
    echo '买个房子';  
}  
//后续代码  
echo '该干嘛干嘛';  
?>
```

在《3.2.2布尔就是易经的知识》这一章中跟大家做了一个猥琐的举例：

例如电视剧里面经常讲到的一句话：

**如果那个漂亮妹子（帅哥）被我泡到手了，我死都愿意。**

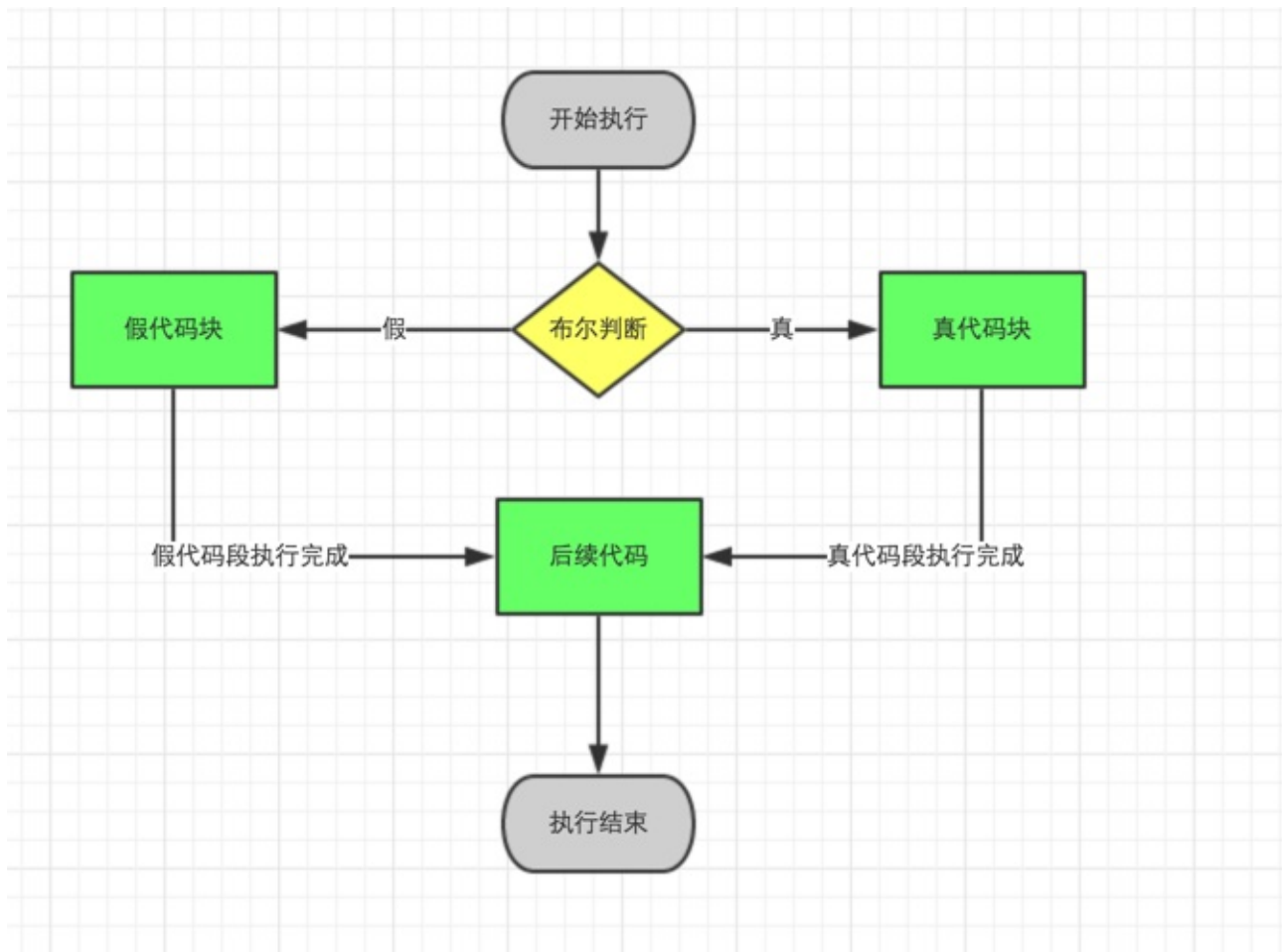
我们拿流程图来看就是如下的样子：



用计算机的代码完全可以写出这句话：

如果我泡到了漂亮妹子  
    则：我愿意去死  
如果没有泡到  
    则：我不愿意去死

如果按流程图来看的话就是这样：



上面的这个例子，我们完全可以用if...else...翻译成代码:

```
<?php
//我们定义一个泡到美女的变量($pao)为false, 意思为没泡到
$pao = false;

if($pao)
    //你可以试试在这儿写多行代码会不会报错。
    echo '我愿意去死';
else
    echo '我不愿意去死';

    //if...else执行结束, 后续代码
?>
```

在if...else中，我们再写一个可以用大括号括起来，多句话的例子：

```
<?php
//我们定义一个泡到美女的变量($pao)为true, 意思为泡到了
$pao = true;

if($pao){
    echo '我愿意去死';
    echo '林志玲, 我爱死你了.';
}else{
    echo '我不愿意去死';
    echo '凤姐, 我肯定不会爱你的';
}
//if...else执行结束, 后续代码
?>
```



## 3.2.6 NULL类型

空在英文里面表示是null，它是代表没有。空(null)不是false，不是0，也不是空格。

【重点】知道null产生的三种情况，学习empty 和 isset两个函数的区别。

主要有以下三空情况会产生空（ null ）类型：

1. 通过变量赋值明确指定为变量的值为NULL
2. 一个变量没有给任何值
3. 使用函数unset()将变量销毁掉

我们用代码来演示一下。

```
<?php
//声明变量为null
$n = null;
var_dump($n);

//var_dump显示输出变量$meiyou，看看结果是什么？
var_dump($meiyou);

//声明一个变量$iphone的值为字符串的手机
$iphone = '手机';
//unset销毁掉一个变量
unset($iphone);
var_dump($iphone);

?>
```

接下来我们来讲解两个跟null相关的函数，这两个函数非常常用，我们将级别定义为【默写级别】。

empty()可以向括号中间传入一个变量。这个变量的值如果为false或者为null的话，返回true。

```
<?php

$apples = null;
if(empty($apples)){
    echo '执行了真区间，凤姐，我爱你';
}else{
    echo '执行了假区间，你想凤姐了';
}
?>
```

上述实验证明，\$apple为null。将apple放至empty中间。结果执行了真区间。

isset()可以向括号中间传入一个或者多个变量，变量与变量间用逗号分开。只要有一个变量为null，则返回false。否则，则返回true。

```
<?php
//待会儿将变量$jia改为null再执行看看结果
$jia = false;

$result = isset($jia);
```



### 3.2.6 NULL类型

```
var_dump($result);
```

```
?>
```

```
<?php
$one = 10;
$two = false;
$three = 0;
$four = null;

$result = isset($one , $two , $three , $four);
//执行看看结果, 是不是
var_dump($result);

?>
```

---

unset()这个函数的功能是毁掉变量。unset(变量)括号中间插入想要毁掉的变量名，这个变量就会被毁掉。

英文说明

unset

读音：[ʌn'set]

解释：复原

## 3.2.7对象以后会学

---

对象类型大家这这里不用学习。

因为以后会有几个专门的章节来学习，你会觉得很爽的：



本章重点：

1. 知道对象是一个复合类型
2. 对象的英文叫object，var\_dump一个变量的时候看到的类型为object的，这个变量就是对象类型

---

所谓复合类型：就是在一个类型中可以同时存入字符串、浮点、整型、布尔等

### **object**

读音：['ɒbdʒɪkt]

解释：对象、物体、目标

## 3.2.8 数组会有单纯的一个章节

---

数组类型我们也不在这里学习，因为以后会也有一个章节来进行讲解。

本章重点：

1. 知道数组是一个复合类型
2. 数组的英文叫array，var\_dump一个变量的时候看到的类型为array的，这个变量就是数组类型
3. 会一个最简单、最基本的数组声明【默写】

接下来我们声明一个数组，了解数组里面可以插入多个值就行。

```
<?php
//定义$shu这个变量
//数组声明是向array里面插入一个或者多个值
//一个或者多个值用逗号分开
$shu = array(1,2,3);

?>
```

---

### array

读音：[ə'reɪ]

解释：数组、队列

## 3.2.9 资源类型

---

资源类型很多初学者觉得比较难以理解。因为资源类型打印出来只能够看到一个英文的resource。其他的什么在电脑上都不能显示出来，而他客观存在。

其实，资源类型很简单。我们针对词来进行说明。

电脑里面的资源是指：

1. word，excel等文件
  2. 有人收藏的美女图片和我们的自拍照片等图片
  3. 音乐
  4. 某些人爱看的AVI小电影
  5. 打开的网页
  6. 数据库
- ... ..

我们打开一个图片，用PHP来操作它，我们就是操作的一个资源。

我们打开的数据库连接，我们需要连接数据库，数据库就是我们操作的资源。

包括网络连接，连接后发送邮件，我们也可以认为是一个资源。

本章重点：理解资源就提操作我们可见和不可见的文件、网络和数据。我们在以后的章节中，操作图片和网络的时候大家会有更为深刻的印象。

---

英文解释

**resource**

读音：['ri:sɔ:rs]

解释：资源

## 3.2.10 眼前了解回调类型即可

---

callback，英文中的call是叫，back是返回，回来的意思。

callback 汉译过来的意思即为：回调。（执行完成，再调一次某个功能执行一次）

我们在讲解函数类型和对象的时候，我们会演示讲解回调函数。

---

### **call**

读音：[kɔl]

解释：呼唤，喊叫; 召唤，叫来，召集

back

读音：[bæk]

解释：后退; 倒退;

## 3.2.11 查看和判断数据类型

我们知道了一个数据的类型，才能进行下一步操作。后面的时候，大家可以学习到更多的知识——自定义功能（函数）。

我们来做一个场景模拟：（注：眼前不用会写这个函数，以后会教大家）

假设，我们可以写一个智能的功能（函数），它让有一个功能，就是打开一个装有学生成绩的电子表格。这个函数非常先进。

1. 只有一个人的时候，就返回一个人的成绩，成绩是整型或者浮点型变量。
2. 有2个或两个以上人的时候，返回一个数组类型变量，数组中装有多个人的成绩
3. 如果没有人的时候，则返回布尔值的变量，变量值为false

通过上面的一个场景模拟，我们知道了，一个函数（功能）有可能返回不同的数据类型。因此，我们可以根据不同的类型来做不同的事情。

判断数据类型很重要，我们需要严格掌握。

【默写级知识点】显示类型的函数、得到类型的函数、判断类型的函数

### 查看数据类型

1. `gettype`(传入一个变量) 能够获得变量的类型
2. `var_dump`(传入一个变量) 输出变类型和值

```
<?php
//声明一个变量88.8，你可以自己多做几次实验换成其他类型看看$type输出是多少
$float = 88.8;
$type = gettype($float);

echo $type;

?>
```

```
<?php

//多换几个类型试试
$str = '你喜欢尊上还是喜欢杀阡陌？';

var_dump($str);

?>
```

### 判断数据类型

我们使用`is_*`系列函数。`is_types`这一系列的函数，来进行判断某个东西是不是某个类型。如果是这个类型返回真，不是这个类型返回假。

`is_int` 是否为整型

`is_bool` 是否为布尔

`is_float` 是否是浮点

`is_string` 是否是字符串

`is_array` 是否是数组

本文档使用 [看云](#) 构建

is\_object 是否是对象

is\_null 是否为空

is\_resource 是否为资源

is\_scalar 是否为标量

is\_numeric 是否为数值类型

is\_callable 是否为函数

```
<?php
//is_* 系列函数有个特点，就是如果是这个类型就返回的是真。不是这个数据类型就返回的是假
//声明类型为假
$fo = false;

if(is_bool($fo)){
    echo '执行真区间';
}else{
    echo '执行假区间';
}

//检查未声明的变量$p是否为空，为空就执行真区间
if(is_null($p)){
    echo '执行真区间';
}else{
    echo '执行假区间';
}

//字符串类型的数值，看看执行的是真还是假
$str = '18.8';
if(is_numeric($str)){
    echo '执行真区间';
}else{
    echo '执行假区间';
}

?>

//把sca的值换成整型、浮点、布尔和字符串试试
$sca = true;
//如果为标量，执行真区间
if(is_scalar($sca)){
    echo '执行真区间';
}else{
    echo '执行假区间';
}

//换成echo,is_int试试，为什么echo执行假区间呢？
if(is_callable('var_dump')){
    echo '执行真区间';
}else{
    echo '执行假区间';
}
```

注：istypes 很好记。is 在前后面跟类型即可。

标量：整型（int）、浮点（float）、布尔（bool）、字符串（string）

混合类型：数组（array）、对象（object）

特殊类型：空（null）、资源（resouce）、回调（callback）

为什么最后的实验中echo执行假区间呢？

答：因为echo 不是函数，是基本语法。大家眼前可以不用理解函数和基本语法的区别。了解和不了解在实际开发中没有影

### 3.2.11 查看和判断数据类型

响。知道有这个知识点就行。



## 3.2.12 数据类型的自动转换和强制转换

PHP在PHP 5.x阶段都是完全的弱类型的编程语言。所谓弱类型，就是在声明变量的时候，不需要指定变量的类型。我要声明一个整型的变量，我不用在前面非得写上类型，再写变量。

而**PHP 7**的性能有很大的提高。实际测试的结果，PHP 7的性能与PHP5.6相比，提升了性能接近200%。在PHP 7 中有些地方，我们可以强制指定类型，也可以不用强制指定类型来声明变量。

我们接来下来讲强制类型转换和自动类型转换两个部份。

【默写级别】布尔值的自动类型转换、强制类型转换的英文单词

### 布尔值的自动类型转换

自动类型转换，就是数据类型在某些情况下，会自动变为其他的类型参与运算。自动类型转换的发生时机是：运算和判断的时候某些值会自动进行转换。

下面的情况是**布尔值判断时的自动类型转换**：

- 1, 整型的0为假，其他整型值全为真
- 2, 浮点的0.0，布尔值的假。小数点后只要有一个非零的数值即为真。
- 3, 空字符串为假，只要里面有一个空格都算真。
- 4, 字符串的0，也将其看作是假。其他的都为真
- 5, 空数组也将其视为假，只要里面有一个值，就为真。
- 6, 空也为假
- 7, 未声明成功的资源也为假

我们针对以上针理的规则一项一项来做实验看看结果。

#### 1，整型的0为假，其他整型值全为真

```
<?php
//整型的0，换成整型的其他值试试
$bool = 0;
if($bool){
    echo '美女美女我爱你';
}else{
    echo '凤姐凤姐爱死我，执行假区间咯';
}
?>
```

#### 2, 浮点的0.0，布尔值的假。小数点后只要有一个非零的数值即为真。

```
<?php
//浮点类型的的0，换成其他值试试
$bool = 0.0;
if($bool){
```

### 3.2.12 数据类型的自动转换和强制转换

```
        echo '美女美女我爱你';
    }else{
        echo '凤姐凤姐爱死我，执行假区间咯';
    }
    ?>
```

### 3，空字符串为假，只要里面有一个空格都算真。

```
<?php
//空字符串，中间没有空格哟。实验完加个空格试试
$str = '';
if($str){
    echo '美女美女我爱你';
}else{
    echo '凤姐凤姐爱死我，执行假区间咯';
}
?>
```

### 4，字符串的0，也将其看作是假。其他的都为真

```
<?php
//0这个字符串哟，试试其他值看看
$str = '0';
if($str){
    echo '美女美女我爱你';
}else{
    echo '凤姐凤姐爱死我，执行假区间咯';
}
?>
```

### 5，空数组也将其视为假，只要里面有一个值，就为真。

```
<?php
//这个数组当中啥也没放
$arr = array();
if($arr){
    echo '美女美女我爱你';
}else{
    echo '凤姐凤姐爱死我，执行假区间咯';
}
?>
```

### 6，空也为假

```
<?php
//声明了一个空的变量$bool
$bool = null;
if($bool){
    echo '美女美女我爱你';
}else{
    echo '凤姐凤姐爱死我，执行假区间咯';
}
?>
```

### 7, 未声成功的资源也为假

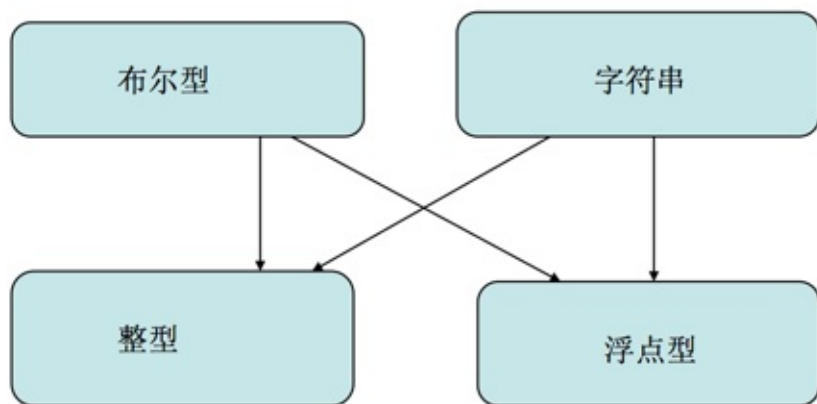
```
<?php
//下面这段代码会显示警告，可忽略。暂时只需要对着实验知道效果即可：未声成功的资源也为假
```

```
//下面这一块了解意思就行：打开adasfasfasfdsa.txt这个不存在的文件
$res = fopen('adasfasfasfdsa.txt','r');
if($res){
    echo '美女美女我爱你';
}else{
    echo '凤姐凤姐爱死我，执行假区间咯';
}
?>
```

## 其他类型的自动类型转换

自动类型转换在运算时也可以发生。跟所有我们总结的规律和观点一样：先总结、后实验。

只有标量在运算时会产生以下的自动类型转换：



```
<?php
//布尔变整型参与运算
$fo = true;

$result = $fo + 10;
//$result 结果为整型的11，因为$fo布尔的true变为了1
//如果$fo的值为0
var_dump($result);

//字符串类型
$str = '419不要爱';
$result = $str + 1;
//结果为420。因为将$str变为了整型的419参与运算
//将419放在字符串中间和结尾试试
var_dump($result);

?>
```

总结：

1. 布尔值的true参与运算是会变成整型或者浮点的1
2. 布尔值的false参与运算是会变成整型或者浮点的0
3. 字符串开始处是整型或浮点类型的字符，会转成对应的类型参与运算

## 强制类型转换

强制类型转换有三种方式：

1. 用后面的三个函数可以完成类型转换，intval()、floatval()、strval()

2. 变量前加上()里面写上类型，将它转换后赋值给其他变量
3. settype(变量，类型) 直接改变量本身

我们来进行实验：

#### intval()、floatval()、strval()转换

```
<?php
    $float = 1.23;
    $result = intval($float);
    //看看结果是不是变了？
    var_dump($result);

    //鸭脖子为整型的5
    $yabozi = 5;
    $re = floatval($yabozi);
    var_dump($re);

    //定义整型的变量
    $yabozi = 23;
    $bian = strval($yabozi);
    //强制变成字符串试试
    var_dump($bian);

?>
```

#### 变量前加上()里面写上类型，将它转换后赋值给其他变量

```
<?php
//定义一个变量，我们来变化一下试试
$transfer = 12.8;
//把浮点变为整型
$jieguo = (int)$transfer;
var_dump($jieguo);

//把浮点变为布尔
$jieguo = (bool) $transfer;
var_dump($jieguo);

//把布尔变整型
$bool = true;
$jieguo = (int)$bool;

//把浮点变数组
$fo = 250;
$jieguo = (array)$fo;
var_dump($jieguo);

//其他的操作方式，按照文字总结的规律你来试试

?>
```

#### settype(变量，类型) 直接改变量本身

```
<?php

//定义浮点变为整型
$fo = 250.18;
//settype第二个参数是int，你实验的时候要记得第二个参数要为字符串类型
settype($fo, 'int');
//输出看看结果
var_dump($fo);
```

```
?>
```

**【你试试】** 以下是强制类型转换时的特点，每一项你做实验看看对不对：

1. 空转为整型会为整型的0
2. 空转为浮点会为浮点的0
3. 空转为字符串会为空字符串''
4. 浮点的123.0转为字符串会为字符串123
5. 浮点的123.2转为字符串会为字符串的123.2
6. 浮点即使小数点再大，它都会被干掉，会舍掉小数点后面的值
7. 如果字符串转为整型的时候，如果数值在前面，会将前面的数值拿出来做为整型的转换值。
8. `settype(变量,'null');` 等价于 `unset()`一个变量
9. `$目标变量 = (类型)$操作变量` 只会改变目标变量的类型，不会改变原变量的类型，`Settype`是改变原值

```
<?php
    //小可爱，记得自己做实验，验证上面的9点哟
    $t=12.9;

    settype($t, 'int');

    var_dump($t);
?>
```

---

#### set

读音：[sɛt]

解释：设置

#### type

读音：[taɪp]

解释：类型、方式

## 3.3 常量和变量

---

[3.3.1 用常量限制用户跳过某些文件](#)

[3.3.2 可变变量](#)

[3.3.3 外部变量](#)

[3.3.4 环境变量](#)

[3.3.5 变量引用](#)

## 3.3.1 用常量限制用户跳过某些文件

常——汉语字面为：长久，经久不变。

常量那就好翻译了：长久不变的值。

【重点知识】知识级别我们定义为：默写级

常量的使用范围非常广泛。我们在以后，定义我们的工作目录、定义一些特点的帐户密码、版本号等我们都会使用到常量。所以这一块的知识，非常重要。

常量在代码中的定义、书写方式：

### define(常量名，常量值)

注：

1. 常量值只能为上一章中我们讲到的标量。
2. 常量名可以小写，但是通常大写
3. 常量名可以不加引号，但是通常加上引号。
4. 在字符串中调用常量时候，必须在引号外面
5. 常量名建议只用字母和下划线

我们用代码来实验一下：

#### 1.定义和调用一次常量试试

```
<?php

define('MY_NAME','PHP学院');

echo MY_NAME;
//下面是错误的调用方式
echo '我的名字是MY_NAME';
//正确的调用方式该这么写
echo '我的名字是' . MY_NAME;
?>
```

#### 2.注意项目实验

```
<?php
//其实可以小写，但是不好区分，所以我们规定通常大写
define('xiaoxie',true);
echo xiaoxie;

//常量可以在外面不加引号
define(YH,'不要对未来迷茫，迷茫的时候静下心来coding');
echo YH;

//只能用标量，我在后面用了一个数组，大家学一下就行，会报错的哟
define('BIAO',array(1,2,3));

?>
```

此外，系统还为我们准备了一些内置的常量。这些常量都是规定好的。我们先熟悉几个，还有更多的系统常量等我们学习完上册，入门后再慢慢的增加和学习。

常量名	说明
LINE	当前所在的行
FILE	当前文件在服务器的路径
FUNCTION	当前函数名
CLASS	当前类名
METHOD	当前成员方法名
PHP_OS	PHP运行的操作系统
PHP_VERSION	当前PHP的版本
TRAIT	Trait 的名字,php5.4新加
DIR	文件所在的目录
NAMESPACE	当前命名空间的名称（区分大小写）

defined()函数来做安全机制

此处知识，可以先不用掌握。等学习完函数后，再来看本部分代码也可以。

defined()我们来学习这种用法，主是是为了防止其他人绕过安全检查文件。

函数：defined(常量)

功能：向函数的括号后面传入常量，如果常量定义了就返回true，否则返回false

【情景模拟】假设，我们的这套在线电子商城的软件需要付钱，检查是否付费是通过对软件授权检查来完成的，而文件version.php中就有检查授权的功能，我们在软件中规定，没有授权检查文件version.php就不能使用这个软件。所有的代码都包含了version.php。并且为了防止有人盗版，我还可以把version.php的代码进行了加密。

我们有两个文件：

- 1. 一个文件中间有版本号，版本声明和授权声明。文件名为version.php
- 2. 一个文件中有具体的业务功能。例如：用户注册、登陆等，文件名为users.php

我们该怎么做呢？——也就是说不包含 version.php文件就不让，执行users.php之后的代码。

我们来进行实验：

version.php文件

```
<?php

//此处是检查是否是否授权的业务部份代码xxxx
define( 'AUTH', true);

//略过模拟代码xxx行

?>
```

users.php



### 3.3.1 用常量限制用户跳过某些文件

```
<?php
//尝试将include 'version.php'这一行代码注释后再执行看看，对比结果
include 'version.php';

if(!defined('AUTH')){
    echo '非法！非法！你尝试跳过授权文件';
    exit;
}

//模拟后面用户注册和用户登陆的代码xxx行
echo '用户注册';
?>
```

实验结果可知：version.php必须要包含，不然不会显示后面的echo '用户注册'；

include

读音：[ɪnˈklud]

解释：包含

version

读音：[ˈvɜːʃn]

解释：版本

user

读音：[ˈjuːə]

解释：用户

复数：users

define

读音：[dɪˈfaɪn]

解释：规定

函数：include('传入文件路径和文件名')

功能：这个函数的功能是传入指定路径的文件，让PHP包含进来执行

注意：在后面的章节中会专门讲解和实验include

## 3.3.2 可变变量

可变变量，这个词解释的太过于高大上。看起来很有“bigger”。它还有一个叫法，叫作变量的变量。

我们认为这些叫法上都不太科学。毕竟都是翻译过来的舶来之物。

可变变量其实就是——已声明的变量前，再上变量符。

举例说明：

```
<?php
//定义了一个变量叫作 $shu 将$shu这个变量的值设为字符串的biao
$shu = 'biao';
//定义了一个【变量】$biao。将他的值设置为鼠标
$biao = '鼠标';

//$$shu 就是可变变量：在已声明的变量$shu前又加上了一个变量符
echo $$shu;
?>
```

上面的过程说明：\$shu的值为字符串的'biao'。我在\$shu前再加上一个\$(美元符号)，可以理解成为以下的变形过程：

```
$$shu
${$shu} 分成两块来看
${'biao'} 把变量$shu解释成了biao
$biao 而$biao也是一个变量对应的值是：鼠标
```

你可以自己写几个可变变量玩玩，请问以下的代码运行结果是多少？

```
<?php
$shu = 'biao';
$biao = 'wo';
$wo = 'test';
$test = 'sina';
$sina = 'zhongguo';
$zhongguo = 'china';
$china = '我爱你';
//别运行，自己去推理一下代码。也写几个可变变量玩玩吧！
echo $$$$shu;
?>
```

## 3.3.3 外部变量

PHP的外部变量是PHP 在使用过程中规定好的一些变量。这个变量的规定是这样规定的，就这样使用。

我们先讲解几个最常用的例子,我们将下面的表单命名为user.html：

```
<html>
  <head>
  </head>

  <body>
    <form action="reg.php" method="get">
      <input type="text" name="username" />
      <input type="password" name="pwd" />
      <input type="submit" value="提交" />
    </form>
  </body>
</html>
```

上现是很基础的一段HTML代码，在这段代码的主要意思是把用户和密码，采用get方法，将数据发送给reg.php（在上面代码的第6行规定的）。reg.php想办法接收用户传过来的username和pwd这两个值。

我们得出我们的第一个外部变量：\$\_GET。

\$\_GET 的主要作用是将得到get传值的数据。

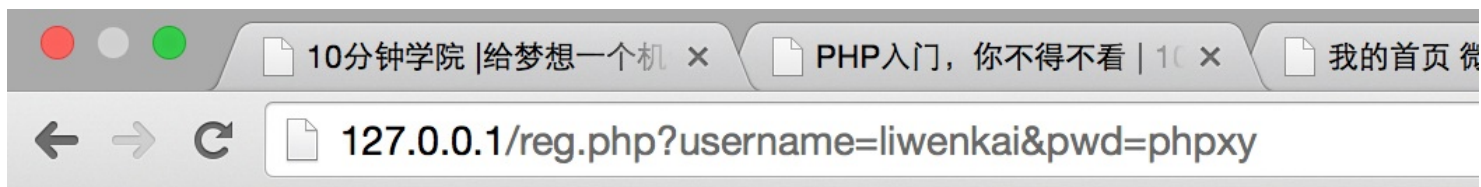
我们写一个reg.php，用\$\_GET来接收值试试：

```
<?php
//$_GET后面加上中括号，将username作为字符串放在中括号里面，就得到了表单里面的<input type="text" name="username" /> 的值
$u = $_GET['username'];
echo $u.'<br />';

//$_GET['pwd'] 得到表单<input type="password" name="pwd" /> 的值
$passwd = $_GET['pwd'];
echo $passwd.'<br />';
?>
```

你可以输出值看一下结果。通过上面的实验我们知道了，通过\$\_GET这个外部变量，可以得到从表单输入的值。

大家在实验的时候会发现地址栏上面有一个特点：



liwenkai  
phpxy

根据上图，观察特点：

1. reg.php后面跟了一个? ( 问号 )
2. 表单里面的username变在了地址栏里面
3. 表单面username的值输入的是liwenkai，在username后面跟了个= ( 等号 ) 输入的值
4. username ( 名字 ) =liwenkai ( 值 ) 后面的密码是password ( 名字 ) =NoAlike ( 值 )，中间有&(and符)分开

密码是可见的，那怎么样保证安全呢。让我在注册的过程当中，密码不在地址栏里面被人看到呢？

这个时候我们需要使用到post传值，post传值是在地址栏中不可见的。

同样上面例子的代码我们进行修改，html代码如下：

```
<html>
  <head>
  </head>

  <body>
    <!-- 这一行method 对应的值改为了post -->
    <form action="reg.php" method="post">
      <input type="text" name="username" />
      <input type="password" name="pwd" />
      <input type="submit" value="提交" />
    </form>
  </body>
</html>
```

PHP的代码里面的\$\_GET全改为了\$\_POST：

```
<?php
//$_POST后面加上中括号，将username作为字符串放在中括号里面，就得到了表单里面的<input type="text" name="username" /> 的值
$u = $_POST['username'];
echo $u.'<br />';

//$_POST['pwd'] 得到表单<input type="password" name="pwd" /> 的值
$passwd = $_POST['pwd'];
echo $passwd.'<br />';
?>
```



观察特点

reg.php后的? ( 问号 ) 不见了。后面的username和password也不见了。那他怎么传递数据的呢？

他是我们看不见的浏览器的请求头文件传递的数据。所以在URL ( 网址 ) 栏不可见。

注：附录中有如何通过火狐浏览器的firebug查看传递结果的演示过程。这一块都是HTTP协议规定的传送方式。

除此之外，我们还有\$\_REQUEST来接收数据。现在我们这样处理：

将php代码段中的\$\_POST全改为\$\_REQUEST，代码如下：

```
<?php
$u = $_REQUEST['username'];
echo $u.<br />';

$passwd = $_REQUEST['pwd'];
echo $passwd.<br />';
?>
```

把网页user.html里面的这一行中的method，改为get执行一次，再改为Post再运行一次，看看结果：

```
<form action="reg.php" method="post">
```

通过上面的实验你会发现\$\_REQUEST即可以接收get传值也可以接收post传值。

另外，我们总结一些外部变量，要求知识点的学习级别：了解含义，默写这个单词的写法和作用。

全局变量名	功能说明
\$_COOKIE	得到会话控制中cookie传值
\$_SESSION	得到会话控制中session的值
\$_FILES	得到文件上传的结果
\$_GET	得到get传值的结果
\$_POST	得到post传值的结果
\$_REQUEST	即能得到get的传值结果，也能得到post传值的结果

请再记一句话：以上这些变量全是超全局的。（以后会讲解超全局的含义）。

注：

1.我们认为从用户输入过来的所有数据都不是可信的。本书的下半部分会专门讲解限制和过滤

2.在提交数据的时候，我们常用的方法有get和post。可以这样理解，get传值在url中可见，而post传值在url中不可见。



而post传值在url中不可见，是通过浏览器的header头部分将数据发送给指定服务器的。需要通过专门的工具才能看到post发送的值为什么。你可以下载火狐浏览器（firefox）的插件（firebug）来查看。

火狐浏览器的图标：



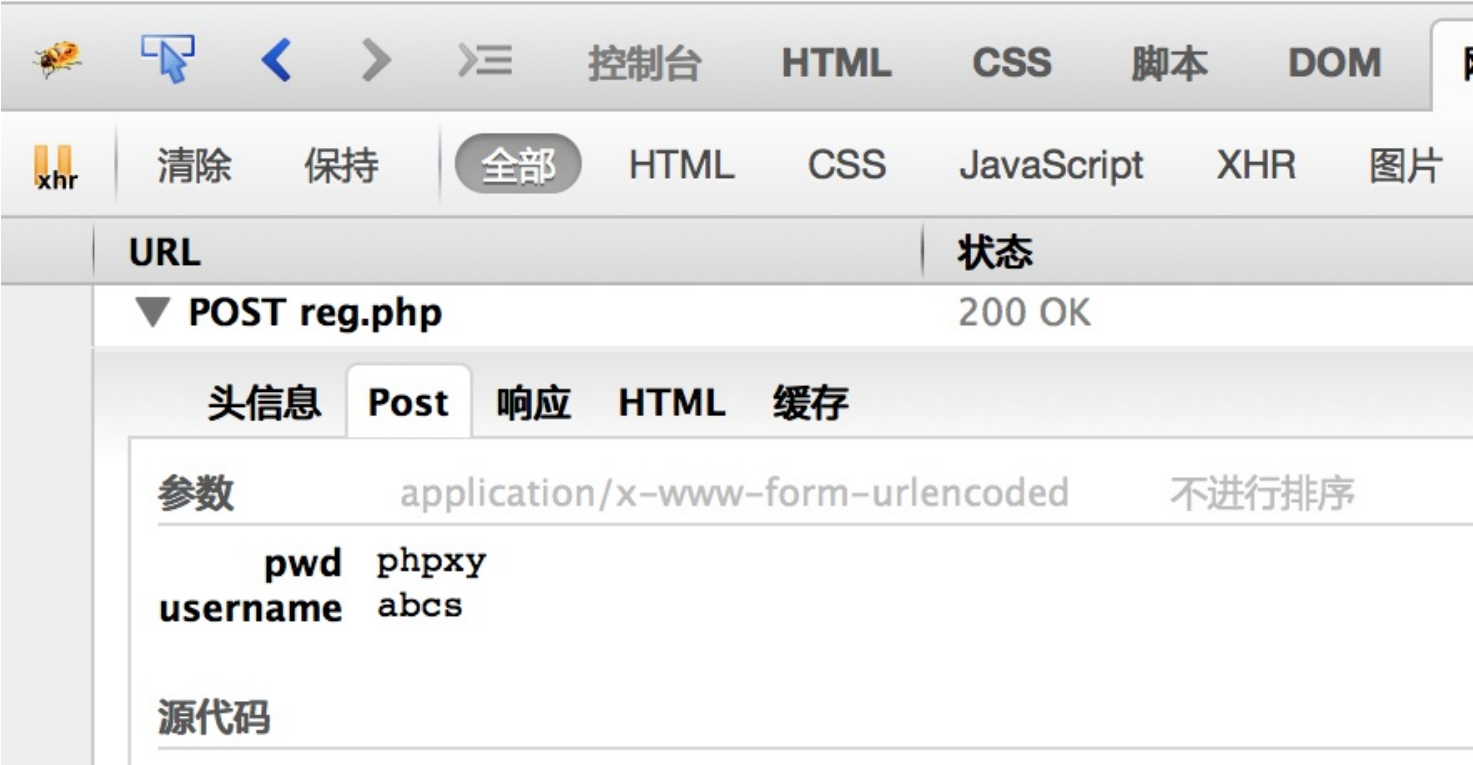
打开firebug：

om/user.html

官方网站 ▾  新手上路  常用网址 ▾



查看header头传递数据（网络，点击POST reg.php 选择Post），就看到了传递的名字和传递的数据值：



1. 若真是使用get传密码，密码在地址栏里面显示过后。浏览器的历史记录会自动记录访问过的地址。恶意用户会通过查看你的浏览器的历史记录，得到你曾经输入的密码。因此，不能使用get方式来作密码的传输方式。



### 3.3.3 外部变量

解释：得到，计算机里是指一种传送数据的方式

post

读音：[pəʊst]

解释：计算机里指一种数据传递方式

request

读音：[rɪ'kwest]

解释：请求

submit

读音：[səb'mɪt]

解释：提交，呈送

action

读音：['ækʃən]

解释：行动、活动

## 3.3.4 环境变量

---

环境变量我们主要用的有\$\_SERVER和\$\_ENV两个环境变量。

不过，\$\_ENV逐渐被PHP的新版本给废弃了。

**【重点】**知道在哪儿查看环境变量的名字（key）和值（value），记住并默写几个常用的环境变量。

查看环境变量，我们在学习PHP的第一天就学习过：

```
<?php
phpinfo();

?>
```

其实环境变量不是不用背的，我了解在哪儿能够找到环境变量的key（键）和值即可。  
我们学了phpinfo();打印出来了一批乱乱的东西，我们今天来学习其中的环境变量部份。

你在执行phpinfo();的这个网址中，将网页向下拉动翻页，看看能不能找到截图中的部份：

## PHP Variables

Variable	Value
_SERVER["USER"]	www
_SERVER["HOME"]	/home/www
_SERVER["FCGI_ROLE"]	RESPONDER
_SERVER["QUERY_STRING"]	no value
_SERVER["REQUEST_METHOD"]	GET
_SERVER["CONTENT_TYPE"]	no value
_SERVER["CONTENT_LENGTH"]	no value
_SERVER["SCRIPT_NAME"]	/reg.php
_SERVER["REQUEST_URI"]	/reg.php
_SERVER["DOCUMENT_URI"]	/reg.php
_SERVER["DOCUMENT_ROOT"]	/alidata/www/buqiu.com/weixin/
_SERVER["SERVER_PROTOCOL"]	HTTP/1.1
_SERVER["GATEWAY_INTERFACE"]	CGI/1.1
_SERVER["SERVER_SOFTWARE"]	nginx/1.7.4
_SERVER["REMOTE_ADDR"]	202.108.31.94
_SERVER["REMOTE_PORT"]	13194
_SERVER["SERVER_ADDR"]	42.96.157.23
_SERVER["SERVER_PORT"]	80
_SERVER["SERVER_NAME"]	weixin.buqiu.com
_SERVER["REDIRECT_STATUS"]	200
_SERVER["PATH_INFO"]	no value
_SERVER["PATH_TRANSLATED"]	/alidata/www/buqiu.com/weixin/
_SERVER["SCRIPT_FILENAME"]	/alidata/www/buqiu.com/weixin//reg.php
_SERVER["HTTP_HOST"]	weixin.buqiu.com
_SERVER["HTTP_ACCEPT"]	text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8
_SERVER["HTTP_UPGRADE_INSECURE_REQUESTS"]	1
_SERVER["HTTP_USER_AGENT"]	Mozilla/5.0 (Macintosh; Intel Mac OS X 10_10_3) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/44.0.2403.107 Safari/537.36
_SERVER["HTTP_DNT"]	1
_SERVER["HTTP_REFERER"]	http://weixin.buqiu.com/user.html
_SERVER["HTTP_ACCEPT_ENCODING"]	gzip, deflate, sdch
_SERVER["HTTP_ACCEPT_LANGUAGE"]	zh-CN,zh;q=0.8,en;q=0.6
_SERVER["HTTP_X_FORWARDED_FOR"]	unknown
_SERVER["HTTP_CACHE_CONTROL"]	max-age=259200
_SERVER["HTTP_CONNECTION"]	keep-alive
_SERVER["PHP_SELF"]	/reg.php
_SERVER["REQUEST_TIME_FLOAT"]	1438498706.4377
_SERVER["REQUEST_TIME"]	1438498706

\_SERVER['中间的值']，是需要我们了解意思的。

如果我们需要显示我们当前访问的这个phpinfo()页面文件放在哪儿，就可以执行：

```
<?php
//我在上图左侧找到的一项，在前面加上了一个$(美元符)，就显示出来了当前文件的路径
echo $_SERVER['SCRIPT_FILENAME'];

?>
```

我们来了解一些常用的环境变量的键名和值对应的意思：

键名	含义
\$_SERVER["REQUEST_METHOD"]	请求当前PHP页面的方法
\$_SERVER["REQUEST_URI"]	请求的URI
\$_SERVER["SERVER_SOFTWARE"]	用的是哪一种服务器
\$_SERVER["REMOTE_ADDR"]	客户的IP地址
\$_SERVER["SERVER_ADDR"]	当前服务器的IP地址
\$_SERVER["SCRIPT_FILENAME"]	主前请求文件的路径
\$_SERVER["HTTP_USER_AGENT"]	当前访问这个网址的电脑和浏览器的情况
\$_SERVER["HTTP_REFERER"]	上级来源（用户从哪个地址进入当前网页的）
\$_SERVER["REQUEST_TIME"]	当前的时间

URI 和URL都是网址，但是URL带有了主机地址部份，而URI不带主机地址部份，例如：

<http://www.phpxy.com/abc.php?username=liwenkai>

上面是一个URL（统一资源定位符），而URI是不带主机和(<http://>)协议的部份：

abc.php?username=liwenkai

time

读音：[taɪm]

解释：时间

file

读音：[faɪl]

解释：文件

name

读音：[neɪm]

解释：名字

software

读音：[ˈso:ftwer]

解释：软件

address(简写addr)

读音：[ˈædres]

解释：地址

remote

读音：[rɪˈmɔʊt]

解释：远程，遥远的

server

读音：[ˈsɜ:və(r)]

解释：服务，服务器

method

### 3.3.4 环境变量

读音：['mɛθəd]

解释：方法

port

读音：[pɔ:rt]

解释：端口

## 3.3.5 变量引用

变量引用很多老师喜欢来用C语言的指针来去讲解。我们作为有这么多年开发和教学经验的人来说——大多数学习PHP的人来说根本不了解C语言。

使用C语言一指针来讲解变量引用，我们觉得画蛇添足。并且，不利于没有C语基础的朋友们学习。

关于变量引用的知识点，请以我们的讲解为理解的基准！

我们来对比两段代码的执行结果：

第一段代码，没有任何区别。跟我们原有的PHP代码一模一样：

```
<?php

$fo = 5;
// $fo的值为5，将5赋值
$bar = $fo;
// $bar的值原来为5，现在将值改为6
$bar = 6;
// $bar的结果为6
echo $bar.'<br />';
// $fo的结果为5
echo $fo.'<br />';

?>
```

第二段代码：

```
<?php

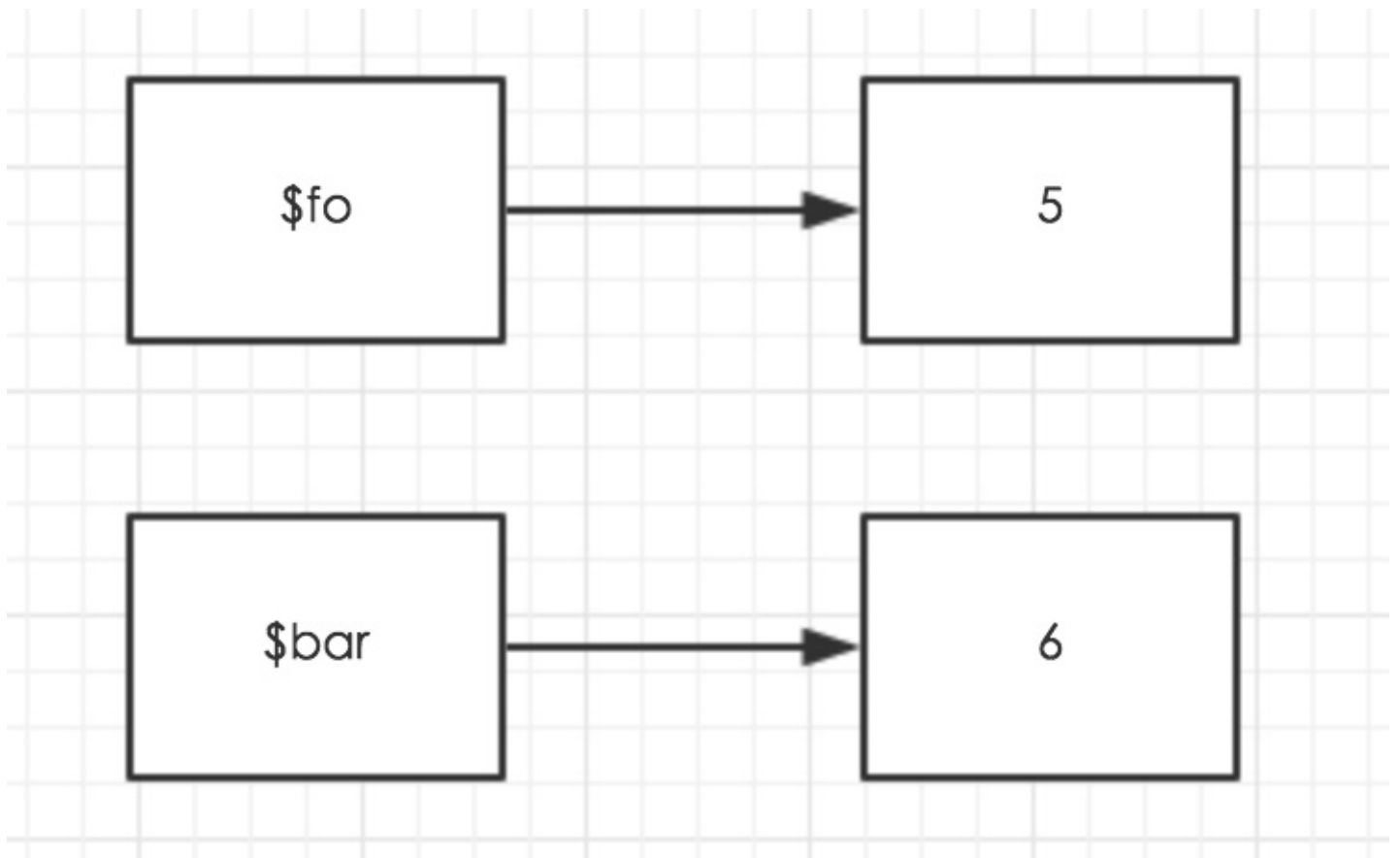
$fo = 5;
// 注意，加上了一个&符号
$bar = &$fo;

$bar = 6;
// $bar的结果为6
echo $bar.'<br />';
// $fo的结果为6
echo $fo.'<br />';

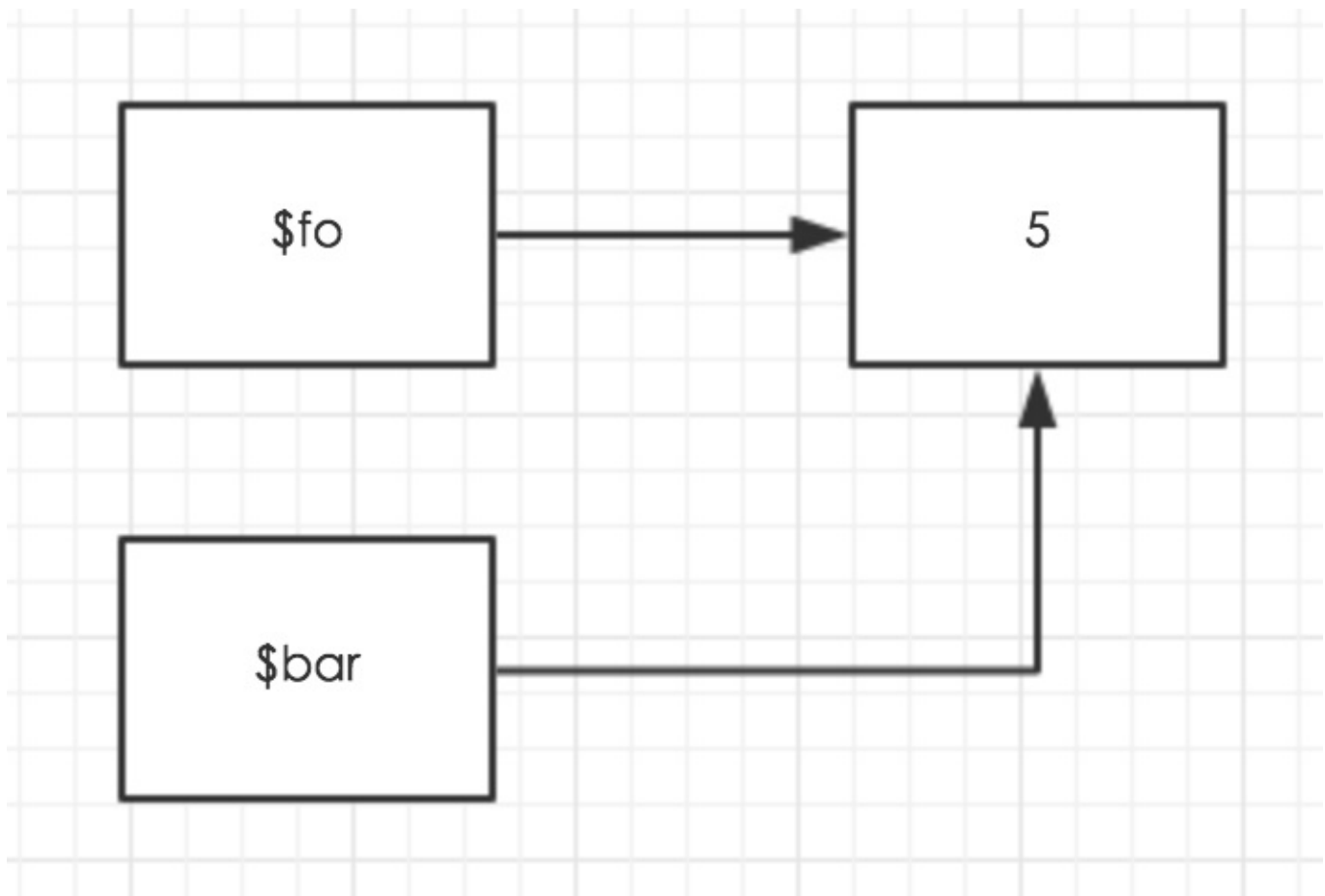
?>
```

为什么两个结果都变成6了呢？

我经常跟大家这样来举例。我们可以这样来想象：一个变量名，对应了一个数据值。如下图：



而加上&（ and 符后），把变量指向同一个存值空间了，如下图：



也就是不论 $\$fo$ 或 $\$bar$ 的值如何发生变化， $\$fo$ 变 $\$bar$ 也变， $\$bar$ 发生变化， $\$fo$ 也会发生变化。

举个不靠谱的例子，方便你来理解：

你家有只小狗，叫作李文凯。又有一只小狗叫作狗蛋。有一天，你把狗蛋给吃了。把狗蛋这个名字给了李文凯

那么，你不论是打狗蛋一下，还是打李文凯一下都是打的同一条狗。

---

注：如果你理解、了解、熟悉C语言的指针。可自行用指针方式来理解，不在本文的讨论范围内。



## 3.4 PHP表达式与运算符

---

3.4.1 算术运算

3.4.2 赋值运算

3.4.3 自加、自减运算

3.4.4 比较运算

3.4.5 逻辑运算

3.4.6 位运算

3.4.7 运算符优先级

3.4.8 三元运算符和其它运算符

# 3.4.1 算术运算

算数运算符，就是大家小学所学绝大多数知识：

符号	说明	举例
+	加号	\$x + \$y
-	减号	\$x - \$y
*	乘号,乘以	\$x * \$y
/	除号,除以	\$x / \$y
%	取余也叫取模、求模	\$x % \$y

```
<?php

$x = 5;

$y = 6;
//5+6为11
echo $x + $y;

?>
```

```
<?php

$x = 20;

$y = 15;
//20 -15 为5
echo $x - $y;

?>
```

```
<?php

$x = 20;

$y = 15;
//20乘以15结果为300
echo $x * $y;

?>
```

```
<?php

$x = 10;

$y = 5;
//10除以5 结果为2
echo $x / $y;

?>
```

### 3.4.1 算术运算

```
<?php

$x = 10;

$y = 3;
//$x 不能整除3, 得到的余数为1, 所以结果输出为1
echo $x % $y;

?>
```

与我们数学所学一样，也有优先级：先乘除，后加减。如果你想更明确的改变优先级，那就不用()【小括号】，将想要优先的运算给括起来。

# 3.4.2 赋值运算

赋值运算，我们已经学过了。

我们在数学里面把 = (一个等号)叫作赋值运算符，即：把等号右边的值，赋值给等号左边的变量，左边的变量就为右边的值。

代码是从上到下运行的，因此赋值就可以从上至下的反复赋值：

```
<?php

$x = 5;

$x = true;

$x = '爱你';

$x = 12.888;

echo $x;
?>
```

上例中x被反复赋值为不同的类型的值了。上面我们已经学习过了很多次了。

那么PHP的赋值运算符还有几个：

符号	举例	等价式
+=	\$x += \$y	\$x = \$x + \$y
-=	\$x -= \$y	\$x = \$x - \$y
*=	\$x *= \$y	\$x = \$x * \$y
/=	\$x /= \$y	\$x = \$x / \$y
%=	\$x %= \$y	\$x = \$x % \$y
.=	\$x .= \$y	\$x = \$x . \$y

上面的例子和等价式都有明确的说明。

\$x += \$y 等价于 \$x = \$x + \$y

```
<?php

$x = 5;

$y = 8;

$x += $y;

echo $x;
?>
```

以上的全部记住即可。都是一些语法上的规定。



# 3.4.3 自加、自减运算

自加自减，就是把自己加1，或者减1。  
如果你学习过其他编程语言。你会发现，此处的用法又是计算机里面的一个规定。可以这样用，用起来更加简洁。

符号	说明
\$x++	先赋值后加
\$x--	先赋值后减
++\$x	先加后赋值
--\$x	先减后赋值

上面的用法其实挺简单的，按照上面的例子。我们分成步骤去，按过程去判断即可。

```
<?php

$x = 5;
//先赋值后加：即先将$x的值赋值给$y。$x的值为5，所以将$x的值赋值给$y。$y也为5
$y = $x++;
//$x的结果输出为6，因为赋值给$y后，$x自己又把自己进行了+1操作。所以，$x的结果为6
echo $x;
?>
```

我们来对比一下先加后赋值，如下：

```
<?php

$x = 5;
//先将$x自加1，$x等于5，自加1后结果为6。因此，$y的结果为6
//自加后，再将结果6偷偷的赋值给自己$x
$y = ++$x;
//$x的结果输出也为6。因为$x执行+1完成后，将5+1的结果赋值给了自己
echo $x;
?>
```

你再做实验，试试\$x-- 和 --\$x的结果是这样吗？

请回答，下面\$water + \$paper的结果为多少？

```
<?php

$x = 5;
$y = 6;

$foo = $x++ + $x--;
$bar = ++$y + ++$x;
$cup = $x-- + $y--;
$paper = ++$x + $x++;
$water = $y-- + $x--;

echo $water + $paper;
?>
```



# 3.4.4 比较运算

比较运算符，在大家小学的时候就学了很多，有：

说明	符号
大于	>
小于	<
大于等于	$\geq$
小于等于	$\leq$
不等于	$\neq$
等于	=

不过在在在PHP里面多出了一种运算符：

说明	符号
大于	>
小于	<
大于等于	>= (电脑打不出 $\geq$ 所以用>=)
小于等于	<= (电脑打不出所以用 $\leq$ <=)
不等于	!= ( 不等的另外一种写法 ) 电脑打不出 $\neq$
等于	== ( 赋值是=号，所以 = = 规定为等于 )
全等 ( 判断类型等于 )	===
全不等 ( 判断类型不等于 )	!==

我们来回顾一下小学时候学的知识：

```
x = 5
y = 6
x<y 是对的，因为x为5，y为6所以x小于y这个判断式是对的
x>y 是错的，因为x为5，y为6所以x大于y这个判断式是错的
```

我们小学的时候就学过类似于这样的判断题。  
而电脑里面的对错，就是bool（布尔）数据类型的真(true)和假(false)。



那，我们是不是可以用之前所学的if...else来判断类型呢？

```
<?php
$x = 5;

$y = 6;
//因为5大于6不成立，所以为错的。即为false执行了假区间
if($x > $y){
    //真区间
    echo '变量x 大于 变量y, 成立';
}else{
    //假区间
    echo '变量x 大于 变量y, 不成立';
}

?>
```

我想，如果小学毕业的你。小于、小于等于、大小等于、不等于都会做实验了，请你实验几次。并且，完全能够默写！

接下来的一个重点是演示，等于（ == ）和全等于也叫判断类型等于（ === ）。

我们来写一段代码，大家看看两段代码就是符号PHP学院，为什么结果有这么大的差距呢？

下面这段代码执行出来的结果，运行了真区间。

```
<?php
$x = 5;
$y = '5';
if($x == $y){
    echo '结果为真';
}else{
    echo '结果为假';
}

?>
```

下面这段代码执行出来的结果，运行了假区间。

```
<?php
$x = 5;
$y = '5';
if($x === $y){
    echo '结果为真';
}else{
    echo '结果为假';
}

?>
```

我们对比区别发现：

下面这段代码是===（三个等号，我们说他还有一个名字是判断类型等于）。而\$x为整型的5.\$y为字符串类型的5.类型不同，所以执行的是假区间。而上面一段代码，是两个等号不判断类型，所以执行的是真区间。

# 3.4.5 逻辑运算

逻辑运算符比较简单，是我们人类进行逻辑思考时的一种方式。

讲出很多屌丝男士的心愿：某一个女人，要不长得漂亮、要不有钱我就娶她。如果一个条件都没有，那就算了。

上面的这种不评价好坏的心理状态，只是来说明这就是典型的计算机思维。

美丽这个条件成立(true) 或者 财富这个条件成立(true)，则执行娶她的行为和动作。否则，就不娶这个女孩。

那我们把这些逻辑关系进行了归纳和总结，下面表格中的：\$x为条件一，\$y 为条件二。进行说明：

逻辑与，中文解释为并且。可理解为，\$x 并且 \$y 都为真(true)的时候执行。

逻辑或，中文解释为或者。可以理解为,\$x或者\$y其中一个为真（ true ）的时候执行。

逻辑非，中文解释取反。如果\$x的执为假（ false ），进行一次非操作。非假（ false ）就为真了，就可以执行真区间了。反推 true，则执行假区间了。

逻辑异或，如果\$x和\$y相同则为false，不相同则为true。

举例	说明	详细说明
\$x and \$y	逻辑与（ 并且关系 ）	\$x 和\$y 为真则返回真
\$x && \$y	同上	同上
\$x or \$y	逻辑或	\$x,\$y均为false时为假，其他情况全为真
<b>\$a    \$b</b>	同上	同上
!\$x	逻辑非	取反，即true变为false，false变为true
\$x xor \$y	逻辑异或	相同取false，相异为true

那我们来举几个例子来试试，你自己也要记得多做几次实验（可结合3.4.4这一章中的比较运算符自己写几个例子哟）。

逻辑与：

```
<?php
    $x = true;
    $y = false;
    //逻辑与（并且），要求两个都为true才执行真区间，所以代码中执行假区间
    if($x && $y){
        echo '执行了真区间';
    }else{
        echo '执行了假区间';
    }
    ?>
```

逻辑或：

```
<?php
```

### 3.4.5 逻辑运算

```
$foo = false;
$bar = true;
//逻辑或, 有一个为真则为真
if($foo || $bar){
    echo '执行真区间';
}else{
    echo '执行假区间';
}

?>
```

逻辑非：

```
<?php

$foo = false;

//逻辑非, 把false变为了true
if(!$foo){
    echo '执行真区间';
}else{
    echo '执行假区间';
}

?>
```

#### 【重点知识】短路

短路就是采用一个懒人模式来思考问题。

逻辑与的特性是：两边为true即为true，其他情况均为假。

逻辑或的特性是：两边为假均为假，其他情况全为真。

我们现在把自己想象成一个懒汉，非常非常懒。来思考逻辑与和逻辑或。可不可以这样理解：

逻辑与：如果前面第一个条件为false了，后面就可以不用执行了。

用代码表示：if(\$x && \$y) 如果\$x已经为false了，后面的\$y 就没有执行必要了。

逻辑或：前面第一个条件为true了，后面就不用执行了。

用代码表示：if(\$x || \$y) 如果\$x已经为true了，后面的\$y 就没有执行必要了。

我们写段代码来证明一下：

```
<?php

$x = false;
$y = 2;
if($x && $y++){
    echo '真';
}else{
    echo '假';
}
//结果还为2, 说明没有执行$y++
echo $y;

?>
```

代码如下，将两个&符改为一个&符试试：

```
<?php

$x = false;
$y = 2;
if($x & $y++){
    echo '真';
}else{
    echo '假';
}
//再看看结果
echo $y;
?>
```

我们来看看短路的逻辑或：

```
<?php

$x = true;
$y = 2;
if($x || $y++){
    echo '真';
}else{
    echo '假';
}
//结果，因为$x已经为true了，肯定执行真区间没有必要执行$y++了
echo $y;
?>
```

改成一个 | 再看看执行结果

```
<?php

$x = true;
$y = 2;
if($x | $y++){
    echo '真';
}else{
    echo '假';
}
//自己运行对比结果
echo $y;
?>
```

通过上例我们知道了&&与&的区别，||和|的区别。我们也了解了什么是短路。那我们在什么地方可以用到短路呢？有一些奇怪的写法，我们必须了清楚。其实就是基础语法的再应用。

回顾3.3.1最后一段内容：

```
<?php
//如果为defined('AUTH')存在AUTH常量则为true，不访问后面的exit了。如果为false则执行exit
defined('AUTH') or exit('存在安全因素不准访问');
```

上面的一段代码就是典型的短路应用的代码

exit 的意思是指在此处停止运行，退出。后面的PHP代码不再执行了。它有两种用法：

1,直接exit; 就是直接退出

本文档使用 [看云](#) 构建

2,exit('提示内容'), 退出的时候还给出一段提示内容

exit

读音：['eksɪt]

解释：退出

# 3.4.6 位运算

写了几年PHP的人都说，没有用过位运算符。所以，此处你看二进制看的头晕，就去T ¥ M ¥ D吧。

位运算符基本不用，我们也将这个知识设置为了解级别。位运算符的知识点，你不想学习也可以。**等以后用到位运算的时候，再来学习吧。**

知识学习级别【了解级，有印象即可】。

举例	说明	详细说明
\$a & \$b	And（按位与）	将把 \$a 和 \$b 中都为 1 的位设为 1。
<b>\$a   \$b</b>	Or（按位或）	将把 \$a 和 \$b 中任何一个为 1 的位设为 1。
\$a ^ \$b	Xor（按位异或）	将把 \$a 和 \$b 中一个为 1 另一个为 0 的位设为 1。
~ \$a	Not（按位取反）	将 \$a 中为 0 的位设为 1，反之亦然。
\$a << \$b	左移	将 \$a 中的位向左移动 \$b 次（每一次移动都表示“乘以 2”）。
\$a >> \$b	右移	将 \$a 中的位向右移动 \$b 次（每一次移动都表示“除以 2”）。

以上的符号，均为二进制的运算。

二进制大家绝大多数情况下不会遇到，遇到了补一下二进制这一块的知识就会了。

```
<?php
// $x二进制值为：
$x = 5;
// $y二进制值为：
$y = 8;
// 结果为13
echo $x ^ $y;
?>
```

变量	二进制值
\$x	0101
\$y	1000
异或结果	1101

异或解释：如果x,y两个值不相同，则异或结果为1。如果x,y两个值相同，异或结果为0。

由此可推1101为\$x和\$y异或出来的结果，1101用二进制转10进制工具转换后的结果就为13。

在线二进制转十进截图：

二进制：

1101

十进制：

13

转换

## 3.4.7 运算符优先级

---

本章的学习级别为了解级别。

因为绝大多数人都不会去记运算符的优先级。

我们在小学的时候，优先级是先乘除后加减。想要改变运算的优先级加括号即可。

**重点：不用记优先级，确定不了的时候号上括号把优先级标出来即可**

# 3.4.8 三元运算符和其它运算符

此外还有一些特殊的运算符和符号，我们先来进行讲解。可能以后我们需要用到。

符号	说明
\$x? 真代码段:假代码段	判断是否为真假？真情况：假情况;
`` ( 反引号 )	反引号中间插代命令，执行系统命令，等价于shell_exec函数
@	单行抑制错误，把这一行的错误不让它显示出来了，效率低不建议使用
=>	数组下标访问符
->	对象访问符
instanceof	判断某个对象是否来自某个类，如果是的返回true，如果不是返回false

三元运算符，相当于是if...else结构。不过三元运算符的写法更加简洁,语法格式如下：

```
$x? 真代码段 ( 只能写一句代码 ) :假代码段 ( 只能写一句代码 );
```

代码如下：

```
<?php
$x = true;

$x ? $y = 5 : $y = 6;
//输出5
echo $y;

?>
```

反引号就更加特殊了，我们经常需要显示IP地址，在PHP里面可不可能显示我们windows的IP地址呢？用反引号，就可以执行我们的命令（不过有些虚拟服务器是禁止执行这些的命令脚本的）：

```
<?php
echo '<pre>';

echo `ipconfig`;
echo '</pre>';

?>
```

上面的代码你执行看看效果，是不是显示出来了你机器的IP地址和一堆和IP相关的参数？

@符是指单行抑制错误，以后的章节我们还会讲解到。此处为了解级别。

```
<?php
//打开一个不存在的文件adfsafasdfasfasdfdsadf.txt，你运行一下会发现报错了。
//再前面再加上一个@符看看效果
$f = fopen('adfsafasdfasfasdfdsadf.txt','r');

//@$f = fopen('adfsafasdfasfasdfdsadf.txt','r');

?>
```



数组下标访问符，以后也会有一章讲解，此处了解一下就可以了：

```
<?php

$data = array('sina' =>'新浪' , 'sohu' => '搜狐');

?>
```

对象访问符->和instanceof 都是了解级别，以后也有专门的章节进行讲解：

```
<?php
//实例化一个对象
$obj = new stdClass();

//判断某对象是某由某个类实例化，如果是的则执行真
if($obj instanceof stdClass){
    echo '真';
}else{
    echo '假';
}
//向obj对象中追加一个成员属性为username
$obj -> username = 'PHP学院';

echo $obj -> username;

?>
```

上面的所有例子中，除了三元运算符、@符、反引号外。其他的学习级别均为了解，在以后讲解的时候。知道有这个符号即可。

## 04. PHP中的流程控制

流程控制就是人类社会的做事和思考和处理问题的方式和方法。通过本章，你将会发现采用计算机的思维去考虑问题，我们在做事的过程当中会更加严谨。

我们通过一个一个的场景来去推理流程：

有一个高富帅，他姓王。他的名字叫——王思聪（cong）。王思聪同学计划要投资一个项目。如果这个项目计划开始，为了这个投资项目每周往返一次北京和大连。什么时候王思聪同学不再往返呢？项目失败后或者万达（da）集团临时除知除外，他就可以不再这么每周往返了。

王思总同学呢，有一个好习惯，就是每次往返的时候，害怕自己到底一年往返了多少次。王同学都会在自己的记事本上记上往返的次数，第一次就写上一，第2次就写上2... ..直至最后项目停止。

王思总同学家里头特别有钱，所以他的行程方式和正常人的又有些不同。不仅有更多的方式，而且王同学还迷信。

他的出行方式呢有6种，如下：

- 1，司机开车
- 2，民航
- 3，自己家的专机
- 4，火车动车
- 5，骑马
- 6，游轮

每次王思总同学，都自己会在骰子上写上1，2，3，4，5，6。摇到哪种方式，王同学就会采用哪种方式进行往返两地。

并且呢，王同学是生活极度充满娱乐化和享受生活的人。他抵达北京或者大连的时候不同，他抵达后做的事情都不同，如下：

半夜到达，先去夜店参加假面舞会  
早上抵达，爱在酒店泡个澡  
中午到达，会吃上一份神户牛肉  
晚上到达，总爱去找朋友去述说一下心中的寂寞

王思总同学在出行和项目中也是极度有计划性。他给自己的生活秘书和工作秘书分别指派了出差的行程：

生活上：  
先查天气，下雨带雨具和毛巾。不下雨要带防晒霜  
雨具、毛巾和防晒霜的情况要提前检查，如果没有要及时买

工作上：  
要提前沟通去大连前的工作计划，准备好了要及时检查，检查合格，要提前打印现来。  
及时没有及时准备好的情况下，要列出主要的项目沟通议题。

本章中的知识点说明

【默写】

if...else  
while  
for  
switch...case  
break,continue,exit

【了解】

goto  
do...while

【跳过学习的知识点】

declare

---

注：

**为了提高流程控制的趣味性，上面的故事全属恶搞，我们用流程控制的想法和注释去实现上面的代码，加强对流程控制的理解**

## 4.1 if条件结构流程

if和else 语句，在之前的3.2.5章节中已经做了说明。我们配合王思聪同学的例子，再次进行说明，方便大家对此章节的理解。

本章的知识点为：【默写级】

基本语法，不能有半点马乎，完全是语法规则规定的，不这么写就错！

王思聪同学决定是投资项目的这个过程就可以这样理解：

```
<?php
/*
如果不投执
    向粑粑汇报，不再投资，停止后续代码执行。
*/

//定义是否投资的变量
$shifoutouzi = false;
if($shifoutouzi)
    exit('粑粑，我不投这个项目了');
?>
```

在之前我们也讲过，因此if的结构可以根据人类思维推理出来两种结构：

```
//if单行判断
if(布尔值判断)
    只写一句话;

后续代码

//if多行判断
if(布尔值判断){
    可以写多句话;
}
后续代码
```

## 4.1.1 if语句

我们为了加强大家对代码的理解，我们串了一个故事恶搞了一个王思聪同学。

在4.1和3.2.5这两个章节中我们都介绍到了if和if...else结构。并且我们讲解的很清楚。

我们现在来用if...else结构来写一个小东西，加强大家对逻辑的理解。

我们配合之前的知识点来写一个计算器：

```
<form>
  <input type="text" name="num1">

  <select name="fh">
    <option value="jia"> + </option>
    <option value="jian"> - </option>
    <option value="c"> x </option>
    <option value="chu"> / </option>
    <option value="qy"> % </option>

  </select>

  <input type="text" name="num2">

  <input type="submit" value="运算" />

</form>

<?php

$num1 = $_GET['num1'];
$num2 = $_GET['num2'];
$fh = $_GET['fh'];

if(!is_numeric($num1) || !is_numeric($num2)){

    echo '请输入数值类型';
}

if($fh == 'jia'){
    echo $num1 . '+' . $num2 . '=' . ($num1+$num2);
}

if($fh=='jian'){
    echo $num1 . '-' . $num2 . '=' . ($num1-$num2);
}

if($fh=='c'){
    echo $num1 . 'x' . $num2 . '=' . ($num1*$num2);
}
if($fh=='chu'){
    echo $num1 . '/' . $num2 . '=' . ($num1/$num2);
}
if($fh=='qy'){
    echo $num1 . '%' . $num2 . '=' . ($num1%$num2);
}

?>
```

作业：

写一个平年、闰年计算器。写一个form表单，通过get将年份给传过来，判断传进来的年份是否为数值类型。并且要求，如本文档使用 [看云](#) 构建

如果是闰年就提示这是闰年，如果是平年就提示这一年是平年。

平年闰年规则：年份能被4整除，但是不能被100整除。或者能被400整除，即为闰年，其他情况全为平年

## 4.1.2 嵌套if...else...elseif结构

还记得本章开篇我们讲了一个王思聪同学的例子：

王同学是生活极度充满娱乐化和享受生活的人。他抵达北京或者大连的时候PHP学院，他抵达后做的事情都PHP学院，如下：

半夜到达，先去夜店参加假面舞会

早上抵达，爱在酒店泡个澡

中午到达，会吃上一份神户牛肉

晚上到达，总爱去找朋友去述说一下心中的寂寞

我们来了解一下他的语法规则【知识点要求：默写】

```
<?php
if (判断语句1) {
    执行语句体1
}elseif(判断语句2){
    执行语句体2
}else if(判断语句n){
    执行语句体n
}else{
    最后的else语句可选
}

//后续代码
?>
```

上述结构表示：

如果判断语句1的值为真，则执行语句体 1。执行完成后进入后续代码段。

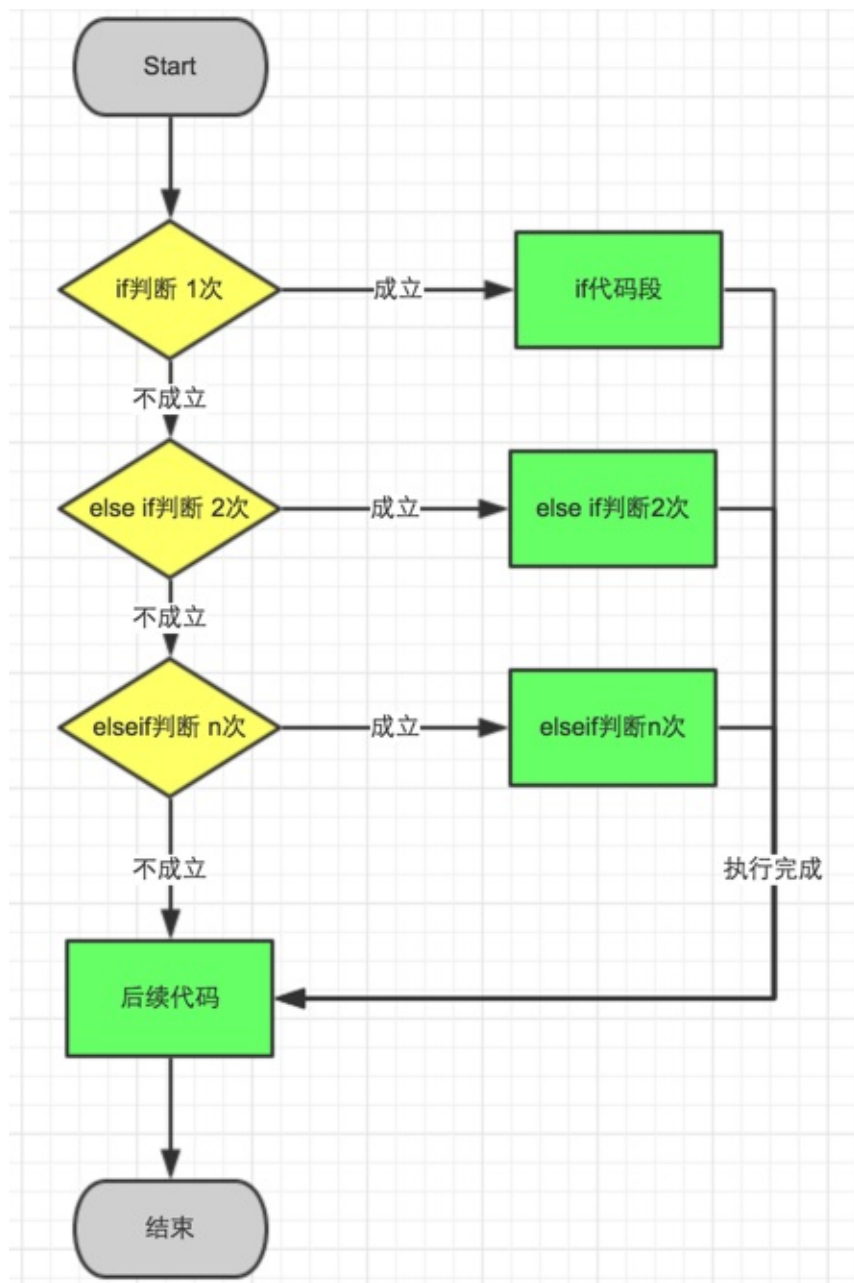
否则转入后面的判断语句2（elseif），判断语句2若为真，则执行语句体 2。

否则转入后面的判断语句n（elseif），判断语句2若为真，则执行语句体 n。

如果均不匹配则执行else语句。这种循环嵌套可以不含else语句，即只含有if、elseif语句。

注：elseif()也可以写成else if()

我们把上面的代码用流程图的形式表示清楚就如下图了：



我们可以把王思聪同学的例子通过PHP代码表示出来，代码表示结果如下：

```
<?php
//定义一个随机变量，抵达时间，随机0点至23点
$dida = rand(0,23);

if($dida > 6 && $dida < 10){
    echo '我爱洗澡';
}else if($dida >10 && $dida < 14){
    echo '吃神户牛肉';
}else if($dida > =19 && $dida < 22){
    echo '找一个朋友聊聊内心的寂寞';
}elseif($dida > 22 && $dida <= 23){
    echo '洗澡';

}elseif($dida >= 1 && $dida <3){
    echo '洗澡';
}else{
    echo '睡觉或者工作';
}
```



```
?>
```

作业：

写一个网页cj.html，向panduan.php页面以post方式提交分数段。分数段不同，显示的结果也不同，要求如下：

1. 0---60以下，不及格
2. 60---70及格了，要努力
3. 70---80 还不错
4. 80---90 上清华有希望
5. 90---100 你这辈子没希望了
6. 100 更没希望了
7. 100分以上 爱因斯坦转世啊，思密达！
8. 不是一个数值类型或者小于0 请输入正确的分数，不然就放李文凯啦

## 4.1.3 if语句多种嵌套

王思聪同学我们在最开始的故事中讲到了他有两个秘书：一个生活秘书、一个工作秘书。

王思聪同学在出行和项目中也是极度有计划性。他给自己的生活秘书和工作秘书分别指派了出差的行程：

生活上：

先查天气，下雨带雨具和毛巾。不下雨要带防晒霜

雨具、毛巾和防晒霜的情况要提前检查，如果没有要及时买

工作上：

要提前沟通去大连前的工作计划，准备好了要及时检查，检查合格，要提前打印现来。

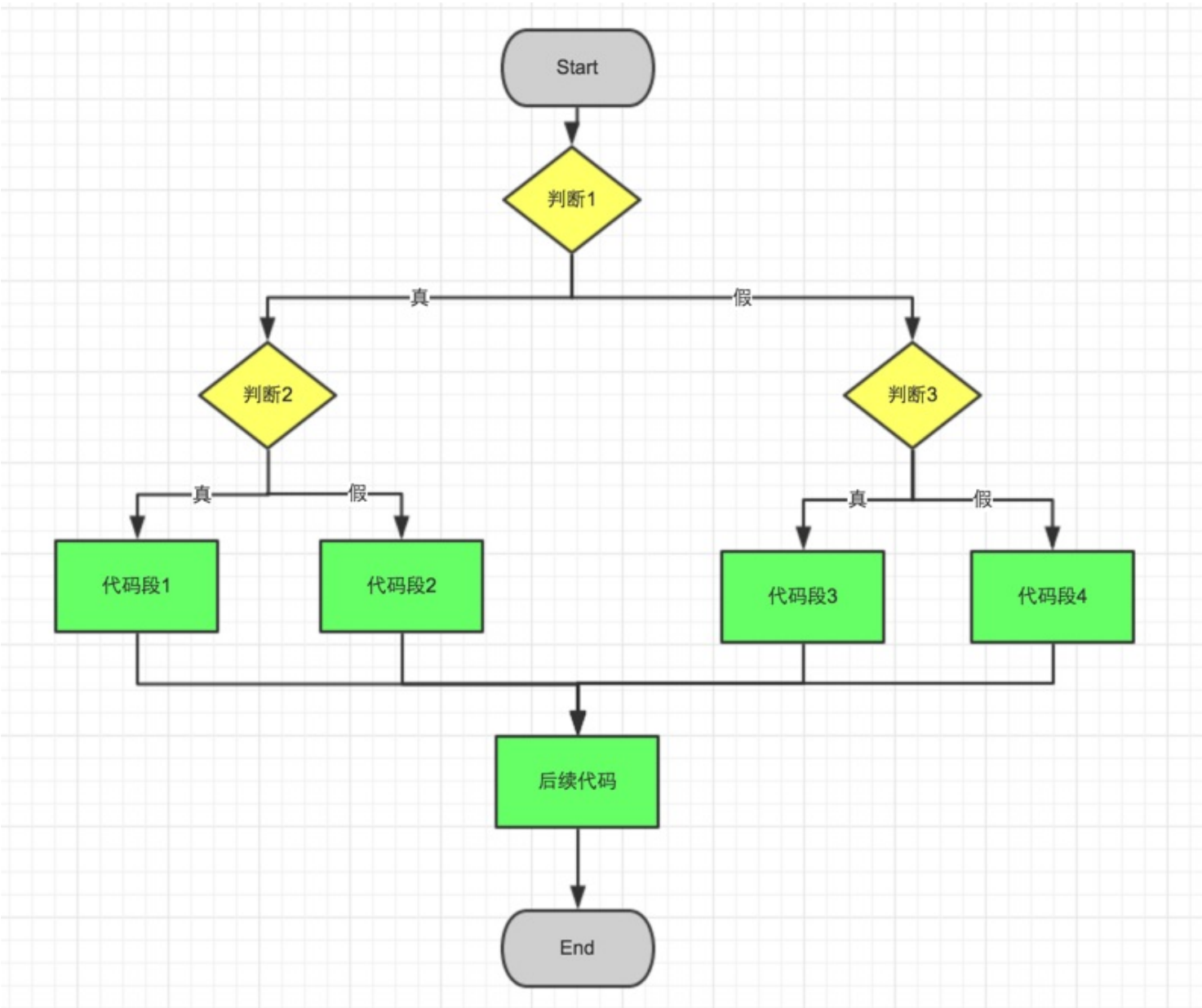
及时没有及时准备好的情况下，要列出主要的项目沟通议题。

类似于上面的这种情况判断，我们就需要用到if...elseif...else反复嵌套的结构了。

在if语句中可以嵌套一个或多个if语句，以实现多个参数的判断，这就是if语句的多种嵌套。其结构形式如下：

```
<?php
if(判断1){
    if(判断2){
        代码段 1
    }else{
        代码段2
    }
}else{
    if(判断3){
        代码段3
    }else{
        代码段4
    }
}
?>
```

我们用流程图来表示如下：



- 注意：
- 1. 我们在代码段1，2，3，4中可以再加入判断。根据实际情况还可以再加入嵌套
  - 2. 注意缩进，缩进的作用只是使代码富有层次感，美观易读，对目标代码的生成毫无影响。

我们把王思聪同学的生活要求，我们可以用代码用嵌套结构表示出来。我们使用到了三层嵌套结构，代码如下：

```
<?php
//0表示工作秘书，1表示生活秘书
//用代码模拟随机产生当前的工作是生活秘书的还是工作秘书的
$mishu = rand(0,1);

if($mishu){
    //下雨和不下雨的状态，随机产生
    //下雨状态为1
    //不下雨状态为0
    $xiyu = rand(0,1);
    if($xiyu){
        //是否购买雨伞
        $you = rand(0,1);
        if($you){
            echo '下雨天，已购买不用买雨伞';
        }
    }
}
```

### 4.1.3 if语句多种嵌套

```
        }else{
            echo '下雨天，未已购买，需要买雨伞';
        }
    }else{
        //是否购买防晒霜
        $you = rand(0,1);
        if($you){
            echo '没下雨，有防晒霜';
        }else{
            echo '没下雨，需要准备防晒霜';
        }
    }
}

}else{
    //是否准备好了会议议程
    $shifou = rand(0,1);

    if($shifou){
        echo '已准备好，可以随时出发';
    }else{
        echo '没有准备好，需要打印，延迟出发';
    }
}

}
```

警告：对于编程新手，在使用这种嵌套if...else循环时，请小心使用。因为太多层的循环容易使设计的逻辑出问题，或者少打了大括号等，都会导致程序出现莫名其妙的问题。

希望你能够默写出来。并且，不能有一丁点的语法错误。在以后，我们随时用，要在大脑当中立马产生反映，动手就写。

## 4.2 分支结构switch语句的使用

还记得我们最开始讲了这么一个故事：

王思聪同学家里头特别有钱，所以他的行程方式和正常人的又有些不同。不仅有更多的方式，而且王同学还迷信。他的出行方式呢有6种，如下：

- 1，司机开车
- 2，民航
- 3，自己家的专机
- 4，火车动车
- 5，骑马
- 6，游轮

他的方式有6种，而骰子也真好有6面。所以，我们用if...elseif...的判断方式可以实现，但是效率太低了。

还有其他更好的方式吗？我们可以使用到一种方式就是:switch...case语法。

switch...case的语法结构如下：

```
<?php

switch(变量){ //字符串，整型

    case 具体值:
        执行代码;
        break;

    case 具体值2:

        执行代码2;
        break;

    case 具体值3:

        执行代码3;
        break;

    default:

}
?>
```

switch后放需要判断的变量，而case后是放结果。switch后变值为多少，case的值写的与switch变量执相同段的代码。

上面的break是可选的

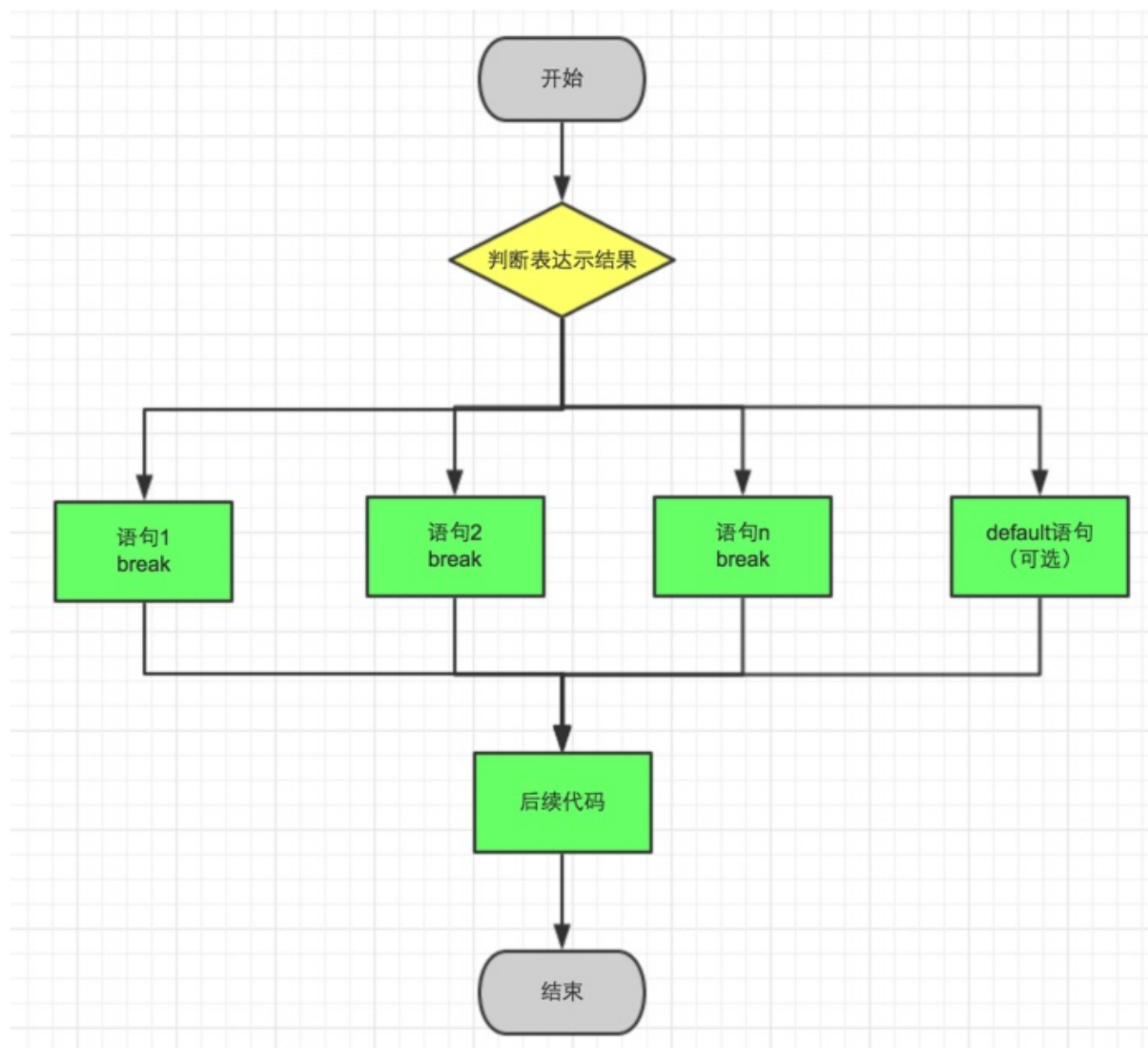
上面的default也是可选的

不要在case 后面写成分号，后面是冒号：

不要在case后面写判断区间，如（ \$foo > 20 或者 \$foo == 30 ）

switch里面在变量最好是 整型，字符串，因为布尔判断更适合if...else..

我们用流程图来表示的话结果就如下图所示：



上节课中我们用到了rand函数，那我们现在来用rand实现王思总同学的问题选择：

```
<?php
//定义出行工具
$tool=rand(1,6);

switch($tool){

    case 1:
        echo '司机开车';
        break;
    case 2:
        echo '民航';
        break;
    case 3:
        echo '自己家的专机';
        break;
    case 4:
        echo '火车动车';
        break;
    case 5:
        echo '骑马';
```

## 4.2 分支结构switch语句的使用

```
        break;
    case 6:
        echo '游轮';
        break;

}

?>
```

上面的代码我们只需要简单修改，就可以实现我们在微信上面玩的一个小游戏简单的家务骰子和剪刀石头布。你想想？

你再做做实验：

我们可以把 case 1 代码段中的break去掉，你再试试，什么效果？

我们再来写一个简单的星期判断，写法也可以变的怪一些哟：

```
<?php
//得到今天是星期几的英文简称
$day = date('D');

switch($day){
    //拿学校举例，我们让星期一、二、三是校长日
    case 'Mon':
    case 'Tue':
    case 'Wed':
        echo '校长日';
        break;

        echo '星期三';
        break;
    case 'Thu':
        echo '星期四';
        break;
    case 'Fri':
        echo '星期五';
        break;
    default:
        echo '周末，周末过的比周一到周五还要累<br />';
}
```

自己做实验试试：

上面的例子发现，不匹配的时候就执行了default了吧？

当然你也可以这样，不过这种写太太累了，没有if...else更加简单。我们不用swith...case来做true和false判断

```
<?php
//用switch...case来完成bool判断
$bool=true;

switch($bool){
    case true:

    case false:

}

/*****分隔线*****/
if($bool){

}else{
```

```
}
```

---

世界上最痴心的等待就是我是case你是switch，我一只默默的等待，可是你却不选我！



## 4.3 循环语句的使用

王思聪同学需要反复往返于北京和大连，就是典型的循环结构。假设王思聪投资这个项目需要往返大连100次，每次往返都王同学都会计数一次。难道我们写一百遍同样的代码？显然对于智商极高的程序员来说不可能这样处理。

我们抽象了人类的这种思维。我们定义一种循环结构

```
<?php

//定义需要往返的次数，老外喜欢从0开始计数，我们也从0开始计
$count = 0;

//while后面接布尔值判断，为真执行，为假停止
// $count 小于100的时候执行，也就是$count为0至99的时候执行
//如果$count不小于100了，循环停止执行后续的代码

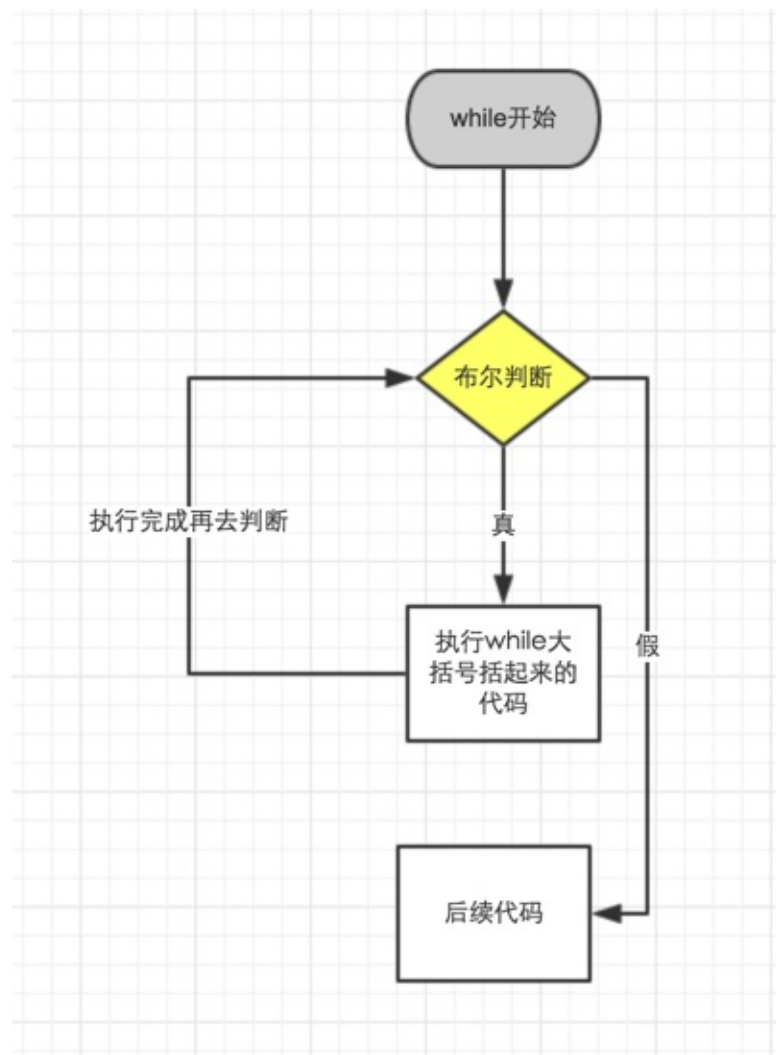
//循环开始处
while($count < 100){

    echo '我是王思总，我是第' . $count . '次出差<br />';
    //每次执行让$count+1，这样的话，就不会产生$count永远小于100的情况了
    $count++;

//循环结束
}

echo '后续代码';
?>
```

我们可以为while循环补一段专门的代码逻辑图：



## 4.3.1 while循环

在循环的开章，我们讲到了循环的逻辑和语法，通过以上的训练。你能轻易的掌握循环的知识点。

while是一个布尔型循环，while(布尔判断)的值为true就执行大括号括起来的代码。如果为假就停出去，执行后续代码。

现在，我们增加一点难度，来做隔行变色。隔行变色，是我们在网页中经常使用到的表现。

要求重点掌握。我们来看看效果：

0	1	2	3	4	5	6	7	8	9
10	11	12	13	14	15	16	17	18	19
20	21	22	23	24	25	26	27	28	29
30	31	32	33	34	35	36	37	38	39
40	41	42	43	44	45	46	47	48	49
50	51	52	53	54	55	56	57	58	59
60	61	62	63	64	65	66	67	68	69
70	71	72	73	74	75	76	77	78	79
80	81	82	83	84	85	86	87	88	89
90	91	92	93	94	95	96	97	98	99

那我们现在要写一个0-99的隔行变色的表格该怎么写呢？

【要求默写】

1. 定义初始值，输出表格标签 和 表格中的列标签

```
<?php
//定义循环的初始值
$i=0;
echo '<table width="800" border="1">';
```

```
while($i<100){
//输出列0-99的列了
echo '

'.$i.'";
//一定要加哟，不然死循环了
$i++;
}
echo "
";
?>
```

2. 加上行产生的逻辑

```
<?php
$i=0;
```

### 4.3.1 while循环

```
echo '  
  
';  
while($i<100){  
    //0 - 9 为一行  
    //10 -19 为一行  
    //因此，每一行都能够被10求默，如为为10的时候，应该显示行开始的标签  
    if($i%10 == 0){  
        //为了隔行变色，每20，40，60每行的颜色是PHP学院的，因此我们又可以再进行一次取余运算  
        if($i%20==0){  
            echo '  
  
';  
        }  
    }  
}
```

```
    echo '<td>'.$i.'</td>';  
  
    $i++;  
    //同理，每一行结束是不是应该有一个</tr>结束标签呢？  
    if($i%10==0){  
        echo '</tr>';  
    }  
}
```

```
echo '  
  
'
```

```
';  
?>
```

以上代码需要多练习，不断的实验才能想象出来。上面的部份，需要默写来锻炼逻辑。

可不可以循环里面再嵌套一次循环（双层循环），来实现隔行变色的表格效果呢？【要求默写】

基本实现逻辑如下

1. 先输出表格标签
2. 通过第一层循环再输出<tr></tr>行标签
3. 在第一层循环里面再插入一层循环输出<td></td>标签

```
<?php  
echo '  
  
'  
  
';  
$i=0;  
while($i<10){  
    echo '  
  
'
```

```
';  
  
    $j=0;
```

### 4.3.1 while循环

```
while($j<10){  
  
    echo '<td>'.$j.'</td>';  
    $j++;  
  
}  
echo '</tr>';  
  
$i++;
```

```
}  
echo '
```

```
';  
;  
?>
```

上面的代码，你自己加上隔行变色的判断和正常的0-99的数值显示哟？

\* \* \* \* \*

你可以学完循环后，再学习一些基本的算法。以后PHP学院的视频中我们会多多的进行讲解。  
这样帮助你面试的成功率更高。

注：不要写死循环（没有退出条件的循环）

```
while(1){  
echo 1111.'  
;  
;  
}
```

## 4.3.2 do...while循环的区别

---

do...while与while的语法结构基本一样，也是一个布尔型循环，功能也基本一样。

基本语法规则如下：

```
do {  
    //代码块  
} while (判断);
```

do...while与while的区别是，它们的值的检查时机不同。

do-while 不论while判断是否成立，先执行一次代码代码块循环语句，保证会执行一次（表达式的真值在每次循环结束后检查）。

然而我们之前的while循环会检查布尔判断区域，成立则执行。不成立则不执行。

我们用代码去验证一下：

```
<?php  
$i = 0;  
do {  
    echo $i;  
} while ($i > 0);  
?>
```

上面代码中，\$i 肯定不大于0，也执行了。

---

当然，你还不能理解，实在想不到应用场景也没关系，可以完全跳过此块。

do...while用的情况比较少。我们有可能用在资源处理如：文件打开等地方。

作业：用do...while单层和双层写一个隔行变色0-99的表格



```
}

```

我们来试一下break,exit和continue来控制一下9\*9乘法口诀表。

语句	作用
exit	exit之前我们讲过了，从当前处停止后续执行
break	之前遇到过，跳出循环或者跳出结构体执行后续代码
continue	跳出此次循环，下次循环继续

我们来演示一下break和continue：

```
<?php
for ($i = 1; $i <= 10; $i++) {

    if($i == 4){
        //待会儿换成contiune试试
        break;
    }

    echo '分手后第'.$i.'年，我全都忘了你的样子<br />';
}
?>
```

\$i 等于4，break效果如下：

分手后第1年，我全都忘了你的样子  
分手后第2年，我全都忘了你的样子  
分手后第3年，我全都忘了你的样子

注：上图中第4之后不再执行

\$i 等于4，continue效果如下：

分手后第1年，我全都忘了你的样子  
分手后第2年，我全都忘了你的样子  
分手后第3年，我全都忘了你的样子  
分手后第5年，我全都忘了你的样子  
分手后第6年，我全都忘了你的样子  
分手后第7年，我全都忘了你的样子  
分手后第8年，我全都忘了你的样子  
分手后第9年，我全都忘了你的样子  
分手后第10年，我全都忘了你的样子

注：上图中第4第丢失了，接着从第5年继续执行了

作业：

- 用for 的单层循环来控制隔行变色的表格
- 用for的双层循环来控制隔行变色的表格
- 默写99乘法口诀表，并在中间\$i、\$j的位置实验continue和break;



## 4.3.4 goto语法

自 PHP 5.3.0 起，还可以使用 goto 来跳出循环。

在本章开始的章节，我们讲解到一个故事，王同学每周往返，但有一个特例：

**项目失败后或者万达（da）集团临时出事除外，他就可以不再这么每周往返了。**

### 基本语法

```
<?php
goto wan;
echo '天王盖地虎';

wan:
echo '小鸡炖蘑菇';
?>
```

通过上例，我们发现直接显示输出了：小鸡炖蘑菇。

我们来实现一下霸道王总裁的代码：

```
<?php
for($i=0; $i<100; $i++) {
    echo '第'. $i .'周往返北京大连<br />';
    if($i == 17){
        goto end;
    }
}

end:
echo '集团公司要求停止此项';
?>
```

这个知识点为了解级别，大家不想学习，可以不用学习此块。

注：

goto 操作符可以用来跳转到程序中的另一位置。

该目标位置可以用目标名称加上冒号来标记，而跳转指令是 goto 之后接上目标位置的标记。

PHP 中的 goto 有一定限制，目标位置只能位于同一个文件和作用域，也就是说无法跳出一个函数或类方法，也无法跳入到另一个函数。也无法跳入到任何循环或者 switch 结构中。可以跳出循环或者 switch，通常的用法是用 goto 代替多层的 break。

## 4.3.5 declare 语法

---

在基础阶段，大家应该跳过此处学习。

涉及到了函数相关的知识，若你有一定的基础，可以继续下面的学习。

## 05.PHP的函数基本语法

---

函数，在之前的课程当中我们做过介绍。函数的英文叫作：function，而function的解释项中有另外一个含义：功能。

函数 就是 功能。调用一个函数就是在调用一个功能。

我们之前也了解过一些函数了,例如：is\_int、phpinfo()等等。我们都介绍过，他们都具有不同的功能。我们不再这里复述。

**PHP学院的凯哥（李文凯）经常说过一句话：函数是条狗，哪里需要哪里吼。**

通过上句话，我们能得到几个不同的规律：

1. 需要用一个函数的时候，就吼一声它的名字
2. 函数可以反复去吼，吼了它的名字它就过来了。也就是函数可以反复调用

我们明白后，那我们接下来来讲解函数的分类，我们将函数分为两大类别：

1. 自定义函数
2. 系统函数

## 5.1 自定义函数

PHP提供了功能强大的函数，但这远远满足不了需要，程序员可以根据需要自己创建函数。本节就开始学习创建函数的方法。

我们在实际开发过程当中需要有很多功能都需要反复使用到，而这些反复需要使用到的功能，我们能定义成功能（函数），就尽可能定义成功能（函数）。使用的时候，吼一下它的名字即可。

那我们来学一下自定义函数的语法规则，语法规则如下：

```
function 函数名([参数名1[=值1], 参数名2[=值2], 参数名n[=值n]])
{
    函数中的功能体
    [return 返回值]
}
```

上面的语法规则中发现了如下特点，产生如下语法规则：

1. 函数以function开始
2. function后面接空格，空格后接函数名
3. 函数名与变量命名规则基本一样，但是不同的是：函数名不区分大小写
4. 所谓参数其实就是变量
5. 函数名后接括号，括号内跟参数，参数全都有[]（中括号）括起来了，代表参数可填可不填
6. 如果有参数的话，参数后可以接（=）等号，等号接默认值。参数值也是用[]（中括号）括起来的，代表选填
7. 函数后的参数变量，主要功能是把函数体外的变量值，传入函数体内来使用，函数体的变量和函数体外的变量通常是两个不同的变量。
8. 函数中的具体功能（功能体）用大括号括起来，代表这是一个函数的功能区间
9. 函数可以有返回值也可以没有返回值，用[]（中括号）括起来的，代表选填。
10. return后接空格，空格后接返回值，若有return,return后的代码均不执行。
11. 函数的执行没有顺序关系，可以在定义处之前的位置调用
12. 函数不能被定义两次，即函数不能被重载

**切记：你也要多写代码来进行实验！**

我们可以通过实验来一点一点证明这些观点。

### 1. 函数是条狗，哪里调用哪里走

```
<?php

function liwenkai(){

    echo '我是一条狗';
}

liwenkai();
liwenkai();
liwenkai();
?>
```

liwenkai这个函数，是不是显示出来了三段：我是一条狗？

### 2. 函数名与变量命名规则一样，但是不同的是：函数名不区分大小写

```
<?php

function 1demo(){

}

?>
```

以上代码会报错。函数命名与变量命名相同。

### 3. 函数名与变量命名规则一样，但是不同的是：函数名不区分大小写

```
<?php

function Demo(){

    echo '如果是写代码的男人，我们需要更加承担来自家庭的责任<br />';
}

demo();
Demo();
DEMO();

?>
```

通过上面的代码会发现，会显示出来三行：如果是写代码的男我人我们需要更加承担来自家庭的责任

。也就是函数名在调用的时候不区分大小写。函数可以反复叫它的名字，可以反复执行。也体现了重用的特点。

### 4. 函数体的参数若是定义了，未传参数，代码会报错

```
<?php

//定义函数名为test，必须要传入一个参数
function test($hello){

}

test();

?>
```

你也自己写一段出来试试，会不会报错？

### 5. 函数后的参数如果有默认值，参数可以不填，代码也不会报错

```
<?php
function test( $arg = 10){

    echo $arg;

}

test();

test(88);

?>
```

是不是没有报错呀？并且显示了 10和88出来了。

## 5.1 自定义函数

说明，参数如果传了，参数会带入函数中。如果函数没有传参数，会用参数后的默认值。

## 6. 函数后的参数可以写多个

```
<?php
function test( $a , $b = 20 , $c = 30){

    echo $a + $b + $c;

}

test( 1 , 2 , 3 );
?>
```

自己写写上面的代码，我们是不是传入了多个参数？

## 7. 函数后如果有默认值和无默认值的参数，通常把无默认值的参数写在最前面

之前的第3和第4条实验我们发现：默认值是代表这个参数可以不用传值进去。而没有默认值的话，代码会报错，也就是无默认值的参数是必传的。我们来看看下面的例子：

```
<?php
function test( $a = 20 , $b = 30 , $c ){

    echo $a + $b + $c;

}

//重点：重点看这一行执行一下
test( , , 8 );
?>
```

通过上例发现执行上面的代码又报错了。也就是上面的语法是不行的。

上面的语法我们希望的是，参数\$a和参数\$b我们不传入任何值。参数\$c是必须要传进去的，我们传了8。可是PHP的语法规则中不准许我们这么写。因此，我们换一种写法，也能达到一样的效果：

```
<?php
function test( $c , $a = 20 , $b = 30){

    echo $a + $b + $c;

}

//重点：重点看这一行执行一下
test( 8 );
?>
```

通过上例实验大家是不是发现，参数\$c我传了，代码还没有报错。而参数\$a = 20 ,参数 \$b = 30 一样带入了代码 \$a + \$b + \$c 了。

## 8. 函数体的变量与函数体外的变量没有关系

```
<?php
//定义变量$hello的值为10
$hello = 10;

//函数后的参数（形式上的参数，参数）处写上变量名为$hello
function demo( $hello ){

    //形参带入了函数体内，函数体内变量$hello 改为了 250
```

## 5.1 自定义函数

```
$hello = 250;

//输入2个250相加的结果
echo $hello + $hello;

}

//将函数体外的变量$hello，传入变量的参数处（实际传入的参数，实参），显示出的结果为500
demo($hello);

//此处$hello的值输出，依然为10
echo $hello;
?>
```

通过上例发现：形参、与实参没有半点关系。而函数体内的传进去的形参，再怎么变化，都不会影响实参\$hello的实际结果。

## 9. 函数体中若有return,return 后的代码不执行

```
<?php

function demo(){

    echo 111;

    return;

    echo 222;

}

demo();
?>
```

发现没？——只输出了111。

## 10. 函数执行完后,return 可把函数体内的值，带带函数体外

```
<?php

//定义一条函数狗
function liwenkai(){

    $foo = 5;

    $bar = 6;

    $result = $foo + $bar;
    //将$result的结果进行返回
    return $result;

}

//调用liwenkai()这个函数，$foo和$bar相加的$result就会返回回来给到变量$piao
$piao = liwenkai();

//输出$piao的结果，果真为11
echo $piao;

?>
```

## 11. 函数的执行没有顺序关系，可以在定义处之前的位置调用

```
<?php

demo();
```

## 5.1 自定义函数

```
function demo(){  
  
    $str = '爸爸妈妈年龄越来越大';  
    $str .= '大多数的孩子都是独生子女，更加应该负起责任';  
  
    echo $str;  
}  
  
demo();  
?>
```

通过上例发现，函数在定义之前还是在定义之后都可以调用。也就是函数可以任意位置调用。

## 11. 函数不能重载

```
<?php  
  
function demo(){  
  
}  
  
function demo(){  
  
}  
//试试会报错吗？
```

通过上例发现，报错了，也就是同名函数不能被定义两次，否则报错

---

若您有民族信仰，有伤害您民族信仰的地方，在这里。我向您道歉！上面的一个例子只是为了让大家理解函数。

作业：

1. 定义一个双层循环，隔行变色的表格
2. 要求这个表格，默认值为10和10，颜色中有一个为默认参数，一个没有默认参数
3. 将表格的字符串return 回去



## 5.2 自定义函数高级调用

---

学完上一章，我们已经知道了函数的规则，我们也学会了如何来定义函数，本章将介绍更多的函数的高级用法。

不过，这些高级用法中，很多用法很奇葩。

## 5.2.1 回调函数

回调函数，可以配合匿名函数和变量函数实现更加优美、复杂的一种函数结构。

回调函数，就是在处理一个功能的时候，我让让这个功能自定义能力再强一些，我准许调用这个函数的时候，还可以传入一个函数配合、协助进行处理。

这是变量函数和回调函数结合的一个章节。

```
<?php

function woziji($one,$two,$func){
    //我规定：检查$func是否是函数，如果不是函数停止执行本段代码，返回false
    if(!is_callable($func)){
        return false;
    }

    //我把$one、$two相加，再把$one和$two传入$func这个函数中处理一次
    // $func是一个变量函数，参见变量函数这一章
    echo $one + $two + $func($one,$two);
}

//我们定义几个函数试试
function plusx2( $foo , $bar){

    $result = ($foo+$bar)*2;

    return $result;
}

function jian( $x , $y ){
    $result = $x - $y;

    return $result;
}

//调用一下函数，woziji, 向里面传入参数试试

echo woziji(20,10,'plusx2');

//将plusx2改成jian试试结果
echo woziji(20,10,'jian');

?>
```

处理过程是这样的:

1. 将20赋值给形参\$one,10赋值给了\$two，而plusx2或者jian这两个变量函数，赋值给了\$func
2. 在woziji这个函数中判断plusx2或者jian是否为函数，不是函数就return false 停止执行了
3. 显示plusx2或者jian是函数。因此\$one = 20, \$two =10相加了，相加后，\$one和\$two又带入了\$func(\$one,\$two)中。
4. 带入至里面后而\$func，是可变的，可以为plusx2或者jian。如果为plusx2的话，\$one = 20,\$two = 10 的这个两个结果又给了plusx2函数里面的\$foo和\$bar
5. \$foo + \$bar 乘以2后将结果返回至woziji这个函数功能体的运算处：\$one + \$two + \$func(\$one,\$two);
6. 这样就得到了运算结果

### 5.2.1 回调函数

现在我们明白了回调函数：在一个函数里面，再传入一个函数名，将函数名加上()括号。识别为变量函数，配合执行。

## 5.2.2 变量函数

---

在之前的变量部份，我们学习了可变变量。可变函数仅仅是可变变量的一个变种、变形表达。

可变函数，我们也会称呼为变量函数。简单回顾一下之前的知识点：

```
<?php

$hello = 'world';
$world = '你好';

//输出的结果为：你好
echo $$hello;

?>
```

因为\$hello先被解释成了world，再world前加上\$符就输出了：你好。

而变量函数的用法是这样的：

```
<?php

function demo(){
    echo '天王盖地虎';
}

function test(){
    echo '小鸡炖蘑菇';
}

$fu = 'demo';

//把$fu变为了demo,把demo后加上了一个括号，就执行函数了
$fu();

//把$fu的值改为test字符串再试试？

?>
```

建议在学基础语法阶段，大家不要去想最终的处理场景，因为最终的处理场景跟大家说了大家没有语法基础，也不会明白。例如：可变函数，也叫变量函数。可以用于以后的MVC，面向对象的设计模式等处。所以，眼前不要去深究。

## 5.2.3 匿名函数

所谓匿名，就是没有名字。

匿名函数，也就是没有函数名的函数。

匿名函数的第一种用法，直接把函数赋值给变量，调用变量即为调用函数。

匿名函数的写法比较灵活。

### 1.变量函数式的匿名函数

```
$greet = function($name)
{
    echo $name.'，你好';
};

$greet('明天');
$greet('PHP学院');
```

上例中的函数体没有函数名，通过\$greet加上括号来调用的，这就是匿名函数。

### 2.回调式的匿名函数

我们将之前的例子拿过来。实际使用场景中，我们要通过一个函数实现更多的功能。但是，我又不想专门定义一个函数。我们回顾一下，我们回调函数的例子：

```
function woziji($one,$two,$func){
    //我规定：检查$func是否是函数，如果不是函数停止执行本段代码，返回false
    if(!is_callable($func)){
        return false;
    }

    //我把$one、$two相加，再把$one和$two传入$func这个函数中处理一次
    // $func是一个变量函数，参见变量函数这一章
    echo $one + $two + $func($one,$two);
}

woziji(20,30,function( $foo , $bar){

    $result = ($foo+$bar)*2;

    return $result;

});
```

仔细推理一下过程哟。只不过在之前的章节当中，plusx2换成了我们的匿名函数：

```
<?php

function( $foo , $bar){

    $result = ($foo+$bar)*2;

    return $result;
```

### 5.2.3 匿名函数

```
}  
?>
```

因此，匿名函数在调用的时候没有函数名。我们可以采用以上的一些方法来使用匿名函数。

## 5.2.4 内部函数

内部函数，是指在函数内部又声明了一个函数。

注意事项：

1. 内部函数名，不能是已存在的函数名
2. 假设在函数a里面定义了一个内部函数，不能定用两次函数a。

我们下面来看代码，你将很快的学习会：

```
<?php
function foo()
{
    echo '我是函数foo哟，调一下我才会执行定义函数bar的过程<br />';
    function bar()
    {
        echo '在foo函数内部有个函数叫bar函数<br />';
    }
}

//现在还不能调用bar()函数，因为它还不存在
bar();

foo();

//现在可以调用bar()函数了，因为foo()函数的执行使得bar()函数变为已定义的函数

bar();

//再调一次foo()看看是不是会报错？
foo();

?>
```

你会发现，在上面foo() 函数内部又定义了一个bar函数，这就是内函数数。

仔细观察和实验后你会得出如下的结论：

1. foo()调用两次会报错
2. 如果不调foo()函数无法执行bar函数，因为bar是在foo的内部

# 5.2.5 变量作用域

我们通过5.1章节的函数定义部份的学习我们知道了几个不同的规矩：

- 1. 函数定义时后括号里面接的变量是形式上的参数（形参），与函数体外的变量没有任何关系。仅仅是在函数内部执行
- 2. 函数内声明的变量也与函数外的变量没关系。

但是，我们实际的处理情况中会遇到这样的一个情况：

- 1. 我想在函数体内定义的变量在函数体外用
- 2. 我想把函数体外的变量拿到函数体内来使用

这个时候我们就需要用到超全局变量。我们来回顾一下之前的知识点：	全局变量名	功能说明
\$_COOKIE	得到会话控制中cookie传值	
\$_SESSION	得到会话控制中session的值	
\$_FILES	得到文件上传的结果	
\$_GET	得到get传值的结果	
\$_POST	得到post传值的结果	
\$_REQUEST	即能得到get的传值结果，也能得到Post传值的结果	

我们来通过实验来观察 一下外部变量（超全局变量）的特点，打破本章开头我们总结的规律：

我们定义一下global.html页现来写HTML内容：

```
<html>
  <head>
    <title>超全局数组实验</title>
  </head>
  <body>
    <!--先用POST来实验，以后你可以改成GET哟 -->
    <form action="glob.php" method="post">
      <input type="text" name="hongniu" /><br />

      <input type="submit" value="提交" />
    </form>

  </body>
</html>
```

我们将html的内容通过form表单提交给了glob.php页面，我们现在来写glob.php：

```
<?php

function demo(){
echo $_POST['hongniu'];
}
```



### 5.2.5 变量作用域

```
demo();  
?>
```

通过这个小例子，你会发现超全局的\$\_POST等这一系列的超全局变量（外部变量）在函数内部也是可以用的。没有本文开始处变量作用域的限制。

其实我们所有声明的变量都放到了\$GLOBALS这个数组下面，举个例子：

```
<?php  
  
$hello = 10;  
  
echo $GLOBALS['hello'].'<br />';  
  
$GLOBALS['hello'] = '我爱你';  
  
echo $hello;  
  
?>
```

通过上例，你会发现\$变量名是等价于\$GLOBALS['变量名']。所有的变量都放到了\$GLOBALS里面了。而\$GLOBALS也是全局的。

因此，我们能够来达到我们的目的：将函数体内的变量（局部变量）在函数外部使用。也可以让函数外的变量到函数里面来使用。

#### 1.通过\$GLOBLAS来读取外部变量

```
<?php  
  
$one = 10;  
  
function demo(){  
    $two = 100;  
  
    $result = $two + $GLOBALS['one'];  
  
    return $result;  
  
}  
//你会发现结果变成了110  
echo demo();  
  
?>
```

上例中：我们将函数体外的变量通过\$GLOBALS拿到了函数体使用。所以，打破了函数外的变量不能在函数体内使用的限定。

#### 2.通过\$GLOBLAS，在函数内修改外部变量

```
<?php  
  
$hongniu = '我是一个兵，来自老百姓';  
  
function test(){  
  
    echo '执行了函数test哟<br />';  
    //调用test()函数，将通过$GLOBALS['hongniu'],把$hongniu的值改变掉
```

### 5.2.5 变量作用域

```
$GLOBALS['hongniu'] = '帮助别人很快乐';  
}  
  
test();  
//发现是不是输出的值变了呀？  
echo $hongniu;  
  
?>
```

通过上例：我们发现通过\$GLOBALS['hongniu'],把\$hongniu的值改变掉，所以在最后输出\$hongniu的时候，值发生了变化。

### 3.通过\$GLOBLAS，在函数内创建全局变量

```
<?php  
  
function hello(){  
  
    $GLOBALS['que'] = '提神喝茶更好哟';  
  
    echo '你调了一下函数hello<br />';  
}  
  
hello();  
  
echo $que;  
  
?>
```

上例中，我们发现\$que是不存在的，可以调用函数hello()后，在函数外部可以执行echo \$que了，能够看到结果：提神喝茶更好。我们明白了，函数内声明的变量，通过\$GLOBALS声明一个变量，也能够显示出来。

下面的知识是**【了解】级别**，我们越来越少的使用这种方式来声明全局变量了。在函数内使用global关键词后面跟一个或多个变量，就把变量变为了全局变量，格式如下：

```
global $变量1[,变量2,....变量n]
```

在global后面可以跟一个或者多个变量，变量用逗号分开。

```
$a = 10;  
  
$b = 100;  
  
function test(){  
  
    global $a , $b;  
  
    echo $a + $b;  
}  
//结果是不是显示出来了？  
test();
```

上例中，你还可以试试在global关键词后面跟着未创建的变量，在函数体内定义变量值，修改变量的值试试。其实与\$GLOBALS一样，只是用法不同而已。

注意：

不可在global 后写 \$变量 = 值。

global

读音：['gləʊbl]

解释：全局的，全球的

## 5.2.6 参数的引用

在变量这个函数中，我们学习了**变量的引用**，我们来回顾一下知识：

```
<?php

$a = 10;

$b = &$a;

$a = 100;

echo $a.'-----'.$b;
?>
```

上述知识点的在变量章节中，变量引用有讲述，是指变量\$a和\$b指向到了同一个存储位置，来存值。

而函数的参数引用，也是这个意思，将形参和实参指向到同一个位置。如果形参在函数体内发生变化，那么实参的值也发生变化。我们来通过实验来看看：

```
<?php

$foo = 100;

//注意：在$n前面加上了&符
function demo(&$n){

    $n = 10;

    return $n + $n;

}

echo demo($foo).'<br />';

//你会发生$foo的值变为了10
echo $foo;

?>
```

通过上例，我们发现实参为\$foo，在调用demo的时候，让\$foo和\$n指向到了同一个存储区域，当\$n的值发生变化的时候。那么\$foo的值也发生变化。

## 5.2.7 递归函数

递归函数，递归只是一个名字，而递归函数的规定：函数体内调用函数自己。

这需要一定的思维理解深度，本章学习过程当中，如果你实在是思维无法跟上的地方，可以跳过本章不用学习。

因为，实际工作中，用递归有用到，但是使用量不会很大。递归在实际工作中主要是用在：文件和文件夹操作的时候有使用到。

解决办法：

万一你的思维跟不上本章，你可以直接了解本块的原理后，用现成的文件和文件夹处理函数或文件处理类就可以。

我说几个思维上的盲区：

1. 代码是从上到下执行的，所有代码没有exit等停止符，函数必须执行完。
2. 如果函数从函数A跳至函数B后，必须把函数B执行完成再执行函数A余下的代码。
3. 递归函数必须要能执行完有结束条件，不然函数就会限入死循环。函数会永远自我执行下去。

我们来写一代码来理解一下：

```
<?php

$num = 10;

//调用一次函数A();
A($num);

function A( $arg ){

    echo $arg;

    //在函数A里面去，跑去执行函数B去了
    B($arg);

    echo '我们需要不断的努力，努力到上天都为我们感动';

    echo $arg.'<br />';
}

function B( $number ){

    echo $number;

    echo '俺是狗蛋，执行完了<br />';

}

?>
```

通过上例大家会发现：

1. 执行函数A到一半的时候，跑去执行了函数B
2. 执行完函数B，先显示出来的是：“俺是狗蛋，执行完了”，接着显示的才是：“我们需要不断的努力，努力到上天都为我们感动”
3. 也就是证明了我们所说思维盲区里面的内容，代码从上到下执行，代码必须执行完。

我们来写一个简单的递归代码,让函数自己调用自己。

```
<?php
$n = 2;

function dg( $n ){

    echo $n.'<br />';

    $n = $n - 1;

    if($n > 0){
        //在函数体内调用了dg自己哟
        dg($n);

    }else{

        echo '-----';

    }

    echo '俺是狗蛋，俺还没执行' . $n . '<br />';

}

dg($n);
?>
```

你猜猜显示结果是什么？为什么这样？

我们来仔细推理一次：

1. 第一次调用dg(), 将数字\$n = 2传到dg中, 先显示出来了2
2. 然后将\$n - 1 \$n的值为1
3. 接着判断\$n 是否大于0,肯定是大于0的, 所以调用递归自己, 再把自己执行一次。
4. 而第二次在执行自己dg()的时候, 而最下面的 **echo '俺是狗蛋，俺还没执行' . \$n . '**  
'还没有执行到。等待执行完成后再来执行
5. \$n此时等于1 , 所以显示出来1。
6. \$n把自己减了一次, \$n的结果为0
7. \$n大于0肯定不成立的, 所以显示了一条:"-----"
8. 而这个时候该执行：echo '俺是狗蛋，俺还没执行' . \$n . '  
';
9. 第二次执行dg()执行完成。第一次dg()的代码还没执行完, 将第4点中的余下代码执行完。

上面的, 我们就将运行结果推导完了, 我们接下来, 看看效果：





## 5.2.8 静态变量

如果我想知道函数被调用了多少次怎么办？在没有学习静态变量的时候，我们没有好的办法来解决。

静态变量的特点是：声明一个静态变量，第二次调用函数的时候，静态变量不会再初始化变量，会在原值的基础上读取执行。

有了这个特点，我们就可以实现，最开始我们的提问：  
函数调用词数的统计。

先执行10次demo()函数试试，再执行10次test()函数试试：

```
<?php
function demo()
{
    $a = 0;
    echo $a;
    $a++;
}
?>

<?php
function test()
{
    static $a = 0;
    echo $a;
    $a++;
}
?>

demo();
demo();
demo();
demo();
demo();
demo();
demo();
demo();
demo();
demo();

/*
for($i = 0 ;$i < 10 ; $i++){
    test();
}
*/
```

上例中你会发现：

test();执行一次数值就会加1，而demo输出的显示结果，始终为0。

通过上例你就会发现，本章开始处说明的静态变量的特点。



## 5.3 使用系统内置函数

PHP学院的李文凯，总结了这么两句话：

1. 如果代码能赚钱，帮你赚钱最多的是基本语法
2. 如果你还在当程序员，你看手册的时候可能比你看老婆的时间还长

现在我们就来学习如何使用系统内置函数。

子曰：授人以鱼不如授人以渔（送你一条鱼，不如教你捕鱼的办法）。

下面我们就来学习捕鱼的办法，在使用系统内置函数前，你必须学会以下几个东西：

1. 下载一个最新的手册
2. 经常更新手册
3. 将PHP手册放在你最容易找到的地方。需要的时候随时可以查手册
4. 学会手册中函数的用法

你可以访问，下载后放到桌面上，随时需要随时打开查即可：

[http://down.phpxy.com/book%2Fphp\\_enhanced\\_zh.chm](http://down.phpxy.com/book%2Fphp_enhanced_zh.chm)（每周更新一次）

使用函数的重点是三块：

1. 了解函数的功能，特别是常用函数的功能
2. 了解函数的参数
3. 了解函数的返回值

我们针对上面的三块，讲解6个函数，这6个函数，概况了函数的基本用法的全部注意事项：

1. 直接返回布尔型，如bool copy ()
2. 带有MIXED参数的函数如何调用。Mixed表示任何类型的数据。如Array\_unshift()
3. 参数中带有&符的参数，一定要传一个变量做为参数。函数里面改变了他的值。
4. 带有[]的参数，表示可选项。
5. 带有...的参数，表示可以传任意多个参数。
6. 带有callback的参数，表示回调函数。需要传一个函数进来。Array\_map()
7. 函数支持的版本你要了解

实验

1，拿copy()这个函数来举例：返回bool值的，通常为操作是否成功、验证是否通过、检查是否正确等。

我们拿copy这个函数来看：

```
bool copy ( string $source , string $dest [, resource $context ] )
```

**这个函数的功能为：**

拷备一个文件

**返回值：**

bool型值，就是成功返回true，失败返回false

**参数：**

两个字符串的值，一个是copy的源文件，一个为目标文件。第三个参数可选的，不常用，我们不管它。

所以，我们就可以推理出下面的实验了：

```
<?php

if(copy('abc.txt','bcd.txt')){
    echo '复制成功';
}else{
    echo '复制失败';
}
```

2，Mixed表示任何类型的数据。如Array\_unshift()

我们来看这个函数：

```
int array_unshift ( array &$array , mixed $value1 [, mixed $... ] )
```

**功能：**

操作一个数组，向数组中之前插入其他类型的参数。

**返回值：**

int 类型，可能就是插入成功最后的个数

**参数：**

第一个参数为&符，也就是在操作的过程中，改变了第一个参数的值。引用传参。也就是操作这个数组，向这个数组中传入参数。会直接改变这个数组的值。

第二个参数为mixed，因为数组可以存入多个不同的类型.mixed是指混合的意思。因此，mixed是指可传入任意类型

第三个参数加了中括号，我们所有遇到中括号的。都是指后面的参数可传，也可以不传。

第四，最后还看到了三个...(省略号)。代表可以传入任意多个参数。

示例：

```
<?php
$queue = array("凤姐", "芙蓉");
array_unshift($queue, "杨幂", "姚晨");
print_r($queue);
?>
```

你可以自己实验一下，看看结果。

3，遇到callback的传函数或者匿名函数进去协助处理，让功能更强大。

```
bool array_walk ( array &$array , callable $callback [, mixed $userdata = NULL ] )
```

我们来看看。

功能：

传入一个回调函数，将数组的原来的组操作，并且发生变化。

返回值：

bool 值 也就是意味着，提示成功或者失败

参数：

第一个参数是要操作的数组。

第二个参数是callback 代表着可以传入函数或者匿名函数。

我们来写个例子，加强理解：

```
<?php

$shuaige = array("a" => "wuyanzhu", "b" => "huangxiaoming", "c" => "ninzetao");

function test_print($item2, $key)
{
    echo $key ." ---". strtoupper($item2) . "<br />\n";
}

echo '<pre>';
var_dump($shuaige);
echo '</pre>';

array_walk($shuaige, 'test_print');

echo '用自定义函数test_print执行后的效果：';

echo '<pre>';
var_dump($shuaige);
echo '</pre>';

?>
```

通过上例，我们发现数组中的每个参数和值都被修改了。上例大家只需要理解看见传callback的，需要传入函数协助处理就可以。不用理解。我们在下一章的数组部份，学习完数组，大家可以再看看。

### 函数支持的版本号很重要

我们来看手册中一个函数的截图：



这是一个系统函数，而不是自定义的函数。系统函数手册中存在这个函数，可是为什么无法调用执行这个函数呢？请注意，（ PHP 5 >= 5.5.0 ），你可以phpinfo()看一下你当前的版本。有的时候可能是因为你的版本太低，或者你所在的版本中没有这个函数，会提示函数不存在。

有事没事查手册，手册比老婆还要亲。



# 5.4 文件包含函数

在实际开发中，常常需要把程序中的公用代码放到一个文件中，使用这些代码的文件只需要包含这个文件即可。这种方法有助于提高代码的重用性，给代码的编写与维护带来很大的便利。在PHP中，有require、require\_once、include、include-once四种方法包含一个文件。

我们来对比他们的不同：

函数	包含失败	特点
Include	返回一条警告	文件继续向下执行。通常用于动态包含
Require	一个致命的错	代码就不会继续向下执行。通常包含极为重要的文件，整个代码甬想执行
Include_once	返回一条警告	除了原有include的功能以外，它还会做once检测，如果文件曾经已经被被包含过，不再包含
Require_once	一个致命的错	除了原的功能一外，会做一次once检测，防止文件反复被包含

注意：

1，少用\_once带once，因为它会消耗更多的资源去做检测的工作。

2，特高级

Include文件只需要编译一次，因为每次包含include都会再执行一次对应的代码，如何减少include再次执行时，需要重新解析的过程。

我们来做一个实验：

1，include包含函数的功能。

创建一个functions.php文件，里面写上两个函数：

```
<?php
//functions.php文件

function demo(){
    echo 'aaaa';
}

function test(){
    echo 'cccdddd';
}

?>
```

在functions.php的同级目录下，我再创建一个user.php文件把functions.php文件包含进来。这样我的函数就可以专门放在functions.php里面，哪儿需要用到这些函数的时候，我就从哪儿包含进来：

```
<?php

//user.php
```

## 5.4 文件包含函数

```
include 'functions.php';

//可以直接调用
demo();

test();

?>
```

我通过过上例我们知道了include的功能。接下来我们对比include和require：

代码中，我们先用include 来包含不存在的test.php文件，

```
<?php

//user.php

include 'functions.php';
include 'test.php';

//可以直接调用
demo();

test();

?>
```

再用require包含 不存在的test.php文件：

```
<?php

//user.php

include 'functions.php';
require 'test.php';

//可以直接调用
demo();

test();

?>
```

通过上例的对比我们发现：

1. 如果test.php文件不存在include 会发出警告继续执行demo()和test()函数。
2. 而require则直接报错，demo()和test()函数无法继续执行。

我们通过表格知道了:include 和include\_once的区别在于，检测是否重复包含。如果重复包含了include\_once不会再包含 对应的文件了，而include 则不管这些。有没引入过文件，都再引入一次。

同样刚刚的user.php我们再实验一下,我们将functions.php包含两次，分别使用include和include\_once：

```
<?php

//user.php

//这儿被包含了两次同样的函数定义文件哟
include 'functions.php';
```

## 5.4 文件包含函数

```
include 'functions.php';

//可以直接调用
demo();

test();

?>
```

改为include\_once再试一次：

```
<?php

//user.php

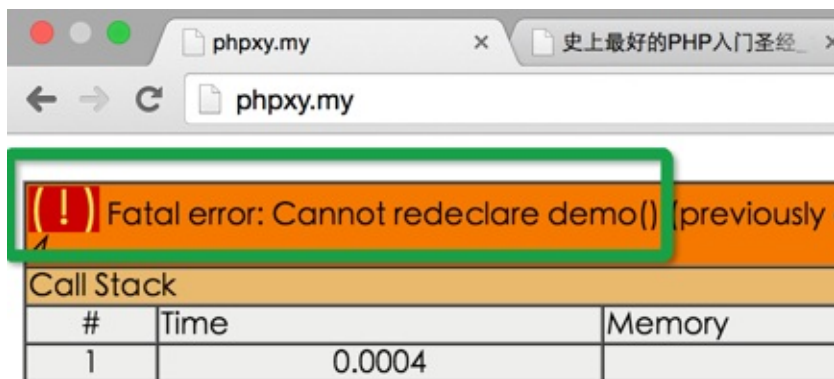
//这儿被包含了两次同样的函数定义文件哟
include_once 'functions.php';
include_once 'functions.php';

//可以直接调用
demo();

test();

?>
```

大家分别执行后会发现——include 这次引入functions.php两次的时候报了如下错误：



上图中的提示是说，不能重新定义函数demo()。

我们在函数定义章节讲过，函数不能定义两次，否则会报错。因为我们将 functions.php包含了两次所以执行了两次，因此报这个错误。

而include\_once不报错的原因是因为：他检测了functions.php曾经包含过，不再进行包含引入了。

而require和require\_once的功能，大家用你最最聪明的小脑袋是不是就能推理出来了呢？require\_once有两个特点：

## 5.4 文件包含函数

1. 包含 的文件必须存在，否则停止执行
2. 会做重复包含检查哟





## 5.5 数学常用函数

---

在PHP程序中常常需要进行数据处理运算，这就需要使用数学函数。数学函数是最简单、最常用的系统函数。本章以数学函数为例讲解系统函数的调用方法。

我们现在来说我们最常用的数学函数。在讲解数学函数之前，我们来说一项。就是——PHP数学函数特别多，如下链接截图：

<http://phpbook.phpxy.com/ref.math.html>

- [abs](#) — 绝对值
- [acos](#) — 反余弦
- [acosh](#) — 反双曲余弦
- [asin](#) — 反正弦
- [asinh](#) — 反双曲正弦
- [atan2](#) — 两个参数的反正切
- [atan](#) — 反正切
- [atanh](#) — 反双曲正切
- [base\\_convert](#) — 在任意进制之间转换数字
- [bindec](#) — 二进制转换为十进制
- [ceil](#) — 进一法取整
- [cos](#) — 余弦
- [cosh](#) — 双曲余弦
- [decbin](#) — 十进制转换为二进制
- [dechex](#) — 十进制转换为十六进制
- [decoct](#) — 十进制转换为八进制
- [deg2rad](#) — 将角度转换为弧度
- [exp](#) — 计算 e 的指数
- [expm1](#) — 返回  $\exp(\text{number}) - 1$ ，甚至当 number 的值接近零也能计算出准确结果
- [floor](#) — 舍去法取整
- [fmod](#) — 返回除法的浮点数余数
- [getrandmax](#) — 显示随机数最大的可能值
- [hexdec](#) — 十六进制转换为十进制
- [hypot](#) — 计算一直角三角形的斜边长度
- [intdiv](#) — Integer division
- [is\\_finite](#) — 判断是否为有限值
- [is\\_infinite](#) — 判断是否为无限值
- [is\\_nan](#) — 判断是否为合法数值
- [lcg\\_value](#) — 组合线性同余发生器
- [log10](#) — 以 10 为底的对数
- [log1p](#) — 返回  $\log(1 + \text{number})$ ，甚至当 number 的值接近零也能计算出准确结果
- [log](#) — 自然对数
- [max](#) — 找出最大值
- [min](#) — 找出最小值
- [mt\\_getrandmax](#) — 显示随机数的最大可能值
- [mt\\_rand](#) — 生成更好的随机数
- [mt\\_srand](#) — 播下一个更好的随机数发生器种子
- [octdec](#) — 八进制转换为十进制
- [pi](#) — 得到圆周率值
- [pow](#) — 指数表达式
- [rad2deg](#) — 将弧度数转换为相应的角度数
- [rand](#) — 产生一个随机整数
- [round](#) — 对浮点数进行四舍五入
- [sin](#) — 正弦
- [sinh](#) — 双曲正弦
- [sqrt](#) — 平方根
- [srand](#) — 播下随机数发生器种子
- [tan](#) — 正切
- [tanh](#) — 双曲正切

这些不用大家全部掌握，你看一遍知道PHP的数学函数里面为大家准备了这么多就可。以后需要的时候，直接来用。

大家只需要记最常用的一些即可，我将最常用的列表列给大家：

函数名	描述	实例	输入	输出
<a href="#">abs()</a>	求绝对值	<code>\$abs = abs(-4.2); //4.2</code>	数字	绝对值数字
<a href="#">ceil()</a>	进一法取整	<code>echo ceil(9.999); // 10</code>	浮点数	进一取整
	舍去法			

	取整			
fmod()	浮点数取余	"\$x = 5.7;\$y = 1.3;\$r = fmod(\$x, \$y); // \$r equals 0.5, because 4 * 1.3 + 0.5 = 5.7 "	两个浮点数,x>y	浮点余数
pow()	返回数的n次方	echo pow(-1, 20); // 1	基础数 n 次方	乘方值
round()	浮点数四舍五入	echo round(1.95583, 2); // 1.96	一个数值	保留小数点后多少位,默认为0 舍入后的结果
sqrt()	求平方根	echo sqrt(9); //3	被开方的数	平方根
max()	求最大值	"echo max(1, 3, 5, 6, 7); // 7 echo max(array(2, 4, 5)); // 5"	多个数字或数组	返回其中的最大值
min()	求最小值	min	多个数字或数组	返回其中的最小值
mt_rand()	更好的随机数	echo mt_rand(0,9); //n	最小/最大,随机数	随机返回范围内的值
rand()	随机数	echo rand()	最小/最大,随机数	随机返回范围内的值
pi()	获取圆周率值	echo pi(); // 3.1415926535898	无	获取圆周率

## 5.6 日期常用函数

---

曾经有位大哥说过：

只要是有人的地方就有江湖。

我说：

要记录数据的地方就有时间。

我们注册的时间、发消息的时间、访问的时间、登陆的时间、抽奖的时间、中奖的时间、搜索关键词的时间... ..

太多太多的时间都在被记录了。而这些东西都需要让PHP协助我们记录到数据库里面。

那好，各位美男子和美女们。我们开始来学习日期函数吧。

---

注：

我们为什么不直接用数据库里面的时间，因为在实际开发过程中，我们需要把时间取出来再排序和显示，所以单纯的用数据库里面的时间达不到我们的业务需求。

## 5.6.1 获取时期时间信息函数

在正式学习日期函数前大家要了解几个概念：

1. 时区
2. 世界时
3. unix时间戳

### 时区

这个概念，之前大家听说过很多。我们来啰嗦两句，我们现实生活中使用的实区，在电脑里面也是一样有规定的。

1884年在华盛顿召开国际经度会议时，为了克服时间上的混乱，规定将全球划分为24个时区。

在中国采用首都北京所在地东八区的时间为全国统一使用时间。

### 世界时

不光是天文学家使用格林尼治时间（英文简写:GMT），就是在新闻报刊上也经常出现这个名词。我们知道各地都有各地的地方时间。如果对国际上某一重大事情，用地方时间来记录，就会感到复杂不便，而且将来日子一长容易搞错。因此，天文学家就提出一个大家都能接受且又方便的记录方法，那就是以格林尼治（英国某地区）的地方时间为标准。

### unix时间戳

电脑本身不认识时间，我们在电脑里面设置一个时间方便运算。于是我们规定了一种计算方式，unix时间戳。

从Unix纪元（1970年1月1日零时）开始到一个时间经过的秒数。

我们学了几个概念，那我们现在可以开始来学习时间函数了。

### 1. 设置时区

如果，我们是跨国的多语方程序，我们通常是通过在配置文件里面来写好了个时区，每次程序运行的时候。都会读取这个时区的设置，来显示时间。

设置时区的函数为：

- 1). date\_default\_timezone\_get()
- 2).date\_default\_timezone\_set()

第一个函数我们就不重点讲解了，比较简单。

用法如下：

```
string date_default_timezone_get ( void )
```

功能如下：

取得一个脚本中所有日期时间函数所使用的默认时区

示例：

```
<?php
echo date_default_timezone_get ();
?>
```

这样就显示出来了当前设定的时间。

本文档使用 [看云](#) 构建

### 5.6.1 获取时期时间信息函数

第二个函数是重点：

用法如下：

```
bool date_default_timezone_set ( string $timezone_identifier )
```

功能如下：

用于所有日期时间函数的默认时区

示例：

```
<?php

//定义一下时区常量，以后你可以放到配置文件里
define('TIME_ZONE','Asia/shanghai');

//执行函数
date_default_timezone_set(TIME_ZONE);

echo date('Y-m-d H:i:s');

?>
```

上例的代码对比试试，再将date\_default\_timezone\_set注释掉，再看看会提示什么。

注：

时区列表请详见官方手册 <http://php.net/manual/zh/timezones.php>

## 2.time()获取当前的unix时间戳

time()函数的功能是获取当前时间的 Unix 时间戳。

以下代码输出当前时间的Unix 时间戳。

```
<?php
$time=time();
print_r( $time);

?>
```

程序运行结果：

1421597858

## 3. “亚麻跌”是PHP学习时间处理的关键

Y 英文是 year，为年份代表年——亚

m 英文代表month，为月份代表——麻

d 英文代表day，为日期 代表——跌

所以我们需要输出前前的年份，月份，日期的话。例如：1997年7月1日，我们就可以用到上面的三个参数。

```
<?php

echo date('Y年m月d日');

?>
```

你可以运行一下代码试试，看看是不是显示出来了。

后面还有几个参数：

H:m:s 代表的是：时分秒

h 的英文为：hour 代表小时

i的英文为：minute 代表分钟

s的英文为：second 代表秒

写全了就是：

```
<?php

//就可以显示出来当前的时间了哟。
echo date('Y-m-d H:i:s');
?>
```

date函数用于将一个时间进行格式化输出，以方便时间的显示或存储。其语法格式如下：

string date ( string \$format [, int \$timestamp] )

在参数列表中:

- 1. \$timestamp是一个时间戳，函数将这个时间戳按\$format规定的格式输出。
- 2. 如果\$timestamp没有输入值，则默认为当前的时间。
- 3. \$format是一个时间输出格式的字符串，需要使用规定的字符构造输出格式。

date函数的格式参数表：

字符	说明	返回值
d	月份中的第几天，有前导零的2 位数字	01 到31
D	英文星期几，3个字母	Mon到Sun
j	月份中的第几天，没有前导零	1 到31
l(字母)	英文星期几	Sunday到 Saturday
N	1格式数字表示的星期	1 ( 表示星期一 ) 到7 ( 表示星期天)
S	每月天数后面的英文后缀，2个字符	st , nd , rd或者th。可以和jg一起用
w	星期中的第几天，数字表示	0 ( 表示星期天 ) 到 6 ( 表示星期六 )
z	一年中的第几天	0到366
W	年份中的第几周，每周从星期一开始	42 ( 当年的第42周 )
F	月份，完整的文本格式	January 到 December
m	数字表示月份，有前导零	01 到 12
M	3个字母缩写表示的月份	Jan 到Dec
n	数字表示月份，没有前导零	1 到 12
t	给定月份所应有的天数	28 到 31

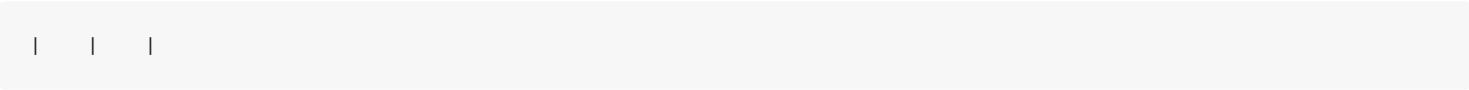
L	是否为闰年	如果是闰年为1，否则为0
o	格式年份数字	例如2007
Y	4 位数字完整表示年份	例如1999或2008
y	2 位数字表示的年份	例如99或08
a	小写的上午和下午值	am或pm
A	大写的上午和下午值	AM或PM
g	小时，12小时格式，没有前导零	1到12
G	小时，24小时格式，没有前导零	0 到 23
i	有前导零的分钟数	00 到 59
s	秒数，有前导零	00到59
e	时区标识	
U	从Unix纪元开始至今的秒数	长整型数字

3. getdate获取当前系统时间

getdate用来获取当前系统的时间，或者获得一个时间戳的具体含义。时间戳是一个长整数，表示getdate的语法格式如下所示。

```
array getdate ([ int $timestamp = time() ] )
```

函数的返回值是一个根据timestamp得到的包含有时间信息的数组。如果没有参数，则会返回当前的时间。getdate返回的数组，键名包括时间和日期的完整信息。



键名	说明	返回值
secnods	秒	数字0到 59
minutes	分钟	数字0到59
hours	小时	数字 0到 23
mday	月份中第几天	数字 1到 31
wday	星期中第几天	数字0（表示星期天）到6（表示星期六）
mon	月份	数字 1 到 12
year	年	4 位数字表示的完整年份
yday	一年中第几天	数字0到365
weekday	星期几的英文	Sunday到 Saturday
month	月份的英文	January 到 December
0	自从Unix纪元开始的秒数	长整型数字

以下代码可以返回getdate 数组的详细信息。

```
<?php
    $mytime= getdate();
    print_r( $mytime);
```



### 5.6.1 获取时期时间信息函数

```
?>
```

print\_r可以输出一个数组中所有的键名与值。运行这段代码，结果如下所示。程序输出当前计算机的时间与日期详细信息：

```
Array
(
    [seconds] => 1          //秒
    [minutes] => 10         //分钟
    [hours] => 17           //小时
    [mday] => 18            //日
    [wday] => 0             //星期中的第几天
    [mon] => 1              //月
    [year] => 2015          //年
    [yday] => 17            //年中的第几天
    [weekday] => Sunday     //星期
    [month] => January      //月份
    [0] => 1421597401       //时间戳
)
```

理解了getdate函数和返回的数组以后，就很容易取得当前的时间信息了。下面的代码就是用getdate函数取得时间信息，调用返回时间数组的值输出时间信息。

```
<?php
$mytime = getdate();
echo "年 :".$mytime['year']."\n";
echo "月 :".$mytime['mon']."\n";
echo "日 :".$mytime['mday']."\n";
echo "时 :".$mytime['hours']."\n";
echo "分 :".$mytime['minutes']."\n";
echo "秒 :".$mytime['seconds']."\n";
echo "一个小时中的第几秒 :".$mytime['minutes']."\n";
echo "这是一分钟的第几秒 :".$mytime['seconds']."\n";
echo "星期名称 :".$mytime['weekday']."\n";
echo "月份名称 :".$mytime['month']."\n";
echo "时间戳 :".$mytime[0]."\n";
?>
```

运行这个程序，会显示当前时间的详细信息。程序的运行结果如下所示。

```
年 :2015
月 :1
日 :18
时 :17
分 :14
秒 :11
一星期中的第几天 :14
一年中的第几天 :11
星期名称 :Sunday
月份名称 :January
时间戳 :1421597651
```

## 5.6.2 日期验证函数

---

checkdate可以判断一个输出的日期是否有效。

在实际的工作中，我们需要经常用于检测常用于用户提交表单的数据验证。

例如：验证用户输入的时间是否正确。

函数的语法格式如下：

```
bool checkdate ( int $month , int $day , int $year )
```

下例中，我们就可以用一个代码来进行实验，写出一段真实的例子。试试2011年有没有2月29日。

如果是有效的时间就返回真，如果不是有效的时间就返回假。

```
<?php
var_dump(checkdate(12, 31, 2018));
var_dump(checkdate(2, 29, 2011));
?>
```

输出结果：

bool(true)

bool(false)

## 5.6.3 获取本地化时间戳函数

在实际的工作中我们还需要经常用到指定某个时间生成。

例如：需要找到昨天到今天此时此刻的注册用户。

那么我们需要做两件事情：

1. 得到当前的时间unix时间戳。用time()函数就可以直接搞定
2. 那么昨天指定时间该怎么生成呢。这个时候我们就需要使用到mktime()函数。简称：make time.创建时间。

生成出来的时间最好是unix时间戳。因为是1970年1月1日0时到现在的时间。我们做一个区间判断，就把昨天到今天注册的用户按照时间筛选出来了。

我们的mktime()函数可以对一个日期和时间获得一个本地化时间戳。其语法格式如下所示：

```
int mktime (int $hour [, int $minute [, int $second [, int $month [, int $day [, int $year [, int $.is_dst [, ] ] ] ] ] ] 31)
```

函数的参数分别表示：时、分、秒、月、日、年、是否为夏令时。在使用这个函数时，需要注意所列的参数要与函数的参数含义相同。例如，下面的代码实现了用mktime构造一个时间戳的功能。

```
<?php
echo  mktime (13 ,15 , 3 0, 8,18, 2008) ;
?>
```

运行程序结果如下所示：

1219036530

mktime函数的返回结果是一个Unix时间戳，对用户的含义不大，常常与date函数一起完成时间的转换。

例如，下面的代码实现对时间的运算：

```
<?php
echo date("m-d-Y h:m:s")."\n";
echo date("m-d-Y h:m:s",mktime(10,15,35,date("m"),date("d"),date("Y")))."\n";
echo date("m-d-Y h:m:s",mktime(10,15,35,date("m"),date("d")-30,date("Y")))."\n";

?>
```

我们使用的时候还经常需要使用到另外一个函数：strtotime()。

它的语法格式如下：

```
int strtotime ( string $time [, int $now = time() ] )
```

它能将英文文本的日期时间描述解析为 Unix 时间戳。

### 5.6.3 获取本地化时间戳函数

参数：

1. 传入一个字符串的时间
2. 可选参数为是否传入unix时间戳，如果不传则是当前的unix时间戳。

我们来实验一下，看看手册中提供的一些例子：

```
<?php
//now为现在的当前时间
echo strtotime("now")."<br />";
//2000年9月10日
echo strtotime("10 September 2000")."<br />";
//当前时间加一天
echo strtotime("+1 day")."<br />";
//当前时间加一周
echo strtotime("+1 week")."<br />";
//当前时间加一周2天4小时2秒
echo strtotime("+1 week 2 days 4 hours 2 seconds")."<br />";
//下一个星期四
echo strtotime("next Thursday")."<br />";
//上一个星期一
echo strtotime("last Monday")."<br />";
?>
```

通过上例发现，把一些时间通通加上了或者减去了英文中所表示的指定时间段。

## 5.6.4 程序执行时间检测

我们有的时经常需要做程序的执行时间执行效率判断。

实现的思路如下：

```
<?php
//记录开始时间

//记录结整时

// 开始时间 减去 (-) 结束时间 得到程序的运行时间

?>
```

可是大家不要忘了，程序的运行速度太快了。快到只有0.00000几秒的一刹那。那这个时候大家要记录一个特函的函数了：

```
mixed microtime ([ bool $get_as_float ] )
```

microtime()这个函数，能够返回当前 Unix 时间戳和微秒数。

参数：

如果你传入true的话，将会返回一个浮点类型的时间，这样方便参与运算。

我们来模拟一个检测函数执行时间的例子，测试某个函数效率的快慢：

```
<?php
//开始时间
$time_start = microtime(true);

//循环一万次
for($i = 0 ; $i < 10000 ; $i++){

    //你可以用上, mktime() 生成一个昨天的时间

    //再用strtotime() 生成一个昨天的时间

    //对比两个函数认的效率高

}

//结整时间
$time_end = microtime(true);
//相减得到运行时间
$time = $time_end - $time_start;

echo "这个脚本执行的时间为 $time seconds\n";
?>
```

最后输出的结果就是我们实际的函数的执行时间。你可以多对比几次，看看最终的结果。

谁的时间短，在实际的工作中，你就可以经常使用哪个函数。

## 5.7 字符串常用函数

---

数组、字符串和数据库是我们函数里面最、最、最常用的三类函数，数组和数据库我们现在还没有讲到，等讲到的时候我们再来和大家细说。

当然PHP的字符串函数也有很多。我们最常使用的两个系列的字符串：

1. 单字节字符串处理函数
2. 多字节字符串处理函数
3. 字符串编码转换函数

我们来说说为什么要学这么多函数：

1. 我们学的是中文，是双字节或者三字节的。老外的函数只能处理英文和数字这些单字节的字符串处理不鸟中文。达不到我们的功能需求
2. 有的时候需要做不同字符编码间的转换，例如：把GBK的转为UTF-8
3. 英文这些字符在电脑里又是必须要处理的

因此，我们要学三个类型的常用字符串函数。

我们来贴个PHP手册的链接给大家看看：

1. 多字节iconv\_\* 系列：<http://phpbook.phpxy.com/book.iconv.html>
2. 多字节mb\_\* 系列：<http://php.net/manual/zh/book.mbstring.php>
3. 处理英文系列：<http://phpbook.phpxy.com/book.strings.html>

看了这三个链接是不是又被吓着了？



当然，你不用学这么多。PHP学院已经帮你把最常用的，需要强制背诵的准备好了。



之前都教过大家用，那你开始背函数吧。

PHP常用函数：

函数名	描述	实例
trim()	删除字符串两端的空格或其他预定义字符	<code>\$str = "\r\nHello World!\r\n"; echo trim(\$str);</code>
rtrim()	删除字符串右边的空格或其他预定义字符	<code>\$str = "Hello World!\n\n"; echo rtrim(\$str);</code>
chop()	rtrim()的别名	同上
ltrim()	删除字符串左边的空格或其他预定义字符	<code>\$str = "\r\nHello World!"; echo ltrim(\$str);</code>
dirname()	回路径中的目录部分（我们把它归在了字符串函数里了）	<code>echo dirname("c:/testweb/home.php");</code>
str_pad()	把字符串填充为指定的长度	<code>\$str = "Hello World"; echo str_pad(\$str,20,".");</code>
str_repeat()	重复使用指定字符串	<code>echo str_repeat(".",13);</code>
str_split()	把字符串分割到数组中	<code>print_r(str_split("Hello"));</code>
strrev()	反转字符串	<code>echo strrev("Hello World!");</code>
wordwrap()	按照指定长度对字符串进行折行处理	<code>\$str = "An example on a long word is: Supercalifragulistic"; echo wordwrap(\$str,15);</code>
str_shuffle()	随机地打乱字符串中所有字符	<code>echo str_shuffle("Hello World");</code>
parse_str()	将字符串解析成变量	<code>"parse_str("id=23&amp;name=John%20Adams",\$myArray); print_r(\$myArray);"</code>
number_format()	通过千位分组来格式化数字	<code>"echo number_format("1000000"); echo number_format("1000000",2); echo number_format("1000000",2,"",".");"</code>
strtolower()	字符串转为小写	<code>echo strtolower("Hello WORLD!");</code>
strtoupper()	字符串转为大写	<code>echo strtoupper("Hello WORLD!");</code>
ucfirst()	字符串首字母大写	<code>echo ucfirst("hello world");</code>
ucwords()	字符串每个单词首字符转为大写	<code>echo ucwords("hello world");</code>
htmlentities()	把字符转为HTML实体	<code>\$str = "John &amp; 'Adams'"; echo htmlentities(\$str, ENT_COMPAT);</code>
htmlspecialchars()	预定义字符转html编码	
nl2br()	\n转义为 标签	<code>echo nl2br("One line.\nAnother line.");</code>
strip_tags()	剥去 HTML、XML 以及 PHP 的标签	<code>echo strip_tags("Hello <b>world!</b>");</code>
addslashes()	在指定的字符前添加反斜线转义字符串中字符	<code>\$str = "Hello, my name is John Adams." echo \$str; echo addslashes(\$str,'m');</code>
stripcslashes()	删除由addslashes()添加的反斜线	<code>echo stripcslashes("Hello, \my na\me is Kai Ji\m.");</code>
addslashes()	指定预定义字符前添加反斜线	<code>\$str = "Who's John Adams?";echo addslashes(\$str);</code>
stripslashes()	删除由addslashes()添加的转义字符	<code>echo stripslashes("Who\'s John Adams?");</code>
quotemeta()	在字符串中某些预定义的字符前添加反斜线	<code>\$str = "Hello world. (can you hear me?)" echo quotemeta(\$str);</code>



## 5.7 字符串常用函数

chr()	从指定的 ASCII 值返回字符	echo chr(052);
ord()	返回字符串第一个字符的 ASCII 值	echo ord("hello");
strcasecmp()	不区分大小写比较两字符串	echo strcasecmp("Hello world!","HELLO WORLD!");
strcmp()	区分大小写比较两字符串	
strncmp()	比较字符串前n个字符,区分大小写	
strncasecmp()	比较字符串前n个字符,不区分大小写	int strncasecmp ( string \$str1 , string \$str2 , int \$len )
strnatcmp()	自然顺序法比较字符串长度,区分大小写	int strnatcmp ( string \$str1 , string \$str2 )
strnatcasecmp()	自然顺序法比较字符串长度,不区分大小写	int strnatcasecmp ( string \$str1 , string \$str2 )
chunk_split()	将字符串分成小块	str chunk_split(str \$body[,int \$len[,str \$end]])
strtok()	切开字符串	str strtok(str \$str,str \$token)
explode()	使用一个字符串为标志分割另一个字符串	array explode(str \$sep,str \$str[,int \$limit])
implode()	同join,将数组值用预订字符连接成字符串	string implode ( string \$glue , array \$pieces )
substr()	截取字符串	string substr ( string \$string , int \$start [, int \$length ] )
str_replace()	字符串替换操作,区分大小写	mix str_replace(mix \$search,,mix \$replace,mix \$subject[,int &\$num])
str_ireplace()	字符串替换操作,不区分大小写	mix str_ireplace ( mix \$search , mix \$replace , mix \$subject [, int &\$count ] )
substr_count()	统计一个字符串,在另一个字符串中出现次数	int substr_count ( string \$haystack , string \$needle [, int \$offset = 0 [, int \$length ] ] )
substr_replace()	替换字符串中某串为另一个字符串	mixed substr_replace ( mixed \$string , string \$replacement , int \$start [, int \$length ] )
similar_text()	返回两字符串相同字符的数量	int similar_text(str \$str1,str \$str2)
strchr()	返回一个字符串在另一个字符串中开始位置到结束的字符串	string strstr ( string \$str, string \$needle , bool \$before_needle )
strrchr()	返回一个字符串在另一个字符串中最后一次出现位置开始到末尾的字符串	string strrchr ( string \$haystack , mixed \$needle )
stristr()	返回一个字符串在另一个字符串中开始位置到结束的字符串,不区分大小写	string stristr ( string \$haystack , mixed \$needle [, bool \$before_needle = false ] )
strtr()	转换字符串中的某些字符	string strtr ( string \$str , string \$from , string \$to )
strpos()	寻找字符串中某字符最先出现的位置	int strpos ( string \$haystack , mixed \$needle [, int \$offset = 0 ] )
stripos()	寻找字符串中某字符最先出现的位置,不区分大小写	int stripos ( string \$haystack , string \$needle [, int \$offset ] )
strrpos()	寻找某字符串中某字符最后出现的位置	int strrpos ( string \$haystack , string \$needle [, int \$offset = 0 ] )

stripos()	寻找某字符串中某字符最后出现的位置,不区分大小写	int stripos ( string \$haystack , string \$needle [, int \$offset ] )
strspn()	返回字符串中首次符合mask的子字符串长度	int strspn ( string \$str1 , string \$str2 [, int \$start [, int \$length ]] )
strcspn()	返回字符串中不符合mask的字符串的长度	int strcspn ( string \$str1 , string \$str2 [, int \$start [, int \$length ]] )
str_word_count()	统计字符串含有的单词数	mix str_word_count(str \$str,[])
strlen()	统计字符串长度	int strlen(str \$str)
count_chars()	统计字符串中所有字母出现次数 (0..255)	mixed count_chars ( string \$string [, int \$mode ] )
md5()	字符串md5编码	\$str = ""Hello""; echo md5(\$str)
iconv		
mb_substr	获取字符串的部分	string mb_substr ( string \$str , int \$start [, int \$length = NULL [, string \$encoding = mb_internal_encoding() ]] )
mb_http_output	设置/获取 HTTP 输出字符编码	mixed mb_http_output ([ string \$encoding = mb_http_output() ] )
mb_strlen	获取字符串的长度	mixed mb_strlen ( string \$str [, string \$encoding = mb_internal_encoding() ] )
iconv	字符串按要求的字符编码来转换	string iconv ( string \$in_charset , string \$out_charset , string \$str )
iconv_substr	截取字符串的部分	
iconv_get_encoding	获取 iconv 扩展的内部配置变量	
mb_substr_count	统计字符串出现的次数	
mb_check_encoding	检查字符串在指定的编码里是否有效	
mb_strrpos	查找字符串在一个字符串中最后出现的位置	
mb_split	使用正则表达式分割多字节字符串	
parse_url	解释URL成为一个数组	

注：mb\* 和iconv\* 他们可以处理多字节字符，例如：中文。

中文主要用的是GBK和utf-8两种编码格式。  
GBK和utf-8是两个不同的编码委员会对于汉字进行的编码的标准。

他们规定GBK是双字节，也就是一个汉字占用2Bytes。  
utf-8是三字节，一个汉字占用三个字节长度的存储空间。

## 06.PHP数组与数据结构

---

数组是PHP中一个灰常、灰常、灰常重要的一个数据类型。

因为他涉及到的知识点太多，太杂，我们的讲解是在数组类型部份告诉了大家一个大概，然后专门用一个章节来跟大家讲解。

学习数组，大家主要学习两部分的知识：

1. 数组的定义，定义中的一些注意的坑
2. 数组的函数使用和（默写）

**又是默写级别的**



## 6.1 数组的定义

数组在之前我们让大家记住两个观点：

1. 数组可以存入多个不同类型的数据，是一个复合数据类型。
2. 数组的英文是array，学一下最简单的数组声明。

那我们来简单的回顾：

```
<?php

$shu = array(1 , 1.5 , true , '天王盖地虎, 小鸡炖蘑菇');

echo '<pre>';
var_dump($shu);
echo '</pre>';

?>
```

在上例中，我们发现我们存入了：

1. 整型
2. 浮点
3. 布尔
4. 字符串

注：上例中echo pre这个标签的主要目的是为原样输出，格式展示的更好看，更清晰。

我们用var\_dump()打印出来显示看一下效果：



我们仔细观察一下上面的这张图，你会发现以下几个特点：

1. array(size = 4) 说明里面有4个元素
2. 0 => int 1 我们知道int是整型的意思，1是一个整型的数值。那前面的0,1,2,3和=>代表什么意思呢？
3. 最新前的0, 1, 2, 3代表的是值的读取标识号，我们称之为下标或者键（英文：key）
4. => 是一个符号标准叫法叫作：键值对应符。因此，以后再看到 0=> int 1可以这样来说。下标访问符0对应整型的1。
5. 我们还称数组里面的键值对为元素，元素就是键值对的组合。

## 6.1 数组的定义

哦耶！数组好像还挺好学的，全是一些规律哟。

我们通过上例，其实你一不小心就完成了数组其中的一种声明方式：索引数组的声明哟。

所谓索引数组：就是下标全为整型的数组。

### 索引数组的下标必须要从0开始吗？

答：这个问题其实不然，索引数组并不一定要从0开始。

那如何能够不从0开始呢？

答：需要用到大家在上面学的一小块知识哟。就是键值对应符。我们来动手写写。

```
<?php

$kele = array('只有不断努力才能博得未来',10 => 'NoAlike', 'PHP学院' , '去PHP学院去PHP', 19 => '凤姐和芙蓉我都爱' , '杨幂臭脚我最爱');

//打印显示$kele
echo '<pre>';
var_dump($kele);
echo '</pre>';
?>
```

上例中，我们一不小心就写了一个索引数组。不过索引数组的下标不是从0 开始的，而是从10开始的。

不过上面的例子中，我们觉得写的不优美，我们可以将代码写的更加优美一些，格式更加清晰一些。

```
<?php

$kele = array(
    '只有不断努力才能博得未来',
    10 => 'NoAlike',
    'PHP学院' ,
    '去PHP学院去PHP',
    19 => '凤姐和芙蓉我都爱' ,
    '杨幂臭脚我最爱'
);

//打印显示$kele
echo '<pre>';
var_dump($kele);
echo '</pre>';
?>
```

这样是不是看得更清楚，一行对应一个数组的值。

我们执行一下代码，看一下效果：

```

← → ↻ phpxy.com

array (size=6)
  0 => string '只有不断努力才能博得未来' (length=36)
  10 => string 'phpxy' (length=5)
  11 => string 'php学院' (length=9)
  12 => string '去PHP学院去PHP' (length=18)
  19 => string '凤姐和芙蓉我都爱' (length=24)
  20 => string '杨幂臭脚我最爱' (length=21)

```

通过上面的效果，我们来总结规律：

1. 索引数组若不强制声明他的下标，他的下标是从0开始的。（我们的第一个数组的值：只有不断努力才能博得未来。这个值的下标为0）。
2. 如果我指定过下标他的下标就为我指定的值。如下标为10和下标为19的，都是我指定过的值。
3. 若某个值（如NoAlike），强制指定了下标（下标为10）。在它后面加上的值（PHP学院），不指定下标的话。他们的下标增长规律为最大值+1。

例如：凤姐和芙蓉我都爱的下标为19.我在后面加上了：杨幂臭脚我最爱。它的下标自动增长为了21。

不知不觉，你已经学会了索引数组的创建，神奇吧？真为你感到高兴！

## 向索引数组中增加元素

学习完了索引数组的创建，接下来我们学习索引数组的**增加、修改和删除**。

```

<?php

$minren = array(
    '杨幂',
    '王珞丹',
    '刘亦菲',
    '黄圣依'
);

//如何向这$minren这个数组中增加元素呢

//猜猜范冰冰的下标是多少？
$minren[] = '范冰冰';

$minren[100] = '范爷';

//它的下标又为几呢？
$minren[] = '李晨';

?>

```

总结：

1. 向索引数组中增加元素用: 数组变量名[]、数组变量名[键值]这两种方式来增加元素
2. 键值的增长规则与之前的规则一样。都是最大值加1的原则。

## 向索引数组中删除元素

我们还拿刚刚的数组来举例：

## 6.1 数组的定义

```
<?php

$minren = array(
    '杨幂',
    '王珞丹',
    '刘亦菲',
    '黄圣依',
    '范冰冰'
);

//假设我不喜欢：黄圣依，如何将黄圣依给删掉呢？

//如果删除掉后范冰冰的下标为多少呢？

//如果在后面再追加一个元素，会填掉：“黄圣依”留下来的空吗？

unset($minren[3]);

$minren[] = '金星';

echo '<pre>';

var_dump($minren);

echo '</pre>';

?>
```

看看效果：



```
array (size=5)
  0 => string '杨幂' (length=6)
  1 => string '王珞丹' (length=9)
  2 => string '刘亦菲' (length=9)
  4 => string '范冰冰' (length=9)
  5 => string '金星' (length=6)
```

1. 使用unset删除变量的方式来删除数组里面的值。
2. 删除了中间的值，并不会让后面的下标向前自动移动。而是原来的值为多少就为多少
3. 删除掉其中的某个值，新加入的值不会替换掉原来的位置，依然遵循最大值加1的原则。

## 修改值

我们学习最简单的创造、增加和删除。我相信大家一定能够推理出如何修改值了。

```
<?php

$minren = array(
    '杨幂',
    '王珞丹',
    '刘亦菲',
    '黄圣依',
    '范冰冰'
);

$minren[5] = '范爷';

$minren[2] = '亦菲，不要架给韩国人好吗？';
```

## 6.1 数组的定义

```
echo '<pre>';

var_dump($minren);

echo '</pre>';

?>
```

执行上面的代码，输出看看结果。

1.用变量名[键] = 新值。就把数组中的值定的值修改了。

### 索引数组的其他声明方式

通过上面的例子，我们学习到了数组的声明。我们来学一下数组的其他声明方式。

#### 一、直接用之前未声明的变量，用变量名后面接中括号的方式声明数组。

```
<?php

//直接写一个变量后面加上中括号，声明变量

$qi[] = '可口可乐';
$qi[10] = '百事';

echo '<pre>';

var_dump($qi);

echo '</pre>';

?>
```

#### 二、每次用array()写的太麻烦了，还可以不用写array哟，更简单。

```
<?php

$minren = [
    '杨幂',
    '王珞丹',
    100 => '刘亦菲',
    '黄圣依',
    '范冰冰'
];

echo '<pre>';

var_dump($minren);

echo '</pre>';

?>
```

上面是另外两种PHP学院的写法。当然，你自己喜欢使用哪一种就使用哪一种来声明。

### 关联数组

索引数组适当的变一下形就出现了关联数组。只要数组里面有一个为字符串的数组，就为关联数组。

通过上例中，我们发现数组的下标只能够为字符串，当然不能够满足我的需求。

假设我想声明一个下标为帅对应李文凯这个值。肯定不能满足我的需求。



## 6.1 数组的定义

那我们来声明一下关联数组。跟索引数组的声明方式完成一样。不过不同的是，必须要指定字符串的下标和必须要使用键值对应符。

```
<?php

//声明一下关联数组
$rela = array(
    '帅' => '李文凯',
    '很帅' => '黄晓明',
    '灰常灰常帅' => '宁泽涛',
    '有男人味的大叔' => '吴秀波',
);

//再来玩玩简洁声明

$drink = [
    '美' => '凤姐',
    '很美' => '芙蓉姐姐',
    'verymei' => '杨幂',
    '心中滴女神呀' => '华妃',
    100 => '孙俪',
    '娘娘',
];

// 输出 $rela
echo '<pre>';

var_dump($rela);

echo '</pre>';

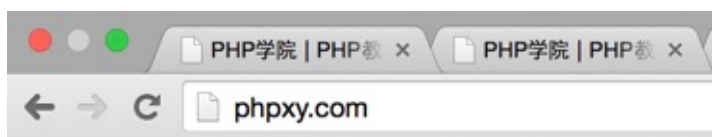
// 输出$drink
echo '<pre>';

var_dump($drink);

echo '</pre>';

?>
```

我们来实验一下看看最终的结果是什么样的：



```
array (size=4)
  '帅' => string '李文凯' (length=9)
  '很帅' => string '黄晓明' (length=9)
  '灰常灰常帅' => string '宁泽涛' (length=9)
  '有男人味的大叔' => string '吴秀波' (length=9)

array (size=6)
  '美' => string '凤姐' (length=6)
  '很美' => string '芙蓉姐姐' (length=12)
  'verymei' => string '杨幂' (length=6)
  '心中滴女神呀' => string '华妃' (length=6)
  100 => string '孙俪' (length=6)
  101 => string '娘娘' (length=6)
```

我们通过实验知道：

## 6.1 数组的定义

1. 声明关联数组是 键名 => 值
2. 在关联数组可以有索引数组的元素
3. 关联数组中的索引数组的元素后再声明了无下标的元素，依然是最大值+1原则。(观察上图中的值为孙俪、娘娘的两个元素)。

### 关联数组的增、删、改

```
<?php
$drink = [
    '美' => '凤姐',
    '很美' => '芙蓉姐姐',
    'verymei' => '杨幂',
    '心中滴女神呀' => '华妃',
    100 => '孙俪',
    '娘娘',
];

//追加方式与索引数组是一样的
$drink['kx'] = '康熙';

//输出试试
echo '<pre>';

var_dump($drink);

echo '</pre>';

//删除一个试试
unset($drink['verymei']);
echo '<pre>';

var_dump($drink);

echo '</pre>';

//将芙蓉姐姐 改成：心里美才是真的美
$drink['很美'] = '心里美才是真的美';

echo '<pre>';

var_dump($drink);

echo '</pre>';

?>
```

你动手做做刚刚的实验，你会发现操作方式与索引的操作方式一样。只不过下标在读取的时候读取的下标PHP学院了。

### 其他的关联数组声明方式

```
<?php

$drink['nf'] = '农夫山泉';

$cocacola = '可口可乐';

//当然可以是变量哟
$drink['k1'] = $cocacola;

$data = array(
    'k1' => $cocacola,

);

?>
```

## 6.1 数组的定义

通过上例我们发现，在关联数组当中也是可以不用array，直接在变量后面接括号。括号里面插入字符串下标，一样也声明成功。

后面插入变量的例子，只不过把字符串变成了变量，当然没有问题。

### 数组当中插入数组

数组可以插入整型、浮点、字符串，那么数组当中可不可以插入数组呢？

当然可以。

在学习数组的时候，我们定义了这样一组名词。

1. 一维数组 数组里面没有其他数组，只有单纯的一些变量或者值。
2. 二维数组 数组里面插入了单层的一个数组，或者多个数组
3. 三维数组 在数组（A）里面插入了一个数组（B），在B数组里面又插入了一层级的数组（C），这种我们就称为三维数组
4. 超过三维的，统统都叫多维数组。

注：索引数组和关联数组当中都可以再插入其他数组，让数组变为多维的。

我们来声明一个一维数组,只有一个维度。索引、关联均可,这个操作大家都很熟。

```
<?php

//一维的索引数组
$data = [1 , 2 , 3 , 4 , 10 => 250];

//一维的关联数组

$srela = [
    'beijing' => '北京',
    'shanghai' => '上海',
    'tj' => '天津',
];

?>

echo '<pre>';
var_dump($srela);
echo '</pre>';

echo '<pre>';
var_dump($data);
echo '</pre>';
```

一维数组大家很熟悉，也不难，就是声明数组嘛。二维也不难，只是在数组里面再插一个或多个数组。二维学好了，多维就学好了。

```
<?php

$person = array(

    'office' => '办公室',

    //注意：插入第一个数组哟
    'family' => array(

        '爸爸',
        '妈妈',
        'yeye' => '爷爷',
```

## 6.1 数组的定义

```
        'nn' => '奶奶',
    ),

    //注意：又插入了一个平级的数组
    'jiaotong' => array(
        '自行车',
        '摩托车',
        '汽车',
        '飞机',
    ),
);

echo '<pre>';
var_dump($person);
echo '</pre>';

?>
```

我们执行一下看看效果：



```
array (size=3)
  'office' => string '办公室' (length=9)
  'family' =>
    array (size=4)
      0 => string '爸爸' (length=6)
      1 => string '妈妈' (length=6)
      'yeye' => string '爷爷' (length=6)
      'nn' => string '奶奶' (length=6)
  'jiaotong' =>
    array (size=4)
      0 => string '自行车' (length=9)
      1 => string '摩托车' (length=9)
      2 => string '汽车' (length=6)
      3 => string '飞机' (length=6)
```

我们发现变量person 是一个数组类型。有三个元素（size=3）。

- 第一个元素为 office 对应的是一个字符串“办公室”，
- 第二个元素为family 里面是一个数组，这个数组又有4个元素，值分别为：爸爸、妈妈、爷爷、奶奶。
- 第三个元素为jiaotong，里面也为一个数组有4个元素，值分别为：自行车、摩托车、汽车、飞机。

好了，二维数组好声明，只要把格式写对了，确定是在一个数组里面再插入一个数组就好。

那如何访问：爷爷和爸爸这两个值呢？

```
<?php
$person = [

    'office' => '办公室',

    //注意：插入第一个数组哟
    'family' => [

        '爸爸',
        '妈妈',
        'yeye' => '爷爷',
        'nn' => '奶奶',
    ],

    //注意：又插入了一个平级的数组
```

## 6.1 数组的定义

```
'jiaotong' => [
    '自行车',
    '摩托车',
    '汽车',
    '飞机',
],
];

//访问“爸爸”这什值
echo $person['family'][0];

echo '<br />-----华丽丽的分割线-----<br />';

//访问“爷爷”这什值
echo $person['family']['yeye'];

echo '<br />-----华丽丽的分割线-----<br />';

//访问“汽车”这什值
echo $person['jiaotong'][2];

?>
```

我们看看结果如下：



上例大家会发现，访问二维数组不过是按照之前的下标读取方式依次向下读取即可。

先写上变量，中括号写上family这个下标，再写上要访问元素的下标。

三维数组我们说了概念，那我们直接上手实验看看效果：

```
<?php

$area = array(

    'china' => array(

        '上海',
        '湖北',
        '天津',
        '北京' => array(
            'hd' => '海淀',
            '朝阳',
            '房山',
            'cp' => '昌平',
        ),

        '广东' => array(
            '深圳',
            '广州',
            '佛山',
            'dg' => '东莞',
```

## 6.1 数组的定义

```
        ),
    ),
    'usa' => array(
        '华盛顿',
        '旧金山',
        '纽约' => array(
            '曼哈顿区',
            '皇后区',
            '布鲁克林区',
        ),
    ),
);

echo '<pre>';
var_dump($area);
echo '</pre>';
?>
```

我们执行一下看看效果：



```
array (size=2)
  'china' =>
    array (size=5)
      0 => string '上海' (length=6)
      1 => string '湖北' (length=6)
      2 => string '天津' (length=6)
      '北京' =>
        array (size=4)
          'hd' => string '海淀' (length=6)
          0 => string '朝阳' (length=6)
          1 => string '房山' (length=6)
          'cp' => string '昌平' (length=6)
      '广东' =>
        array (size=4)
          0 => string '深圳' (length=6)
          1 => string '广州' (length=6)
          2 => string '佛山' (length=6)
          'dg' => string '东莞' (length=6)
  'usa' =>
    array (size=3)
      0 => string '华盛顿' (length=9)
      1 => string '旧金山' (length=9)
      '纽约' =>
        array (size=3)
          0 => string '曼哈顿区' (length=12)
          1 => string '皇后区' (length=9)
          2 => string '布鲁克林区' (length=15)
```

说明：

在变量\$area下有二个数组，一个为china，一个为usa。

在china这个数组里面插入了上海、湖北、天津，又插入了北京和广东。而北京和广东又是一个数组。在北京和广东这两个数组里面分别有不同的元素。

本文档使用 [看云](#) 构建

## 6.1 数组的定义

在这usa 这个数组里面插美国的华盛顿、旧金山和纽约。而纽约下面又是一个数组，说明了纽约下面的几个区。

所以说，三维数组就是在数组里面再插入一个数组(A)，在A数组里面插入一个数组。

我们接下来看看，如何来读取里面的值。

```
<?php

$area = array(

    'china' => array(

        '上海',
        '湖北',
        '天津',
        '北京' => array(
            'hd' => '海淀',
            '朝阳',
            '房山',
            'cp' => '昌平',
        ),
        '广东' => array(
            '深圳',
            '广州',
            '佛山',
            'dg' => '东莞',
        ),
    ),

    'usa' => array(

        '华盛顿',
        '旧金山',
        '纽约' => array(
            '曼哈顿区',
            '皇后区',
            '布鲁克林区',
        ),
    ),

);

//读取华盛顿
echo $area['usa']['0'];

//读取：布鲁克林
echo $area['usa']['纽约'][2];

//读取：昌平
echo $area['china']['北京']['cp'];

//修改cp下标的值改为：西城区

$area['china']['北京']['cp'] = '西城区';

//输出看看原来昌平的值是否发生了变化
echo $area['china']['北京']['cp'];

?>
```

通过上面我们发现数组学习起来不难。

## 6.1 数组的定义

学习多维数组的难点：

注意格式，将每个维度的换行、缩进弄整齐。就不容易出错了。

### 【切记】

数组元素间的分割符为逗号，在数组中插入数组的时候，不要在结尾处写成了分号（；）

下图是错误的截图展示：

```
3 $area = array(  
4     'china' => array(  
5         '上海',  
6         '湖北',  
7         '天津',  
8         '北京' => array(  
9             'hd' => '海淀',  
10            '朝阳',  
11            '房山',  
12            'cp' => '昌平',  
13        );  
14     '广东' => array(  
15         '深圳',  
16         '广州',  
17         '佛山',  
18         'dg' => '东莞',  
19     );  
20 );  
21  
22 'usa' => array(  
23     '华盛顿',  
24     '旧金山',  
25     '纽约' => array(  
26         '曼哈顿区',  
27         '皇后区',  
28         '布鲁克林区',  
29     ),  
30 ),  
31 );  
32 )
```



## 6.2 数组的操作

---

6.2.1 数组的计算

6.2.2 for循环遍历索引数组

6.2.3 foreach遍历关联数组

6.2.4 list、each函数遍历数组

6.2.5 常用操作数组函数

## 6.2.1 数组的计算

数组是我们最常用到的类型，那如何计算某个一维数组的个数呢。其实我们可以用到我们之前学过的数学函数里面的一个：count()。

我们来看看count函数的用法：

```
int count ( mixed $变量)
```

注：

1. 参数\$变量 要求是一个数组或者一个可以被统计的对象

那我们可以来尝试使用统计函数来统计一下数组的个数。

```
<?php
$a[0] = 1;
$a[1] = 3;
$a[2] = 5;
$result = count($a);
// $result == 3

$b[0] = '迪奥和奥迪我都爱';
$b[5] = '努力开创未来';
$b[10] = '为了未来而努力';
$result = count($b);

$data = [
    'baidu' => '百度',
    'ali' => '阿里',
    'tencent' => '腾讯',
];
echo count($data);

$erwei = [
    [
        'baidu' => '百度',
        'ali' => '阿里',
        'tencent' => '腾讯',
    ],
    [
        'netease' => '网易',
        'sohu' => '搜狐',
        'sina' => '新浪',
    ]
];

//试试输出一个二维数组个数
echo count($erwei);

//试试输出二维数组中某个元素的个数
echo count($erwei[1]);
?>
```

通过上例，我们发现即可以输出索引数组，也可以输出关联数组的个数。

如果是一个二维数组，这个函数只会统计当前的这一个维度的数组元素个数。如上例中的：\$erwei。所以count(\$erwei)的时候结果是2。而count(\$erwei[1])的时候，结果为3。



## 6.2.2 for循环遍历索引数组

遍历二字，从字面解释就是一个接一个全读访问一次，显示出来。

因为for循环是一个单纯的计数型循环，而索引数组的下标为整型的数值。因此，我们可以通过for循环来遍历索引数组。

我们知道索引数组下标为整型。我们定义下面的一个数组：

```
<?php

//声明一个数组，值为1到10
$num = array(1,2,3,4,5,6,7,8,9,10);

//按照索引数组的特点，下标从0开始。所以1的下标为0，10的下标为9
echo $num[0].'<br />';
echo $num[9].'<br />';

//我们可以得到数组中元素的总个数，为10
echo count($num);

//遍历这个索引数组的话，我们就可以定义一个变量为$i
//$i 的值为0，从0开始
//可以设定一个循环条件为：$i 在下标的(9)最大值之内循环
for($i = 0 ; $i < count($num) ; $i++){

    echo $num[$i].'<br />';

}

?>
```

通过上面的例子，我们就把数组进行了循环。

因为下标是从0开始的，定义*\$i*=0。每次循环的时候让*\$i* 加1，但是必须要小于10，因为数组下标的最大值为9。

这样，我们就学会了对索引连续下标数组的遍历。

那么问题来了：

那关联数组怎么办？如果索引数组的下标不连续怎么办？

答：咱们下个章节讲，小伙子别急嘛。

## 6.2.3 foreach遍历关联数组

### foreach的基本语法

我们通过上一章的内容学会了遍历连续下标的索引数组。可是，我们发现我们遍历不了关联数组，也遍历不了下标不连续的索引数组。

那我们其实在学循环的时候，有一个布尔型循环是专门用来循环数组的。这个循环的基本语法就是foreach基本语法。

语法格式如下：

```
foreach( 要循环的数组变量 as [键变量 =>] 值变量){

    //循环的结构体

}
```

### 遍历关联数组

这是一个固定用法，将要循环的数组放进去。

as 是一个固定的关键字

后面的键变量是可选的，随意定义一个变量，每次循环的时候，foreach这个语法会把键取出来赋值到键变量里面  
后面的值变量是必填的。每次循环的时候，会把值放到值变量里面。

我们下面用代码来举例子，加强对这个语法的理解。

```
<?php

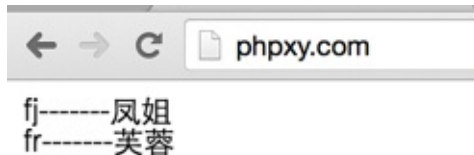
$data = [
    'fj' => '凤姐',
    'fr' => '芙蓉',
];

foreach($data as $key => $value){
    echo $key . '-----' . $value . '<br />';
}

//如果我们只想读取值的话，就可以把下面的$key => 给删除掉，读取的时候，就只读取值了。做完上面的实验，你可以打开下面的代码再实验几次。

/*
foreach($data as $value){
    echo $value . '<br />';
}
*/
?>
```

我们运行一下看看结果：



通过上面的运行结果我们得出下面的结果：

1. 每次循环的时候，把下标赋值给了变量\$key，把值的变量赋值给了变量\$value
2. 循环一次读一次键和值。如上例中，读完“凤姐”再读取“芙蓉”，读到最后，发现没有可以读取的数组元素后，停止循环遍历数据。

注意：\$key 和\$value并不是变量名必须得为这两个名字。你命名为其他的也可以，如 \$kai => \$wen是一样的。你要清楚键赋值给了哪个变量，值赋值给了另外的哪个变量。

### 遍历索引数组

foreach还挺好学的。因此，我们可以通过foreach遍历连续的索引数组，如下例：

```
<?php

$data = array(
    0 => '中国',
    100 => '美国',
    20 => '韩国',
    300 => '德国',
);

//待会儿可以自己做做实验，循环遍历一下下面的这个数组
//$data = array(1,2,3,4,5,6,7,8,9,10);

foreach($data as $k => $v){

    echo $k . '-----' . $v . '<br />';

}

?>
```

运行看一下结果：



按照foreach的结果推理与刚刚做的关联数组的结果是一样的。

不同的是不连续的索引数组。每读一次数组的元素，将当次循环的下标赋值给了变量\$k，值赋值给了变量\$v。每读将键和值输出，然后显示出来。循环一次向后移动一次下标。读取至最后，退出执行。

### 遍历多维数组

数组当中还有一个数组我们该怎么遍历呢？我们来做一个实验：

```
<?php

$data = array(

    0 => array(
        '中国' => 'china',
        '美国' => 'usa',
```

### 6.2.3 foreach遍历关联数组

```
        '德国' => ' Germany',
    ),

    1 => array(
        '湖北' => 'hubei',
        '河北' => 'hebei',
        '山东' => 'shandong',
        '山西' => 'sanxi',
    ),

);

//注：我们在使用foreach循环时，第一次循环将键为0和键为1的两个数组赋值给一个变量($value)。然后，再套一个循环遍历这个$value变量，$value中的值取出来，赋值给$k和$v。

foreach($data as $value){

    //第一次循环把国家的数组赋值给了$value
    //第二次循环把中国的省份的数组又赋值给了$value
    //因此，我在循环的时候把$value再遍历一次

    foreach($value as $k => $v){
        echo $k . '-----' . $v . '<br />';
    }

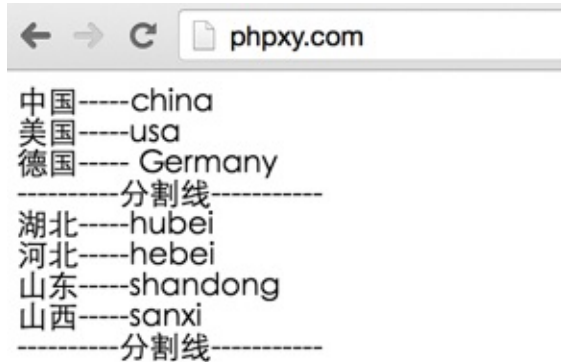
    //为了看的更清晰，我在中间加上华美丽的分割线方便你来分析

    echo '-----分割线-----<br />';

}

?>
```

结果就出来了哟：



总结：

1. 第一次循环的时候，将数组赋值给了\$value，然后用foreach循环\$value。将二维的子数组中的键给到\$k，值赋值给变量\$v。
2. 第一次循环退出子数组的循环，执行后续代码显示分割线。
3. 依此类推，第二次循环也是这样。

## 作业

将如下数组：

```
<?php
$arr=array(
    '教学部'=>array(
```

```
        array('李某','18','人妖'),
        array('高某','20','男'),
        array('张某','21','妖人'),
    ),
    '宣传部'=>array(
        array('李某','18','人妖'),
        array('高某','20','男'),
        array('张某','21','妖人'),
    ),
    '财务部'=>array(
        array('李某','18','人妖'),
        array('高某','20','男'),
        array('张某','21','妖人'),
    ),
);
?>
```

遍历显示出来，效果如下：

## 教学部

李某	18	人妖
高某	20	男
张某	21	妖人

## 宣传部

李某	18	人妖
高某	20	男
张某	21	妖人

## 财务部

李某	18	人妖
高某	20	男
张某	21	妖人



## 6.2.4 list、each函数遍历数组

我们来研究两个比较有意思的数组函数。这两个函数学起来不难。但是，有些同学会遇到一点点小困难。困难在于，找到list函数和each函数的操作特点。

### list函数

我们先来讲list函数：

```
list ( mixed $变量1 [, mixed $变量n ] )
```

它的功能：将索引数组下标为0的对应我变量1，下标1的对应变量2，依此类推。

我们来通过实验来看一下：

```
<?php

list($one , $two , $three) = array('张三' , '李四' , '王五');

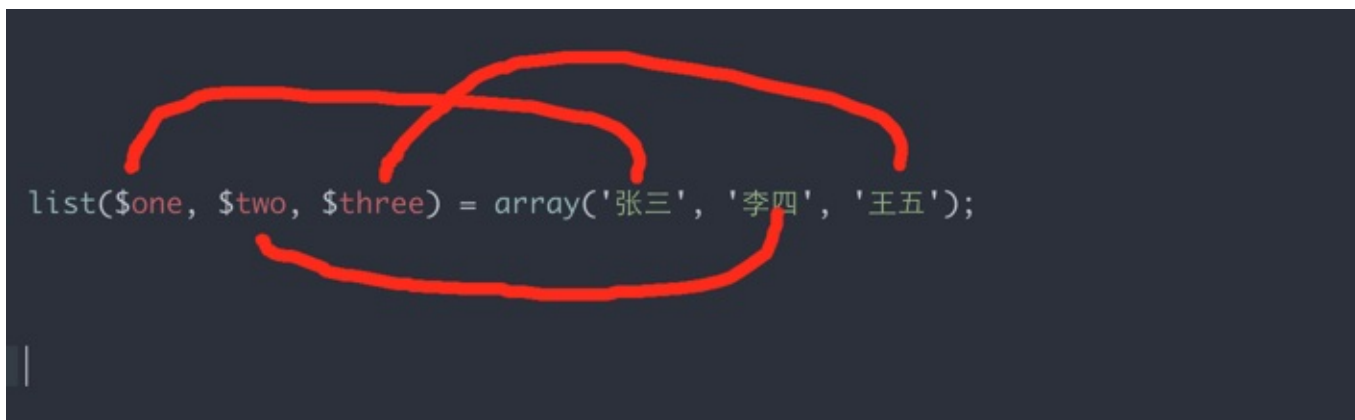
//再次声明：单引号不注释变量，所以输出的是字符串$one
echo '$one----'.$one.'<br />';
echo '$two----'.$two.'<br />';
echo '$three----'.$three.'<br />';

?>
```

我们来看一下实验结果：



分析结果如图：



结论：

本文档使用 [看云](#) 构建

## 6.2.4 list、each函数遍历数组

1. 将下标为0的张三赋值给了\$one
2. 将下标为1的李四赋值给了\$two
3. 将下标为2的王五赋值给了\$three

因此，我们知道了list的功能就是从左到右，一一对应索引数组从0开始的下标值。

list的另外一种用法：

```
<?php

list( , , $three) = array('张三' , '李四' , '王五');

echo '$one----' . $one . '<br />';
echo '$two----' . $two . '<br />';
echo '$three----' . $three . '<br />';

?>
```

运行结果：



结论：

1. list当中的第一、二个放变量的地方留空，我只写了\$three。
2. 按照一一对应原则，张三和李四没有变量可以对应。
3. 所以只有王五有变量对应

请严格记住：索引数组的一一对应原则。list第一个变量对应的是下标为0的数组元素，下标为1的对应的是list里面的第二个数组元素。

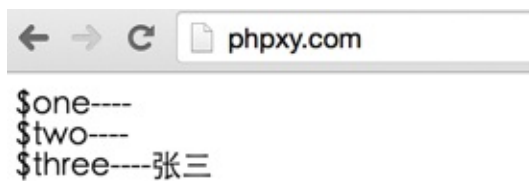
请猜猜下面的结果是多少？为什么？

```
<?php
list($one, $two, $three) = array(2 => '张三', '李四', '王五');

echo '$one----' . $one . '<br />';
echo '$two----' . $two . '<br />';
echo '$three----' . $three . '<br />';

?>
```

运行结果如下图：



总结：

1. 因为是一一对应原则，\$one找不到下标为0的数组元素，\$two找不到下标为1的数组元素，只有\$three找到了下标为2的数组元素
2. 在list(\$one, \$two, \$three)，我只写了三个变量。对应完成，无需再对应后面的变量了，丢弃李四和王五。

## each函数

each 函数的规律性上面就更有特点了，并且比较有趣。

```
array each ( array &$array )
```

功能：传入一个数组。它会将其中的一个元素拆为个新数组。每次执行这样操作一个元素。执行一次先后移动一次，同样的方式操作下一个数组元素。执行到最后，返回false。

我们先来看看each是怎么操作的数组元素。

```
<?php

//定义一个变量叫$kongjie(空姐)
$kongjie=[
    'gao'=>'穿黑衣服的',
    'shou'=>'腿特别长特别细',
    'mei'=>'好白',
    'pl'=>'五官端正',
    'type'=>'那就是女神',
    '我是屌丝不敢跟女神搭讪'
];

//第一次each
$data = each($kongjie);

echo '<pre>';
var_dump($data);
echo '</pre>';

?>
```

我们来看看第一次执行each的结果：

```

array (size=4)
  1 => string '穿黑衣服的' (length=15)
  'value' => string '穿黑衣服的' (length=15)
  0 => string 'gao' (length=3)
  'key' => string 'gao' (length=3)

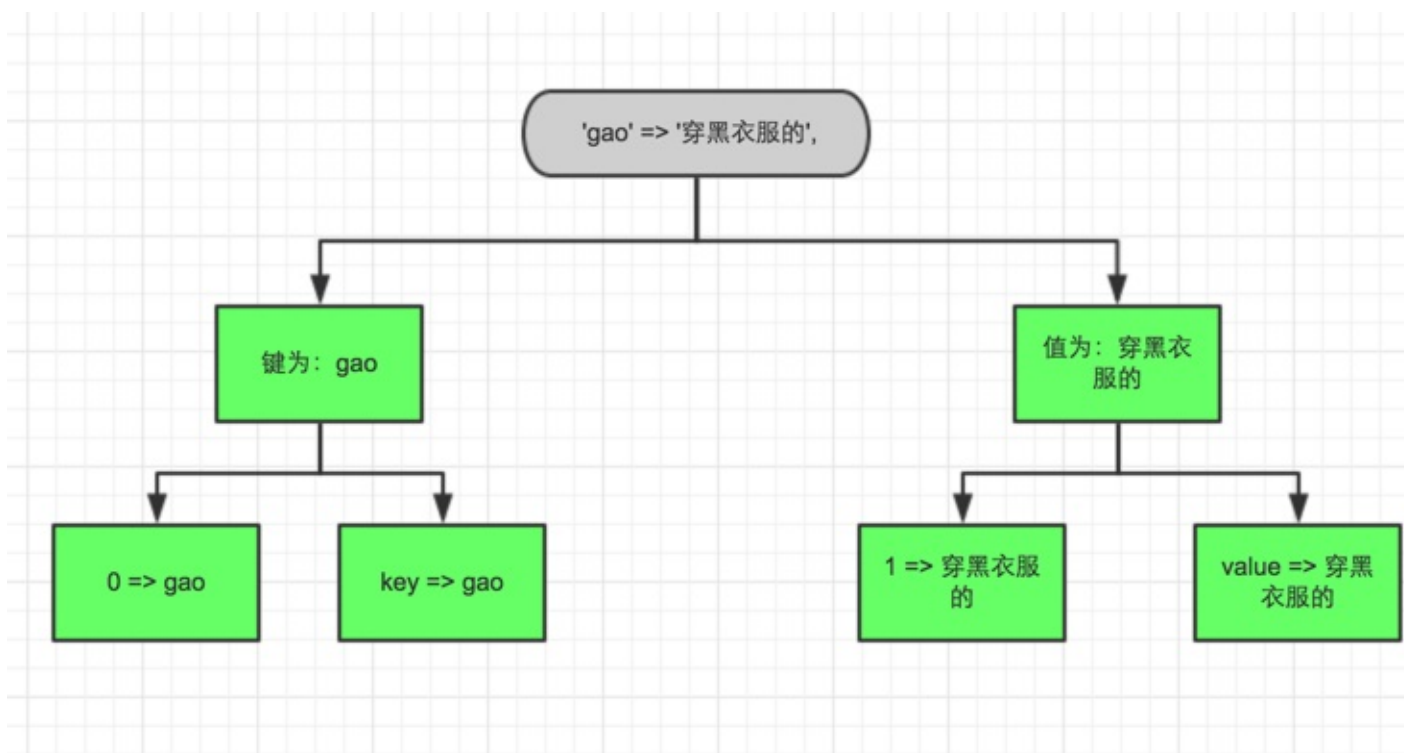
```

总结：

1. 读取了\$kongjie中的第一个元素，将第一个元素（'gao'=>'穿黑衣服的'）分解开了。

1. 分解后第一个元素变成了一个新数组。
2. 在新数组里面，将原值（穿黑衣服的）放了索引下标1里面，同时放到了关联下标value里面。
3. 在新数组里面，将原键（gao），放到了关联下标key里面，放到了索引下标0里面。

我们用图来表示：



这样我们一下子就看明白了。

接下来我们说说each的另外一个特性。读一次，向后移动一个元素。

```

<?php
//定义一个变量叫$kongjie(空姐)
$kongjie=[
    'gao'=>'穿黑衣服的',
    'shou'=>'退特别长特别细',
    'mei'=>'好白',
];

```

## 6.2.4 list、each函数遍历数组

```
//第一次each
$data = each($kongjie);

echo '<pre>';
var_dump($data);
echo '</pre>';

echo '-----华丽丽分割线-----<br />';

//第2次each
$data = each($kongjie);

echo '<pre>';
var_dump($data);
echo '</pre>';

echo '-----华丽丽分割线-----<br />';

//第3次each【执行到了最后一个元素了】
$data = each($kongjie);

echo '<pre>';
var_dump($data);
echo '</pre>';

echo '-----华丽丽分割线-----<br />';

//第4次【此时，后面已没有可操作的元素了，看返回什么】
$data = each($kongjie);

echo '<pre>';
var_dump($data);
echo '</pre>';

echo '-----华丽丽分割线-----<br />';

?>
```

执行结果：

```

← → ↺ phpxy.com

array (size=4)
  1 => string '穿黑衣服的' (length=15)
  'value' => string '穿黑衣服的' (length=15)
  0 => string 'gao' (length=3)
  'key' => string 'gao' (length=3)

-----华丽丽分割线-----

array (size=4)
  1 => string '退特别长特别细' (length=21)
  'value' => string '退特别长特别细' (length=21)
  0 => string 'shou' (length=4)
  'key' => string 'shou' (length=4)

-----华丽丽分割线-----

array (size=4)
  1 => string '好白' (length=6)
  'value' => string '好白' (length=6)
  0 => string 'mei' (length=3)
  'key' => string 'mei' (length=3)

-----华丽丽分割线-----

boolean false

-----华丽丽分割线-----

```

总结：

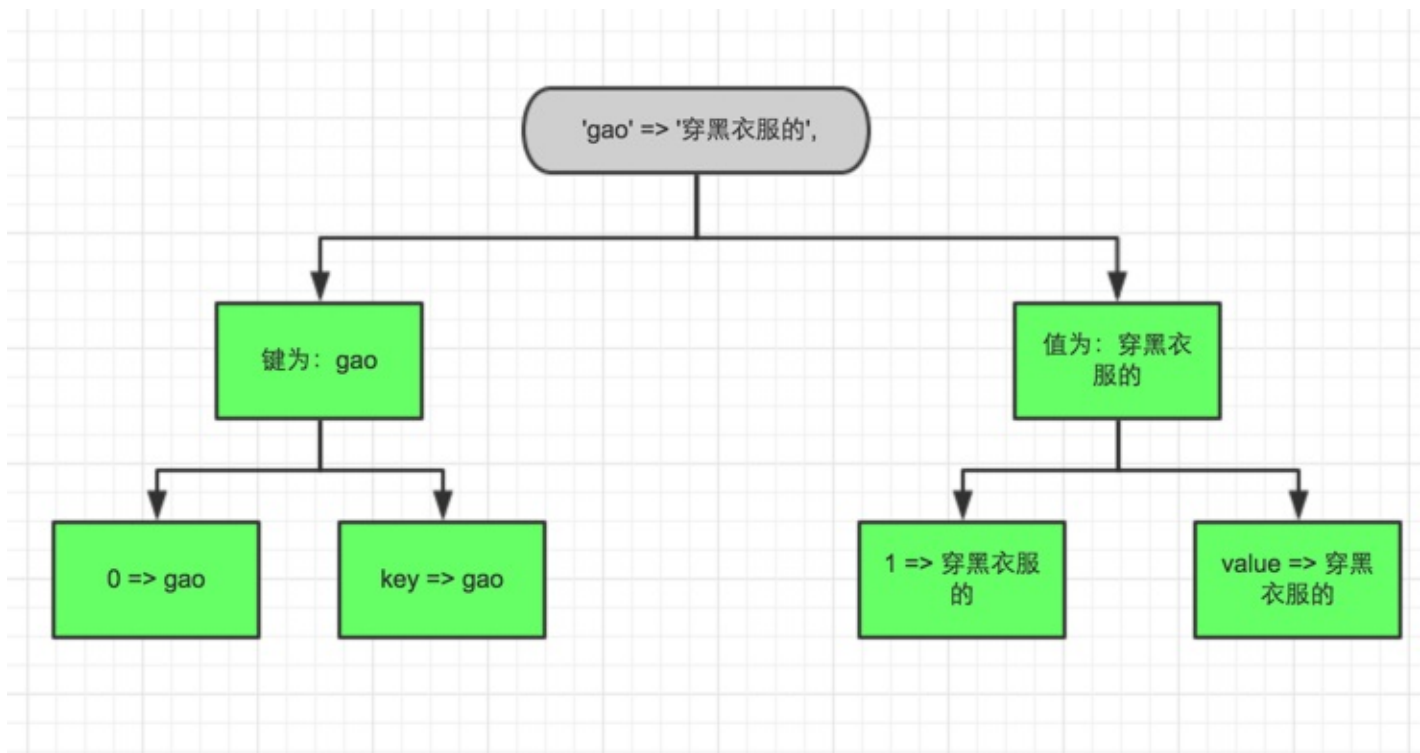
1. 读一次向后移动一次【可以想象有一个记录的箭头在移动】，将其中的每个元素拆解成一个新数组
2. 读取到最后，没有可操作的元素了，所以返回了false。

## list 和 each 配合

我们知道了list的特性，也知道了each的特性。那list是不是可以和each配合起来一起来完成一些工作呢？

```
list($key,$value) = each($array);
```

我们来看之前说到的这个图：



而list中的第一个变量会去找索引下面0的元素去赋值，第二个变量会找索引下标为1的元素对变量赋值。

我们来看看下面例子：

```

<?php

//定义一个变量叫$kongjie(空姐)
$kongjie=[
    'gao'=>'穿黑衣服的',
    'shou'=>'退特别长特别细',
    'mei'=>'好白',
];

list($key,$value) = each($kongjie);

echo $key. '-----' . $value . '<br />';

?>
  
```

运行结果如下：



总结：

1. each 把变量拆成了4个元素
2. 而list把0 =>gao 赋值给了变量\$key
3. list把1 => 穿黑衣服的 赋值给了变量 \$value

each到最后会返回false，因此我可以用布尔型循环while来配合完成数组的循环。

把上面的代码，轻轻一改就实现了如下效果：

```
<?php

//定义一个变量叫$kongjie(空姐)
$kongjie=[
    'gao'=>'穿黑衣服的',
    'shou'=>'退特别长特别细',
    'mei'=>'好白',
];

while(list($key,$value) = each($kongjie)){

    echo $key. '-----' . $value . '<br />';

}

?>
```

执行演示：



总结：

1. 循环一次，执行一次each，执行代码，然后向后移动一个元素
2. 执行到最后返回false，因此停止执行。
3. 可以通过each和list配合实现foreach一样的效果。

作业：

使用list和each配合，将如下数组：

```
<?php
$arr=array(
    '教学部'=>array(
        array('李某','18','人妖'),
        array('高某','20','男'),
        array('张某','21','妖人'),
    ),
    '宣传部'=>array(
        array('李某','18','人妖'),
        array('高某','20','男'),
        array('张某','21','妖人'),
    ),
    '财务部'=>array(
        array('李某','18','人妖'),
        array('高某','20','男'),
        array('张某','21','妖人'),
    ),
);
?>
```



遍历显示出来，效果如下：

## 教学部

李某	18	人妖
高某	20	男
张某	21	妖人

## 宣传部

李某	18	人妖
高某	20	男
张某	21	妖人

## 财务部

李某	18	人妖
高某	20	男
张某	21	妖人

# 6.2.5 常用操作数组函数

我们有很多操作数组的元素，我们这一节先讲一些。在6.3里面我们会总结更多的数组常用函数。

下面的几个主要是移动数组指针和压入弹出数组元素的和个函数。

函数	功能
array_shift	弹出数组中的第一个元素
array_unshift	在数组的开始处压入元素
array_push	向数组的末尾处压入元素
array_pop	弹出数组末尾的最后一个元素
current	读出指针当前位置的值
key	读出指针当前位置的键
next	指针向下移
prev	向上移
reset	指针到开始处
end	指针到结束处

## array\_shift

mixed array\_shift ( array &\$array )

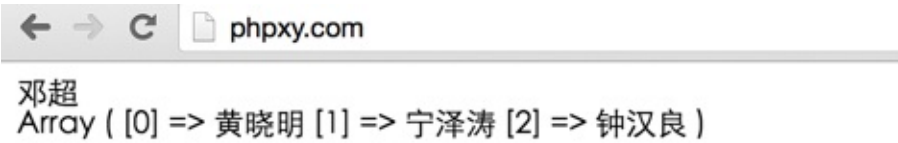
功能：弹出数组中的第一个元素

```
<?php
$mingren = array("邓超", "黄晓明", "宁泽涛", "钟汉良");
$dc = array_shift($mingren);

echo $dc . '<br />';

print_r($mingren);
?>
```

执行结果：



结论：

- 1. 将第一个数组元素弹出，改变了原数组的结果
- 2. 弹出的值赋值给了\$dc

## array\_unshift

```
int array_unshift ( array &$数组 , mixed $值1 [, mixed $... ] )
```

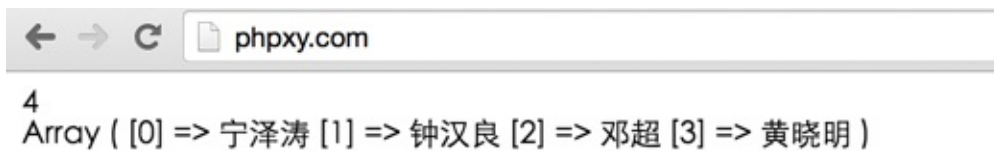
功能：向指数组的开始处压入一个或多个元素，返回的是总个数。

```
<?php
$mingren = array("邓超", "黄晓明");
$dc = array_unshift($mingren, "宁泽涛", "钟汉良");

echo $dc . '<br />';

print_r($mingren);
?>
```

执行结果如下：



```
4
Array ( [0] => 宁泽涛 [1] => 钟汉良 [2] => 邓超 [3] => 黄晓明 )
```

## array\_pop

```
mixed array_pop ( array &$array )
```

功能：弹出数组末尾的一个元素

```
<?php
$mingren = array("邓超", "黄晓明", "宁泽涛", "钟汉良");
$dc = array_pop($mingren);

echo $dc . '<br />';

print_r($mingren);
?>
```

执行结果：



```
钟汉良
Array ( [0] => 邓超 [1] => 黄晓明 [2] => 宁泽涛 )
```

## array\_push

```
int array_push ( array &$array , mixed $value1 [, mixed $... ] )
```

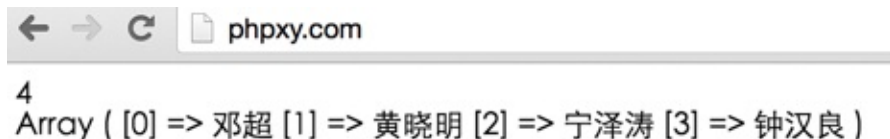
功能：向指数组末尾处压入一个或多个元素，返回的是总个数。

```
<?php
$mingren = array("邓超", "黄晓明");
$dc = array_push($mingren, "宁泽涛", "钟汉良");

echo $dc . '<br />';

print_r($mingren);
?>
```

执行结果：



4  
Array ( [0] => 邓超 [1] => 黄晓明 [2] => 宁泽涛 [3] => 钟汉良 )

## current,key,prev,next,reset 功能演示

这几个函数功能已经说的很清楚了。我们通过代码来进行演示：

```
<?php
$t=array(
    '我们',
    'yy'=>'永远',
    'dbg'=>'需要不断奋进',
    'djn'=>'才能开创未来'
);

//读取数组的值
echo current($t). '<br />';
//读取数组的键
echo key($t). '<br />';

//向后移动一下
next($t);

//再读值和键
echo current($t). '<br />';

echo key($t). '<br />';

//向后移动一下
next($t);
echo current($t). '<br />';

echo key($t). '<br />';

//向前移动一下
prev($t);
echo current($t). '<br />';
echo key($t). '<br />';

//移到末尾
end($t);
echo current($t). '<br />';
echo key($t). '<br />';

//移至开始处
reset($t);
echo current($t). '<br />';

echo key($t). '<br />';
```

```
//销毁数组  
unset($t);  
var_dump($t);  
?>
```

# 6.3 数组的常用函数

因为数组的函数特别多，很多全是英文标识的。还有一些数组的函数不太常用。因此，我们把最最常用的数组函数跟大家总结好了，记大家务必提到哪个函数的时候，就能随时想起来。直接使用。

请将这些函数对着手册的示例多用几次。要求达到：**默写级别**。

以后在看和写任何代码的时候，不用再反映半天。上手就直接使用这些函数，是一个程序员的基本素养。

这些函数，也是面试中基础面试中最爱问到的问题。

函数名	功能
array_combine()	生成一个数组,用一个数组的值作为键名,另一个数组值作为值
range()	创建并返回一个包含指定范围的元素的数组。
compact()	创建一个由参数所带变量组成的数组
array_fill()	用给定的值生成数组
array_chunk()	把一个数组分割为新的数组块
array_merge()	把两个或多个数组合并为一个数组
array_slice()	在数组中根据条件取出一段值，并返回
array_diff()	返回两个数组的差集数组
array_search()	在数组中搜索给定的值，如果成功则返回相应的键名
array_splice()	把数组中的一部分去掉并用其它值取代
array_sum()	计算数组中所有值的和
in_array()	检查数组中是否存在某个值
array_key_exists()	检查给定的键名或索引是否存在于数组中
shuffle()	将数组打乱,保留键值
count()	计算数组中的单元数目或对象中的属性个数
array_flip()	返回一个键值反转后的数组
array_keys()	返回数组所有的键,组成一个数组
array_values()	返回数组中所有值，组成一个数组
array_reverse()	返回一个元素顺序相反的数组
array_count_values()	统计数组中所有的值出现的次数
array_rand()	从数组中随机抽取一个或多个元素,注意是键名
array_unique()	删除重复值，返回剩余数组
sort()	按升序对给定数组的值排序,不保留键名
rsort()	对数组逆向排序,不保留键名
asort()	对数组排序,保持索引关系
arsort()	对数组逆向排序,保持索引关系
ksort()	按键名对数组排序

ksort()	将数组按照键逆向排序
natsort()	用自然顺序算法对数组中的元素排序
natcasesort()	自然排序,不区分大小写
array_filter()	去掉数组中的空元素或者预定元素
extract	将键变为变量名，将值变为变量值

## 07. PHP中的正则表达式

---

我们常说这么一句话：

世界上最难懂的是道士的鬼符和医生的处方

其实我们还要加上一句：

世界上最难懂的是道士的鬼符、医生的处方和程序员的正则表达式。

我们先看一个正则表达式的样子：

```
/^a-z@([a-z0-9] [_]?[a-z0-9]+)+[.][a-z]{2,3}([.][a-z]{2})?$/i
```

呵呵，看着就觉得揪心。提前声明，别被吓着，学习一下后你会发现没那么困难。

正则表达式我们其实之前经常看到，它主要用在以下一些地方：

1. 匹配邮箱、手机号码、验证码
  2. 替换敏感的关键词。例如：涉及政治和骂人的话
  3. 文章采集。
  4. 早期的表情替换技术,ubb文件编码、markdown编辑器替换等
  5. 以后自己写模板引擎也需要用到正则表达式
- 其他....

我们学习本章，先通过一点一点来去学习和了解正则表达式需要掌握的基本技术。



# 7.1 正则表达式的定界符

在学习正则表达式前，我们先要学习正则表达式的定界符。

**定界符，就是定一个边界，边界已内的就是正则表达式。**

PHP的正则表达式定界符的规定如下：

定界符，不能用a-zA-Z0-9\ 其他的都可以用。必须成对出现，有开始就有结束。

我们来举几个例子：

例子	说明
/中间写正则/	正确
\$中间写正则\$	正确
%中间写正则%	正确
^中间写正则^	正确
@中间写正则@	正确
(中间写正则)	错误
A中间写正则A	错误

注：\ 是转义字符，如果在以后正则表达式里面需要匹配/，如下图：

///

这个时候真要匹配/ 的时候，需要把定界符里面的/ 用转义字符转义一下，写成下面的例子：

/\\/

如果你觉得麻烦，遇到这种需要转义的字符的时候可以把两个正斜线（//）定界，改为其他的定界符（##）。

## 7.2 正则表达式中的原子

### 原子

原子是正则表达式里面的最小单位，原子说白了就是需要匹配的内容。一个成立的正则表达式当中必须最少要有一个原子。

所有可见不可见的字符就是原子

说明：我们见到的空格、回车、换行、0-9、A-Za-z、中文、标点符号、特殊符号全为原子。

在做原子的实例前我们先来讲解一个函数, preg\_match:

```
int preg_match ( string $正则 , string $字符串 [, array &$amp;结果] )
```

功能：根据\$正则变量，匹配\$字符串变量。如果存在则返回匹配的个数，把匹配到的结果放到\$结果变量里。如果没有匹配到结果返回0。

注：上面是preg\_match常用的主要几个参数。我在上面将另外几个参数没有列出来。因为，另外两个参数太不常用了。

我们来通过实验来证明：

```
<?php
//定义一个变量叫zz，放正则表达式。为了方便大家记忆，如果你英文比较ok，建议把变量名还是写成英文的$pattern。
```

```
<?php
$zz = '/a/';

$string = 'ddfdjji2jfvkwkfi24';

if(preg_match($zz, $string, $matches)){
    echo '匹配到了，结果为：';
    var_dump($matches);
}else{
    echo '没有匹配到';
}

?>
```

因为我希望的是匹配a，而\$string当是是不存在a的，所以不成功。

```
<?php
$zz = '/wq/';

$string = 'ssssswqaaaaa';

if(preg_match($zz, $string, $matches)){
    echo '匹配到了，结果为：';
    var_dump($matches);
}else{
    echo '没有匹配到';
}

?>
```

上面的字符串中s后存在wq，因此匹配成功。

接下来我们匹配一个空格试试：

```
<?php
$zz = '/ /';

$string = 'ssssw aaaa';

if(preg_match($zz, $string, $matches)){
    echo '匹配到了，结果为：';
    var_dump($matches);
}else{
    echo '没有匹配到';
}

?>
```

执行结果如下：



因上，\$string这个变量的w字符后存在一个空格。所以匹配成功，输出了字符串类型，长度为1。只不过我们的肉眼不可见，看不到这个字符串而已。

特殊标识的原子

原子	说明
\d	匹配一个0-9
\D	除了0-9以外的所有字符
\w	a-zA-Z0-9_
\W	除了0-9A-Za-z_以外的所有字符
\s	匹配所有空白字符\n \t \r 空格
\S	匹配所有非空白字符
[ ]	指定范围的原子

这个个需要记住，最好达到默写级别。记忆的时候成对记忆，\d是匹配一个0-9，那么\D 就是除了0-9以外的所有字符。上面已经说明的很清楚了，我们进行实验一步一步对这些进行学习。

请你在学习的时候，对于这些原子务必达到默写级别。因为，我们以后做实验的时候，一点一点你就学会了。

\d匹配一个0-9

```
<?php
$zz = '/\d/';

$string = '我爱喝9你爱不爱喝';

if(preg_match($zz, $string, $matches)){
```

## 7.2 正则表达式中的原子

```
        echo '匹配到了，结果为：';
        var_dump($matches);
    }else{
        echo '没有匹配到';
    }

    ?>
```

### \D匹配一个非0-9的值

```
<?php
$zz = '/\D/';

$string = '121243中23453453';

if(preg_match($zz, $string, $matches)){
    echo '匹配到了，结果为：';
    var_dump($matches);
}else{
    echo '没有匹配到';
}

?>
```

匹配成功，匹配到了中。因为它不是0-9之间的字符。

### \w匹配一个a-zA-Z0-9\_

```
<?php
$zz = '/\w/';

$string = '新中_国万岁呀万岁';

if(preg_match($zz, $string, $matches)){
    echo '匹配到了，结果为：';
    var_dump($matches);
}else{
    echo '没有匹配到';
}

?>
```

匹配成功，匹配到了下划线。

### \W匹配一个非a-zA-Z0-9\_

```
<?php
$zz = '/\W/';

$string = 'afasABCWEQR44231284737';

if(preg_match($zz, $string, $matches)){
    echo '匹配到了，结果为：';
    var_dump($matches);
}else{
    echo '没有匹配到';
}

?>
```

匹配失败。因为，上面上面全是a-zA-Z0-9，没有非a-zA-Z0-9。

### \s 匹配所有空白字符\n \t \r 空格

## 7.2 正则表达式中的原子

```
<?php
$zz = '/\s/';

$string = "中国万
岁";

if(preg_match($zz, $string, $matches)){
    echo '匹配到了, 结果为:';
    var_dump($matches);
}else{
    echo '没有匹配到';
}

?>
```

匹配成功，因为有一个回车。

### \S 非空字符

```
<?php
$zz = '/\s/';

$string = "
    a    ";

if(preg_match($zz, $string, $matches)){
    echo '匹配到了, 结果为:';
    var_dump($matches);
}else{
    echo '没有匹配到';
}

?>
```

匹配成功。虽然上面有空格，回车和缩进。但是，有一个非空白字符a。因此，匹配成功。

### [] 指定范围的原子

```
<?php

$zz = '/[0-5]\w+/';

$string = '6a';

$string1 = '1C';

if(preg_match($zz, $string, $matches)){
    echo '匹配到了, 结果为:';
    var_dump($matches);
}else{
    echo '没有匹配到';
}

?>
```

结论：

上例中0-5匹配\$string失败，而\$string1成功。因为，\$string中的第一个数值为6，不在[0-5]的范围之内。

```
<?php

$zz = '/[a-zA-Z0-9_]\w/';

$string = 'ab';
```

## 7.2 正则表达式中的原子

```
$string1 = '9A';

if(preg_match($zz, $string, $matches)){
    echo '匹配到了, 结果为:';
    var_dump($matches);
}else{
    echo '没有匹配到';
}

?>
```

结论：

\$string和\$string1都匹配成功。因为\w就是[a-zA-Z0-9\_]

```
<?php

$zz = '/[abc]\d+/';

$string = 'a9';

$string1 = 'b1';

$string2 = 'c5';

$string3 = 'd4';

if(preg_match($zz, $string, $matches)){
    echo '匹配到了, 结果为:';
    var_dump($matches);
}else{
    echo '没有匹配到';
}

?>
```

结论：

\$string、\$string1、\$string2匹配成功，而\$string3不成功。因为\$string3超过了[abc]的范围，它是从d开始的。

### [^ 字符] 不匹配指定区间的字符

```
<?php

$zz = '/[^0-9A-Za-z_]/';

$string = 'aaaaab311dd';

$string1 = '!$@!#%$#^##!';

if(preg_match($zz, $string, $matches)){
    echo '匹配到了, 结果为:';
    var_dump($matches);
}else{
    echo '没有匹配到';
}

?>
```

结论：

1. 匹配\$string不成功，但是匹配\$string1的时候成功。因为中括号里面有个抑扬符。

2. ^ 抑扬符在中括号里面的作用是不准以中括号里面的字符进行匹配。

总结：

原子	等价式
\w	[a-zA-Z0-9_]
\W	[^a-zA-Z0-9_]
\d	[0-9]
\D	[^0-9]
\s	[ \t\n\f\r]
\S	[^ \t\n\f\r]

# 7.3 正则表示中的元字符

## 元字符

抛出问题：\d 代表匹配一个字符。而我现在想要匹配十个八个，任意多个数字肿么办？

这个时候我们就要用到元字符。在使用原子的时候，发现只能够匹配一个字符，可是要匹配多个字符就出现了问题。这个时候，我们需要借助元字符来帮我们修饰原子，实现更多的功能。

先不要被下面的这些给吓到。我们一点一点做实验，就全明白了。主要是这几个得多用用。自己准备个小卡片，帮助自己记忆是最好的。

我们来看：

元字符	功能说明
*	是代表匹配前面的一个原子，匹配0次或者任意多次前面的字符。
+	匹配一次或多前前面的一个字符
?	前面的字符可有可无【可选】 有或没有
.	更标准一些应该把点算作原子。匹配除了\n以外的所有字符
	或者。注：它的优先级最低了。
^	必须要以抑扬符之后的字符串开始
\$	必须要以\$之前的字符结尾
\b	词边界
\B	非边界
{m}	有且只能出现m次
{n,m}	可以出现n到m次
{m,}	至少m次，最大次数不限制
()	改变优先级或者将某个字符串视为一个整体，匹配到的数据取出来也可以使用它

### + 匹配最少1次前面的字符

```
<?php
$zz = '/\d+/';

$string = "迪奥和奥迪250都是我最爱";

//待会儿再试试中间没有0-9的情况
//$string = "迪奥和奥迪都是我最爱";

if(preg_match($zz, $string, $matches)){
    echo '匹配到了，结果为：';
    var_dump($matches);
}else{
    echo '没有匹配到';
}

?>
```

匹配成功，证明了\d+中的+。 \d是匹配数字，而+是最少匹配一次前面的字符。



**\* 匹配0次或者任意多次前面的字符**

```
<?php
$zz = '/\w*/';

$string = "!!@!!@##!$@#!";

//待会儿再试试中间没有0-9的情况
//$string1 = "!!@#!@#!abcABC#@#!";

if(preg_match($zz, $string, $matches)){
    echo '匹配到了, 结果为:';
    var_dump($matches);
}else{
    echo '没有匹配到';
}

?>
```

说明，注释掉的\$string1和\$string都匹配成功了。因为，\w是匹配的0-9A-Za-z\_，而\*是说明前面的\w可以不存在。如果存在可以有1个或者多个。

**? 前面的字符出现0次或者1次，可有可无**

```
<?php

$zz = '/ABC\d?ABC/';

$string = "ABC1ABC";

//待会儿再试试中间没有0-9的情况
//$string1 = "ABC888888ABC";
//$string2 = "ABCABC";

if(preg_match($zz, $string, $matches)){
    echo '匹配到了, 结果为:';
    var_dump($matches);
}else{
    echo '没有匹配到';
}

?>
```

匹配\$string,\$string2成功，但是匹配\$string1失败。

因为匹配前后都是ABC，中间是一个0-9。0-9可有可无，但是不能有多。

**. (点) 匹配除\n以外的所有字符**

```
<?php

$zz = '/gg.+gg/';

$string = "ABC1ABC";

if(preg_match($zz, $string, $matches)){
    echo '匹配到了, 结果为:';
    var_dump($matches);
}else{
    echo '没有匹配到';
}

?>
```

匹配\$string,\$string2成功，但是匹配\$string1失败。

因为匹配前后都是ABC，中间是一个0-9。0-9可有可无，但是不能有多个。

### |（竖线），或者，优先级最低

我们通过实验来看优先级和或者的匹配

```
<?php

$zz = '/abc|bcd/';

$string1 = "abccd";
$string2 = "ggggbcd";

if (preg_match($zz, $string1, $matches)) {
    echo '匹配到了，结果为：';
    var_dump($matches);
} else {
    echo '没有匹配到';
}

?>
```

我们来看看：

1. 最开始我匹配的想法是想匹配的是abccd或者是abbcd。可是，匹配\$string1和\$string2，匹配出来的结果却是abc和bcd.
2. 实现了或者匹配，匹配出来了abc或者是bcd。它还没有字符串连续在一起的优先级高。

### 那么问题来了，我要匹配上例中的abccd或者是abbcd怎么办？

需要使用到() 来改变优先级。

```
<?php

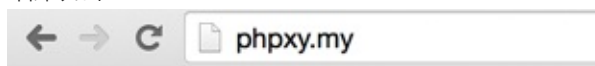
$zz = '/ab(c|b)cd/';

$string1 = "起来abccd阅兵";
$string2 = "ggggbcd";
$string3 = "中国abbcd未来";

if (preg_match($zz, $string1, $matches)) {
    echo '匹配到了，结果为：';
    var_dump($matches);
} else {
    echo '没有匹配到';
}

?>
```

结果如下：



匹配到了，结果为：

```
array (size=2)
  0 => string 'abccd' (length=5)
  1 => string 'c' (length=1)
```

结论：

1. 确实匹配了了abccd或者abbcd ( \$string1 or \$string3 ) 。
2. 但是匹配的数组里面多了一个元素，这个元素的下标为1
3. ()中的内容只要匹配成功，会把匹配到的数据放到下标为1的这个数组元素中。

### ^（抑扬符），必须要以^之后的字符串开始

```
<?php

$zz = '/^李文凯好帅\w+/' ;

$string1 = "李文凯好帅abccdaaaasds";
//$string2没有以李文凯好帅开始
$string2 = "帅abccdaaaasds";

if (preg_match($zz, $string1, $matches)) {
    echo '匹配到了，结果为：';
    var_dump($matches);
} else {
    echo '没有匹配到';
}

?>
```

通过实验发现以下结论：

1. \$string1匹配成功，\$string2没有匹配成功
2. 因为\$string1是以指定的字符开始的
3. 而\$string2并没有以^之后的字符开始
4. 翻译这个正则的意思就是：以要李文凯好帅开始后面接a-zA-Z0-9\_最少一个字符。

### \$（美元符）必须要以\$之前的字符结束

```
<?php

$zz = '/\d+努力$/';

$string1 = "12321124333努力";
//$string2
$string2 = "12311124112313力";

if (preg_match($zz, $string1, $matches)) {
    echo '匹配到了，结果为：';
    var_dump($matches);
} else {
    echo '没有匹配到';
}

?>
```

我们运行一下看一下结果，得出来的结论：

1. \$string1 匹配成功，而\$string2匹配不成功
2. \$之前的字符是\d+，后面接着中文的努力。
3. 因此，匹配的是这一个整体。\\d指的是0-9的整型,+号代表最少一个0-9

### \\b和\\B 词边界和非词边界

我们来讲解什么是边界：

1. 正则表达示是有边界的，这个边界是定界符的开始和结尾是正则的边界。

2. this是一个英文单词，后面加上一个空格，意味着这个词结束了，到达了这个词的边界

\b词边界，就是指必须要在最前或者最后。

\B非边界，就是不能在一个正则表达示的最前或者最后。

```
<?php

$zz = '/\w+\b/';

$string1 = "this is a apple";
$string2 = "thisis a apple";
$string3 = "thisisaapple";

if (preg_match($zz, $string1, $matches)) {
    echo '匹配到了，结果为：';
    var_dump($matches);
} else {
    echo '没有匹配到';
}

?>
```

结论：

1. \$string1、\$string2和\$string3都匹配成功。
2. \$string1匹配的时候this 空格是边界
3. \$string2匹配的时候thisis是边界
4. \$string3匹配的时候，thisisaapple到了整个正则表达示的最后，因此也是边界。所以匹配成功。

我们来实验一下非词边界：

```
<?php

$zz = '/\Bthis/';

$string1 = "hellothis9";

//$string2 = "hello this9";
//$string2 = "this9中国万岁";

if (preg_match($zz, $string1, $matches)) {
    echo '匹配到了，结果为：';
    var_dump($matches);
} else {
    echo '没有匹配到';
}

?>
```

总结：

1. 匹配\$string1成功而\$string2不成功。
2. 因为\B后接的是this，所以this不能在词边界（空格和开始结尾）的位置出现。

### {m}有且只能出现m次

```
<?php

$zz = '/喝\d{3}酒/';
```

## 7.3 正则表示中的元字符

```
$string1 = "喝988酒";

//$string2 = "喝98811酒";

if (preg_match($zz, $string1, $matches)) {
    echo '匹配到了, 结果为:';
    var_dump($matches);
} else {
    echo '没有匹配到';
}

?>
```

结论：

上例中\d{3}我规定了0-9只能出现3次，多一次少一次都不行。

### {n,m} 可以出现n到m次

```
<?php

$zz = '/喝\d{1,3}酒/';

$string1 = "喝9酒";

//$string2 = "喝988酒";

if (preg_match($zz, $string1, $matches)) {
    echo '匹配到了, 结果为:';
    var_dump($matches);
} else {
    echo '没有匹配到';
}

?>
```

结论：

上例中\d{1,3}我规定了0-9只能出现1次，2次或者3次。其它次数都是错的

### {m,} 至少m次，最大次数不限制

```
<?php

$zz = '/喝\d{2,}/';

$string1 = "喝9";

//$string2 = "喝98";
//$string3 = "喝98122121";

if (preg_match($zz, $string1, $matches)) {
    echo '匹配到了, 结果为:';
    var_dump($matches);
} else {
    echo '没有匹配到';
}

?>
```

结论：

上例中\d{2,}我规定喝后面的0-9最少出现两次，最多次数不限。因此\$string1是匹配不成功的，\$string2是匹配成功的。  
\$string3是匹配成功的。



# 7.4 正则达达示中的模式修正符

我们通过元字符和原子完成了正则表达式的入门。但有一些特殊情况我们依然需要来处理。

如果abc在第二行的开始处如何匹配？  
我不希望正则表达示特别贪婪的匹配全部，只匹配一部分怎么办？

这个时候，我们就需要用到下面的这些模式匹配来增强正则的功能。

常用的模式匹配符有：

模式匹配符	功能
i	模式中的字符将同时匹配大小写字母.
m	字符串视为多行
s	将字符串视为单行,换行符作为普通字符.
x	将模式中的空白忽略.
A	强制仅从目标字符串的开头开始匹配.
D	模式中的美元元字符仅匹配目标字符串的结尾.
U	匹配最近的字符串.

模式匹配符的用法如下：

```
/ 正则表达示/模式匹配符
```

模式匹配符是放在这句话的最后的。例如：

```
/\w+/s
```

格式我们清楚了，接下来最主要的是加强对于模式匹配符使用的理解和记忆。我们通过代码来理解加上和不加模式匹配符有何区别。

## i 不区分大小写

```
<?php
//在后面加上了一个i
$pattern = '/ABC/i';

$string = '8988abc12313';

$string1 = '11111ABC2222';

if(preg_match($pattern, $string, $matches)){
    echo '匹配到了，结果为：';
    var_dump($matches);
}else{
    echo '没有匹配到';
}
?>
```

结论，不论是\$string还是\$string1全都匹配成功了。因此，在后面加上了i之后，在匹配的时候可以不分大小写。

### m 视为多行

正则匹配的时候，要匹配的目标字符串我们通常视为一行。

“行起始”元字符（`^`）仅仅匹配字符串的起始，“行结束”元字符（`$`）仅仅匹配字符串的结束。

当设定了此修正符，“行起始”和“行结束”除了匹配整个字符串开头和结束外，还分别匹配其中的换行符的之后和之前。

**注意：如果要匹配的字符串中没有“`\n`”字符或者模式中没有`^`或`$`，则设定此修正符没有任何效果。**

我们通过实验和代码来验证一下这个特点：

第一次匹配，你会发现匹配不成功：

```
<?php

$pattern = '/^a\d+/';

$string = "我的未来在自己手中我需要不断的努力
a9是一个不错的字符表示
怎么办呢，其实需要不断奋进";

if (preg_match($pattern, $string, $matches)) {
    echo '匹配到了，结果为：';
    var_dump($matches);
} else {
    echo '没有匹配到';
}
```

第二次匹配，我们加上m 试试：

```
<?php

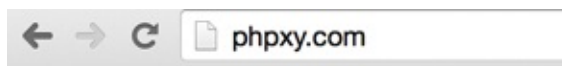
$pattern = '/^a\d+/m';

$string = "我的未来在自己手中我需要不断的努力
a9是一个不错的字符表示
怎么办呢，其实需要不断奋进";

if (preg_match($pattern, $string, $matches)) {
    echo '匹配到了，结果为：';
    var_dump($matches);
} else {
    echo '没有匹配到';
}
```

结果：

哦耶！匹配成功了。`/^a\d+/` 匹配的内容是a9，必须得在行开始处。在第二行也被匹配成功了。



匹配到了，结果为：

```
array (size=1)
  0 => string 'a9' (length=2)
```



### s 视为一行

如果设定了此修正符，模式中的圆点元字符 (.) 匹配所有的字符，包括换行符。

第一次，不加模式匹配符s：

```
<?php

$pattern = '/新的未来.+\\d+/';

$string = '新的未来
987654321';

if (preg_match($pattern, $string, $matches)) {
    echo '匹配到了，结果为：';
    var_dump($matches);
} else {
    echo '没有匹配到';
}

?>
```

第二次，在正则表达式后面加上模式匹配符s:

```
<?php

$pattern = '/新的未来.+\\d+/s';

$string = "新的未来
987654321";

if (preg_match($pattern, $string, $matches)) {
    echo '匹配到了，结果为：';
    var_dump($matches);
} else {
    echo '没有匹配到';
}

?>
```

结果如下，匹配成功！



```
匹配到了，结果为：
array (size=1)
  0 => string '新的未来
987654321' (length=22)
```

### 结论：

1. 因为在新的未来，未来后面有一个换行
2. 而.(点)是匹配非空白字符以外的所有字符。因此，第一次不成功
3. 第二次，加上了s模式匹配符。因为，加上后.(点)能匹配所有字符。

### x 忽略空白字符

## 7.4 正则表达式中的模式修正符

1. 如果设定了此修正符，模式中的空白字符除了被转义的或在字符类中的以外完全被忽略。
2. 未转义的字符类外部的#字符和下一个换行符之间的字符也被忽略。

我们先来实验一下忽略空白行等特性：

```
<?php

$pattern = '/a b c /x';

$string = '学英语要从abc开始';

if (preg_match($pattern, $string, $matches)) {
    echo '匹配到了，结果为：';
    var_dump($matches);
} else {
    echo '没有匹配到';
}

?>
```

这样也能匹配成功。



匹配到了，结果为：

```
array (size=1)
  0 => string 'abc' (length=3)
```

在\$pattern里面有空格，每个abc后面有一个空格。而\$string里面没有空格。所以x忽略空白字符。

而第二句话从字面上比较难理解，

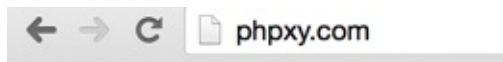
```
<?php
//重点观察这一行
$pattern = '/a b c #我来写一个注释
/x';

$string = '学英语要从abc开始';

if (preg_match($pattern, $string, $matches)) {
    echo '匹配到了，结果为：';
    var_dump($matches);
} else {
    echo '没有匹配到';
}

?>
```

结果也匹配成功了！



匹配到了，结果为：

```
array (size=1)
  0 => string 'abc' (length=3)
```

我们发现，x的第二个特性是忽略：#字符和下一个换行符之间的字符也被忽略。

## e 将匹配项找出来，进行替换

- e模式也叫逆向引用。主要的功能是将正则表达式括号里的内容取出来，放到替换项里面替换原字符串。
- 使用这个模式匹配符前必须要使用到preg\_replace()。

```
mixed preg_replace ( mixed $正则匹配项, mixed $替换项, mixed $查找字符串)
```

- preg\_replace的功能：使用\$正则匹配项变，找到\$查找字符串变量。然后用\$替换项变量进行替换。

在正式讲解前我们回顾一下之前的知识，我们故意把每个要匹配的原子外面都加上括号：

```
<?php
//加上了括号
$pattern = '/(\d+)([a-z]+)(\d+)/';

$string = '987abc321';

if (preg_match($pattern, $string, $match)) {
    echo '匹配到了，结果为：';
    var_dump($match);
} else {
    echo '没有匹配到';
}
?>
```

我们来看看结果：



```
array (size=4)
  0 => string '987abc321' (length=9)
  1 => string '987' (length=3)
  2 => string 'abc' (length=3)
  3 => string '321' (length=3)
```

这是我们之前讲括号的时候：匹配到的内容外面有括号。会把括号里面的内容，也放到数组的元素里面。如图中的：987、abc、321。

我们接下来看正则表达式中的e模式：

```
<?php
$string = "{April 15, 2003}";

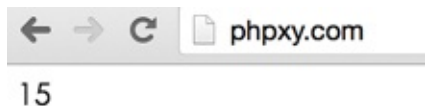
// 'w' 匹配字母，数字和下划线，'d' 匹配0-99数字，'+' 元字符规定其前导字符必须在目标对象中连续出现一次或多次
$pattern = "/{(\w+) (\d+), (\d+)}/i";

$replacement = "\$2";

// 字符串被替换为与第 n 个被捕获的括号内的子模式所匹配的文本
echo preg_replace($pattern, $replacement, $string);
```

```
?>
```

我们看看执行结果：



结论：

1. 上例中\\$\$ 指向的是正则表达式的第一个(\d+)。相当于把15又取出来了
2. 替换的时候，我写上\\$\$。将匹配项取出来，用来再次替换匹配的结果。

## U 贪婪模式控制

正则表达式默认是贪婪的，也就是尽可能的最大限度匹配。

我们来看看正则表达式是如何贪婪的：

```
<?php
$pattern = '/<div>.*</div>/';

$string = "<div>你好</div><div>我是</div>";

if (preg_match($pattern, $string, $match)) {
    echo '匹配到了，结果为：';
    var_dump($match);
} else {
    echo '没有匹配到';
}

?>
```

我们来看看结果，得到如下结论。它从“

你好”直接匹配到了“我是”。



同样一段代码我们再加大写的U，再看看效果：

```
<?php
$pattern = '/<div>.*</div>/U';

$string = "<div>你好</div><div>我是</div>";

if (preg_match($pattern, $string, $match)) {
    echo '匹配到了，结果为：';
    var_dump($match);
} else {
    echo '没有匹配到';
}

?>
```

```
    echo '没有匹配到';  
}  
  
?>
```



匹配到了，结果为：

```
array (size=1)  
  0 => string '<div>你好</div>' (length=17)
```

我们发现，只匹配出来了：

```
<div>你好</div>
```

这样，把正则的贪婪特性取消掉。让它找到了最近的匹配，就OK了。

### A 从目标字符串的开头开始匹配

此模式类似于元字符中的^（抑扬符）效果。

```
<?php  
  
$pattern = '/this/A';  
  
$string = 'hello this is a ';  
//$string1 = 'this is a ';  
  
if (preg_match($pattern, $string, $match)) {  
    echo '匹配到了，结果为：';  
    var_dump($match);  
} else {  
    echo '没有匹配到';  
}  
  
?>
```

结论：

1. 如果加A模式修正符的时候匹配不出来\$string，不加时能匹配出来
2. 如果加上了A模式修正符的时候能匹配出来\$string1,因为必须要从开始处开始匹配

### D 结束\$符后不准有回车

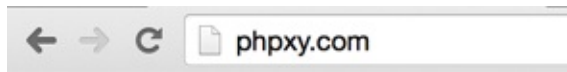
如果设定了此修正符，模式中的美元元字符仅匹配目标字符串的结尾。没有此选项时，如果最后一个字符是换行符的话，美元符号也会匹配此字符之前。

```
<?php  
  
$pattern = '/\w+this$/';  
  
//$pattern1 = '/\w+this$/D';  
  
$string = "hellothis  
";
```

## 7.4 正则表达式中的模式修正符

```
if (preg_match($pattern, $string, $match)) {  
    echo '匹配到了, 结果为: ';  
    var_dump($match);  
} else {  
    echo '没有匹配到';  
}  
  
?>
```

结果展示：



匹配到了, 结果为:

```
array (size=1)  
  0 => string 'hellothis' (length=9)
```

结论：

1. 如pattern 在匹配\$string的时候，\$string的字符串this后有一个回车。在没有加D匹配符的时候也能匹配成功
2. 如pattern 在匹配\$string的时候，加上了D。\$string的字符串this后有空格，匹配不成功。

# 7.5 写正则的诀窍和常用正则

## 写正则的诀窍

女孩和男孩恋爱时的诀窍通常是：测试你是不是对我好一点，如果是的，咱们的关系就更好一点。

而正则的诀窍和恋爱的诀窍基本一致：写一点、测一点。

因为，我们需要不断的正则，用preg\_match对比是不是能匹配成功。成功了，再写后面的一点。直到写完，全部匹配成功为止！

例如，我要写一个邮箱的正则。我先要做的事情，将常用的邮箱格式全部列出来。例如：

liwenkai@phpxy.com  
liwenkai@corp.baidu.cm  
liwenkai@126.com  
l\_w\_k@xxx.com  
12345@qq.com

常用的格式主要有这样一些。那我们就可以来分析：

- 1. 先匹配@之前的字符 \w+（ 因为是0-9A-Za-z\_ ）
- 2. 第二个跟一个@符号
- 3. 第三个再写上[a-zA-Z0-9-]+ 因为qq和126这些主域名是不能有下划线的
- 4. corp.baidu. 或者是126. 通常邮箱后缀都是这样的。所以我们可以写成：([a-zA-Z0-9-]+){1,2}
- 5. 上面的是将.转义，让它是本身的意思。括号重复的区间最少一次，最多两次。
- 6. 后面接下com|cn|org|gov.cn|net|edu.cn等就可以了

因此，我们的正则表达示在我使用：

```
^w+@([a-zA-Z0-9-]+){1,2}(com|cn|org|gov.cn|net|edu.cn)/
```

邮箱的正则就被我写成功了。

## 常用正则函数

我们常用的正则函数有：	函数名	功能
preg_filter	执行一个正则表达式搜索和替换	
preg_grep	返回匹配模式的数组条目	
preg_match	执行一个正则表达式匹配	
preg_match_all	执行一个全局正则表达式匹配	
preg_replace_callback_array	传入数组，执行一个正则表达式搜索和替换使用回调	
preg_replace_callback	执行一个正则表达式搜索并且使用一个回调进行替换	
preg_replace	执行一个正则表达式的搜索和替换	
preg_split	通过一个正则表达式分隔字符串	

大家针对这这些函数，对着手册用一下。有问题或遇到问题可以来我们的官网提问。

### 正则关于面试常遇到的问题

面试中经常考到的几个正则达达示是：

1. 匹配邮箱
  2. 匹配手机号
  3. 匹配一个网址
  4. 用正则匹配某个格式，取出某个个例
  5. 写一个采集器
- 其他....

面试的第4题和第5题我对大家不担心，因为只要大家认真学习了我给的前五节的内容。第4，5题推理就行。

因为，通常在技术答题面试环节，是时候准许查手机的！

### 常用正则表达示

下面的是快速查找的正则表达示，大家一定得知道具体意思。

需要的时候直接复制即可：



## 7.6 用正则写一个UBB文本编辑器

我们来看一下UBB编辑器。这是网站当中经常用到的文本处理技术。因为使用UBB文件编辑器，我指定的格式才能存在。我不指定的格式，用户是无法在网站中展现的。

我们来看看效果：

```
<?php
$string='[b]为你写诗[/b]
[i]为你做不可能事[/i]
[u]哎呀，哥不是写情诗[/u]
[color=Red]哥是在说歌词[/color]
[size=7]吴克群[/size]
[qq]1378353651[/qq]';

//匹配UBB字符
$pattern=array(
    '/\[b\](.*)\[\/b\]/i',
    '/\[i\](.*)\[\/i\]/iU',
    '/\[u\](.*)\[\/u\]/i',
    '/\[color=(.*?)\](.*)\[\/color\]/',
    '/\[size=(\d+)\](.*)\[\/size\]/',
    '/\[qq\](\d{5,12})\[\/qq\]/',

    );

//需要替换的UBB字符
$replace=array(
    '<b>\1</b><br />',
    '<i>\1</i><br />',
    '<u>\1</u><br />',
    '<font color="\1">\2</font><br />',
    '<font size="\1">\2</font><br />',
    '<a href="http://wpa.qq.com/msgrd?v=1&uin=\1&site=[Discuz!]&menu=yes"
target="_blank"></a>',
    );

//使用正则匹配$string，将$string当中的值变为$replace的效果
$ubb=preg_replace($pattern,$replace,$string);

echo $ubb;
?>
```

实现的更高级，你可以让用户传过来的是form表单中的结果，让用户传值过来。

你转换输出成UBB格式。

哦耶，成功了！

**下一章节中我们学习了文件系统的知识，再跟大家讲解更加有趣的：网页采集器。**

## 08.文件系统

---

我们会点鼠标右键删除文件、会control+c（或右键）复制、粘贴文件，会新建一些文件，检测这个文件是不是只读文件。

在电脑里面进行的这些操作，在代码里面如果能操作就好了。

因为，如果有了这些操作。我们能做很多事情了：

1. 可不可以写入修改配置文件？
2. 是不是可以做PHP安装的时候检测文件的权限
3. 是不是可以做生成Html文件等等很多不同的操作

..... 其他太多太多的地方用到了文件操作。

学习文件处理本质上面就是学习文件处理的函数。再结合之前写的代码，完善自己的业务处理能力。

## 8.1 读取文件

我们在上一节当中，我们讲到了可以进行文件操作。而读取文件是一个最最常用的功能。

### readfile读取文件

那如何读取一个文件呢？我们先学一个函数。

```
int readfile ( string $文件名)
```

功能：传入一个文件路径，输出一个文件。

下面的这一段代码中，只要传入文件名或者指定的文件路径就把文件读取出来了。

```
<?php
//linux类的读取方式
readfile("/home/paul/test.txt");
//windows类的读取方式
readfile("c:\\boot.ini");
?>
```

注意：上面的代码中windows的斜线是\斜线，可能会转义掉一些字符。因此，我们写的时候写上两个斜线。

### file\_get\_contents打开文件

上面的是单纯打文件就直接输出了，有没有打开文件后，能够赋值给一个变量的操作方式呢。

PHP当然会提供这种方式。这个方式就是PHP打开文件并返回内容的方式之一，我们来看看函数：

```
string file_get_contents ( string filename)
```

功能：传入一个文件或文件路径，打开这个文件返回文件的内容。文件的内容是一个字符串。

```
<?php

$filename = 'NoAlike.txt';

$filestring = file_get_contents($filename);
echo $filestring;

?>
```

上面的代码，就打开了一个文件，并且将文件当中的内容进行了输出。

我们来针对之前的知识把代码进行一下扩展。用一用之前的知识点。

```
<?php
//假设我们有一个多行的文件叫NoAlike.txt，没有的话你可以新建一个这个文件
$filename = 'NoAlike.txt';

//打开这个文件，将文件内容赋值给$filestring
$filestring = file_get_contents($filename);
```

```
//因为每一行有一个回车即\n, 我用\n来把这个字符串切割成数组
$filearray = explode("\n", $filestring);

//把切割成的数组, 下标赋值给$key, 值赋值给$val, 每次循环将$key加1。
while (list($key, $val) = each($filearray)) {
    ++$key;
    $val = trim($val);

    //用的单引号, 单引号不解释变量进行了拼接而已
    print 'Line' . $key . ':' . $val . '<br />';
}

?>
```

上面，我们就把之前所学的知识进行了组合。

fopen、fread、fclose操作读取文件

上面file\_get\_contents打开文件的方式简单、粗暴。下面的

resource fopen ( string \$文件名, string 模式)

string fread ( resource \$操作资源, int 读取长度)

bool fclose ( resource \$操作资源 )

通过上面的函数我们来讲解资源类型的通常操作方式：

1. 打开资源

2. 使用相关函数进行操作

3. 关闭资源

fopen函数

fopen函数的功能是打开文件，参数主要有两个：

1. 文件打开的路径

2. 打开文件的模式

返回类型是一个资源类型，我们第一次遇到了之前基础类型的时候讲到的资源类型。

资源类型需要其他的函数来操作这个资源。所有的资源有打开就要有关闭。

fread函数

函数的功能是读取打开的文件资源。读取指定长度的文件资源，读取一部分向后移动一部份。至到文件结尾。

fclose函数

fclose函数的功能是关闭资源。资源有打开就有关闭。

了解完函数，后两个函数比较简单。而fopen函数的模式到底是什么，fopen的模式有下面几个，我们来讲一下fopen的模式：

模式	说明
r	只读方式打开，将文件指针指向文件头。
r+	读写方式打开，将文件指针指向文件头。

8.1 读取文件

w	写入方式打开，将文件指针指向文件头并将文件大小截为零。如果文件不存在则尝试创建
w+	读写方式打开，将文件指针指向文件头并将文件大小截为零。如果文件不存在则尝试创建
a	写入方式打开，将文件指针指向文件末尾。如果文件不存在则尝试创建
a+	读写方式打开，将文件指针指向文件末尾。如果文件不存在则尝试创建之
x	创建并以写入方式打开，将文件指针指向文件头。如果文件已存在，则 fopen() 调用失败并返回 FALSE，并生成一条 E_WARNING 级别的错误信息。如果文件不存在则尝试创建
x+	创建并以读写方式打开，将文件指针指向文件头。如果文件已存在，则 fopen() 调用失败并返回 FALSE，并生成一条 E_WARNING 级别的错误信息。如果文件不存在则尝试创建

接下来，我们只来学习r模式，下结课我们在将写入的时候再来讲讲其他几个模式。

我们先会读取文件了，才能很好的掌握写入文件。

1.打开文件

```
<?php
//你可以创建一个NoAlike.txt，以只读模式打开
$fp = fopen('NoAlike.txt', "r");

//var_dump()操作一下$fp看看效果，输出的是不是只有类型提示的是resource
var_dump($fp);
?>
```

2.读取文件

```
<?php
$fp = fopen('NoAlike.txt', "r");

//打开一个文件类型后，读取长度
$content = fread($fp, 1024);
?>
```

3.关闭文件

```
<?php
$fp = fopen($filename, 'r');
$content = fread($fp, 1024);
fclose($fp);
echo $content;
?>
```

其他注意事项：

模式	说明
t	windows下将\n转为\r\n
b	二进制打开模式

使用示例：

```
<?php
```

## 8.1 读取文件

```
$fp = fopen($filename, 'ab');  
$contents = fwrite($fp, '可爱的很\n哟');  
fclose($fp);  
echo $contents;  
  
?>
```

说明：

实验无法让肉眼看到这个实验效果。大家记住有这个特点即可。

Windows 下提供了一个文本转换标记 ( 't' ) 可以将 \n 转换为 \r\n。

与此对应还可以使用 'b' 来强制使用二进制模式，这样就不会转换数据。要使用这些标记，要么用 'b' 或者用 't' 作为 mode 参数的最后一个字符。

## 8.2 创建和修改文件内容

上一节当中我们学习了读取文件特别好掌握。这一节当中我们来讲解的是写入文件。

### file\_put\_contents写入文件

我们先来学习第一种写入文件的方式：

```
int file_put_contents ( string $文件路径, string $写入数据)
```

功能：向指定的文件当中写入一个字符串，如果文件不存在则创建文件。返回的是写入的字节长度

```
<?php
    $data = "我是一个兵，来自老百姓";

    $numbytes = file_put_contents('binggege.txt', $data);

    if($numbytes){

        echo '写入成功，我们读取看看结果试试：';

        echo file_get_contents('binggege.txt');

    }else{
        echo '写入失败或者没有权限，注意检查';
    }
?>
```

我们发现写入文件也挺简单的。按照这个函数的格式，指定文件，写入字符串数据就可以了。

### fwrite配合fopen进行写入操作

```
int fwrite ( resource $文件资源变量, string $写入的字符串 [, int 长度])
```

注：fwrite的别名函数是fputs

**我们上节课试了r模式，只道是读取的时候使用，接下来我们用fwrite加上fopen中的w，写入模式来进行文件写入。**

我们来看一下特点：

写入方式打开，将文件指针指向文件头并将文件大小截为零。如果文件不存在则尝试创建。

注意：在下面的实验中，你可以试试新建个test.txt文件向里面写入内容。然后，可以试试把test.txt删除。看看有什么提示。

```
<?php
    $filename = 'test.txt';
    $fp= fopen($filename, "w");
    $len = fwrite($fp, '我是一只来自南方的狼，一直在寻找心中的花姑娘');
    fclose($fp);
    print $len . '字节被写入了\n';
?>
```

总结：

- 1.不论有没有新建都会打开文件重新写入
- 2.原有的文件内容会被覆盖掉
- 3.文件不存在会创建

那我们来对比一下以下几个模式的不同：

模式	说明
r	只能读不能使用fwrite写
r+	可操作读、写
w	只可以写功能
w+	即可读又可以写

我们来通过实验证明一下：

```
<?php
    $filename = 'test.txt';
    $fp= fopen($filename, "r+");
    $len = fwrite($fp, '我是一只来自南方的狼，一直在寻找心中的花姑娘');
    fclose($fp);
    print $len . '字节被写入了\n';
?>
```

实验时可以把r后面的+号去掉。

我们通过实验，确实发现使用r模式，在文件存的时候可以进行写入数据，只用r的话，写入是不成功的。

## a模式和w模式的不同

同样是下面的这段代码，我们改为a模式。

```
<?php
    $filename = 'test.txt';
    $fp= fopen($filename, "a");
    $len = fwrite($fp, '读大学迷茫了， PHP学院PHP给你希望');
    echo  $len . '字节被写入了\n';
?>
```

打开网页执行这段代码，你会发现：每刷新一次，文件中就会多一段  
：读大学迷茫了， PHP学院PHP给你希望。

总结：

模式	总结
x	每次写入会干掉原有文件的内容，文件不存在都会创建
a	每次写入都会向文件的尾端追加内容

注：a+ 是增强的追加功能。可以读取时也可以使用。

## x模式和w模式的不同

这段代码我们再实验一次，改为x模式：



## 8.2 创建和修改文件内容

```
<?php
    $filename = 'test.txt';
    $fp= fopen($filename, "x");
    $len = fwrite($fp, '读大学迷茫了, PHP学院PHP给你希望');
    echo  $len . '字节被写入了\n';
?>
```

我们会发现：

1. 文件存在的时候会报错
2. 如果把\$filename 改成其他的文件名,就可以了。但是，再次刷新的时候又报错了
3. x+ 是增强的x模式。读取时也可以使用。

## 8.3 创建临时文件

我们之前创建的文件都是永久文件。

而创建临时文件在我们平时的项目开发中也非常有用。创建临时文件的几个好处：

1. 文件使用完后即删除
2. 不需要去维护这个文件的删除状态

例如：我需要把A的文件内容转存B里面，把B的文件内容转存到C里面。

就跟现实生活中一样，我可以先用一个临时的瓶子把B的水装起来，然后把A的数据写入到B里面。把临时瓶子的水追加到C里面。

我们来学习一下这个函数：

```
resource tmpfile ( )
```

功能：创建一个临时文件，返回资源类型。关闭文件即被删除。

```
<?php
//创建了一个临时文件
$handle = tmpfile();

//向里面写入了数据
$numbytes = fwrite($handle, '写入临时文件');

//关闭临时文件，文件即被删除
fclose($handle);

echo '向临时文件中写入了' . $numbytes . '个字节';
?>
```

## 8.4 移动、拷贝和删除文件

我们日常在处理文件的时候，可以删除文件、重命名文件也可以复制文件。

那在这一节，我们就来讲解我们日常生活当中最常用到的一些操作。

我们先来说重命名，重命名的函数是：

### 重命名文件

```
bool rename($旧名,$新名);
```

这个函数返回一个bool值，将旧的名字改为新的名字。

```
<?php
//旧文件名
$filename = 'test.txt';

//新文件名
$filename2 = $filename . '.old';

//复制文件
rename($filename, $filename2);
?>
```

我们打开目录，我们可以看看效果，你会发现把指定的文件，复制了一份到了目标路径。

### 复制文件

复制文件，就相当于是克隆技术，将一个原来的东西再克隆成一个新的东西。两个长得一模一样。

```
bool copy(源文件,目标文件)
```

功能：将指定路径的源文件，复制一份到目标文件的位置。

我们来通过实验和代码来玩玩：

```
<?php
//旧文件名
$filename = 'copy.txt';

//新文件名
$filename2 = $filename . '_new';

//修改名字。
copy($filename, $filename2);
?>
```

总结：

你会通过上面的例子，发现多出来了一个文件。

### 删除文件

删除文件就是将指定路径的一个文件删除，不过这个删除是直接删除。使用的是windows电脑，你在回收站看不到这个文

本文档使用 [看云](#) 构建

件。

你只会发现，这个文件消失了。

`bool unlink(指定路径的文件)`

```
<?php
    $filename = 'test.txt';

    if (unlink($filename)) {
        echo "删除文件成功 $filename!\n";
    } else {
        echo "删除 $filename 失败!\n";
    }
?>
```

# 8.5 检测文件属性函数

有些同学特别好奇，检测文件属性用在什么地方。检测文件属性能用到的地方太多了。

我们来举例子：

- 1. 我们在做软件安装的时候，大家会发现如果文件存在了，就跳转到了其他的地方。
- 2. 如果在安装的过程当中，某些文件没有写入权限，就不让进行安装。

我们来举一个国内非常出名的软件，discuz的安装过程截图给大家看看：

/install/index.php?step=1&uchidden=&submit=我同意

微盘空间	10M	个限制	555/2M
目录、文件权限检查			
目录文件	所需状态	当前状态	
./config/config_global.php	✓ 可写	✓ 可写	
./config/config_ucenter.php	✓ 可写	✓ 可写	
./config	✓ 可写	✓ 可写	
./data	✓ 可写	✓ 可写	
./data/cache	✓ 可写	✓ 可写	
./data/avatar	✓ 可写	✓ 可写	
./data/plugindata	✓ 可写	✓ 可写	
./data/download	✓ 可写	✓ 可写	
./data/addonmd5	✓ 可写	✓ 可写	
./data/template	✓ 可写	✓ 可写	
./data/threadcache	✓ 可写	✓ 可写	
./data/attachment	✓ 可写	✓ 可写	
./data/attachment/album	✓ 可写	✓ 可写	
./data/attachment/forum	✓ 可写	✓ 可写	
./data/attachment/group	✓ 可写	✓ 可写	
./data/attachment/group	✓ 可写	✓ 可写	

上面的例子，就是典型的文件检测的用法。

我们来学习一下下面的一批函数。然后，我们来通过一个例子进行学习。

```
bool file_exists ( $指定文件名或者文件路径)
功能：文件是否存在。

bool is_readable ( $指定文件名或者文件路径)
功能：文件是否可读

bool is_writable ( $指定文件名或者文件路径)
功能：文件是否可写

bool is_executable ( $指定文件名或者文件路径)
功能：文件是否可执行

bool is_file ( $指定文件名或者文件路径)
功能：是否是文件
```

## 8.5 检测文件属性函数

bool is\_dir ( \$指定文件名或者文件路径)

功能：是否是目录

```
void clearstatcache ( void )
```

功能：清楚文件的状态缓存

上面的功能一看就清楚了。那实验，我们就来写这个最开始我们举的这个例子。

我们来讲第一个例子，文件锁。如果已经安装了，存在安装锁就提示已安装，否则就继续安装。

我们假设安装界面的网址是：install.php,安装的锁文件是install.lock。我们就可以检测install.lock文件是否存在。

```
<?php

if(file_exists('install.lock')){

    echo '已安装, 请不要再次进行安装';
    exit;

}

?>
```

我们接下来做一个文件安装检测的实验来检测文件或目录是否有写入或者读取权限。如果没有则不能进行安装。

处理这件事情的思路如下：

1. 定义一批需要检测权限的数组
2. 可以检测是文件夹还是文件
3. 做一个标置位变量，如果标置位变量一旦为false则不显示下一步的安装

```
<?php  
//可以定义一批文件是否存在  
$files = [  
    'config.php',  
    'img/',  
    'uploads/',  
];  
  
//定义标志位变量  
$flag = true;  
foreach($files as $v){  
    echo $v;  
  
    //判断是文件还是文件夹  
  
    if(is_file($v)){  
        echo '是一个文件&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&';  
    }else if(is_dir($v)){  
        echo '是一个文件夹&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&~';  
    }  
  
    if(is_readable($v)){  
        echo '可读';  
    }else{  
        echo '<font color="red">不可读</font>';  
    }  
  
    if(is_writeable($v)){  
        echo '可写';  
    }else{
```

## 8.5 检测文件属性函数

```
        echo '<font color="red">不可写</font>';
    }

    echo '<br />';
}

if($flag){
    echo '<a href="step1">下一步</a>';
}
else{
    echo '不能进行安装';
}
?>
```

通过上面的例子，我们就做到了。实现PHP某个软件安装过程当中的安装检测。

也就是我们上面思路的实现。

# 8.6 文件常用函数和常量

## 文件操作的常量

下面这一个常量是最常用的。是文件目录的分割符的常量。

我们来看看格式：

平台	分割符
windows	\
linux	/

windows 的路径格式为 d:\xxx\xxx 注意：windows支持 d:/xxx/xxx  
linux 的路径格式为 /home/xxx/xxx 注意：如果\home\xxx\xxx 在linux上是错误的  
所以当你开启转义之类的话，转义字符\用一起的话 d:\xxx\xxx 是一样的。判断时候有两个\有的话 再转成一个\ 再把\替换成 /当路径分割，这样在linux上或者windos上的路径就能保持统一了。

我们会使用到一个常量：

### DIRECTORY\_SEPARATOR

我们来写一个小的实例，定义当前文件所在的路径：

由于**FILE**是PHP的预定义常量，所以没办法改变，如果需要让**FILE**也自适应操作系统。  
那么就是不要用**FILE**,可以用自定义的常量，并且把**FILE**处理一下，如下：

```
<?php
$current_file = str_replace(array('/', '\\'), DIRECTORY_SEPARATOR, __FILE__);
define('__CUR_FILE__', $current_file);

echo __CUR_FILE__;

?>
```

## 文件指针操作函数

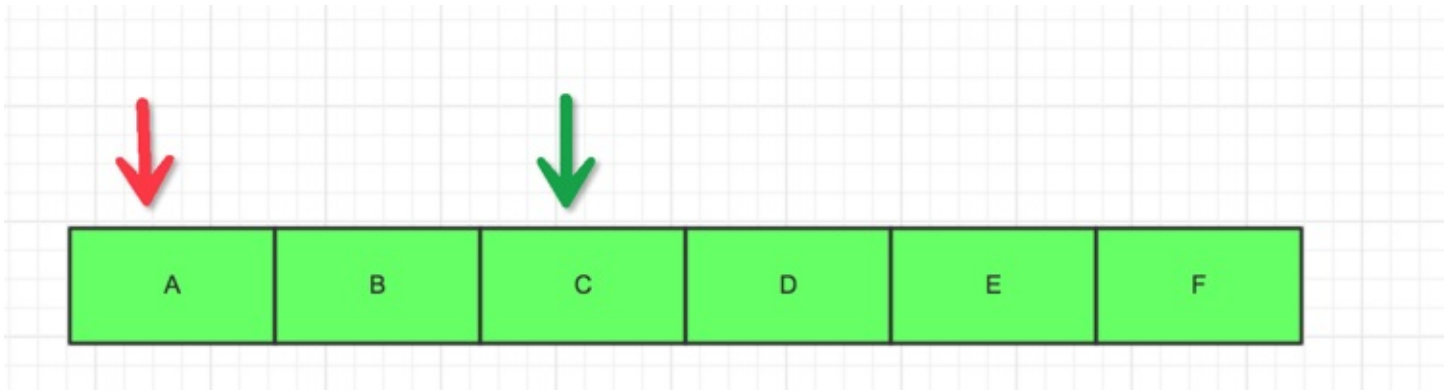
rewind ( resource handle)

功能：指针回到开始处

fseek ( resource handle, int offset [, int from\_where])  
功能：文件指针向后移动指定字符

我们在之前的读取当中我们发现fread读取指定长度的数据。读取指定长度的内容，下次再读取的时候从原位置开始再接着向后读取。





如上图，我们可以想象：

1. 文件刚打开的时候读取到的是红色的图标
2. 文件假设从A读取到了C
3. 下次打开的时候可再从C绿色箭头处开始读取。

我们在demo.txt文件中写入一批文件：

```
abcdeefghijklk  
opqrst  
vwxyz  
12345678
```

我们可开始来实验一次。

```
<?php  
$fp = fopen('output.txt', 'r+');  
//读取10个字符  
echo fread($fp,10);  
  
//指针设置回到开始处  
rewind($handle);  
//再读取10次看看输出的是什么  
echo fread($fp,10);  
  
//文件指针向后移动10个字符  
echo fseek($fp,10);  
  
//再看看文件中输出的是什么  
echo fread($fp,10);  
  
fclose($handle);  
?>
```

上面的例子，你会发现fseek指定多长就移动多少个字节。而rewind每次都是回到文件的开始处。

那如何移动到最末尾呢？我们可以统计字节数。在fseek的时候直接移到回后。

下面我们来讲filesize统计字节数。

## filesize 检测文件的大小

```
<?php
```

```
$filename = 'demo.txt';
echo $filename . '文件大小为: ' . filesize($filename) . ' bytes';

?>
```

## 其它操作文件的函数

其实还有一些其他操作文件的函数，读取文件

函数名	功能
file	把整个文件读入一个数组中
fgets	从文件指针中读取一行,读到最后返回false
fgetc	从文件指针中读取一个字符，读到最后返回false
ftruncate	将文件截断到给定的长度

我们通过一个实例来把上面几个函数都用到。

我们在demo.txt文件中写入一批文件：

```
abcdeefghjklk
opqrst
uvwxyz
12345678
```

```
<?php

//以增加的r模式打开
$fp = fopen('demo.txt','r+');

//你分发现每次只读一个字符
echo  fgetc($fp);

//我要全部读取可以, 读取一次将结果赋值一次给$string
while($string = fgetc($fp)){

    echo $string;

}
?>
```

fgets每次打开一行：

```
<?php

//以增加的r模式打开
$fp = fopen('demo.txt','r+');

//你分发现每次只读一个字符
echo  fgets($fp);
echo  fgets($fp);
echo  fgets($fp);
echo  fgets($fp);

?>
```

上面的代码，你会发现每次读取一次打开一行。读取最后返回的是false。

我们接下来看文件截取函数：

```
<?php

//打开我们上面的demo.txt文件
$file = fopen("demo.txt", "a+");

//你可以数数20个字有多长，看看是不是达到效果了
echo ftruncate($file,20);
fclose($file);
?>
```

上例中我们发现，截取多长就能显示多长的内容。

文件的时间函数

函数	功能说明
filectime	文件创建时间
filemtime	文件修改时间
fileatime	文件上次访问时间

```
<?php

$filename = 'demo.txt';

if (file_exists($filename)) {
    echo "$filename文件的上次访问时间是： " . date("Y-m-d H:i:s", fileatime($filename));

    echo "$filename文件的创建时间是： " . date("Y-m-d H:i:s", filectime($filename));

    echo "$filename文件的修改时间是： " . date("Y-m-d H:i:s", filemtime($filename));
}
?>
```

# 8.7 文件锁处机制

文件锁机制一般在单一打开文件的时候根本看不到效果。这一块的学习有一点点抽象。

大家不要去思考怎么实现的呀？

为什么看不到效果呀？

答：因为电脑的操作太快了，基本上是毫秒级的。所以这个实验其实是看不到效果的。

这一章了解文件锁的基本概念即可，熟悉文件锁函数和锁机制。

文件锁的用途：

若一个人在写入一个文件，另外一个人同时也打了个了这个文件进行写入文件。

这情况下，如果遇到一定的碰撞概率的话，不知道到底谁的操作为准。

因此，这个时候我们引入锁机制。

若用户A在写入或者读取这个文件的时候，将文件加上共享锁。我可以读，其他人也可以读。

但是，我如果这与时的时候。我使用独占锁。这个文件归我了，你们都别动，除非我将文件锁进行释放。

注意：加上了文件锁后要注意释放。

我们来看看这个函数：

```
bool flock ( resource $handle , int $operation)
```

功能：轻便的咨询文件锁定

我们来看看锁类型：

锁类型	说明
LOCK_SH	取得共享锁定（读取的程序）
LOCK_EX	取得独占锁定（写入的程序）
LOCK_UN	释放锁定（无论共享或独占）

我们接下来把demo.txt加上一个独占锁，进行写入操作。

```
<?php

$fp = fopen("demo.txt", "r+");

// 进行排它型锁定
if (flock($fp, LOCK_EX)) {

    fwrite($fp, "文件这个时候被我独占了哟\n");

    // 释放锁定
    flock($fp, LOCK_UN);
} else {
    echo "锁失败，可能有人在操作，这个时候不能将文件上锁";
}
```

## 8.7 文件锁处机制

```
fclose($fp);
```

```
?>
```

说明：

1. 上例中我为了写入文件，把文件加上了独占锁。
2. 如果我操作完成，写入完成后，解除掉了独占锁。
3. 如果是在读取文件的时候，大家可加按照同样的处理思路加上共享锁。

# 8.8 目录处理函数

之前我们处理的全都是文件，那目录和文件夹怎么处理呢？

我们就来学习目录或者称为文件夹的处理相关函数。

处理文件夹的基本思想如下：

- 1. 读取某个路径的时候判断是否是文件夹
- 2. 是文件夹的话，打开指定文件夹，返回文件目录的资源变量
- 3. 使用readdir读取一次目录中的文件，目录指针向后偏移一次
- 4. 使用readdir读取到最后，没有可读的文件返回false
- 5. 关闭文件目录

我们来学习一比常用函数：

函数名	功能
opendir	打开文件夹，返回操作资源
readdir	读取文件夹资源
is_dir	判断是否是文件夹
closedir	关闭文件夹操作资源
filetype	显示是文件夹还是文件，文件显示file，文件夹显示dir

```
<?php
//设置打开的目录是D盘
$dir = "d:/";

//判断是否是文件夹，是文件夹
if (is_dir($dir)) {
    if ($dh = opendir($dir)) {

        //读取一次向后移动一次文件夹指针
        echo readdir($dh).'\<br />';
        echo readdir($dh).'\<br />';
        echo readdir($dh).'\<br />';
        echo readdir($dh).'\<br />';

        //读取到最后返回false

        //关闭文件夹资源
        closedir($dh);
    }
}
?>
```

即便是读取一次向后移动一次，我们是不是可以

```
<?php
//设置打开的目录是D盘
$dir = "d:/";

//判断是否是文件夹，是文件夹
if (is_dir($dir)) {
```

## 8.8 目录处理函数

```
if ($dh = opendir($dir)) {  
  
    //读取到最后返回false, 停止循环  
    while (($file = readdir($dh)) != false) {  
        echo "文件名为: $file : 文件的类型是: " . filetype($dir . $file) . "<br />";  
    }  
  
    closedir($dh);  
}  
}  
?>
```

# 8.9 文件权限设置

文件权限设置的函数在系统管理级别的软件中很常用。例如：某个文件不准许guest组（来宾用户）查看呀。

在企业管理中，某些用户或者某些用户文件只准读取不准修改。这都是非常常用的功能。

注：

- 1. **本章为了解章节**。如果没有学习过linux会有些吃力，可以跳过本章，了解有这个东西即可。
- 2. 在实际生产环节中用处较少。
- 3. 主要针对在linux下有过全面知识体系的同学可以重点学习。
- 4. windows下面有些功能无法实现。

函数	功能说明
chmod	修改读取模式
chgrp	修改用户组
chown	修改权限

上面的函数用法与linux的权限操作的用法一样。

对于学过linux的同学来说学起来比较简单。没有学过的会有些吃力。

我只讲一个例子，看看如何来修改权限：

## chmod 主要是修改文件的的权限

```
<?php
//修改linux 系统/var/wwwroot/某文件权限为755
chmod("/var/wwwroot/index.html", 755);
chmod("/var/wwwroot/index.html", "u+rw,go+r");
chmod("/somedir/somefile", 0755);
?>
```



# 8.10文件路径函数

我们经常会遇到处理文件路径的情况。

例如：

- 1. 文件后缀需要取出来
- 2. 路径需要取出名字不取目录
- 3. 只需要取出路径名中的目录路径
- 4. 或者把网址中的各个部份进行解析取得独立值
- 5. 甚至是自己组成一个url出来
- ... ..

很多地方都需要用路径处理类的函数。

我们把常用的路径处理函数为大家做了标注，大家对着这个路径处理函数进行处理即可：

函数名	功能
pathinfo	返回文件的各个组成部分
basename	返回文件名
dirname	文件目录部分
parse_url	网址拆解成各部分
http_build_query	生成url 中的query字符串
http_build_url	生成一个url

## pathinfo

```
array pathinfo ( string $路径)
功能：传入文件路径返回文件的各个组成部份
```

我们用具体的示例来用一下：

```
<?php
$path_parts = pathinfo('d:/www/index.inc.php');

echo '文件目录名：' . $path_parts['dirname'] . "<br />";
echo '文件全名：' . $path_parts['basename'] . "<br />";
echo '文件扩展名：' . $path_parts['extension'] . "<br />";
echo '不包含扩展的文件名：' . $path_parts['filename'] . "<br />";
?>
```

结果如下：

```
文件目录名：d:/www
文件全名：lib.inc.php
文件扩展名：php
```

不包含扩展的文件名：lib.inc

### basename

string basename ( string \$路径[, string \$suffix ])

功能：传入路径返回文件名

第一个参数传入路径。

第二个参数，指定文件名到了指定字符停止。

```
<?php

echo "1: ".basename("d:/www/index.d", ".d").PHP_EOL;
echo "2: ".basename("d:/www/index.php").PHP_EOL;
echo "3: ".basename("d:/www/passwd").PHP_EOL;

?>
```

执行结果如下

```
1: index
2: index.php
3: passwd
```

### dirname

dirname(string \$路径)

功能：返回文件路径的文件目录部分

```
<?php
dirname(__FILE__);
?>
```

结论：你可以执行看看是不是返回了文件的目录部分。

### parse\_url

mixed parse\_url ( string \$路径 )

功能：将网址拆解成各个部分

```
<?php
$url = 'http://username:password@hostname:9090/path?arg=value#anchor';

var_dump(parse_url($url));

?>
```

结果如下：

```
array(8) {
```

```
["scheme"]=>
string(4) "http"
["host"]=>
string(8) "hostname"
["port"]=>
int(9090)
["user"]=>
string(8) "username"
["pass"]=>
string(8) "password"
["path"]=>
string(5) "/path"
["query"]=>
string(9) "arg=value"
["fragment"]=>
string(6) "anchor"
}
```

### http\_build\_query

string http\_build\_query ( mixed \$需要处理的数据)

功能：生成url 中的query字符串

```
<?php
//定义一个关联数组
$data = [
    'username'=>'liwenkai',
    'area'=>'hubei'
];

//生成query内容
echo http_build_query($data);
?>
```

结果如下：

```
username=liwenkai&area=hubei
```

```
http_build_url()
```

功能：生成一个url

注：

PHP\_EOL 常量

在 windows平台相当于 echo "\r\n";

在unix\linux平台相当于 echo "\n";

在mac平台相当于 echo "\r";



# 8.11 小小文件留言本

我们讲完了这么多关于文件处理的系统，可是我们连一个最基本的小东西都写不出来。

从这一节开始，大家会发现大家能够写越来越多的东西了。

接下来我们我们来看看演示效果：

在下面一个界面中写入留言内容的表单界面：

## 基于文件的留言本演示

用户名：

留言内容：

提交

存在留言后的展示界面：

用户名为**我相信**内容为**我相信只有我不断努力，我就能够创造明天。**时间为2015-09-09 15:30:09

## 基于文件的留言本演示

用户名:

留言内容:

我们来看一下**文件结构**：

index.php ---展示输入框和留言内容

write.php ---向message.txt写入数据

message.txt ---存入聊天内容

index.php文件

```
<?Php
//设置时区
date_default_timezone_set('PRC');

//读了内容
@$string = file_get_contents('message.txt');

//如果$string 不为空的时候执行，也就是message.txt中有留言数据
if (!empty($string)) {

    //每一段留言有一个分格符，但是最后多出了一个&^。因此，我们要将&^删掉
    $string = rtrim($string, '&^');

    //以&^切成数组
    $arr = explode('&^', $string);

    //将留言内容读取
    foreach ($arr as $value) {

        //将用户名和内容分开
        list($username, $content, $time) = explode('#', $value);

        echo '用户名为<font color="gree">' . $username . '</font>内容为<font color="red">' . $content . '</font>时间为' . date('Y-m-d H:i:s', $time);
        echo '<hr />';
    }
}
```

本文档使用 [看云](#) 构建

```
}

?>

<h1>基于文件的留言本演示</h1>
<form action="write.php" method="post">
    用户名: <input type="text" name="username" /><br />

    留言内容: <textarea name="content"></textarea><br />

    <input type="submit" value="提交" />

</form>
```

看了刚刚的展示内容，我们知道文件存储时：

1. 段与段之间进行了分割
2. 内容与用户之前用一个特殊的符号进行了分割

下面我们来写write.php写入留言至文件的代码：

```
<?php
//追加方式打开文件
$fp=fopen('message.txt','a');

//设置时间
$time=time();

//得到用户名
$username=trim($_POST['username']);
//得到内容
$content=trim($_POST['content']);

//组合写入的字符串：内容和用户之间分开，使用$#
//行与行之间分开，使用&^
$string=$username.'$#'.$content.'$#'.$time.'&^';

//写入文件
fwrite($fp,$string);

//关闭文件
fclose($fp);

header('location:index.php');

?>
```

## 8.12 修改配置文件的实例

我们来看看国内知名论坛discuz的安装界面：

Discuz!

安装向导

Discuz!X3.2 简体中文 UTF8 版 20150609

3.

安装数据库

正在执行数据库安装

检查安装环境

设置运行环境

创建数据库

安装

填写数据库信息

数据库服务器:

localhost

数据库服务器地址, 一般为 localhost

数据库名:

ultrax

数据库用户名:

root

数据库密码:

root

数据表前缀:

pre\_

同一数据库运行多个论坛时, 请修改前缀

系统信箱 Email:

admin@admin.com

用于发送程序错误报告

填写管理员信息

管理员账号:

admin

管理员密码:

管理员密码不能为空

重复密码:

管理员 Email:

admin@admin.com

下一步

在这儿安装，它怎么就修改了config.inc.php文件呢？

下面我们来通过几个简单的技术来揭开它所谓的神秘面纱！

文件规划：

- 1. index.php -- 展示修改界面
- 2. edit.php -- 修改功能代码
- 3. config.php -- 实际的修改部份



index.php 展示修改界面。将config.php中的配置项展示出来。展示到表单中：

```
<?php

    include 'config.php';

?>

<form action="edit.php" method="post">
<input type="text" name="DB_HOST" value="<?php echo DB_HOST;?>" /><br />
<input type="text" name="DB_USER" value="<?php echo DB_USER;?>" /><br />
<input type="text" name="DB_PWD" value="<?php echo DB_PWD;?>" /><br />
<input type="text" name="DB_NAME" value="<?php echo DB_NAME;?>" /><br />

<input type="submit" value="修改" />

</form>
```

2.edit.php 读取config.php文件，将这个文件视为字符串。我然后使用正则表达式匹配来修改内容。

```
<?php

$string=file_get_contents('config.php');

//DB_HOST    localhost

foreach($_POST as $key=>$val){

    //定义正则来查找内容，这里的key为form表单里面的name
    $yx="/define\('$key', '.*?'\);"/;

    //将内容匹配成对应的key和修改的值
    $re="define('$key', '$val');";

    //替换内容
    $string=preg_replace($yx,$re,$string);
}

//写入成功
file_put_contents('config.php',$string);

echo '修改成功';

?>
```

config.php 实际存储配置文件的部分，存储了真实的config.php文件内容：

```
<?php

define('DB_HOST','localhost');

define('DB_USER','liwenkai');

define('DB_PWD','facai');

define('DB_NAME','hehaoduopijiu');

?>
```

你是不是发现，这些其实并没有想象中的那么难。结合一下正则表达式和文件的知识就做到了！



## 09.PHP文件上传

---

在我们日常使用中经常会遇到很多种这样的情况：

1. QQ空间里面上传图片呀
2. 微信朋友圈上传图片
3. 发邮件里面上传邮件资料附件
4. 认证的时候要求上传照片或身份证

还有各种产品汪（gou）们提出的需求来分析，上传不同的东西。

而产品汪提出来的需求我们需要实现。

实现文件上传，是一个PHP程序员必备的技能之一。

通过学习文件上传，你将会看到文件上传的本质！



# 9.1 文件上传需要注意php.ini文件

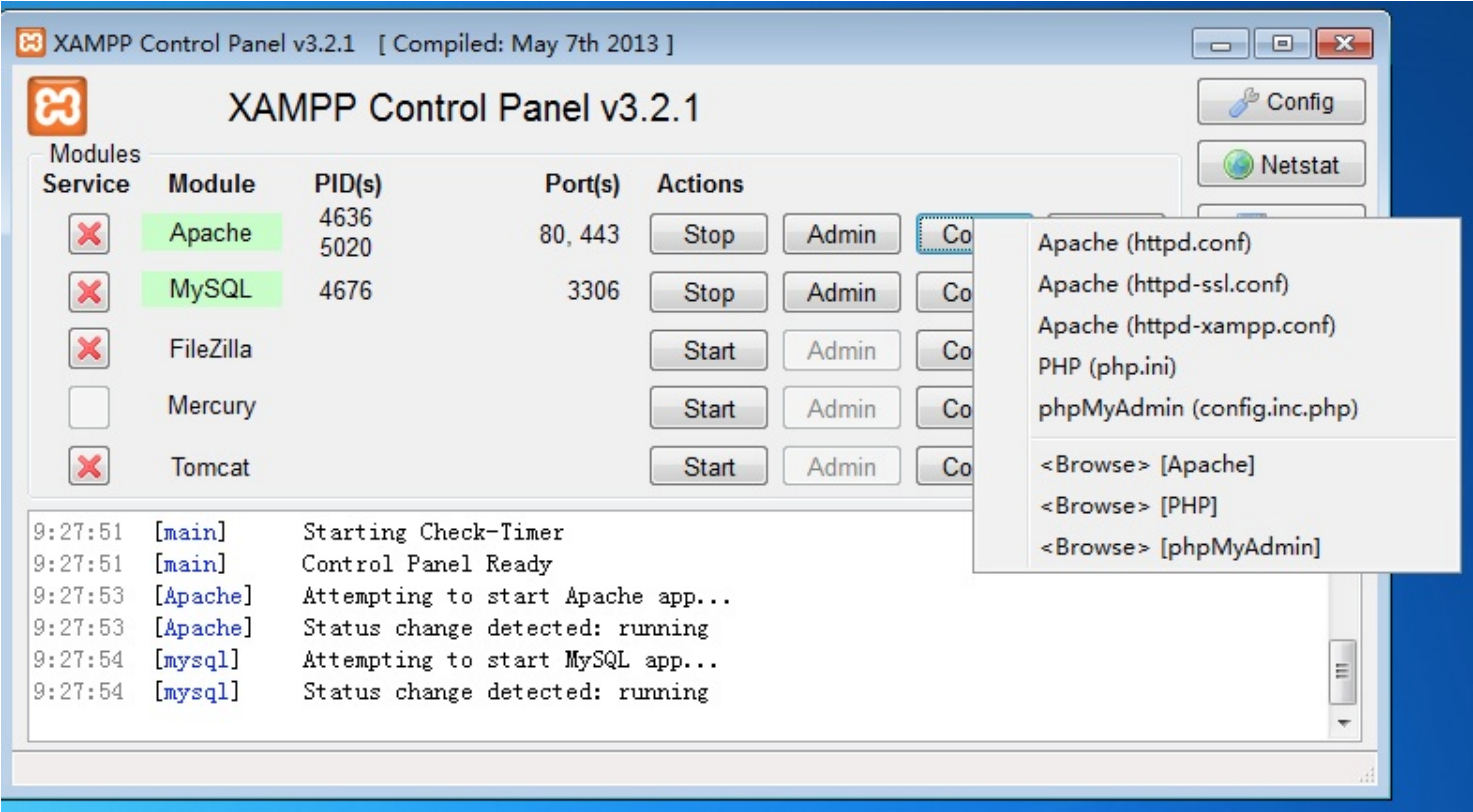
在正式讲解本章之前，我们第一件事情是要注意观察php.ini文件。

我们第一次开始接触到如何修改php.ini文件，如果你的配置项与我们说的不一致，请注意修改。

我们来了解每一个配置项。

我们看一下如何修改php.ini。

打开XAMPP中的php.ini配置文件。



php.ini的文件太多，找不到的时候你可以使用ctrl+f 搜索相关配置项。

配置项	功能说明
file_uploads	on为 开启文件上传功能，off为关闭
post_max_size	系统允许的POST传参的最大值
upload_max_filesize	系统允许的上传文件的最大值
memory_limit	内存使用限制

**建议尺寸：**

file\_size(文件大小) < upload\_max\_filesize < post\_max\_size < memory\_limit

另外，需要注意的是脚本执行时间。

max\_execution\_time，这什参数的单位为秒。

这个参数是设定脚本的最大执行时间。

本文档使用 看云 构建

## 9.1 文件上传需要注意php.ini文件

也可以根据需求做适当的改变。通常不需要来修改，系统默认值即可。超大文件上传的时候，可能会涉及到这一项参数的修改。

上传时间太长了，会超时。如果你将此项参数设为0，则是不限制超时时间，不建议使。

完成了php.ini的相关配置，我们就可以开始试着完成第一次文件上传了。

## 9.2 文件上传的步骤

为了更好的学习PHP，我们将极为复杂的PHP的文件上传归纳总结成为了6个步骤。

在实际使用过程中，你按照这6个步骤就能够很好的完成PHP的文件上传：

### 一、判断是否有错误码

系统返回的错误码详解：

错误码	说明
0	无误，可以继续进行文件上传的后续操作。
1	超出上传文件的最大限制，upload_max_filesize = 2M php.ini中设置，一般默认为2M。可根据项目中的实际需要来修改
2	超出了指定的文件大小,根据项目的业务需求指定上传文件的大小限制
3	只有部分文件被上传
4	文件没有被上传
6	找不到临时文件夹，可能目录不存在或没权限
7	文件写入失败，可能磁盘满了或没有权限

注：错误码中没有5。

### 二、自定义判断是否超出文件大小范围

在开发上传功能时。我们作为开发人员，除了php.ini中规定的上传的最大值外。

我们通常还会设定一个值，是业务规定的上传大小限制。

例如：

新浪微博或者QQ空间只准单张头像图片2M。而在上传图册的时候又可以超过2M来上传。

所以说，它的系统是支持更大文件上传的。

此处的判断文件大小，我们用于限制实际业务中我们想要规定的上传的文件大小。

### 三、判断后缀名和mime类型是否符合

在网络世界里面也有坏人。他们会把图片插入病毒，在附件中上传病毒，他们会在网页中插入病毒或者黄色图片。

我们需要对于上传的文件后缀和mime类型都要进行判断才可以。

MIME(Multipurpose Internet Mail Extensions)是多用途互联网邮件扩展类型。是设定某种扩展名的文件用一种应用程序来打开的方式类型，当该扩展名文件被访问的时候，浏览器会自动使用指定应用程序来打开。多用于指定一些客户端自定义的文件名，以及一些媒体文件打开方式。

在判断后缀和MIME类型的时候，我们会用到PHP的一个函数in\_array(),该函数传入两个参数。

第一个参数是要判断的值；

第二个参数是范围数组。

我们用这个函数来判断文件的后缀名和mime类型是否在允许的范围内。

### 四、生成文件名

我们的文件上传成功了，不会让它保存原名。

因为，有些人在原名中有敏感关键词会违反我国的相关法律和法规。

我们可以采用date()、mt\_rand()或者unique()生成随机的文件名。

### 五、判断是否是上传文件

文件上传成功时，系统会将上传的临时文件上传到系统的临时目录中。产生一个临时文件。

同时会产生临时文件名。我们需要做的事情是将临时文件移动到系统的指定目录中。

而移动前不能瞎移动，或者移动错了都是不科学的。移动前我们需要使用相关函数判断上传的文件是不是临时文件。

is\_uploaded\_file()传入一个参数(\$\_FILES中的缓存文件名)，判断传入的名称是不是上传文件。

### 六、移动临时文件到指定位置

临时文件是真实的临时文件，我们需要将其移动到我们的网站目录下面了。

让我们网站目录的数据，其他人可以访问到。

我们使用：move\_uploaded\_file()。

这个函数是将上传文件移动到指定位置，并命名。


传入两个参数：

第一个参数是指定移动的上传文件；

第二个参数是指定的文件夹和名称拼接的字符串。

## 9.3 文件上传表单注意事项

我们开始正式的学习如何来上传文件。上传文件必须在网页中准备好一个form表单。

这是一个简单的HTML页面表单，form表单为文件内容准备了一个专用的类，当选择  的 type=file 时，默认为上传文件内容。

我们来看一下表单的代码和注意项

```
<html>
  <head>
    <meta charset="utf-8" />
    <title>单文件上传</title>
  </head>
  <body>
    <form action="file.php" method="post" enctype="multipart/form-data">
      <input type="file" name="file">
      <input type="submit" value="上传">
    </form>
  </body>
</html>
```

注意事项：

1. form 表单中的参数method 必须为post。若为get是无法进行文件上传的
2. enctype须为multipart/form-data



## 9.4 按照数组和步骤完成文件上传

form表单提交的文件内容指向了file.php。

我们在file.php中，通过PHP代码，来处理上传文件。

我们选择一个图片进行上传。假设图片的名字为：psu.jpg，点击上传。

PHP为文件类数据准备了一个专用的系统函数\$\_FILES，上传文件的所有相关数据，都保存在这个系统函数中。

在PHP文件中，我们打印 \$\_FILES，来观察这个数组的结构：

```
<?php
//var_dump()或print_r()
//打印变量的相关信息,将变量的信息详细的展示出来
var_dump($_FILES);
?>
```

打印出来的结果的数组结构如下：

```
array (size=1)
  'file' =>
    array (size=5)
      //文件名
      'name' => string 'psu.jpg' (length=7)
      //文件的mime类型
      'type' => string 'image/jpeg' (length=10)
      //缓存文件，上传的图片即保存在这里
      'tmp_name' => string 'E:\wamp\tmp\phpC32A.tmp' (length=23)
      //错误码，详见上面错误码介绍
      'error' => int 0
      //上传的文件大小
      'size' => int 225824
```

得到了上面的数组结构。

我们就可以开始文件的处理过程了。

### 第一步，判断错误码：

```
<?php
if($_FILES['file']['error'] > 0){
    switch ($_FILES['file']['error']) { //错误码不为0，即文件上传过程中出现了错误
        case '1':
            echo '文件过大';
            break;
        case '2':
            echo '文件超出指定大小';
            break;
        case '3':
            echo '只有部分文件被上传';
            break;
        case '4':
            echo '文件没有被上传';
            break;
        case '6':
            echo '找不到指定文件夹';
            break;
```

```
        case '7':
            echo '文件写入失败';
            break;
        default:
            echo "上传出错<br/>";
    }
}
}
//错误码为0，即上传成功，可以进行后续处理，处理流程见下文
}
?>
```

上面的代码详细的介绍了错误码和对应的错误，我们可以根据错误码，来生成准确的错误提示。

### 第二步，判断文件是否超出大小。

在实际项目中，由于系统硬件的限制，以及存储设备的限制，不可能让用户无限制的上传文件，所以我们要对用户上传的文件大小进行限制。定义一个合适的限制大小，能让我们的应用更稳定的运行。

```
//判断错误
if($_FILES['file']['error'] > 0){
    //有错误可停止执行
}else{
    //当前上传文件无误，运行本段代码

    //判断文件是否超出了指定的大小
    //单位为byte
    $MAX_FILE_SIZE = 100000;
    if($_FILES['file']['size'] > $MAX_FILE_SIZE){
        //判断，如果上传的文件，大小超出了我们给的限制范围，退上传并产生错误提示
        exit("文件超出指定大小");
    }
}
```

将我们指定的文件大小，定义为\$MAX\_FILE\_SIZE，该变量的计数单位为byte，对应上传文件的\$\_FILES['file']['size']大小。示例代码中，限制大小约为100K及以下的文件。

### 第三步，判断文件的mime类型是否正确。

更多的时候，我们的文件上传功能，都需要判断用户上传的文件，是否符合要求，不可用的文件上传以后，对于线上应用的整体展示效果，会造成恶劣的影响。所以我们需要通过mime类型和后缀名，来判断用户上传的文件是否符合要求。

下面的示例代码中，我们假设当前的项目需求为指定上传图片，要求上传后缀名为GIF或者jpg的文件，当用户上传不符合要求的文件时，返回错误提示。

```
/*判断后缀名和MIME类型是否符合指定需求

例如：
当前项目指定上传后缀为.jpg或.gif的图片，则$allowSuffix = array('jpg','gif');
*/

//定义允许的后缀名数组
$myImg = explode('.', $_FILES['file']['name']);

/*
explode() 将一个字符串用指定的字符切割，并返回一个数组，这里我们将文件名用'.'切割，结果存在$myImg中，文件的后缀名即为数组的最后一个值
*/

$myImgSuffix = array_pop($myImg);

/*
根据上传文件名获取文件的后缀名
使用in_array()函数，判断上传文件是否符合要求
当文件后缀名不在我们允许的范围内时退出上传并返回错误信息
*/
```

## 9.4 按照数组和步骤完成文件上传

```
*/

if(!in_array($myImgSuffix, $allowSuffix)){
    exit("文件后缀名不符");
}

/*
mime类型和文件后缀名的对应关系，我们可以通过很多途径查询到，为了避免用户自主修改文件后缀名造成文件无法使用。
mime类型也必须做出限制检查mime类型，是为了防止上传者直接修改文件后缀名
导致文件不可用或上传的文件不符合要求。
*/

//数组内容为允许上传的mime类型
$allowMime = array(
    "image/jpg",
    "image/jpeg",
    "image/pjpeg",
    "image/gif"
);

if(!in_array($_FILES['file']['type'], $allowMime)){ //判断上传文件的mime类型是否在允许的范围内
    exit('文件格式不正确，请检查');
    //如果不在允许范围内，退出上传并返回错误信息
}
```

### 第四步，生成指定的路径和文件名。

按照项目的文件安排，生成文件存储路径，为了避免文件名重复而产生的错误，按照一定的格式，生成一个随机文件名。

```
//指定上传文件夹
$path = "upload/images/";

/*
根据当前时间生成随机文件名，本行代码是使用当前时间 + 随机一个0-9的数字组合成文件名，后缀即为前面取到的文件后缀名
*/

$name = date('Y').date('m').date('d').date('H').date('i').date('s').rand(0,9).'.'.$myImgSuffix;
```

### 第五步，判断是否是上传文件。

is\_uploaded\_file()函数是专用的函数，来判断目标文件是否是上传文件。

```
<?php

//使用is_uploaded_file()判断是否是上传文件,函数介绍见上文
if(is_uploaded_file($_FILES['file']['tmp_name'])){

}
?>
```

### 第六步，移动文件到指定位置。

使用move\_uploaded\_file()函数，将文件移动到指定的位置，并命名。需要注意的是，Linux系统中对目标目录是否有权限及磁盘空间是否足够，否则会导致上传操作失败。

```
/*
使用move_uploaded_file()移动上传文件至指定位置,第一个参数为上传文件，第二个参数为我们在前面指定的上传路径和名称。
*/

if(move_uploaded_file($_FILES['file']['tmp_name'], $path.$name)){
```

## 9.4 按照数组和步骤完成文件上传

```
        //提示文件上传成功
        echo "上传成功";
    }else{
/*
文件移动失败，检查磁盘是否有足够的空间，或者linux类系统中文件夹是否有足够的操作权限
*/
        echo '上传失败';
    }
}
}
}
}
}
}
?>
```

我们将这个文件片段整理成一整个文件：

```
<?php
if ($_FILES['file']['error'] > 0) {
    switch ($_FILES['file']['error']) {
        //错误码不为0，即文件上传过程中出现了错误
        case '1':
            echo '文件过大';
            break;
        case '2':
            echo '文件超出指定大小';
            break;
        case '3':
            echo '只有部分文件被上传';
            break;
        case '4':
            echo '文件没有被上传';
            break;
        case '6':
            echo '找不到指定文件夹';
            break;
        case '7':
            echo '文件写入失败';
            break;
        default:
            echo "上传出错<br/>";
    }
} else {

    $MAX_FILE_SIZE = 100000;
    if ($_FILES['file']['size'] > $MAX_FILE_SIZE) {
        exit("文件超出指定大小");
    }

    $allowSuffix = array(
        'jpg',
        'gif',
    );

    $myImg = explode('.', $_FILES['file']['name']);

    $myImgSuffix = array_pop($myImg);

    if (!in_array($myImgSuffix, $allowSuffix)) {
        exit("文件后缀名不符");
    }

    $allowMime = array(
        "image/jpg",
        "image/jpeg",
        "image/pjpeg",
        "image/gif",
    );
```

## 9.4 按照数组和步骤完成文件上传

```
if (!in_array($_FILES['file']['type'], $allowMime)) {
    exit('文件格式不正确, 请检查');
}

$path = "upload/images/";
$name = date('Y') . date('m') . date("d") . date('H') . date('i') . date('s') . rand(0, 9) . '.' . $myImgSuffix;

if (is_uploaded_file($_FILES['file']['tmp_name'])) {

    if (move_uploaded_file($_FILES['file']['tmp_name'], $path . $name)) {
        echo "上传成功";
    } else {
        echo '上传失败';
    }

} else {
    echo '不是上传文件';
}

}
?>
```

## 9.5 多文件上传

介绍了PHP上传单个文件的过程。但是有些时候，为了使用方便，我们需要满足同时上传多个文件的需求。多文件上传原理相同，不过在处理数据时，需要对上传数据进行特殊处理。

```
<html>
  <head>
    <meta charset="utf-8" />
    <title>单文件上传</title>
  </head>
  <body>
    <form action="morefile.php" method="post" enctype="multipart/form-data">
      <input type="file" name="file[]">
      <input type="file" name="file[]">
      <input type="submit" value="上传">
    </form>
  </body>
</html>
```

这里是一个简易的上传页面，form表单同时提交了两个文件。我们 可以通过这个页面提交内容。

注意：

1. input type="file" name="file[]"与之前相比file后多加了一个中括号
2. 写了2个或者多个input type="file" name="file[]"

我们使用 \$\_FILES 来接收文件信息，打印并查看数组：

```
<?php
var_dump($_FILES); //打印$_FILES查看数组结构
?>
```

数组结构如下

```
array (size=1)
  'file' =>
    array (size=5)
      'name' =>
        array (size=2)
          //文件名
          0 => string 'psu.jpg' (length=7)
          1 => string 'qwe.jpg' (length=7)
        //文件mime类型
      'type' => array (size=2)
        0 => string 'image/jpeg' (length=10)
        1 => string 'image/jpeg' (length=10)
      //缓存文件
      'tmp_name' =>
        array (size=2)
          0 => string 'E:\wamp\tmp\phpF6D5.tmp' (length=23)
          1 => string 'E:\wamp\tmp\phpF6F5.tmp' (length=23)
      //文件错误信息
      'error' =>
        array (size=2)
          0 => int 0
          1 => int 0
```

```
//文件大小
'size' =>
    array (size=2)
    0 => int 225824
    1 => int 151651
```

我们可以看到，两个文件被存储在一个数组中，键名和上传单文件是相同。所以，需要我们用for()循环，来分别取出两个文件的需要用到的数据。

在\$\_FILES中同时保存了两个文件的数据，我们需要使用一个简单的循环，来读取单个文件的信息，并将文件移动到我们想要放的位置。

```
for ($i=0; $i < count($_FILE['file']['name']); $i++) {

/*
用is_uploaded_file()函数判断是上传文件
并且没有出现错
*/

if(is_uploaded_file($_FILES['file']['tmp_name'][$i]) && $_FILES['file']['error'][$i] == 0){
    if(move_uploaded_file($_FILES['file']['tmp_name'][$i], 'upload/'.$_FILE['file']['name'][$i])){
//用move_uploaded_file()函数移动文件到指定的位置并使用文件原名
echo "上传成功";

        }else{

            echo '上传失败';

        }

    }else{

        echo '上传失败';

    }

}
```

详细的判断过程，参见单文件上传，这里只做了基本的判断，并没有对文件的大小及格式提醒。请按照业务，自行判断文件大小及格式进行错误提醒。

# 9.6 文件上传进度处理

当文件过大，或者用户网络状态一般，通常上传过程需要一段时间，如果这时候让用户白屏等待的话，相信大部分用户都会直接关掉应用，所以一个监控上传进度，并实时向用户报告的需求就被产品汪摆在了桌面上。一个高逼格的上传进度提示，会让你的应用瞬间变成被仰望的存在。

PHP在5.4之前，总是需要安装额外的扩展才能监控到文件上传进度。而从5.4开始，引入session.upload\_progress的新特性，我们只需要在php.ini中开启配置，即可通过session监控文件上传进度。在php.ini中。

**注意：本章学习需要有session基础和javascript和ajax基础。**

我们需要配置，注意查看和修改php.ini文件：

配置项	说明
session.upload_progress.enabled	是否启用上传进度报告(默认开启) 1为开启，0为关闭
session.upload_progress.cleanup	是否在上传完成后及时删除进度数据(默认开启, 推荐开启)
session.upload_progress.prefix[=uploadprogress]	进度数据将存储在_SESSION[session.upload_progress.prefix . _POST[session.upload_progress.name]]
session.upload_progress.name[=PHP_SESSION_UPLOAD_PROGRESS]	如果_POST[session.upload_progress.name]没有被设置, 则不会报告进度.
session.upload_progress.freq[=1%]	更新进度的频率(已经处理的字节数), 也支持百分比表示'%'.
session.upload_progress.min_freq[=1.0]	更新进度的时间间隔(秒级)

开启了配置，我们可以通过session，来记录一个完整的文件上传进度。在session中，会出现一个如下结果的数组：

```
$_SESSION["upload_progress_test"] = array(  
    //请求时间  
    "start_time" => 1234567890,  
    // 上传文件总大小  
    "content_length" => 57343257,  
    //已经处理的大小  
    "bytes_processed" => 453489,  
    //当所有上传处理完成后为TRUE, 未完成为false  
    "done" => false,  
  
    "files" => array(  
        0 => array(  
            //表单中上传框的名字  
  
            "field_name" => "file1",  
  
            //上传文件的名称  
            "name" => "test1.avi",  
  
            //缓存文件，上传的文件即保存在这里  
            "tmp_name" => "/tmp/phpxxxxxx",  
  
            //文件上传的错误信息  
            "error" => 0,
```



## 9.6 文件上传进度处理

```
//是否上传完成，当这个文件处理完成后会变成TRUE
"done" => true,

//这个文件开始处理时间
"start_time" => 1234567890,

//这个文件已经处理的大小
"bytes_processed" => 57343250,
),

1 => array(
    "field_name" => "file2",
    "name" => "test2.avi",
    "tmp_name" => NULL,
    "error" => 0,
    "done" => false,
    "start_time" => 1234567899,
    "bytes_processed" => 54554,
),
)
);
```

这个数组详细记录了文件上传的进度，已经处理完的文件状态为true。下面，我们通过一个jQuery的AJAX实例，来学习一下文件上传进度的流程。

首先，在表单中，需要添加一个type=hidden 的 input 标签，标签 value 为自定义（建议使用有一定意义的值，因为这个值将要在后台用到）

```
<form id="upload-form" action="upload.php" method="POST" enctype="multipart/form-data" style="margin:15px 0" target="hidden_iframe">
    <input type="hidden" name="<?php echo ini_get("session.upload_progress.name"); ?>" value="test" />
    <p><input type="file" name="file1" /></p>
    <p><input type="submit" value="Upload" /></p>
</form>
<div id="progress" class="progress" style="margin-bottom:15px;display:none;">
    <div class="label">0%</div>
</div>
```

这里，添加了一个ID为progress的div，作为展示上传进度的容器。我们通过js的setTimeout()，定时执行ajax来获取文件上传进度，后台文件返回文件上传的进度百分比。

```
<script src="../jquery/1.8.2/jquery.min.js"></script>
<script type="text/javascript">
function fetch_progress(){
    $.get('progress.php',{ '<?php echo ini_get("session.upload_progress.name"); ?>' : 'test'}, function(data){
        var progress = parseInt(data);
        $('#progress .label').html(progress + '%');
        if(progress < 100){
            setTimeout('fetch_progress()', 100);    //当上传进度小于100%时，显示上传百分比
        }else{
            $('#progress .label').html('完成!');    //当上传进度等于100%时，显示上传完成
        }
    }, 'html');
}

$('#upload-form').submit(function(){
    $('#progress').show();
    setTimeout('fetch_progress()', 100);//每0.1秒执行一次fetch_progress()，查询文件上传进度
});
</script>
```

上面这段代码，就是通过JQ的ajax，每0.1秒返回一次文件上传进度。并把进度百分比在div 标签中显示。

后台代码，需要分为两个部分，upload.php处理上传文件。progress.php 获取session中的上传进度，并返回进度百分比。

这里文件上传就不再赘述，详细步骤参见上文，upload.php:

```
<?php
if(is_uploaded_file($_FILES['file1']['tmp_name'])){                                //判断是否是上传文件
    //unlink($_FILES['file1']['tmp_name']);
    move_uploaded_file($_FILES['file1']['tmp_name'], "./{$_FILES['file1']['name']}");    //将缓存文件移动到指定位置
}
?>
```

主要关注progress.php：

```
<?php
/*
开启session。请注意在session_start()之前，请不要有想浏览器输出内容的动作，否则可能引起错误。
*/

session_start();

//ini_get()获取php.ini中环境变量的值
$i = ini_get('session.upload_progress.name');

//ajax中我们使用的是get方法，变量名称为ini文件中定义的前缀 拼接 传过来的参数
$key = ini_get("session.upload_progress.prefix") . $_GET[$i];
//判断 SESSION 中是否有上传文件的信息
if (!empty($_SESSION[$key])) {
    //已上传大小
    $current = $_SESSION[$key]["bytes_processed"];
    //文件总大小
    $total = $_SESSION[$key]["content_length"];

    //向 ajax 返回当前的上传进度百分比。
    echo $current < $total ? ceil($current / $total * 100) : 100;
}else{
    echo 100;
}
?>
```

到这里，文件进度的代码就已经完成了，配合前端，我们就可以做一个炫酷的文件上传功能啦！

## 10.PHP图像处理

图像处理也是PHP在实际工作中经常到遇的一个就是处理图片。

处理图片的场景很多：

1. 生成验证码
2. 图片缩放
3. 图片水印
4. 密保口令卡
5. 柱状图

... ..等

### 一、生成验证码

请先验证图片验证码

X

验证码



确定

### 二、图片缩放

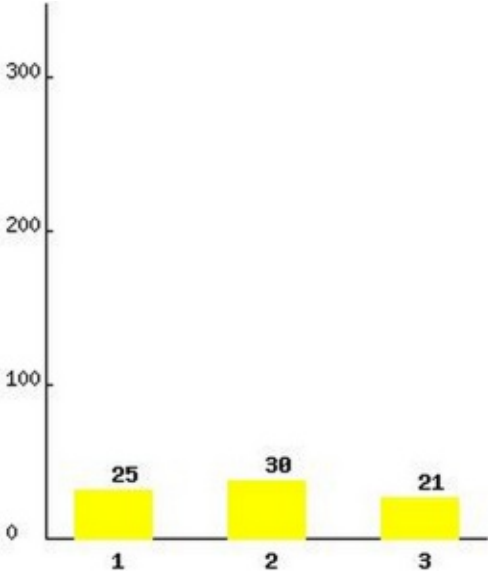


注：实验上的头像原是大图，在微博中变成了小头像。

### 三、图片水印



四、柱状图



五、密保口令卡

SRC:000017755

	T	A	J	M	L	V	U	C	G	H
1	700	551	611	016	326	975	258	188	209	955
2	659	448	897	695	134	051	518	636	194	592
3	220	044	842	219	808	468	217	357	396	301
4	759	951	386	273	415	872	141	483	681	699
5	362	015	377	070	388	278	994	934	263	207
6	837	580	521	832	564	704	591	084	671	401
7	663	754	949	082	321	028	010	904	249	951
8	107	794	503	342	401	432	862	624	151	655

# 10.1 学习前的准备工作

我们通过开篇知道了图像处理的主要作用，功能和未来使用的地方。

学习图像处理重点就是学习PHP的GD系统的函数。

它也可以创建和处理包括 GIF ， PNG ， JPEG ， WBMP 以及 XPM 在内的多种格式的图像。更加方便的是，PHP可以直接将图像数据流输出到浏览器。要想在PHP中使用图像处理功能，你需要连带GD库一起来编译 PHP。GD 库和PHP可能需要其他的库，这取决于你要处理的图像格式。

你可以使用 PHP 中的图像函数来获取下列格式图像的大小： JPEG ， GIF ， PNG ， SWF ， TIFF 和 JPEG2000。

查看我们当前的机器是否安装了GD扩展，我们可以写我们第一天学习到的第一句话：

```
<?php

phpinfo();

?>
```

看下图，如果存在了gd这个选项，就说明我们当前机器支持GD来处理图像：

gd

GD Support	enabled
GD headers Version	2.1.1-dev
FreeType Support	enabled
FreeType Linkage	with freetype
FreeType Version	2.5.2
GIF Read Support	enabled
GIF Create Support	enabled
JPEG Support	enabled
libJPEG Version	8
PNG Support	enabled
libPNG Version	1.2.50
WBMP Support	enabled
XPM Support	enabled
libXpm Version	30411
XBM Support	enabled
WebP Support	enabled

如果没有看到，也没有搜索到。可以按照如下步骤打开gd来处理图像：

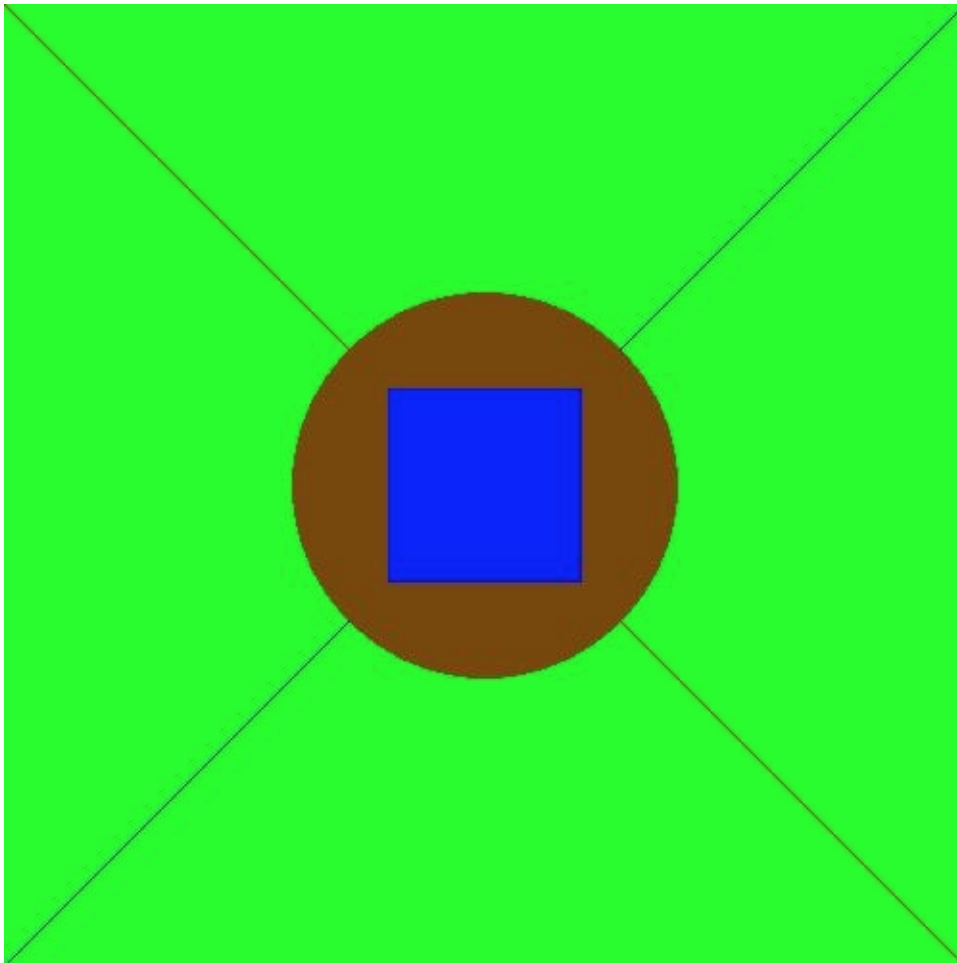
1. 打开php.ini（在文件上传这一章已有说明）
2. 在 Windows 中，需要将GD2的DLL文件php\_gd2.dll作为一个扩展包含在 php.ini 中。修改;extension = php\_gd2.dll，将前面的分号去掉
3. 重新启动apache服务器
4. 再次执行phpinfo尝试查看是否存在gd扩展

注：如果还遇到问题，您也可以在官方网站中提问。



## 10.2 用图片处理函数画一张图

一起来看看下面有一张图：



我们该怎么把这张图给画出来呢。

我们按照步骤能够分析出来：

1. 画图
2. 准备好画这张图需要的颜色
3. 填充背景颜色
4. 画两条对角线
5. 在上面画一个圆
6. 在圆上面画一个矩型
7. 保存图片
8. 销毁资源

**一、我们根据这张图推出出来步骤。我们来分析需要使用到的函数：**

```
//使用imagecreate函数创建图片，返回资源  
$img = imagecreate(500,500);
```

## 二、图片创建完成我们需要向图片资源填加颜色，需要使用到函数

```
$颜色变量 = imagecolorallocate ( resource $图片资源 , int $红 , int $绿 , int $蓝 )
```

红、绿、蓝是计算机里面操作图片的三个基本色。由这三个颜色来组合成我们肉眼所看到的所有颜色。

因此 imagecolorallocate 先输入图片资源，操作这个资源。为这个图片资源准备颜色。

就相当于在画画的时候，先把画布准备好，再准备颜料。

根据这张图，我们需要准备出来的颜色有：

1. 绿
2. 蓝
3. 黑
4. 棕

按照计算机的配色原则分配的话，我们下面的颜色分配的代码就要写成下面的样子：

```
//红
$red = imagecolorallocate($img, 255, 0, 0);
//绿
$green = imagecolorallocate($img, 0, 255, 0);
//蓝
$blue = imagecolorallocate($img, 0, 0, 255);
//棕
$yellow = imagecolorallocate($img, 121, 72, 0);
```

这中图片中需要用到的几个颜色的色值。

## 三、将颜色添加到背景进行填充

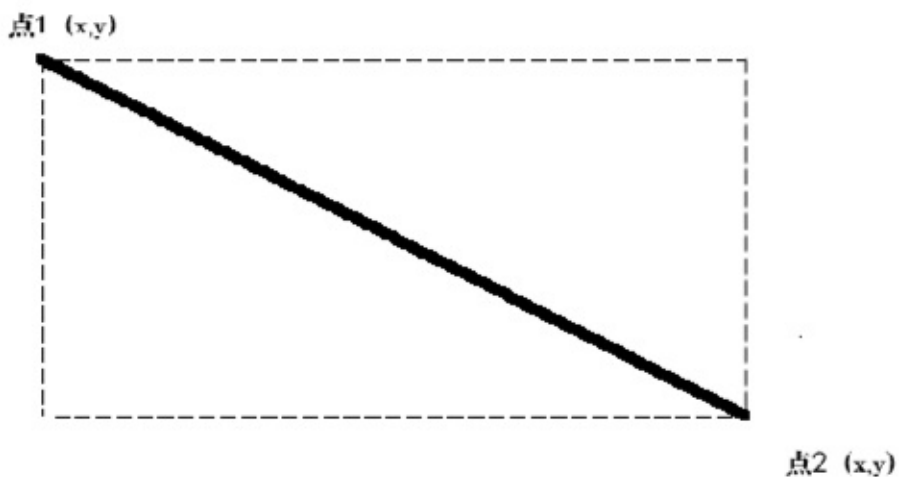
```
imagefilledrectangle ( resource $图片资源 , int $点1x轴 , int $点1y轴 , int $点2x轴 , int $点2y轴 , int $color )
```

这个函数需要涉及到几何的一点点知识。

1. 一个点由x 坐标和y 坐标组成一个点
2. 两个点可以组成一个直线
3. 这条线如果不是水平或者垂直的线可以组成一个矩形

如下图：





点1和点2，可以变成一个矩形。因此，我们输出两个坐标点，可以对画布进行填充。

如果要填充整个画布的话：

点1 为x轴为画布的0位置，点1的y轴也为画布的0位置。

点2 为x轴为画布的500位置，点2的y轴也为画布的500位置。

#### 四、画两条对角线

画一条对角线，对角线是红色。

第一条对角线的坐标为0和0，500和500

第二条对角线的坐标为500和0，0和500

```
imageline($img, 0, 0, 500, 500, $red);
imageline($img, 500, 0, 0, 500, $blue);
```

#### 五、在上面画一个圆

`bool imagefilledellipse ( resource $图片资源 , int $圆心x , int $圆心y , int $圆的宽 , int $圆的高 , int $圆的颜色 )`

```
imagefilledellipse($img, 250, 250, 200, 200, $yellow);
```

操作这个资源，写上圆心的坐标。然后写上长和宽。如果长宽一致为正圆，不一致则为椭圆。

#### 六、在圆上面画一个矩型

```
imagefilledrectangle($img, 200, 200, 300, 300, $blue);
```

这个我们在上面一个中讲过，我们就不细说了。

#### 七、保存图片

```
bool imagejpeg ( resource $image [, string $filename])
```

## 八、销毁图片资源

```
imagedestroy($img);
```

我们来看一下最终组合出来的代码：

```
<?php
//创建图片
$img = imagecreatetruecolor(500, 500);

//分配颜色
$red = imagecolorallocate($img, 255, 0, 0);

$green = imagecolorallocate($img, 0, 255, 0);

$blue = imagecolorallocate($img, 0, 0, 255);

$pur = imagecolorallocate($img, 255, 0, 255);

$yellow = imagecolorallocate($img, 121, 72, 0);

//填充背景
imagefilledrectangle($img, 0, 0, 500, 500, $green);

//画对角线
imageline($img, 0, 0, 500, 500, $red);
imageline($img, 500, 0, 0, 500, $blue);

//画圆
imagefilledellipse($img, 250, 250, 200, 200, $yellow);

//圆中间画矩形
imagefilledrectangle($img, 200, 200, 300, 300, $blue);

//保存图片，图片名为haha.jpg
imagejpeg($img, 'haha.jpg');

//销毁资源
imagedestroy($img);

?>
```

## 10.3 生成验证码

通过上一节，我们已经很好的掌握了图像处理的精髓。

通过我们所掌握的知识。进行一些简单的扩展，我们就可以写出：网上最常用的一个小东西——验证码。

验证码是为了防止机器自动注册、机器自动刷票最常用的手段之一。

我们来看一下验证码的效果：



我们根据上面的效果来推理实现的过程。

实现过程：

1. 生成指定宽高的画布
2. 准备好字好需要生成的字符串
3. 每次执行，让背景填充随机的颜色（浅色系）
4. 在画布上画上随机的干扰元素（随机点、随机线、随机弧形等均可，但不可过份影响用户的视觉）
5. 写上4个文字
6. 输出header头，告知浏览器按照某类型显示
7. 输出图像
8. 销毁图像资源

上面的步骤把实现验证码的每一步都讲清楚了。下面我们来根据这个过程来推理实现。

### 第一步、创建画布

```
$img = imagecreate($width, $height);
```

我们可以定义一个自定义的宽和高。做一个函数将，可以通过函数传入宽和高。这样，可以修改验证码图像的大小。

```
function check_code($width = 100, $height = 50) {  
    $img = imagecreate($width, $height);  
}
```

### 第二步、生成验证码显示的文字

验证码的文字通常有数字、字母。而为了区别验证码字符。我们可以用0-9A-Za-Z。但是0和o，l和I有的时候区分不清楚。我们如果不需要处理，可以想办法去除掉：

方案一：

循环4个ascii码，而ascii码，而chr或者sprintf('%c',第二个参数传ascii码) 将对应的字符转为指定的字符。

## 10.3 生成验证码

```
for ($i = 0; $i < $num; $i++) {  
    $rand = mt_rand(0, 2);  
    switch ($rand) {  
        case 0:  
            $ascii = mt_rand(48, 57); //0-9  
            break;  
        case 1:  
            $ascii = mt_rand(65, 90); //A-Z  
            break;  
  
        case 2:  
            $ascii = mt_rand(97, 122); //a-z  
            break;  
    }  
    //chr()  
    $string .= sprintf('%c', $ascii);  
}
```

方案二：

方案一对很多人来说有点复杂，很多人理解不了ascii码。有没有更简单的方案呢。当然有。我可以准备好字符，然后用str\_shuffle打乱字符后使用substr进行截取。

```
//没有0,i,l,o  
$str = 'abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ123456789';  
  
$str = str_shuffle($str);  
  
$string = substr($str,0 ,3);
```

## 三、 每次执行，让背景填充随机的颜色（浅色系）

文字是深色的看得清楚，而背景是潜色的。

背景的RGB颜色表示通常是这样的，RGB颜色是三个色值，这三个色值都是从0-255。

而：

0-120 低数值是深色系。

130 - 255 通常为浅色系。

使用到图片颜色的地方挺多的。因此，我可以定义分配颜色的功能：

```
//浅色的背景函数  
function randBg($img) {  
    return imagecolorallocate($img, mt_rand(130, 255), mt_rand(130, 255), mt_rand(130, 255));  
}  
  
//深色函数，深色的字或者点这些干 扰元素  
function randPix($img) {  
    return imagecolorallocate($img, mt_rand(0, 120), mt_rand(0, 120), mt_rand(0, 120));  
}
```

## 四、在画布上画上随机的干扰元素

我们可以随机的在图片中画上50个像素点。最小的位置为0，0。最大的位置为最大的宽或者最大的高。

然后使用mt\_rand(0,最大宽)、mt\_rand(0,最大高)。再使用randPix针对我们创建的画布来分配颜色。

```
//画干扰元素

for ($i = 0; $i < 50; $i++) {

    imagesetpixel($img, mt_rand(0, $width), mt_rand(0, $height), randPix($img));

}
```

## 五、写上4个文字

\$string是一个字符串，字符串\$string[0]为字符的第一个字符，依此类类推。

因此，我可在使用imagechar函数在图像中写入文字。

而写入文字的x,y的坐标我们可以用一个公式推倒出来。

X位置 = 图像宽/字符个数(4) \* 循环次数。得到的结果取整后再乘以每次循环的次数。假设图像为100个宽度，那么：第一次写在0的位置，第二次为 25，第三次为50，第四次为75。

Y位置 = mt\_rand(0,图像高- 15)。

我们可以推导出下面的代码，就可以写出对应的文字了：

```
for ($i = 0; $i < $num; $i++) {
    $x = floor($width / $num) * $i;
    $y = mt_rand(0, $height - 15);

    imagechar($img, 5, $x, $y, $string[$i], randPix($img));

}
```

## 六、输出header头，告知浏览器按照某类型显示

我们知道图像类型的输出函数都有：imagejpeg,imagepng,imagegif等特点。

而图像的mime类型为image/jpeg,image/png,image/gif等。

因此，我们可以声明一个变量：

```
$imagetype = 'jpeg';
$imagetype = 'png';
$imagetype = 'gif';
```

输出header类型的时候执行：

```
$header = 'Content-type:image/' . $imagetype;
```

执行函数输出的可以拼接出一个变量函数：

```
$func = 'image' . $type;
```

如果我们的系统支持个类型，则使用function\_exists检测函数是否存在。存在则系统支持该类型，不存在则不支持该类型。

所以代码可以写成这样：

```

$func = 'image' . $type;

$header = 'Content-type:image/' . $type;

if (function_exists($func)) {
    header($header);
    //变为了imagejpeg等
    $func($img);
} else {

    echo '图片类型不支持';
}

```

## 八、销毁资源，返回字符

以后在验证的时候，大家需要把验证码显示出来。并且，用户输出的验证字符还需要与图像中的验证字符匹配。

所以要将验证字符返回回来，供以后保存使用。

```

imagedestroy($img);
return $string;

```

我们来看一下全部的准备完成的PHP函数文件，我们将上面的代码封装成一个统一的函数供大家来学习使用：

```

<?php

check_code();

function check_code($width = 100, $height = 50, $num = 4, $type = 'jpeg') {

    $img = imagecreate($width, $height);
    $string = '';
    for ($i = 0; $i < $num; $i++) {
        $rand = mt_rand(0, 2);
        switch ($rand) {
            case 0:
                $ascii = mt_rand(48, 57); //0-9
                break;
            case 1:
                $ascii = mt_rand(65, 90); //A-Z
                break;

            case 2:
                $ascii = mt_rand(97, 122); //a-z
                break;
        }
        //chr()
        $string .= sprintf('%c', $ascii);
    }
    //背景颜色
    imagefilledrectangle($img, 0, 0, $width, $height, randBg($img));

    //画干扰元素

    for ($i = 0; $i < 50; $i++) {

        imagesetpixel($img, mt_rand(0, $width), mt_rand(0, $height), randPix($img));
    }
    //写字
    for ($i = 0; $i < $num; $i++) {
        $x = floor($width / $num) * $i + 2;
        $y = mt_rand(0, $height - 15);
    }
}

```

### 10.3 生成验证码

```
        imagechar($img, 5, $x, $y, $string[$i], randPix($img));

    }

    //imagejpeg

    $func = 'image' . $type;

    $header = 'Content-type:image/' . $type;

    if (function_exists($func)) {
        header($header);
        $func($img);
    } else {

        echo '图片类型不支持';
    }
    imagedestroy($img);
    return $string;

}
//浅色的背景
function randBg($img) {
    return imagecolorallocate($img, mt_rand(130, 255), mt_rand(130, 255), mt_rand(130, 255));
}
//深色的字或者点这些干 扰元素
function randPix($img) {
    return imagecolorallocate($img, mt_rand(0, 120), mt_rand(0, 120), mt_rand(0, 120));
}

?>
```

# 10.4 图像缩放和裁剪技术

上一章我们学习了图片的水印技术，水印技术和我们这一章要学习到的缩放、裁剪技术基本一致，只不过使用的函数发生了一点点变化。

常用的两个函数如下：

函数名	函数说明
imagecopyresampled	重采样拷贝部分图像并调整大小
imagecopyresized	拷贝部分图像并调整大小

我们来看看这两个函数，这两个函数用起来不难。就是参数略有些复杂。并且来完成图像的缩放或者裁剪：

```
bool imagecopyresampled ( resource $目标图, resource $来源图, int $目标开始的x位置, int $目标开始的y位置, int $来源开始的x位置, int $来源开始的y位置, int $目标图片的宽, int $目标图片的高, int $来源图片的宽, int $来源图片的高 )
```

请注意，上下两个图片的参数都是一样的。

```
bool imagecopyresized ( resource $目标图, resource $来源图, int $目标开始的x位置, int $目标开始的y位置, int $来源开始的x位置, int $来源开始的y位置, int $目标图片的宽, int $目标图片的高, int $来源图片的宽, int $来源图片的高 )
```

图片缩放和裁剪的方法都是一样的，不同的是拷贝时拷贝的是整张图片还是部份部片。

操作方式说明：

从来源图片的开始点(x,y)起，指定的宽高的大小图片。放至到目标图片的起点(x,y)，指定宽高大小的图片中。

## 一、缩放图片，我们将操作步骤化：

- 1. 打开来源图片
- 2. 设置图片缩放百分比（缩放）
- 3. 获得来源图片，按比调整大小
- 4. 新建一个指定大小的图片为目标图
- 5. 将来源图调整后的大小放到目标中
- 6. 销毁资源

我们将范冰冰进行缩放：





```
<?php

//打开来源图片
$image = imagecreatefrompng('fbb.png');

//定义百分比，缩放到0.1大小
$percent = 0.1;

// 将图片宽高获取到
list($width, $height) = getimagesize('fbb.png');

//设置新的缩放的宽高
$new_width = $width * $percent;
$new_height = $height * $percent;

//创建新图片
$new_image = imagecreatetruecolor($new_width, $new_height);

//将原图$image按照指定的宽高，复制到$new_image指定的宽高大小中
imagecopyresampled($new_image, $image, 0, 0, 0, 0, $new_width, $new_height, $width, $height);

header('content-type:image/jpeg');
imagejpeg($new_image);
?>
```

我们来看看缩放后的最终的结果：



## 二、裁剪图片，我们将操作步骤化：

1. 打开来源图片和目标图片
2. 截取来源图片中的点，设置宽高。放至到目标图片中。（裁剪）
3. 保存图片输入
4. 销毁资源

我们将下图中的“度”字裁减出来放至到哈士奇的脸上：

百度logo:



哈士奇图片：



我们来梳理一下思路：

1. 度的开始的坐标x,y轴为：407，154
2. 度这个字从x,y坐标开始的宽、高为：80，89
3. 图片中哈士奇脸的位置的x,y轴位置为：281，71
4. 图片中哈士奇脸的x,y坐标宽、高为：132，160

坐标和宽高我们都清楚了我们开始按照函数的用法来，使用代码来操作图片：

```
<?php
$dst = imagecreatefrompng('hsq.png');
$src = imagecreatefrompng('du.png');

imagecopyresized($dst, $src, 281, 71, 407, 154, 132, 160, 80, 90);
```



## 10.4 图像缩放和裁剪技术

```
header('content-type:image/jpeg');  
imagejpeg($dst);  
  
imagedestroy($dst);  
imagedestroy($src);  
  
?>
```

我们来看看实验效果：



## 10.5 图片水印处理

---

生成水印是整个技术里面最简单的一步。定位水印位置的时候涉及到一点点、最初浅的几何知识。

而上一章我们学习了图片的裁剪技术。水印只不过是图片裁剪技术的一点点的小变形的体现。

一点点几何上的重点知识：

1. 图片大小
2. 图片放在哪个坐标上
3. 图片的宽高

图片水印技术的核心相当于是两张图片：一张大图；一张小图。将小图放置在大图的某个位置上。

水印技术是这个里面最为简单的一个技术，实现方式：

1. 打开原图（也叫操作的目标图片）
2. 打开水印图（也叫水印来源图片）
3. 使用 `imagecopymerge` 将小图合并至大图的指定位置
4. 输出图片
5. 销毁资源

### 一、简单图片水印

需要加水印的目标图片(假设存储在我电脑的d:/www/img/meinv.jpg)，图片如下：



需要加上的logo图片(假设存储在我电脑的d:/www/img/logo.png)，图片如下：



最主要的是要使用这个函数：

```
bool imagecopymerge ( resource $目标图片, resource $来源图片, int $目标开始的x, int $目标开始的y, int $来源的x,
int $来源的y, int $来源的宽, int $来源的高, int $透明度)
```

注意：

透明度的值为0-100的整数。imagecopy和imagecopymerge的区别在于一个有透明度，一个没有透明度。

按照总结的步骤，做一个简单的方法：

```
<?php
//打开目标图片
$dst = imagecreatefrompng('d:/www/img/meinv.jpg');

//打开Logo来源图片
$src = imagecreatefrompng('d:/www/img/logo.png');

//得到目标图片的宽高
$dst_info = getimagesize('fbb.png');

//得到logo图片的宽高
$src_info = getimagesize('logo.png');

//放到最右下角可得出图片水印图片需要开始的位置即：
//x点位置：需要大图的宽 - 小图的宽；
//y点位置：放大图的高 - 小图的高

$dst_x = $dst_info[0] - $src_info[0];

$dst_y = $dst_info[1] - $src_info[1];

//要将图片加在右下角
imagecopymerge($dst, $src, $dst_x, $dst_y, 0, 0, $src_info[0], $src_info[1], 100);

header('Content-type:image/png');
imagepng($dst);

imagedestroy($dst);

imagedestroy($src);

?>
```



我们看看最终的效果如下：



## 二、做一个智能的图片水印函数

一、我们可以做一个自动化打开图片的函数

之前创建图片或者打开图片的函数我们都学习过：

1. imagecreate
2. imagecreatetruecolor
3. imagecreatefromjpeg等



我们来推理下。我们如果能够想办法得到图片的MIME类型，根据MIME类型找到打开该文件的函数就行了。

因此，做这一步分为两块来完成：

1. 得到文件MIME类型，返回类型。
2. 传入路径，打开函数，返回资源。

因此，上面两块，我们都可以做成两个函数。

传入图片的路径，将图片的宽、高、图片的MIME类型全部返回一个数组，需要的时候使用对应的参数即可。

我们可以将mime类型传到到\$data当中的type关联数组中。代码如下：

```
function getImageInfo($path) {
    $info = getimagesize($path);
    $data['width'] = $info[0];
    $data['height'] = $info[1];
    $data['type'] = $info['mime'];
    return $data;
}
```

打开文件的函数，传入一个图片的类型，传入一个图片的路径就打开了图片，返回成了资源类型。

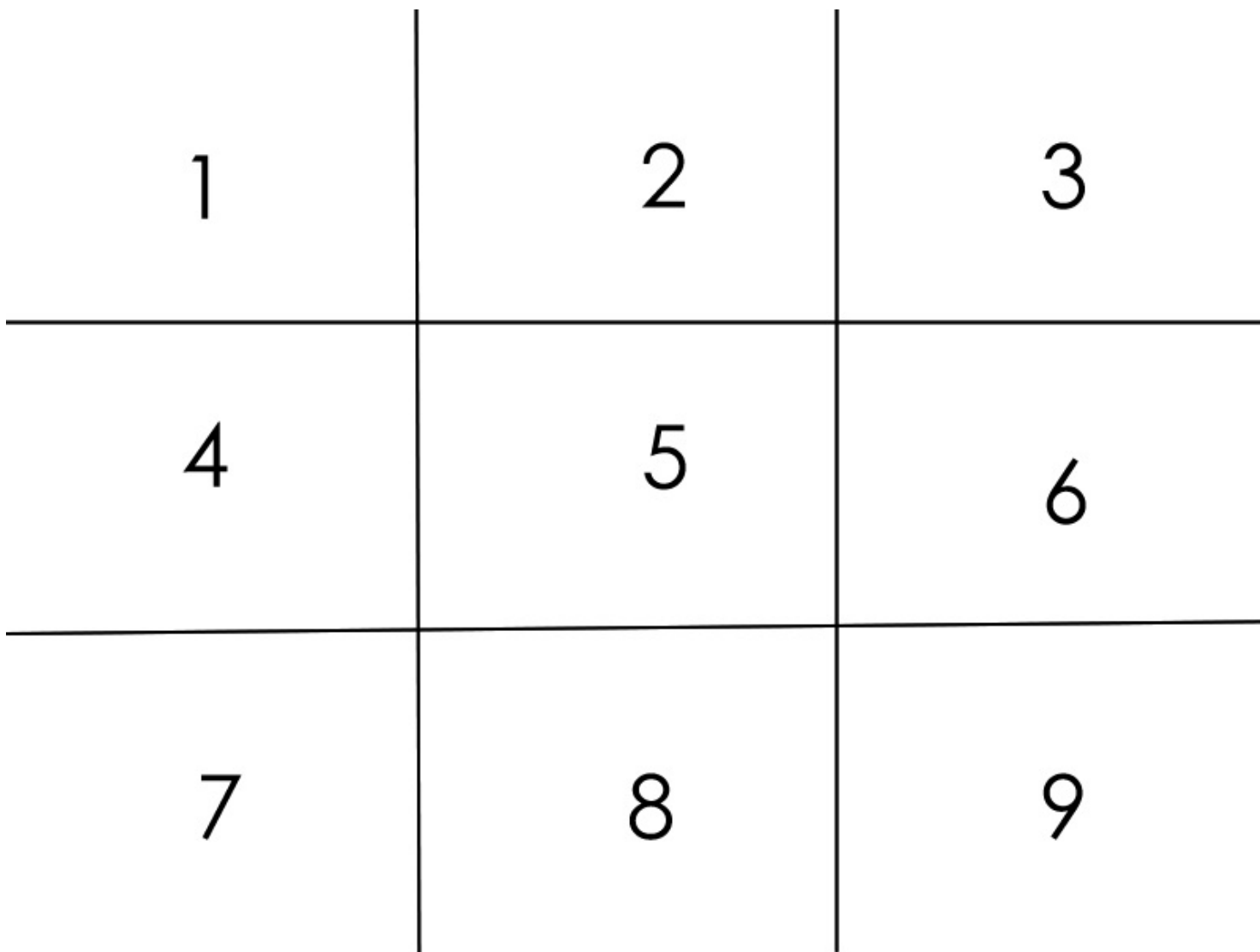
下面的例子中，\$type使用switch...case进行判断，如果是imagejpeg就使用imagecreatefromjpeg来打开\$path中路径指定的文件。最后，返回一个资源类型。

```
function openImg($path, $type) {
    switch ($type) {
        case 'image/jpeg':
        case 'image/jpg':
        case 'image/pjpeg':
            $img = imagecreatefromjpeg($path);
            break;
        case 'image/png':
        case 'image/x-png':
            $img = imagecreatefrompng($path);
            break;
        case 'image/gif':
            $img = imagecreatefromgif($path);
            break;
        case 'image/wbmp':
            $img = imagecreatefromwbmp($path);
            break;
        default:
            exit('图片类型不支持');
    }
    return $img;
}
```

自动计算位置：

我们可将位置分为10个值，分别为0-9。

我们用画图来表示位置：



注：

0为随机位置，可出现在页面中的任意处。但是不能超过图片的范围。

0的位置为：

```
x = 0 至 (大图宽 - 小图宽)
y = 0 至 (大图高 - 小图高)
```

1的位置为：

```
x = 0
y = 0
```

2的位置为：

```
x = (大图宽 - 小图宽) / 2
y = 0
```

3的位置为：

## 10.5 图片水印处理

```
x = 大图宽 - 小图宽  
y = 0
```

4的位置为：

```
x = 0  
y = (大图高 - 小图高) / 2
```

... ..依此类推。

我们来推理0-9的实现代码：

```
switch($pos){  
  case 1:  
    $x=0;  
    $y=0;  
    break;  
  case 2:  
    $x=ceil(($info['width']-$logo['width']/2);  
    $y=0;  
    break;  
  case 3:  
    $x=$info['width']-$logo['width'];  
    $y=0;  
    break;  
  case 4:  
    $x=0;  
    $y=ceil(($info['height']-$logo['height']/2);  
    break;  
  case 5:  
    $x=ceil(($info['width']-$logo['width']/2);  
    $y=ceil(($info['height']-$logo['height']/2);  
    break;  
  case 6:  
    $x=$info['width']-$logo['width'];  
    $y=ceil(($info['height']-$logo['height']/2);  
    break;  
  
  case 7:  
    $x=0;  
    $y=$info['height']-$logo['height'];  
    break;  
  case 8:  
    $x=ceil(($info['width']-$logo['width']/2);  
    $y=$info['height']-$logo['height'];  
    break;  
  case 9:  
    $x=$info['width']-$logo['width'];  
    $y=$info['height']-$logo['height'];  
    break;  
  case 0:  
  default:  
    $x=mt_rand(0,$info['width']-$logo['width']);  
    $y=mt_rand(0,$y=$info['height']-$logo['height']);  
    break;  
}
```

最后调用一下图片的合并、输出和销毁代码即可：

```
imagecopymerge($dst,$src,$x,$y,0,0,$logo['width'],$logo['height'],$tm);
```

我们将最终的代码整合好后给大家实验看效果：

```
<?php

water('zxy.jpg', 'logo.gif', 0, 50);

function water($img, $water, $pos=9, $tm=100){

    $info=getImageInfo($img);

    $logo=getImageInfo($water);

    $dst=openImg($img, $info['type']);
    $src=openImg($water, $logo['type']);

    switch($pos){
        case 1:
            $x=0;
            $y=0;
            break;
        case 2:
            $x=ceil(($info['width']-$logo['width'])/2);
            $y=0;
            break;
        case 3:
            $x=$info['width']-$logo['width'];
            $y=0;
            break;
        case 4:
            $x=0;
            $y=ceil(($info['height']-$logo['height'])/2);
            break;
        case 5:
            $x=ceil(($info['width']-$logo['width'])/2);
            $y=ceil(($info['height']-$logo['height'])/2);
            break;
        case 6:
            $x=$info['width']-$logo['width'];
            $y=ceil(($info['height']-$logo['height'])/2);
            break;

        case 7:
            $x=0;
            $y=$info['height']-$logo['height'];
            break;
        case 8:
            $x=ceil(($info['width']-$logo['width'])/2);
            $y=$info['height']-$logo['height'];
            break;
        case 9:
            $x=$info['width']-$logo['width'];
            $y=$info['height']-$logo['height'];
            break;
        case 0:
        default:
            $x=mt_rand(0, $info['width']-$logo['width']);
            $y=mt_rand(0, $info['height']-$logo['height']);
            break;

    }
    imagecopymerge($dst, $src, $x, $y, 0, 0, $logo['width'], $logo['height'], $tm);

    imagejpeg($dst);

    imagedestroy($dst);
    imagedestroy($src);

}

function openImg($path, $type){
    switch($type){
```

```
        case 'image/jpeg':
        case 'image/jpg':
        case 'image/pjpeg':
            $img=imagecreatefromjpeg($path);
            break;
        case 'image/png':
        case 'image/x-png':
            $img=imagecreatefrompng($path);
            break;
        case 'image/gif':
            $img=imagecreatefromgif($path);
            break;
        case 'image/wbmp':
            $img=imagecreatefromwbmp($path);
            break;
        default:
            exit('图片类型不支持');
    }
    return $img;
}

?>
```

本文仅为技术人员交流学习、交流技术使用。

本文中所使用到的图像：

1. 范冰冰女士的形象照片不可用于商业使用。所有权均为范冰冰女士及相关机构所有。
2. 本文中所使用到的logo归百度公司所有。

特此声明！

## 11.错误处理

在之前我们在写代码的时候经常会看到：函数名写错了，忘加分号了，函数被重新定义了都会报各种不同样的错。

在开发中，显示错误对我们的开发非常有利。因为，显示错误后能帮我们快速定位错误、解决问题。

而在生产环境（即公网）给其他人访问的网站、微网站、手机网站、手机接口等等。

如果错误显示出来了，就容易暴露：

1. 服务器的文件路径和文件存储规范
  2. 有些人喜欢用个人名命名，通过社会工程学可以反向推理出密码
  3. 有时还会暴露mysql数据库服务器的地址
- ... .. 等等

上面这些信息特别容易被网上别有用心的一些人给利用。

例如下面这段代码，我们不加分号就全面暴露了我们的服务器端文件存放路径、框架信息等。如下：

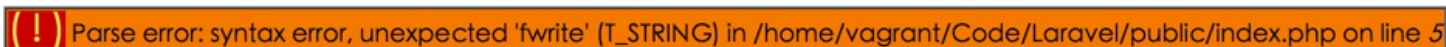
```
<?php

$fp = fopen('abc.txt','a+')

fwrite($fp,'abc');

fclose($fp);
?>
```

报错了：



错误提示中文翻译过来为：

解析错误：语法错误。意外发生在/home/vagrant/Code/Laravel/public/index.php 文件的第5行fwrite附近。

### 那，我们该如何解决了？

——这就需要学习我们的错误处理章节。

## 11.1 禁止显示错误

---

在php.ini配置文件中（见9.1章打开php.ini）。我们可以控制php的错误显示状态。

php.ini中有一个专门的配置项：

### **display\_errors**

这个选项设置是否将错误信息输出到网页，或者对用户隐藏而不显示。

这个值的状态为on 或者 off，也可以设值为1 或者0。

display\_error的值设为0或者off则不在页面中显示错误，如果设为1或者on则显示错误信息。

### **问题：如果没有修改服务器php.ini的状态权限怎么办？**

可以使用ini\_set。

```
<?php
ini_set('display_errors' , 0 );
?>
```

上面的代码也相当于修改了php.ini中display\_errors的值。不过，仅仅在当前php代码中生效。

### **问题：想取得php.ini的配置项状态怎么办？**

可以使用ini\_get(参数项) 得到参数的值。

演示例子：

```
<?php
echo '服务器中display_errors的状态为' . ini_get('display_errors');
?>
```

---

注：修改完php.ini文件，需要重启服务器。

# 11.2 错误报告级别

## 错误类型

php大家最常见的错误显示截图，本书帮大家进行了总结：

### 【掌握级别的错误类型】

我们将最常用的错误分为了三种：

错误类型	说明
E_ERROR	错误，文件直接中断
E_WARNING	警告，问题比较严重。但是还会继续向下运行
E_NOTICE	提示，有些小问题不会影响到程序。常发生在项目未定义
E_PARSE	编译时语法解析错误。解析错误仅仅由分析器产生。
E_ALL	所有的错误
E_STRICT	启用PHP对代码的修改建议，以确保代码具有最佳的互操作性和向前兼容性。
E_DEPRECATED	启用后将会对在未来版本中可能无法正常工作的代码给出警告。

在上面的几种类型中：

- 1. error最严重，必须要解决。不然程序无法继续向下执行
- 2. warning也很重要。通常也必须要解决。如果明确的，故意的可以不用处理。
- 3. notice 你可以不用管。但是在有些公司，项目标准特别高。在高标准要求的项目中也必须要解决。因为，notice会影响到PHP的执行效率。通常发生在函数未定义等。
- 4. parse错误，是指语法错写错了，必须要解决
- 5. 代表全部类型的所有错误

### 【了解级别的错误类型】

再从这三种扩展出来了另外一些需要了解的错误项：

错误类型	错误说明
E_CORE_ERROR	在PHP初始化启动过程中发生的致命错误。该错误类似E_ERROR，但是是由PHP引擎核心产生的
E_CORE_WARNING	PHP初始化启动过程中发生的警告 (非致命错误)。类似 E_WARNING，但是是由PHP引擎核心产生的。
E_COMPILE_ERROR	致命编译时错误。类似E_ERROR,但是是由Zend脚本引擎产生的。
E_COMPILE_WARNING	编译时警告(非致命错误)。类似E_WARNING，但是是由Zend脚本引擎产生的
E_USER_ERROR	用户自定义错误
E_USER_WARNING	用户自定义警告
E_USER_NOTICE	用户自定义提示
E_USER_DEPRECATED	用户产少的警告信息。类似E_DEPRECATED,但是是由用户自己在代码中使用PHP函数 trigger_error()来产生的。
E_RECOVERABLE_ERROR	可被捕捉的致命错误。它表示发生了一个可能非常危险的错误，但是还没有导致PHP引擎



E_RECOVERABLE_ERROR	处于不稳定的状态。
---------------------	-----------

在学习过程中，上面的类型了解即可。因为基本不会遇到，遇到了大家查一下本书或者查一下手册就清楚了。

## error\_reporting 报告错误类型

error\_reporting 是指错误报告。在php.ini中也有这样一个参数。这个参数。决定了PHP引擎记录、报告、显示哪些错误类型。

一、在php.ini中error\_reporting参数。如若error\_reporting参数设置为0。整个PHP引擎发错误均不会显示、输出、记录。在下一章将要讲到的日志记录中，也不会记录。

**如果我们想显示所有错误可以写上：**

```
error_reporting = E_ALL
```

**想要显示所有错误但排除提示，可以将这个参数写为：**

```
error_reporting = E_ALL & ~ E_NOTICE
```

**显示所有错误，但排除提示、兼容性和未来兼容性。可写为：**

```
error_reporting = E_ALL & ~E_NOTICE & ~E_STRICT & ~E_DEPRECATED
```

二、在有些情况下我们无权限操作php.ini文件，又想要控制error\_reporting怎么办呢？

在运行的xxxx.php文件中开始处，我们可以使用error\_reporting()函数来达到目标。

演示代码如下：

```
<?php
//关闭了所有的错误显示
error_reporting(0);

//显示所有错误
//error_reporting(E_ALL);

//显示所有错误，但不显示提示
//error_reporting(E_ALL & ~ E_NOTICE);
?>
```

上面的代码你可以试试，故意写错代码看看。在当前文件中还会不会显示指定的错误。

**[扩展、了解知识点]：**

@ 符是我们之前学习过的单行不显示错误，请不用或者少用@符。

我们拿读取一个不存在的文件，这样的php代码来演示实现过程：

```
<?php
//读取一个不存在的adsaf.txt文件，用@符抑制错误
@$fp = fopen('adsaf.txt','r');
?>
```

@符效率较低，它在php内核中的实现过程是：

```
<?php
//关闭错误
error_reporting(0);

//读取一个不存在的文件，显示错误

//显示错误
error_reporting(E_ALL & ~ E_NOTICE);
?>
```

# 11.3 错误记录日志

在一些公司里面，有专门的日志收集系统。日志收集系统会在背后默默的帮你收集错误、警告、提示。

也有些公司没有专门的日志收集系统，通过文件来服务器当中的运行日志。

其中：PHP的错误，警告这些是必须要收集的。

那么问题来了——不让用户看到，设置好的错误报告级别号，如何将错误收集到日志系统中呢？

这里有需要使用到php.ini的相关配置项。这两个配置项为：

参数	配置项	说明
log_errors	on/off	是否开启日志记录
log_errors_max_len	整型，默认1024	单行错误最大记录长度
error_log	syslog或者指定路径	错误日志记录在什么地方

说明：

- 1. 在表格中的log\_errors和log\_errors\_max\_len非常好理解。
- 2. 而error\_log 指定将错误存在什么路径上。配置项中的syslog可能有点不太好理解。syslog是指系统来记录。windows系统在电脑的日志收集器里面。linux默认在：/etc/syslog.conf

## [扩展]

了解知识点。若Linux系统启动或修改了日志收集。可能存储在第三方专用的日志收集服务器中。

此外，PHP还为我们专门准备了一个自定义的错误日志函数：

```
bool error_log ( string $错误消息 [, int $错误消息类型 = 0 [, string $存储目标]] )
```

这个函数可以把错误信息发送到web服务器的错误日志，或者到一个文件里。

常用的错误消息类型：

错误消息类型	说明
0	发送至默认的错误_log指定位置
1	发送到指定的邮件位置
3	发送至指定的文件位置

示例：

```
<?php

//无法连接到数据库服务器，直接记录到php.ini 中的error_log指定位置
error_log("无法连接到数据库服务器服务器");

//可以发送邮件，但是php.ini必须配置过邮件系统
error_log('可以用邮件报告错误，让运维人员半夜起床干活',1 , 'liwenkai@phpxy.com');
```

## 11.3 错误记录日志

```
//记录在指定的位置
error_log("我是一个错误哟", 3, "d:/test/my-errors.log");

?>
```

---

注意：

error\_log 中发送邮件可能对初学者不熟，您可以不用掌握这一块知识。

## 11.4 自定义错误处理函数

这一块知识起点有些高。大多数人没有软件工程、自定义错误处理的经验，很难想象出使用的场景。若你想跳过此块的学习，完全可以，并且我们支持。

此块知识点对于实际应用中应用场景不多。如果，有计划开始要自己写框架时、或者您做完了本书的第一个项目。

您可以回头再来看11.4这一章的内容。

用户自定义错误经常用到的两个函数：

`set_error_handler ( callable $回调的错误处理函数)`

设置一个用户定义的错误处理函数

`trigger_error ( string $error_msg)`

产生一个用户级别的 error/warning/notice 信息

```
<?php
//定义一个自定义的错误处理函数
function customError($errno, $errstr, $errfile, $errline) {
    //输出错误消息
    echo "<b>Custom error:</b> [$errno] $errstr<br />";
    //输出错误文件和错误行
    echo "Error on line $errline in $errfile<br />";
    echo "Ending Script";
    //中止程序运行
    exit;
}

//使用set_error_handler 绑定用户自定义函数
set_error_handler("customError");

$test=2;

//触发自定义错误
if ($test > 1) {
    trigger_error("A custom error has been triggered");
}
?>
```

## 12.MySQL 入门

### MySQL 学习的要点

**MySQL对于PHP程序员来说就是将业务转化成表结构。做好业务中的增、删、改、查。**

如果你不知道什么是MySQL我们来介绍一下MySQL吧。

MySQL数据库简称MySQL，是一款由瑞典MySQL AB公司开发并且应用广泛的数据管理系统，MySQL数据库因其体积小、速度快、总体拥有成本低受到很多的热捧。现在MySQL归世界上最著名的数据库企业——Oracle所有。

MySQL的应用，在国内的企业包括：百度、阿里、腾讯、新浪、搜狐、网易等等企业，全部都在使用MySQL数据库。

MySQL是一种开放源代码的关系型数据库管理系统（RDBMS），MySQL数据库系统使用最常用的数据库管理语言--结构化查询语言（SQL）进行数据库管理。

由于MySQL是开放源代码的，因此任何人都可以在GPL的许可下下载并根据个性化的需要对其进行修改。MySQL因为其速度、可靠性和适应性而备受关注。大多数人都认为在不需要事务化处理的情况下，MySQL是管理内容最好的选择。

PHP与很多数据库结合都很紧密。由于，PHP和MySQL都是开源免费的。所以PHP一直对于MySQL等数据库都有很好的支持。

最开始学PHP的时候，通常我们也将数据放在MySQL数据库里面。因此，PHP程序员对于MySQL来说是必学课程。

很多朋友在学MySQL的初期，还走了很多不必要的弯路，学了很多原本不需要掌握的知识点，耽误了大量宝贵的学习时间。

在最开始学习MySQL的时候，并不需要学习MySQL数据库，学到DBA（数据库管理员）的级别。

写过两年代码后。很多朋友会遇到一个瓶颈期。感触最深刻的是：

业务代码里面写的最多的就是增、删、改、查。

为了帮大家快速学习和入门，我帮大家总结最精华的干货。本章节可以说是一本：《mysql 精华快速入门》。

只需要学好这些内容，mysql你就入门了！

学好增、删、改、查。写业务我都不怕！

ps：你能想象像新浪、百度、搜狐等。其中的某些大公司使用分库分表和数据库中间件技术后后，有些甚至不准使用到联合查询吗？



## 12.1 请进入《MySQL入门》

---

我们专门写了一本MySQL入门的书，帮你来学习MySQL

### **《MySQL入门》地址**

<http://mysql.phpxy.com>



# 13. PHP操作mysql数据库

通过上一章的学习，我们学习完了MySQL。PHP向MySQL发送数据、PHP操作MySQL数据库是重点中的重点。

为大家举一些例子：

- 1. 要注册一个用户，是将表单的数据POST发送给PHP写入数据库
- 2. 购买一个商品，是将商品信息和用户信息通过PHP写入到数据库
- 3. 在线付费，是将用户的充值信息通过PHP写入数据库
- 4. 修改头像上传的头像地址得到后，通过PHP修改数据库里头像字段的值

... 太多太多的应用场景。从网页、移动端、QQ微信公众号都在使用PHP连接数据库进行操作。

为了方便大家的学习，我们将连接数据库的知识进行了步骤化。你会发现，你很轻松的就学会了PHP连接数据库的知识。

我们准备的数据库连接的知识，几乎适用于所有的数据库连接的方式。

您也可以使用这一套方案和步骤用于PDO，PgSQL等数据库连接函数使用。

在正式开始学习前，我们需要开启mysqli扩展，使用phpinfo()你可以看到如下展示就说明开启成功：

## mysqli

Mysqli Support	enabled	
Client API library version	mysqlnd 5.0.11-dev - 20120503 - \$Id: 3c688b6bbc30d36af3ac34fdd4b7b5b787fe5555 \$	
Active Persistent Links	0	
Inactive Persistent Links	0	
Active Links	0	

Directive	Local Value	Master Value
mysqli.allow_local_infile	On	On
mysqli.allow_persistent	On	On
mysqli.default_host	no value	no value
mysqli.default_port	3306	3306
mysqli.default_pw	no value	no value
mysqli.default_socket	/var/run/mysqld/mysqld.sock	/var/run/mysqld/mysqld.sock
mysqli.default_user	no value	no value
mysqli.max_links	Unlimited	Unlimited
mysqli.max_persistent	Unlimited	Unlimited
mysqli.reconnect	Off	Off
mysqli.rollback_on_cached_plink	Off	Off

若没有mysqli这个扩展模块。在《10.PHP图像处理》章节跟大家一起学习过，若没有看到mysqli扩展在windows服务器下，打开php.ini文件，将php\_mysqli.dll打开即可。

注意：

从PHP7开始默认不再支持mysql扩展，即不再支持mysql\_\*系列函数。请使用mysqli连接数据库。

mysqli即支持php5也支持php7。

# 13.1 数据库连接步骤

我们为大家将数据库连接整理成了最重要的8个步骤，我戏称它为：“数据库连接天龙八部”。

这八个步骤如下，并且将每一步使用的函数都做了说明：

## 第一步: 连接数据库服务器

类型	说明
函数	mysqli_connect
功能	连接到mysql数据库服务器
参数1	主机
参数2	数据库服务器登陆名
参数3	密码
参数4	数据库的名称
参数5	数据库服务器端口不填默认3306

若参数4，数据库名称在此步已填，则不需要再执行第三步。

## 第二步: 判断错误

类型	说明
函数	mysqli_errno
功能	返回连接错误号，无错误返回0
参数1	传入mysqli_connect返回的资源

类型	说明
函数	mysqli_error
功能	返回连接错误字符串
参数1	传入mysqli_connect返回的资源

## 第三步: 选择数据库

类型	说明
函数	mysqli_select_db
功能	选择本连接中的数据库
参数1	传入mysqli_connect返回的资源
参数2	需要连接的数据库名

若在第一步已填数据库，不需要更换成其他数据库，则不需要执行第三步。

## 第四步: 设置字符集

类型	说明
函数	mysqli_set_charset

功能	设置与mysql服务器连接，结果，校验字符集
参数1	传入mysqli_connect返回的资源
参数2	字符集类型

更多注意项，请关注本书《13.6 数据显示乱码终极解决方案》

## 第五步: 准备SQL语句

其实就是一个SQL语句的字符串。

例如：

```
$sql = "insert into user(username,password) values('$username','$password');"
```

我们通常要把变量赋值在SQL语句中使用。可是变量或者SQL语句出错了，非常不好排查。

我们根据实际工作经验增加了这一步。

如果在执行此步的时候报错了，我们可以把SQL语句打印出来，粘贴到phpMyAdmin或者相关工具中。

排错时，如果执行成功就说明不是SQL语句的问题。如果执行失败，请仔细检查SQL语句。

## 第六步: 发送SQL语句

类型	说明
函数	mysqli_query
功能	发送SQL语句
参数1	传入mysqli_connect返回的资源
参数2	传入发送的SQL语句

SQL语句准备完成，需要通过mysqli\_query将SQL语句发送给MySQL服务器。

MySQL服务器会执行发送过来的SQL语句进行执行。

## 第七步: 判断是否执行正常或者遍历数据

读取

第6步中，发送的是select类别的语句，通常需要将结果输出显示出来。就需要用到遍历显示数据的函数。

类型	说明
函数	mysqli_fetch_array
功能	得到result结果集中的数据，返回数组进行便利
参数1	传入查询出来的结果变量
参数2	传入MYSQLI_NUM返回索引数组，MYSQLI_ASSOC返回关联数组，MYSQLI_BOTH返回索引和关联

类型	说明
函数	mysqli_fetch_assoc
功能	得到result结果集中的数据，返回关联数组进行便利
参数1	传入查询出来的结果变量

类型	说明
函数	mysqli_fetch_row
功能	得到result结果集中的数据，返回索引数组进行便利
参数1	传入查询出来的结果变量

类型	说明
函数	mysqli_fetch_object
功能	得到result结果集中的数据，返回对象进行遍历
参数1	传入查询出来的结果变量

类型	说明
函数	mysqli_num_rows
功能	返回查询出来的结果总数
参数1	传入查询出来的结果变量

类型	说明
函数	mysqli_num_rows
功能	返回查询出来的结果总数
参数1	传入查询出来的结果变量
注	实际工作中用得非常少，了解

写入

第6步中，如果发送的是insert的语句，通常需要得到是否执行成功，或者同时拿到自增的ID。

类型	说明
函数	mysqli_fetch_field
功能	遍历数据行
参数1	传入查询出来的结果变量

修改和删除

第6步中，如果发送的是update和delete类别的语句。只需要判断是否执行成功即可。

我们将这些常用函数列出数据表给大家查看。

第八步: 关闭数据库

类型	说明
函数	mysqli_close
功能	关闭数据库连接
参数1	传入mysqli_connect返回的资源

数据库连接是一个资源类型。我们在之前的章节中讲解资源类型的时候跟大家说过。凡是涉及到数资源类型的有打开就有关闭。这样能够保证PHP更高效的处理和回收资源。

因此，数据库连接成功后，不需要使用的时候。我们可以关闭这个连接。

其他：显示服务器信息函数

类型	说明
函数	mysqli_get_server_info
功能	返回服务器信息
参数1	传入mysqli_connect返回的资源

类型	说明
函数	mysqli_get_server_version
功能	返回服务器版本
参数1	传入mysqli_connect返回的资源

注意：  
mysqli只学过程化的方法即可。在面向对象阶段实际工作中完全抛弃了mysqli的对象用法，而是使用的是PDO对象连接数据库的方式。

# 13.2 通过步骤做一个用户注册

我们做一个最简单的注册页面。注册页面中有三个参数：

- 1. 用户名
- 2. 密码
- 3. 重复密码

用户写好三个参数后，点击提交的时候向connect.php页面中传入POST记录。

我们可以把POST记录处理后写入到MySQL数据库中，即完成了用户注。

代码如下：

```
<form action="connect.php" method="post">

    用户名:<input type="text" name="username"><br />

    密码:<input type="password" name="password"><br />

    重复密码:<input type="password" name="repassword"><br />

    <input type="submit" value="提交">

</form>
```

为了更快的表现我们的代码界面没有进行美化。以最快的速度带大家完成用户注册。

## 一、判断重复密码

由于有重复密码，如果用户两次输入的密码不一致也就是有没有进行下一步的任何意义。

在网页中很多地方还是使用到了重复密码。因为，害怕的是用户产生手误。将密码填写出错。

用户在输入密码的时候可能在左右两边多打两个空格。因此，我们会使用trim将密码和重复密码的两边去掉空格。

```
if(trim($_POST['password']) != trim($_POST['repassword'])){

    exit('两次密码不一致,请返回上一页');

}
```

## 二、准备好写入的数据

我们需要把用户的输入数据和隐藏的数据都写入到数据库。

可见数据有：

变量	说明
\$_POST['username']	用户名
\$_POST['password']	密码

## 13.2 通过步骤做一个用户注册

1. 我们需要把用户名去掉两边的空格，这样避免输入不必要的这些信息。
2. 在mysql这一章节我们讲过，用户的密码不要让包括公司内部人员可见。保证密码是不可逆向的。在初级阶段大家学习一下MD5即可。以后我们再教大家其他的加密方式。

不可见数据有：

变量	说明
\$time	用户的注册时间
\$_SERVER['REMOTE_ADDR']	用户的注册IP

1. time返回的unix时间戳
2. REMOTE\_ADDR返回的是IP地址，我们可以用ip2long将其转为整型存储。

```
$username = trim($_POST['username']);  
  
$password = md5(trim($_POST['password']));  
  
$time = time();  
  
$ip = ip2long($_SERVER['REMOTE_ADDR']);
```

### 三、连接数据库、判断错误、选择库和字符集

1. 我们使用mysqli\_connect连接到数据库服务器。
2. 如果有错误，使用mysqli\_errno得到错误号
3. 如何时存在错误mysqli\_error打印出所有的错误，并且退出程序执行
4. 选择数据库并且设置字符集为utf8.

```
//连接数据库  
$conn = mysqli_connect('localhost','root','liwenkaihaha');  
  
//如果有错误，存在错误号  
if(mysqli_errno($conn)){  
  
    echo mysqli_error($conn);  
  
    exit;  
}  
  
mysqli_select_db($conn, 'user');  
  
mysqli_set_charset($conn, 'utf8');
```

### 四、组合SQL语句

我们需要把得到的信息写入到数据库里面去，用户名、密码、创建时间、IP我们都得到了。

将对应的变量插入到SQL语句中即可。组合出来的SQL语句如下：

```
$sql = "insert into user(username,password,createtime,createip) values('" . $username . "','" . $password . "','" . $time .  
"','" . $ip . "')";
```

而我们的创建表的语句如下：

```
CREATE TABLE IF NOT EXISTS user (
  id int(11) NOT NULL,
  username varchar(30) NOT NULL,
  password char(32) NOT NULL,
  createtime int(11) NOT NULL,
  createip int(11) DEFAULT NULL
) ENGINE=InnoDB DEFAULT CHARSET=utf8;
```

表的格式（字段对应说明）：	id	username	password	createtime	createip
用户编号	用户名	密码	创建时间	创建IP	

五、发送语句，判断状态

mysqli\_query我们在上面说过，需要传入两个参数：

- 1. 连接的资源，在这里对应的变量是\$conn。
- 2. 发送的SQL语句。在上面已经准备好了\$sql。

SQL语句可以通过mysqli\_query发送给MySQL服务器。发送成功\$result则为true。否则为false。

成功的话，我们就可以提示用户注册成功啦。

有些情况下，可能还需要使用到mysqli\_insert\_id()。在这里把自增的主键ID打印出来。

大家记住这个知识点，避免以后需要的时候忘记了。

mysqli\_insert\_id应用场景：新加的一个行的数据。我们需要得到自动增长的ID值，将这个ID值插入到另外一个表里面去时。就需要用到这个函数。

```
$result = mysqli_query($conn,$sql);

if($result){
    echo '注册成功';
}else{
    echo '注册失败';
}

echo '当前用户插入的ID为'.mysqli_insert_id($conn);
```

六、关闭数据库连接

将资源变量传到到mysqli\_close这个函数里面即可。

```
mysqli_close($conn);
```

用户注册的基本实现代码就写完了。我们上面讲的都是代码片段。

我们整实现的connect.php代码如下：

```
<?php
```



## 13.2 通过步骤做一个用户注册

```
if (trim($_POST['password']) != trim($_POST['repassword'])) {  
    exit('两次密码不一致,请返回上一页');  
}  
  
$username = trim($_POST['username']);  
$password = md5(trim($_POST['password']));  
  
$time = time();  
  
$ip = $_SERVER['REMOTE_ADDR'];  
  
$conn = mysqli_connect('localhost', 'root', 'liwenkaihaha');  
  
//如果有错误, 存在错误号  
if (mysqli_errno($conn)) {  
    echo mysqli_error($conn);  
    exit;  
}  
  
mysqli_select_db($conn, 'book');  
  
mysqli_set_charset($conn, 'utf8');  
  
$sql = "insert into user(username,password,createtime,createip) values('" . $username . "','" . $password . "','" . $time . "','" . $ip . "')";  
  
$result = mysqli_query($conn, $sql);  
  
if ($result) {  
    echo '成功';  
} else {  
    echo '失败';  
}  
  
echo '当前用户插入的ID为' . mysqli_insert_id($conn);  
  
mysqli_close($conn);  
  
?>
```

## 13.3 通过步骤做一个列表显示

上一章我们按照我们的“数据库连接天龙八部”，很顺利的就完成了用户注册。

我们来做一个后台的用户列表展示。在实际的管理过程当中，我们通过后台，可以个修改用户的密码和用户的相关资料。

在后台需要将所有用户以表格的形式展示出来就是用户列表。

### 一、连接数据库、判断错误和设置字符集

连接、错误判断和字符集选择都在上面已经讲过。老规矩，第一步使用mysqli\_connect连接数据库。在第一节我们讲过，可以在第四个参数中加上库选择。就可以不用使用mysqli\_select\_db函数在后面再次选择一个数据库了。

返回的类型就是一个连接资源。我们在mysqli\_errno、mysqli\_error和mysqli\_set\_charset都要传入资源，才能确定我们操作的是哪个连接。

```
$conn = mysqli_connect('localhost', 'root', 'secret', 'book');

if (mysqli_errno($conn)) {
    mysqli_error($conn);
    exit;
}

mysqli_set_charset($conn, 'utf8');
```

### 二、准备并发送SQL语句

我们需要查询的的将候将用户ID，用户名、时间和IP都查出来。并且使用order by id 进行降序排序。

按照人的思维人们一般喜欢看最新注册的一批用户。而ID自增，也就是ID在越大，就是时间注册越新的用户。因此我们在写SQL语句的时写上的是order by id desc。

```
$sql = "select id,username,createtime,createip from user order by id desc";

$result = mysqli_query($conn, $sql);
```

### 三、判断结果

查询出来的结果只要SQL语句正确结果变量\$result就为真。因此，在实现的时候我们需要多加一步判断，不仅判断\$result。而且，判断查询出来的行数。

查询出来的行数可以使用mysqli\_num\_rows。这个函数要求传入\$result查询的结果变量。

如果有结果则显示列表，如果没有结果我们产生一句提示即可。

代码片段如下：

```
if($result && mysqli_num_rows($result)){

    //显示列表代码段

}else{

    //提示没有结果的代码段
```

```
}

```

四、循环显示数据

所有结果我们需要使用列表的形式展示出来。表格的行和列和数据表的行和列是一样的。所示展示起来很方便。

先声明一个表格，每次循环的时候输出一行。将结果展示到各个列里面。

使用到的函数是mysql\_fetch\_assoc，返回的会是一个关联数组。

这个函数读取一个结果集，会向后移动一次。读取到最后没有结果的时候会返回bool值的false。因此，我们选择while来配合mysql\_fetch\_assoc。

每次循环的结果赋值给\$row，\$row中是关联数组。因此我在这次循环中，可以将行和列都显示出来。

```
echo '<table width="800" border="1">';

while ($row = mysql_fetch_assoc($result)) {

    echo '<tr>';

    echo '<td>' . $row['username'] . '</td>';
    echo '<td>' . date('Y-m-d H:i:s', $row['createtime']) . '</td>';
    echo '<td>' . long2ip($row['createip']) . '</td>';

    echo '</tr>';
}

echo '</table>';

```

五、增加编辑和删除控制

- 1. 在删除的时候我们分为单选删除和多选删除。
- 2. 而编辑的时候，我们会选择一个用户

我们在上一步的代码中增加几个小东西就在页面中实现了删除和编编。

我们来看看实际的效果图,来推理具体的实现过程，效果如下：

<input type="checkbox"/>	范玮琪	2015-10-12 14:20:53	0.2.238.168	<a href="#">编辑用户</a>	<a href="#">删除用户</a>
<input type="checkbox"/>	范冰冰	2015-10-12 14:20:41	0.2.238.168	<a href="#">编辑用户</a>	<a href="#">删除用户</a>
<input type="checkbox"/>	张柏芝	2015-10-12 14:20:17	0.2.238.168	<a href="#">编辑用户</a>	<a href="#">删除用户</a>
<input type="checkbox"/>	谢霆锋	2015-10-12 14:20:07	0.2.238.168	<a href="#">编辑用户</a>	<a href="#">删除用户</a>
<input type="checkbox"/>	井柏然	2015-10-12 14:19:52	0.2.238.168	<a href="#">编辑用户</a>	<a href="#">删除用户</a>
<input type="checkbox"/>	王二小	2015-10-12 14:19:36	0.2.238.168	<a href="#">编辑用户</a>	<a href="#">删除用户</a>
<input type="checkbox"/>	李文凯	2015-10-12 14:19:22	0.2.238.168	<a href="#">编辑用户</a>	<a href="#">删除用户</a>

删除

在实现过程当中有几个要点：

- 1. 单选择删除和编辑时需使用get方法传入ID，我们才知道要编辑或者删除的是哪个用户。
- 2. 多选删除时，需要使用传入多个用户。因此，我们可以使用form表单，使用post方法来提交这批用户ID。

单选删除我们可以在delete.php后面我们跟上?加上id和值就点击时进行删除请求即可。

```
echo '<td><a href="delete.php?id=' . $row['id'] . '">删除用户</a></td>';
```

编辑用户也是同理，我们在edit.php加上?写上id和值，点击时就知道是需要编辑的哪个用户了。

```
echo '<td><a href="edit.php?id=' . $row['id'] . '">编辑用户</a></td>';
```

而多选删除，我们需要使用到html中的checkbox，传入多个用户ID的时候需要在name 后加上id[]。使用form表单将表格包起来，在表格外加上一个submit标签就实现了多选删除。

```
<form action="delete.php" method="post">

echo '<td><input type="checkbox" name="id[]" value="' . $row['id'] . '" /></td>';

echo '<input type="submit" value="删除" />';

echo '</form>';
```

代码如下：

```
echo '<form action="delete.php" method="post">';
echo '<table width="800" border="1">';

while ($row = mysqli_fetch_assoc($result)) {

    echo '<tr>';

    echo '<td><input type="checkbox" name="id[]" value="' . $row['id'] . '" /></td>';
    echo '<td>' . $row['username'] . '</td>';
    echo '<td>' . date('Y-m-d H:i:s', $row['createtime']) . '</td>';
    echo '<td>' . long2ip($row['createip']) . '</td>';
    echo '<td><a href="edit.php?id=' . $row['id'] . '">编辑用户</a></td>';
    echo '<td><a href="delete.php?id=' . $row['id'] . '">删除用户</a></td>';

    echo '</tr>';
}

echo '</table>';

echo '<input type="submit" value="删除" />';
echo '</form>';
```

## 六、关闭数据库连接

我们操作完数据库，关闭掉这个数据库连接。

```
mysqli_close($conn);
```

我们整实现的用户列表list.php代码如下：

```
<?php

$conn = mysqli_connect('localhost', 'root', 'secret', 'book');

if (mysqli_errno($conn)) {
    mysqli_error($conn);
    exit;
}

mysqli_set_charset($conn, 'utf8');
```

### 13.3 通过步骤做一个列表显示

```
$sql = "select id,username,createtime,createip from user order by id desc";

$result = mysqli_query($conn, $sql);

if ($result && mysqli_num_rows($result)) {

    echo '<table width="800" border="1">';

    while ($row = mysqli_fetch_assoc($result)) {

        echo '<tr>';

        echo '<td>' . $row['username'] . '</td>';
        echo '<td>' . date('Y-m-d H:i:s', $row['createtime']) . '</td>';
        echo '<td>' . long2ip($row['createip']) . '</td>';
        echo '<td><a href="edit.php?id=' . $row['id'] . '">编辑用户</a></td>';
        echo '<td><a href="delete.php?id=' . $row['id'] . '">删除用户</a></td>';

        echo '</tr>';
    }

    echo '</table>';

} else {
    echo '没有数据';
}

mysqli_close($conn);
```

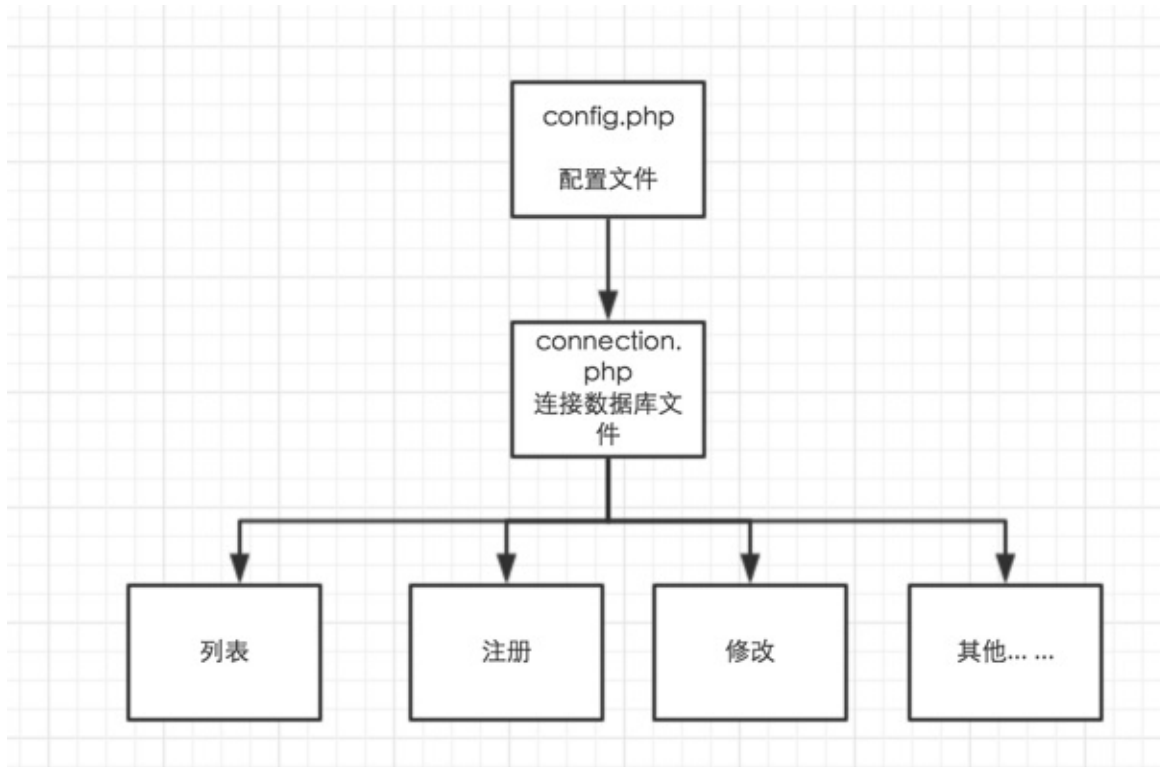
## 13.4 把用户做个分页

在真实的项目中，我们是将主机、用户名、密码、库都写在配置文件当中。

如果在代码中写死了，万一数据库服务器的相关信息发生了变化了，要把所有代码修改一次显然不符合程序员的思维。

此外，在每一个需要连接数据库的页面中。我们都需要写上连接、判断错误、设置字符集、太过于麻烦。并且不利于重复使用这些代码。

我们可以用上之前讲过的include系列函数达成目标。示例图如下：



因此，我们可以做一个配置文件config.php。将需要使用到的配置全部设置为常量，代码如下：

```
<?php
<?php
//数据库服务器
define('DB_HOST', 'localhost');

//数据库用户名
define('DB_USER', 'root');

//数据库密码
define('DB_PWD', 'secret');

//库名
define('DB_NAME', 'book');

//字符集
define('DB_CHARSET', 'utf8');
```

我们将connection.php页面抽取出来，以后需要连接数据库的时候只需要包含connection.php文件即可。代码如下：

13.4 把用户做个分页

```
<?php

include 'config.php';

$conn = mysqli_connect(DB_HOST, DB_USER, DB_PWD, DB_NAME);

if (mysqli_errno($conn)) {
    mysqli_error($conn);
    exit;
}

mysqli_set_charset($conn, DB_CHARSET);
```

我们在以后每个文件使用中直接包含 connection.php文件就可以实现数据库连接了：

```
include 'connection.php';
```

把上面的准备工作完成，接下来完成分页。分页效果如下：

范玮琪	2015-10-12 14:20:53	0.2.238.168	<a href="#">编辑用户</a>	<a href="#">删除用户</a>
范冰冰	2015-10-12 14:20:41	0.2.238.168	<a href="#">编辑用户</a>	<a href="#">删除用户</a>
张柏芝	2015-10-12 14:20:17	0.2.238.168	<a href="#">编辑用户</a>	<a href="#">删除用户</a>
谢霆锋	2015-10-12 14:20:07	0.2.238.168	<a href="#">编辑用户</a>	<a href="#">删除用户</a>
井柏然	2015-10-12 14:19:52	0.2.238.168	<a href="#">编辑用户</a>	<a href="#">删除用户</a>
<a href="#">首页</a> <a href="#">上一页</a> <a href="#">下一页</a> <a href="#">尾页</a> 当前是第 1 页 共2页				

页要实现分页中包含以下几个基本元素：

元素	说明	备注
首页	最开始进入到页面的第一页	用get传参才进去时默认为1
上一页	当前页减1	如果页码为第一页时减1，为应该为第一页
下一页	当前页加1	如果页码为最后一页时减1，应该为最后一页
尾页	最后一页	总条数除以每页显示数得到总页数
当前页	当前所在的页码	就是当前的页码
总页数	一共有多少个页面	总条数除以每页显示数

我们在控制页码的时候，都是通过URL地址栏传入页码值来实现的页码控制。在page.php后面接上页码的相关信息，我们就能够算出更多的有效信息。url控制分页的效果如下：



在代码实现中，是通过limit后的偏移量(offset)和数量(num)，这两个值真正实现的分页。

limit offset , num

页码	url中get值	limit偏移量,数量
第1页	1	0 , 5
第2页	2	5,5
第3页	3	10,5
第n页	n	(n-1)*5,5

假设每页显示5条。最终得到的分页中控制limit公式如下：

offset的值为  $(n-1)*5$   
num 为规定的5

我们通过代码来实现业务：

## 一、计算出分页所需的参数

### 总数

通过查询user表的count(id),得到总数\$count。

```
$count_sql = 'select count(id) as c from user';
$result = mysqli_query($conn, $count_sql);
$data = mysqli_fetch_assoc($result);
//得到总的用户数
$count = $data['c'];
```

### 当前页

刚进入page.php页时，url为http://www.phpxy.com/page.php，后面是不存在?page=1 页面标识号的。

因此我们需要手动创建一个页面标识号传给当前页码变量\$page。

我们害怕用户传的页面中存在小数等，所以我们做一次强制的类型转换：`(int) $_GET['page']`。



## 13.4 把用户做个分页

第一种写法：

```
$page = isset($_GET['page']) ? (int) $_GET['page'] : 1;
```

第二种写法

```
if (isset($_GET['page'])) {  
    $page = (int) $_GET['page'];  
} else {  
    $page = 1;  
}
```

### 最后一页

每一页一定是一个整数。就跟小学的时候数学一样。平均有5.6个人应该准备几个苹果。答案一定是6个。

如果页面出来了20.3个页面，一定是使用进一法取整函数ceil。让分页数变为21。

我们用总数除以每页显示的数据条数，就得到了总页数了。

```
//每页显示数  
$num = 5;  
  
$total = ceil($count / $num);
```

### 上一页、下一页异常情况控制

如果用户的在第一页点击了上一页，在最后一页点击了下一页怎么办呢？

这样的话数据会超出范围，而造成我们分页时无数据显示。

显然这种异常情况需要考虑到。因此，如果在分页时在第一页减一时，我们就让他为第一页。在最后一页加一时，我们就让他为最后一页，即完成了异常控制。

```
if ($page <= 1) {  
    $page = 1;  
}  
  
if ($page >= $total) {  
    $page = $total;  
}
```

## 二、SQL语句

我们之前说过分页的核心是通过SQL语句中的offset和num来控制每页显示数。

我们在上面还列了具体的公式，我们将公司转化为代码如下：

```
$num = 5;  
  
$offset = ($page - 1) * $num;
```

### 13.4 把用户做个分页

我们将\$num和\$offset应用于SQL语句中：

```
$sql = "select id,username,createtime,createip from user order by id desc limit $offset , $num";
```

### 控制好URI中的分页值

```
echo '<tr>
    <td colspan="5">
        <a href="page.php?page=1">首页</a>
        <a href="page.php?page=' . ($page - 1) . '">上一页</a>
        <a href="page.php?page=' . ($page + 1) . '">下一页</a>
        <a href="page.php?page=' . $total . '">尾页</a>
        当前是第 ' . $page . '页 共' . $total . '页
    </td>
</tr>';
```

我们最后将整体业务串联起来实现最终效果，代码如下：

```
include 'connection.php';

$count_sql = 'select count(id) as c from user';

$result = mysqli_query($conn, $count_sql);

$data = mysqli_fetch_assoc($result);

//得到总的用户数
$count = $data['c'];

$page = isset($_GET['page']) ? (int) $_GET['page'] : 1;

/*
if (isset($_GET['page'])) {
    $page = (int) $_GET['page'];
} else {
    $page = 1;
}
*/

//每页显示数

$num = 5;

//得到总页数
$total = ceil($count / $num);

if ($page <= 1) {
    $page = 1;
}

if ($page >= $total) {
    $page = $total;
}

$offset = ($page - 1) * $num;

$sql = "select id,username,createtime,createip from user order by id desc limit $offset , $num";

$result = mysqli_query($conn, $sql);

if ($result && mysqli_num_rows($result)) {

    //存在数据则循环将数据显示出来

    echo '<table width="800" border="1">';
```

### 13.4 把用户做个分页

```
while ($row = mysqli_fetch_assoc($result)) {

    echo '<tr>';

    echo '<td>' . $row['username'] . '</td>';
    echo '<td>' . date('Y-m-d H:i:s', $row['createtime']) . '</td>';
    echo '<td>' . long2ip($row['createip']) . '</td>';
    echo '<td><a href="edit.php?id=' . $row['id'] . '">编辑用户</a></td>';
    echo '<td><a href="delete.php?id=' . $row['id'] . '">删除用户</a></td>';

    echo '</tr>';
}

echo '<tr><td colspan="5"><a href="page.php?page=1">首页</a>  <a href="page.php?page=' . ($page - 1) . '">上一页</a>  <a href="page.php?page=' . ($page + 1) . '">下一页</a>  <a href="page.php?page=' . $total . '">尾页</a>  当前是第 ' . $page . '页 共' . $total . '页 </td></tr>';

echo '</table>';

} else {
    echo '没有数据';
}

mysqli_close($conn);
```

## 13.5 批量和指定删除用户

我们在13.3这一节《通过步骤做一个列表显示》。在删除前，有删除单行数据和删除多行数据。

### 判断是单选还是多选删除

1. 单行是通过get传参的方式向delete.php文件中写上对应的ID。
2. 而多个删除是通过POST的方式向delete.php页面中传递对应的ID。
3. 如果这两个都不符合的话，那我们可以视为数据不合法。

```
if (is_array($_POST['id'])) {  
    $id = join(',', $_POST['id']);  
} elseif (is_numeric($_GET['id'])) {  
    $id = (int) $_GET['id'];  
} else {  
    echo '数据不合法';  
    exit;  
}
```

### 组合SQL语句

我们之前给大家在MySQL这一章时讲解过删除时可以使用到in的子语句。

同样在这里，我们就可以用in的子语句来达到效果。

join函数将多选删除传过来的id变为了3,4,5的格式，最终多选删除的SQL语句执行出来的效果就是：

```
delete from user where id in(3,4,5,6,8);
```

而单选删除的语句效果就是：

```
delete from user where id in(3)
```

这样我们就实现了单选和多选自适应效果。

```
$sql = "delete from user where id in($id)";
```

最终配套而成的整体代码演示如下：

```
<?php  
include 'connection.php';  
  
if (is_array($_POST['id'])) {  
    $id = join(',', $_POST['id']);
```

```
} elseif (is_numeric($_GET['id'])) {  
  
    $id = (int) $_GET['id'];  
  
} else {  
    echo '数据不合法';  
    exit;  
}  
  
$sql = "delete from user where id in($id)";  
  
$result = mysqli_query($conn, $sql);  
  
if ($result) {  
    echo '删除成功';  
} else {  
    echo '删除失败';  
}
```

# 13.6 修改用户信息

在真正的后台管理中管理员可以修改用户的很多信息。如果开放权限，管理员连用户的用户名这些信息都可以修改掉。

在真正的操作中，往往是：

1. 选择要修改的用户

<input type="checkbox"/>	范玮琪	2015-10-12 14:20:53	0.2.238.168	<a href="#">编辑用户</a>	<a href="#">删除用户</a>
<input type="checkbox"/>	范冰冰	2015-10-12 14:20:41	0.2.238.168	<a href="#">编辑用户</a>	<a href="#">删除用户</a>
<input type="checkbox"/>	张柏芝	2015-10-12 14:20:17	0.2.238.168	<a href="#">编辑用户</a>	<a href="#">删除用户</a>
<input type="checkbox"/>	谢霆锋	2015-10-12 14:20:07	0.2.238.168	<a href="#">编辑用户</a>	<a href="#">删除用户</a>
<input type="checkbox"/>	井柏然	2015-10-12 14:19:52	0.2.238.168	<a href="#">编辑用户</a>	<a href="#">删除用户</a>

删除

选择要修改信息的用户

2. 修改相关内容

用户名：

密码：

修改用户信息

3. 执行修改语句，并产生提示

← → ↺

www.phpxy.com/update.php

修改成功

产生修改提示

在做用户列表页的时候，我们已经向大家完整的展示过了如何在列表中显示编辑用户和删除用户的功能。

从列表中点击选择要修改的用户，应该有一个专门的页面来显示需要修改的内容。我们在上面的第二张图中也为大家做了展示。

可是代码实现的时候如何把用户信息放进来呢？

## edit.php展示用户信息

实现过程：

一. 我们可以将用户的ID在get中进行传参，得到用户信息。使用SQL语句将用户信息查询出来。

```
<?php

if (is_numeric($_GET['id'])) {

    $id = (int) $_GET['id'];

}

$sql = "select id,username from user where id = " . $id;

$result = mysqli_query($conn, $sql);

$data = mysqli_fetch_assoc($result);

?>
```

二、将用户的信息分配到form表单中。当用户点击提交时，我们在update.php提交用户在form表当中修改的值。因为在update中修改的where条件中需要指定修改哪个用户。所以，我们在input隐藏表单中放入用户的ID。当点击提交的时候，隐藏的ID也会传入到update页面中。

用户名通常是不让修改的。因此，我在username这一个input表单最后增加了一个：readonly参数，不准修改用户名。

代码如下：

```
<form action="update.php" method="post">

    用户名:<input type="text" name="username" value="<?php echo $data['username'];?>" readonly><br />

    密码:<input type="password" name="password"><br />

    <input type="hidden" value="<?php echo $data['id'];?>" name="id" />

    <input type="submit" value="提交">

</form>
```

## update.php修改操作用户数据

实际上，我们只能修改用户的密码。有两种情况：

1. 用户修改了密码
2. 用户没有修改密码

其实我们可以欺骗前面的不懂技术的普通操作员。

1. 如果他没有修改密码，也给他提示成功
2. 如果他修改了用户的密码。我们真正的修改掉用户的密码，也提示修改成功。

## 获取用户ID和密码

我们实现的过程当中需要获取用户ID。不然update语句生成的时候，没有where条件会将整个表的数据全部修改掉的。

密码之前是用md5存储的。因此，用户修改了密码，密码也应该用md5来存储。

```
$id = (int)$_GET['id'];

$password = md5(trim($_POST['password']));
```

## 生成SQL语句

将用户ID和密码放至到修改的SQL语句中，发送至MySQL服务器执行。即完成了修改密码的操作。

```
$sql = "update user set password='" . $password . "' where id = $id";

$result = mysqli_query($conn, $sql);

if ($result) {
    echo '修改成功';
}
```

## 整体代示演示

### 在form表单中显示用户信息的源代码

```
<?php

if (is_numeric($_GET['id'])) {

    $id = (int) $_GET['id'];

}

$sql = "select id,username from user where id = " . $id;

$result = mysqli_query($conn, $sql);

$data = mysqli_fetch_assoc($result);

?>

<form action="update.php" method="post">

    用户名: <input type="text" name="username" value="<?php echo $data['username'];?>"><br />

    密码: <input type="password" name="password"><br />

    <input type="hidden" value="<?php echo $data['id'];?>" name="id" />

    <input type="submit" value="提交">

</form>

<?php

mysqli_close($conn);

?>
```

### update.php修改的源代码

```
<?php
include 'connection.php';

$id = (int) $_POST['id'];

if (trim($_POST['password'])) {

    $password = md5(trim($_POST['password']));

    $sql = "update user set password='" . $password . "' where id = $id";

} else {
```



## 13.6 修改用户信息

```
        echo '修改成功';  
    }  
  
    $result = mysqli_query($conn, $sql);  
  
    if ($result) {  
        echo '修改成功';  
    }
```

## 13.7 数据显示乱码终极解决办法

---

php连接mysql乱码是开发中新手经常遇到的问题。根据实际过程中大家所遇到的，将乱码的问题，总结成了9个要点来彻底解决链接mysql乱码的问题。

解决乱码问题的核心思想，就是：多个不同的文件系统中一定要统一编码。

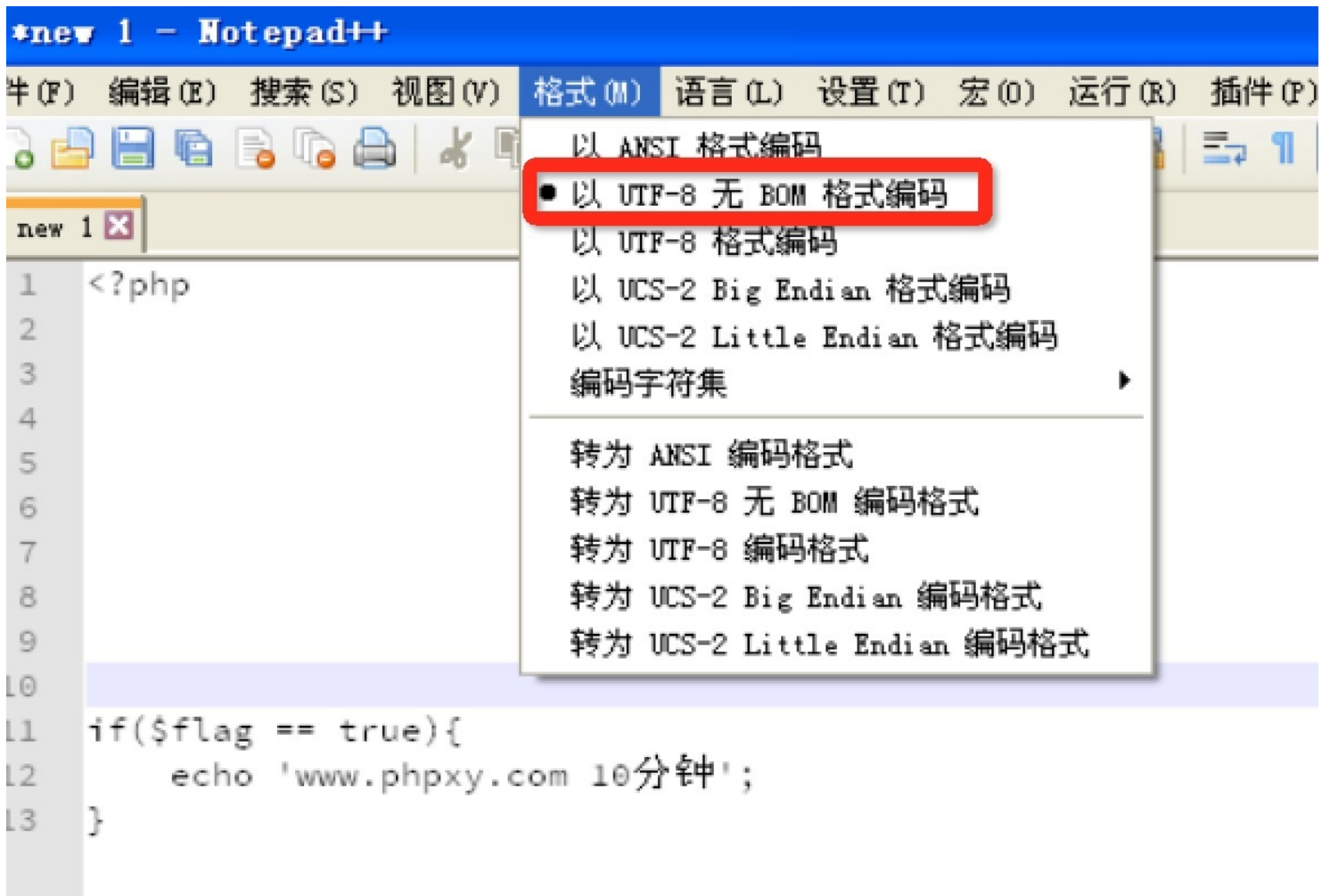
这9个要点分别是：

1. html编码与MySQL编码一致
2. PHP编码与MySQL编码一致
3. 若有header头发送字符集，请与数据库一样
4. 要和页面的文字编码一致
5. 数据库建库的字符集要统一
6. 表的字符集要统一
7. 列的字符集要统一(表设了,列的字符集就默认是表的)
8. 连接,校验的字符集要统一
9. 结果集的字符集要统一

### 一，文件编码

html和PHP文件的编码，示例中：拿notepad++来演示。将PHP和html都要设为这种字符集。

一定要设置为utf-8无BOM格式。



## 二、header头编码

如果php中有header头，一定要是utf-8的

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

## 三、网页头文件编码

如果html文件编码是utf-8的也要设置为一致

## 四、创建数据库的编码

# 数据库

新建数据库

phpxy

注意：在此启用数据库统

utf32\_turkish\_ci

utf32\_unicode\_ci

utf8

utf8\_bin

utf8\_czech\_ci

utf8\_danish\_ci

utf8\_esperanto\_ci

utf8\_estonian\_ci

utf8\_general\_ci

utf8\_general\_mysql500\_ci

创建

五、表和列创建的时候表为utf-8

INT

INT

INT

INT

表注释：

PARTITION definition:

utf32\_slovak\_ci

utf32\_slovenian\_ci

utf32\_spanish2\_ci

utf32\_spanish\_ci

utf32\_swedish\_ci

utf32\_turkish\_ci

utf32\_unicode\_ci

utf8

utf8\_bin

utf8\_czech\_ci

utf8\_danish\_ci

utf8\_esperanto\_ci

utf8\_estonian\_ci

utf8\_general\_ci

utf8\_general\_mysql500\_ci

utf8\_hungarian\_ci

utf8\_icelandic\_ci

存储引擎

InnoDB

Unicode (多语言)

六、连接，结果、校验字符集设置

通过mysql\_set\_charset('utf8')来MySQL连接、结果和校验的字符集设置。

注：数据库的字符集声明和文件中的略有不同。utf8为mysql数据库的，utf-8为文件中使用的。

## 14.会话管理和控制

---

这节我们学习的是会话管理和控制，首先要知道的当然是什么是会话了。

会话，通过字面意思理解就是两个人通话和交流的一个过程。

浏览器打开一个网站，和网站的服务器交互。是两台电脑间的交流。因此，我们也可以认为是两个设备间的会话。不过特殊的是，因为浏览器是电脑里面的一个软件，这个软件没有长相，没有声音，没身份证.....我们不知道到底是谁访问了这个网站。

如果只认IP的话，我们在一个公司，家里IP都是一样的。可能一个IP对应多个电脑。或者是用着用着换了IP，如果用IP来识别用户的唯一身份的话肯定不现实。

其实我们人类社会的思维模式，已经解决了这个问题。我跟大家讲两个关于开会的故事：

1. 在几十年前人们开会的时候，都需要带上一个参会证。这个参会证上有这个人的职务、姓名、单位、照片等信息。在开会的时候，会议安保人员、组织者只需要检查相关信息就行了。
2. 几十年后，越来越先进了。你参会前，给你发一个卡片或者工牌。这个工牌，有一个唯一的号码。拿到号码，再用摄像头自动拍一张你的照片就完成了入场的对比工作。

在电脑里面也有这两种验证方式：

1. 人们在参会证上带上了所有的相关信息的这种会话方式我们叫做cookie。这种模式，信息是保存在用户电脑上的。
2. 人们只需要带一个号码或者磁卡，通过这个信息去验证你的照片、姓名等信息的模式我们叫做session。它只在用户本地存一组小小的值，其他信息全是存在服务器上的。

用专业一点的语言来解释：

HTTP协议是无状态的。何谓无状态？你输入我的网址，我就给你传送数据。我可不管你是金三胖，安倍或者是奥巴马呢。

而我们在现实情况中，往往大家登陆后就知道大家是谁了。这两种方式是HTTP协议中的：

1. cookie会话技术
2. session会话技术

# 14.1 Cookie概述

我们之前拿开会的例子讲了一个小例子：

在几十年前人们开会的时候，都需要带上一个参会证。这个参会证上有这个人的职务、姓名、单位、照片等信息。在开会的时候，会议安保人员、组织者只需要检查相关信息就行了。

这个小例子主要说明一点，人们自己带着自己的参会证，带着自己的信息。这种模式就是cookie。

而电脑将这段cookie信息存在了电脑的硬盘里。

## Cookie存在哪儿？

Cookie的本质是一小段数据，一小段存储在你电脑硬盘中的数据。可是它存在哪里呢？来，我们找一下。

Chrome浏览器的Cookie文件的存放路径是：

C:\Users\你的用户名\AppData\Local\Google\Chrome\User Data\Default\Cookies

Firefox浏览器的Cookies文件存放路径是：

C:\Users\你的用户名\AppData\Roaming\Mozilla\Firefox\Profiles\rdgp36vl.default\cookies.sqlite 每个人可能略有不同 (PS:rdgp36vl.default这个前缀可能会PHP学院)。

用文本编辑器直接打开Cookies文件看到的是乱码，我们得用工具查看，如下图：

Domain	Path	Name	Value	Secure	Http Only	Expires
www.dreams...	/	ai_user		No	No	9999/12/31 23:
acs86.com	/	ubt		No	No	9999/12/31 23:
.tanx.com	/	cap	9516	No	No	9999/12/31 23:
.tanx.com	/	cad	iDop yy8ap7SzhKN2N/1Hw==	No	No	9999/12/31 23:
icast.cn	/	ic5		No	No	2999/12/30 23:
.7pud.com	/	var_yt_cookie_f	pic183025	No	No	2332/6/16 11:3
hack50.com	/	safedog-flow-it		No	No	2151/12/2 13:4
.weste.net	/	safedog-flow-it	0C255D2CC20351AA7074AB	No	No	2151/5/3 16:35
.tudou.com	/	yseidcount	2	No	No	2115/6/21 10:5
.tudou.com	/	ystep	3	No	No	2115/6/21 10:5
.tudou.com	/	ysestep	1	No	No	2115/6/21 10:5
.tudou.com	/	juid	019plvoetg2hcg	No	No	2115/6/14 12:4
.163.com	/	_ntes_nnid	0b5c6f6e9c87d78d40011752c32	No	No	2115/3/3 11:19
163.com	/	_ntes_nnid		No	No	2115/3/1 21:30

我们来看一下需要关注的几个列，Domain代表的是cookies所属的网站，Name代表的是这个Cookie的名字，Value代表的是Cookie的值，Expires代表的是这个Cookie的有效期。

用一个我们熟悉的网站举例，tudou.com，图上我们可以看到有4个关于 tudou.com 的Cookie，那么当我们访问 tudou.com 的时候，浏览器会自动把这4个Cookie的 Name 和 Value 发送到 tudou.com 这个网址所指向的服务器(PS:必须在有效期内，超出有效期的话是不会被发送到服务器的，有效期我们可以依据需求来定)，如此一来，服务器就可以根据这些信息来保持与客户端的连接了，通俗点，就是可以通过这些数据来知道你就是你。当服务器收到这些Cookies后，会根据他们的值来做一些处理，做什么处理？这就取决于开发人员想根据这些信息来干嘛了！

以上稍微介绍了Cookies和他的用途。接下来我们将学习在PHP中使用Cookie。



## 14.2PHP中的Cookie

这节我们通过一个用户首次登陆网站后，再次访问不需要重复输入用户名和密码的例子来学习Cookie。

首先介绍一下php中设置cookie的方法。

php中提供了一个函数来让我们设置cookie，这个函数是：

```
bool setcookie (
    string $名字
    , [ string $值]
    , [ int $过期时间 = 0]
    , [ string $路径]
    , [ string $域名]
    , [ bool $安全 = false]
    , [ bool $http只读 = false]
);
```

参数 描述

\$名字 必需。规定 cookie 的名称。

\$值 可选。规定 cookie 的值。

\$有效期 可选。规定 cookie 的有效期。

\$路径 可选。规定 cookie 的服务器路径。

\$域名 可选。规定 cookie 的域名。

\$安全 可选。规定是否通过安全的 HTTPS 连接来传输 cookie。

\$http安读 可选。如果true，那么js就无法读取改cookie，增加安全性。

一般来说，我们其实用不到上面那么多参数，对于这个函数，我们一般这么用：setcookie(cookie名,cookie值,cookie有效期);

没错，就那么3个。如此一来，我们就可以在服务端通过\$\_COOKIE['name'] 来读取cookie了。

以下是示例：

我们将文件名命名为：cookie.php。

我们来模拟我们在互联网上见到最常见的例子：输入用户名和密码，登陆成功的过程。

我们来建一个数据库login,其中有表user,有username和passwd这两个字段。

```
<?php
//第一次登陆的时候，通过用户输入的信息来确认用户
if ( ( $_POST['username'] != null ) && ( $_POST['password'] != null ) ) {
    $userName = $_POST['username'];
    $password = $_POST['password'];
    //从db获取用户信息
    //PS：数据库连接信息改成自己的 分别为主机 数据库用户名 密码
    $conn = mysqli_connect('host','user','passwd');

    mysqli_select_db($conn,'login');

    $sql = "select * from user where `username` = '$userName' ";
    $res = mysqli_query($conn,$sql);
    $row = mysqli_fetch_assoc($res);
    if ($row['passwd'] == $password) {
        //密码验证通过，设置cookies，把用户名和密码保存在客户端
        setcookie('username',$userName,time()+60*60*24*30);//设置时效一个月，一个月后这个cookie失效
        setcookie('password',$password,time()+60*60*24*30);
```



```

        //最后跳转到登录后的欢迎页面
        header('location: welcome.php' . "?username=$UserName");
    }
}

//再次访问的时候通过cookie来识别用户
if ( ($_COOKIE['username'] != null)  && ($_COOKIE['password'] != null) ) {
    $UserName = $_COOKIE['username'];
    $password = $_COOKIE['password'];

    //从db获取用户信息
    //PS：数据库连接信息改成自己的 分别为主机 数据库用户名 密码
    $conn = mysqli_connect('host','user','passwd','login');
    $res = mysqli_query($conn,"select * from user where `username` =  '$UserName' ");
    $row = mysqli_fetch_assoc($res);
    if ($row['passwd'] == $password) {
        //验证通过后跳转到登录后的欢迎页面
        header('location: welcome.php' . "?username=$UserName");
    }
}

?>
<html>
<head>

</head>
<body>
    <form action="" method="POST">
        <div>
            用户名:<input type="text" name="username" />
            密 码:<input type="text" name="password" />
            <input type="submit" value="登录">
        </div>
    </form>
</body>
</html>

```

跳转到的welcome.php代码

```

<?php
$user = $_GET['username'];
?>
<html>
<head>

</head>
<body>
    welcome,<?php echo $user;?>
</body>
</html>

```

这样，当我第一次访问cookie.php的时候，我需要输入用户名和密码，输入完毕后跳转到了welcome.php。然后我关闭浏览器，再次打开cookie.php，这次没有要求我输入用户信息，而是直接跳转到了welcome.php,因为之前我们存的cookie信息被浏览器自动发送到了服务端.服务端做完处理直接跳转到了welcome.php,服务器认识我们了！知道我是之前那个登陆过的用户，这样我们就通过cookie技术让无状态的HTTP协议保持了状态。

照着这个做一遍，我相信你会用cookie了。

只不过！！！只不过！！！只不过！！！重要的事要说3遍，我们一般是不会把用户名和密码放到cookie中的，因为这并不安全，容易泄露自己的信息，请不要把重要的信息放到cookie中。我们这个只是一个学习cookie的例子。

## 14.3 session概述

---

这一部分我们要学习的是 session技术.

通过上一节的学习，我们知道Cookie是通过将数据保存在客户端来实现与服务端保持连接的，而Session是通过将数据保存在服务器端来实现保持连接的。我们通过一个例子来了解session的机制。

我们去饮料店买饮料，下单以后服务员会给我们一个号码牌，然后你走到一旁，服务员并不认识你是谁，如果你想拿到你的饮料，你必须提供你的号码牌给服务员才可以，服务员通过号码牌来查记录，来确认你是顾客，确认你点了什么饮料，然后才会把你点的饮料给你。

了解了session原理，再回到Web技术中，我们有2种方法让客户端拿到“号码牌”，一种是通过cookie，一种是通过把值嵌入网页传给客户端。我们也有2种方法来让客户端把号码牌传给服务器来拿属于自己的资料，一种是cookie，一种是标准的Query String/POST。

而我们常用的是cookie，因为现在的浏览器都支持cookie，默认也都开启。客户端与服务端彼此都会将cookie发送给对方。来个过程说明一下：打开浏览器输入 `www.taobao.com` 并回车，由于是第一次与这个网站建立连接，服务端还没有设置过cookie(这里假设当前浏览器是第一次访问这个网址，之前这个网址没有向当前客户端写过cookie)，所以没有cookie发送到服务端，服务端在处理完数据返回的时候，会将一个name为sessionid，value为一连串N位字符的cookie发送给客户端，后续客户端再次访问服务端的时候，也会带上这个cookie来访问服务端。于是，他们就这样通过sessionid互相“认识”了。

## 14.4 PHP中使用session

了解了session的原理后，我们来学习如何在PHP中使用session。

### 1.开启session

首先我们要开启session，那么第一个要学习的函数就是  
bool session\_start()了，这个函数没有参数。在php文件的开始使用

```
session_start();
```

就可以启用新会话或者重用现有会话了。

### 2.添加session数据

开启会话之后，那么在接下来的处理中，我们就可以使用\$\_SESSION变量来存取信息了。我们要知道的是\$\_SESSION变量是个数组。当我们要把信息存入session的时候应该这么写：

```
$_SESSION['userName'] = 'wang';
```

### 3.读取session数据

读取很简单，就像我们使用数组一样，如下：

```
$userName = $_SESSION['userName'];
```

当然也可以 \$\_SESSION['userName'] 来用。和数组一样的使用。

### 4.销毁session数据

我们可以使用很多种方式来销毁session数据。

#### a) unset函数

我们通过使用类似

```
unset($_SESSION['xxx']);
```

来销毁session中的 XXX 变量。PS:请不要！请不要！请不要unset(\$\_SESSION),会导致后续无法使用\$\_SESSION这个变量！！！！

#### b) 空数组赋值给session变量

```
$_SESSION = array();
```

之前我们说过\$\_SESSION变量是个数组，那么空数组赋值的话也是相当于将当前会话的\$\_SESSION变量中的值销毁。

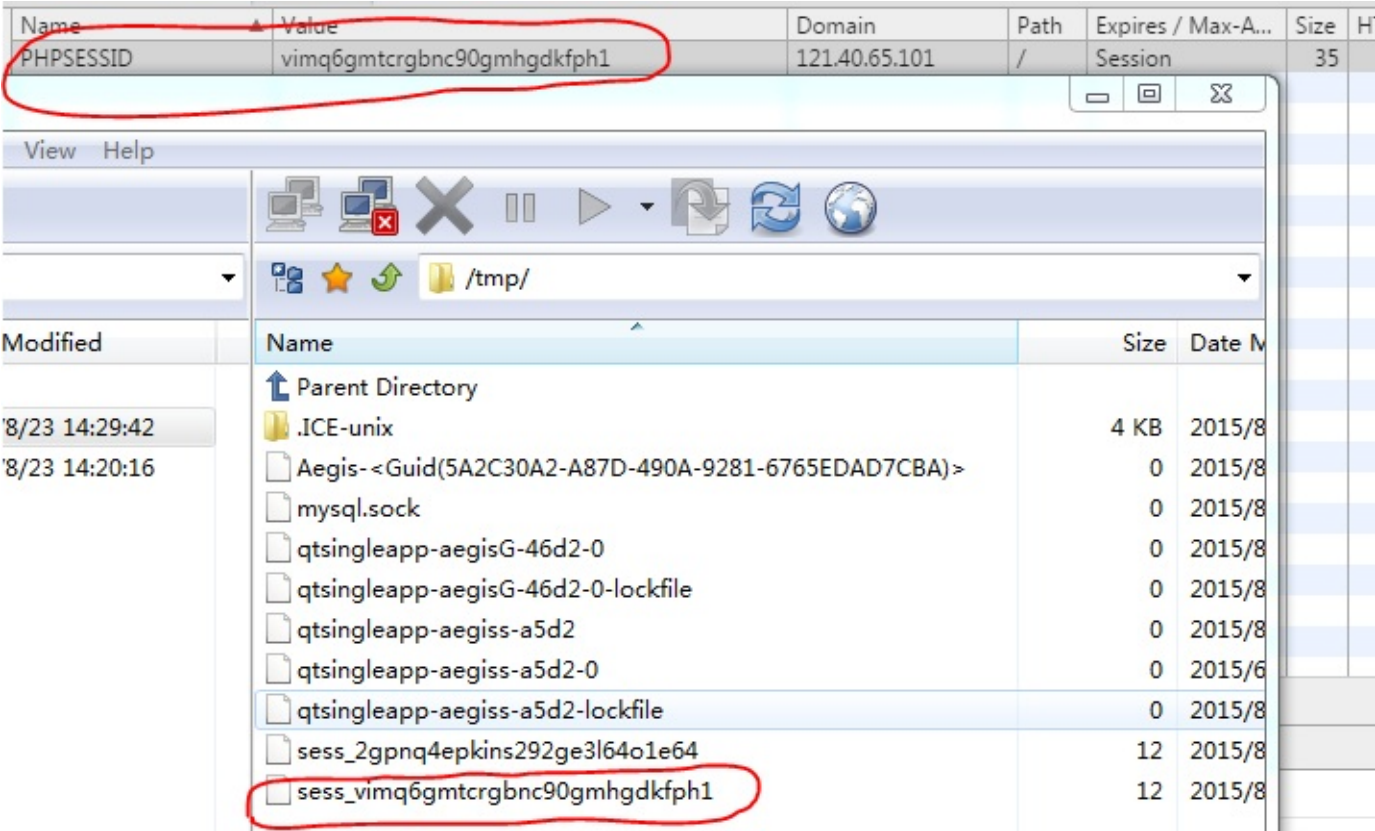
#### c) session\_destroy() 函数

这个函数会销毁当前会话中的全部数据，并结束当前会话。但是不会重置当前会话所关联的全局变量，也不会重置会话cookie。

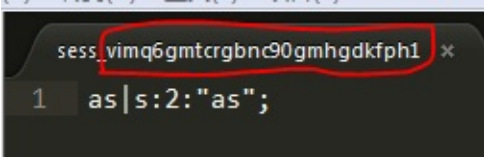
5.通过session来检查用户是否登陆和一个简单的购物车例子  
例子请查看12.5章节

6.session的扩展：默认session存储在哪里。

在php.ini配置文件中有这么一行 session.save\_handler = files, files，说明了php默认的是用文件读写的方式来保存session的。那么在哪个目录呢？继续看。session.save\_path = "/tmp", 这一行前面有个；，说明是被注释的，不过即便这样，php默认的session 也是保存在这里的，/tmp目录。上图：



从图中我们可以看到确实在这个目录下面，我们顺便看看里面的内容



我的写入session的语句是：

```
$_SESSION['as'] = 'as';
```

解读一下，第一个as代表的是\$\_SESSION['as']中的as，|后面的s表示的是这是一个字符串类型的数据，2表示的是这个字符串所占的字节数，最后双引号引起来的是值as。

PS：还可以自己动手试试存数字的话，|后面显示什么字母。还有，你还可以试试如果你存中文的话，字节数是多少？自己试！别看了！我不会告诉你的！！！！

## 14.5 SESSION应用实例

登陆例子：(请注意一定要自己敲一遍，不要CV大法)

首先上一下成果图，激起同学们写的欲望,登录页如下：

---

用户名：	<input type="text" value="gege"/>	密 码：	<input type="text" value="password"/>	<input type="button" value="登录"/>
------	-----------------------------------	------	---------------------------------------	-----------------------------------

点击登陆之后如下：

welcome, gege

说明哦了，么问题。接下来自己实现一下。

首先数据库信息：

新建一个名为 login 的数据库，再建一个 user 表，表的结构如下图：

Column	Type
username	varchar(20)
passwd	varchar(20)

然后开始编码。

login.php

```
<?php
session_start();
if ( ( $_POST['username'] != null ) && ( $_POST['password'] != null ) ) {
    $userName = $_POST['username'];
    $password = $_POST['password'];

    //从db获取用户信息  数据库信息改成自己的
    $conn = mysql_connect('host','username','password','login');
    $res = mysqli_query($conn,"select * from user where `username` = '$userName' ");
    $row = mysqli_fetch_assoc($res);
    if ($row['passwd'] == $password) {
        //密码验证通过，设置session，把用户名和密码保存在服务端
        $_SESSION['userName'] = $userName;
        $_SESSION['password'] = $password;

        //最后跳转到登录后的欢迎页面 //注意：这里我们没有像cookie一样带参数过去
        header('Location: welcome.php');
    }
}

?>
<html>
<head>
<!-- 这里指明页面编码 -->
<meta charset="utf-8">
</head>
<body>
    <form action="" method="POST">
        <div>
            用户名:<input type="text" name="username" />
            密 码:<input type="text" name="password" />
            <input type="submit" value="登录">
        </div>
    </form>
</body>
</html>
```

welcome.php 这里我们用的是session中的信息,而不是像cookie一样在url中带参数过来

```
<?php
session_start();
$userName = $_SESSION['userName'];
?>
<html>
<head>

</head>
<body>
    welcome,<?php echo $userName;?>
</body>
</html>
```

购物车的例子：(请注意一定要自己敲一遍，不要CV大法)

数据库信息：建立名为test的数据库，库中有个shop表，表结构如下图：

Column	Type
id	int(10)
name	varchar(20)
price	varchar(10)

开始编码吧！

goodsList.php 这个是商品展示页，效果图如下：

```
商品名 phone 价格 200购买
商品名 TV 价格 100购买
商品名 camera 价格 300购买
商品名 computer 价格 300购买
商品名 ipad 价格 260购买
查看购物车
```

说明一下，如果是第一次购买某物品，则在购物车中加入该商品信息和计算总价，如果再次点击购买，则已购商品数量加1，总价重新计算，查看购物车链接可以到购物车页面。

```
<?php
$goods = array();
//从数据库获取商品信息存入$goods二维数组
$i = 0;
//这里请换上自己的数据库相关信息
mysqli_connect('host','username','password','test');
$res = mysqli_query($conn,'select * from shop');
//这里把商品信息放到$goods二维数组，每一维存的是单个
//商品的信息，比如商品名、价格。
while ($row = mysqli_fetch_assoc($res)) {
    $goods[$i]['id'] = $row['id'];
    $goods[$i]['name'] = $row['name'];
    $goods[$i]['price'] = $row['price'];
    $i++;
}

?>
<!DOCTYPE html>
<html>
<head>
    <meta http-equiv="Content-Type" content="text/html; charset=utf-8">
</head>
<body>
    <?php
    //取出商品信息显示在页面上，并添加购买功能
    foreach ($goods as $value) {
        echo '商品名 ' . $value['name'] . ' 价格 ' . $value['price'];
```

```

        echo "<a href=buy.php?name=" . $value['name'] . '&price=' . $value['price'] . ">购买</a>";
        echo '<br />';
    }

    ?>
    <a href="shoppingCart.php">查看购物车</a>
</body>
</html>

```

buy.php 此页完成购买功能，然后再次跳转到商品列表。主要是做了在session中处理购买商品操作。

```

<html>
<head>
    <meta http-equiv="Content-Type" content="text/html; charset=utf-8">
</head>
<body>
<?php
    //开启session
    session_start();

    //获取传过来的商品名和价格
    $name = $_GET['name'];
    $price = $_GET['price'];

    //把session中的商品信息和传过来的(刚买的)商品信息对比
    $goods = $_SESSION['goods'];
    if ($name == $goods[$name]['name']) {
        //买过的话，则总价格增加，相应商品数量增加
        $_SESSION['totalPrice'] += $price;
        $goods[$name]['number'] += 1;
    } else {
        //第一次买的话，将相应的商品信息添加到session中
        $goods[$name]['name'] = $name;
        $goods[$name]['price'] = $price;
        $goods[$name]['number'] += 1;
        $_SESSION['totalPrice'] += $price;
    }

    $_SESSION['goods'] = $goods;
    //购买处理完后跳转到商品列表
    header('location: goodsList.php');
?>
</body>
</html>

```

shoppingCart.php 此页展示购物车中的商品、价格、总价等信息。

效果图如下：

```

您买了：
phone 价格 200 数量 2
camera 价格 300 数量 1
computer 价格 300 数量 1
总价：1000
返回商品列表

```

```

<html>
<head>
    <meta http-equiv="Content-Type" content="text/html; charset=utf-8">
</head>
<body>
<?php
    session_start();
    //将session中的商品信息(即购物车中的商品)和总价显示到页面

```

```
$goods = $_SESSION['goods'];
echo '您买了:<br />';
foreach ($goods as $value) {
    echo $value['name'] . ' 价格 ' . $value['price'] . ' 数量 ' . $value['number'] . '<br />';
}
echo '总价:' . $_SESSION['totalPrice'] . '<br />';

?>
<a href="goodsList.php">返回商品列表</a>
</body>
</html>
```

购物车的例子完成了，自己亲自完成一遍后是不是很有成就感！！你很厉害！！



## 15.通过cURL来做小偷程序

微信微博公众号、QQ公众号、地图和很多的第三方功能提供都是采用http 的API（应用程序）形式向开发人员提供的功能。

如果你只希望能够做些简单的微信公众号、QQ公众号等小应用，不想再深入学习PHP。学习到本章，你就可以完全掌握微信公众号服务端的所需知识了。

如何请求微信、地图等第三方API呢？

这个时候就需要使用到——cURL。cURL中文翻译过来叫做基于URL的函数库。它的主要功能是：使用相关的函数模拟协议请求。

例如：

1. 模拟表单向某个地址发送数据
  2. 在无验证码的情况下模拟表单完成用户登陆
  3. 上传某个文件到远程服务器
  4. 请求远程服务器提供的某些功能
- ... ..

curl支持dict, file, ftp, ftps, gopher, http, https, imap, imaps, ldap, ldaps, pop3, pop3s, rtmp, rtsp, smtp, smtps, telnet和tftp协议。

curl同时也支持HTTPS认证、HTTP的POST、HTTP的PUT、FTP上传(这个也能通过PHP的FTP扩展完成)、HTTP基于表单的上传、代理、cookies和用户名+密码的认证。

我们在使用过程当中，如果没有cURL基础根本不知道如何开发微信公众号。

实际工作中，在cURL使用最多的协议就就是http协议中的get和post请求。其他协议和请求方式用的相对较少。当然，在特定的应用中也有使用。

在开发前请保证你已开启cURL模块。开启办法与之前所讲的《PHP图像处理》这一章的开启办法一样，开启的模块名称叫：php\_curl.dll。

看到下图，就说明你的curl模块开启成功：

curl

cURL support	enabled
cURL Information	7.35.0
Age	3
Features	
AsynchDNS	Yes
CharConv	No
Debug	No
GSS-Negotiate	Yes
IDN	Yes
IPv6	Yes
krb4	No
Largefile	Yes
libz	Yes
NTLM	Yes
NTLMWB	Yes
SPNEGO	No
SSL	Yes
SSPI	No
TLS-SRP	Yes
Protocols	dict, file, ftp, ftps, gopher, http, https, imap, imaps, ldap, ldaps, pop3, pop3s, rtmp, rtsp, smtp, smtps, telnet, tftp
Host	x86_64-pc-linux-gnu
SSL Version	OpenSSL/1.0.1f
ZLib Version	1.2.8

# 15.1 curl的使用步骤

本书特别新手的一点，是将重要操作的核心思路都为大家提供了。在数据库、图片、文件上传我们都把真正的步骤。以及步骤如何组何成为我们的代码和实现过程都做过讲解。

我们将curl的步骤分为以下7步：

- 1. 初使化curl资源
- 2. 参数设置请求的协议地址
- 3. 参数设置是否返回请求结果
- 4. 设置发送数据（无发送数据可不设置）
- 5. 其他的参数信息设置（按实际工作需要决定）
- 6. 执行或执行得到返回结果
- 7. 关闭curl资源

我们为了更好的让大家学习和入门，将第2步至第5步本应该归为一步的插成了4个步骤。

第二步至第五步本质上，应该统成为curl参数设置。

所有curl设置参数设置都是通过curl\_setopt都完成

**curl\_setopt这一步最为重要，一切玄妙均在此。**

curl\_setopt这一步完成了包括连接、参数和一切请求的细节设置。要一次性全部看完并理解可能比较困难，所以我们只试一下那些更常用也更有用的选项。

我们为了更好的入门。并且在php中我们用的最多的是http请求。因此，我们将一些不需要大家使用到的全部不作讲解。

如果感兴趣的朋友可以参考curl\_setopt中复杂的参数设置。

地址如下：<http://php.net/manual/zh/function.curl-setopt.php>

## 一、初使化curl资源

这只有一句话，就是使用的是curl\_init函数。这个参数法面要传入任何参数。返回curl的操作资源。

因为，我们在后面是通过curl\_setopt向curl的操作资源变量压入数据的。

例：

```
$ch = curl_init();
```

## 二、参数设置请求的协议地址

curl\_setopt函数的详细使用如下：

类型	说明
函数	curl_setopt
参数1	curl资源变量

参数2	curl参数选项
参数3	curl参数值

### CURLOPT\_URL

这个参数选项规定了请求的url地址。

```
curl_setopt($ch, CURLOPT_URL, "http://www.phpxy.com");
```

### 三、参数设置是否返回请求结果

我们希望curl请求后返回对应的结果。我们要得到对应的结果，也需要设置一个参数，这个参数名为：CURLOPT\_RETURNTRANSFER。

若需要返回值即为1。不需请求后返回的结果可设置为0。

```
curl_setopt($ch, CURLOPT_RETURNTRANSFER, 1);
```

### 四、设置发送数据

如果是get请求，我们不需要设置发送的参数。在post等请求的时候，我们需要设置发送方法为post方法。并设置发送的数据。

**CURLOPT\_POST** 值设为1是使用POST方法，0为不使用POST方法

**CURLOPT\_POSTFIELDS**设置传递的数据

```
//声明使用POST方式来进行发送
curl_setopt($ch, CURLOPT_POST, 1);

//发送什么数据呢
curl_setopt($ch, CURLOPT_POSTFIELDS, $data);
```

### 五、其他的参数信息设置

如果是https，有的时候我们需要忽略https的安全证书。

CURLOPT\_SSL\_VERIFYPEER和CURLOPT\_SSL\_VERIFYHOST 两个参数改为false即忽略了证书。

```
curl_setopt($ch, CURLOPT_SSL_VERIFYPEER, false);
curl_setopt($ch, CURLOPT_SSL_VERIFYHOST, false);
```

CURLOPT\_HEADER这个参数又决定是是否处理http的头信息，我们不想接收处理的话可将这个值设置为0。

```
curl_setopt($ch, CURLOPT_HEADER, 0);
```

此外，我们还可以设置请求的超时时间，参数为：CURLOPT\_TIMEOUT。

```
curl_setopt($ch, CURLOPT_TIMEOUT, 10);
```

其他参数非常多，我们在之前讲过，如果你想了解更多的内容可以访问：  
地址如下：<http://php.net/manual/zh/function.curl-setopt.php>

### 六、执行或执行得到返回结果

我们第三步中，将CURLOPT\_RETURNTRANSFER 参数中将值设为1。如果执行的结果有数据。使用curl\_exec执行后，会将结果返回给\$output变量。

```
$output = curl_exec($ch);
```

### 七、关闭curl资源

关闭curl资源。因为资源类型，我们反复强调过，有打开就有关闭。

如果不需要使用，使用curl\_close关闭后立刻释放内存。

```
curl_close($ch);
```

## 15.2 自定义get方法抓取网页

假设我们使用get方法请求一个网页，得到网页内容后可以匹配出对应的内容。

我们可以使用curl封装一个函数，假设函数名就为get。传入url就能请求指定的网页，将指定网页的HTML代码返回回来。代码如下：

```
function get($url) {  
  
    //初使化curl  
    $ch = curl_init();  
  
    //请求的url, 由形参传入  
    curl_setopt($ch, CURLOPT_URL, $url);  
  
    //将得到的数据返回  
    curl_setopt($ch, CURLOPT_RETURNTRANSFER, 1);  
  
    //不处理头信息  
    curl_setopt($ch, CURLOPT_HEADER, 0);  
  
    //连接超过10秒超时  
    curl_setopt($ch, CURLOPT_TIMEOUT, 10);  
  
    //执行curl  
    $output = curl_exec($ch);  
  
    //关闭资源  
    curl_close($ch);  
  
    //返回内容  
    return $output;  
}
```

我们现在使用我们所写的get方法，请求网易的一个列表，将标题和url抓取出来。

我们可以先用get方法中传入一个URL。得到这个网址所对应网页的html。

网址为新媒体观察网的新闻列表页：<http://www.xmtnews.com/events>

。

将红色区域采集下来：



## 一、得到彩红色区间的html

这个区间从下面的HTML代码开始：

```
<section class="ov">
```

在以下代码结束：

```
<div class="hr-10"></div>
```

使用preg\_match写一个正则表达式就匹配就得到了红色区间的HTML。将匹配到的HTML赋值给变量\$area。

匹配的正则表达式如下：

```
<section class="ov">(.*?)<div class="hr-10"></div>/mis'
```

## 二、在红色区域匹配标题和标题的URL

我们发现所有的标题都在 <h3> 标签里面。我们使用preg\_match\_all写一个正则表达式。

```
preg_match_all('/<h3><a href="(.*?)" title="(.*?)" class="headers" target="_blank">(.*?)</a>'
```

本文档使用 看云 构建

将url和内容匹配出来的内容放置到\$find中，将\$find数组，打印出来就可以看到匹配的结果了。

如果需要，也可以循环读取显示每一行标题和每一行URL。

全部代码演示如下：

```
<?php

$content = get('http://www.xmtnews.com/events');

preg_match('/<section class="ov">(.*?)<div class="hr-10"><\div>/mis', $content, $match);

//将正则匹配到的内容赋值给$area
$area = $match[1];

preg_match_all('/<h3><a href="(.*?)" title=".*?" class="headers" target="_blank">(.*?)<\a><\h3>/', $area, $find);

var_dump($find);

function get($url) {

    //初使化curl
    $ch = curl_init();

    //请求的url, 由形参传入
    curl_setopt($ch, CURLOPT_URL, $url);

    //将得到的数据返回
    curl_setopt($ch, CURLOPT_RETURNTRANSFER, 1);

    //不处理头信息
    curl_setopt($ch, CURLOPT_HEADER, 0);

    //连接超过10秒超时
    curl_setopt($ch, CURLOPT_TIMEOUT, 10);

    //执行curl
    $output = curl_exec($ch);

    //关闭资源
    curl_close($ch);

    //返回内容
    return $output;
}
```



## 15.3 使用post发送数据

如果我们要发送POST数据怎么办？又需要使用curl帮我们来发送数据。

按照步骤我们自定义了一个函数，函数名为：post。post这个方法中需要传入两个参数：

1. 请求的URL地址
2. 发送的数据

发送的数据全为数组，以键值对的形式用POST方法发送给指定的接口地址即可。

我们只需要把《15.1 curl的使用步骤组合一下》就可以完成对应的代码。

在开发微信公众号创建自定义菜单的时候就需要使用到POST方法向微信的自定义菜单接口发送自定义菜单数据。

post的自定义函数，全部代码如下：

```
function post($url, $data) {  
  
    //初使化init方法  
    $ch = curl_init();  
  
    //指定URL  
    curl_setopt($ch, CURLOPT_URL, $url);  
  
    //设定请求后返回结果  
    curl_setopt($ch, CURLOPT_RETURNTRANSFER, 1);  
  
    //声明使用POST方式来进行发送  
    curl_setopt($ch, CURLOPT_POST, 1);  
  
    //发送什么数据呢  
    curl_setopt($ch, CURLOPT_POSTFIELDS, $data);  
  
    //忽略证书  
    curl_setopt($ch, CURLOPT_SSL_VERIFYPEER, false);  
    curl_setopt($ch, CURLOPT_SSL_VERIFYHOST, false);  
  
    //忽略header头信息  
    curl_setopt($ch, CURLOPT_HEADER, 0);  
  
    //设置超时时间  
    curl_setopt($ch, CURLOPT_TIMEOUT, 10);  
  
    //发送请求  
    $output = curl_exec($ch);  
  
    //关闭curl  
    curl_close($ch);  
  
    //返回数据  
    return $output;  
}
```

以后在调用微信公众平台或者其他第三方API系统。它们需要使用POST方法要求你发送数据的时候。你需要使用POST来发送数据的时候，只需要调一下这个post方法就可以了。

## 16. 用PHP写一个论坛

---

16.1 web2.0始于论坛

16.2 需求：开发前你要知道他的样子

16.3 核心业务流程

16.4 数据库表设计

16.5 文件和代码规范

16.6 核心功能说明

## 16.1 web2.0始于论坛

---

优酷、百度贴吧、问答、朋友圈评论、商品评论、QQ空间，可以不夸张的讲，都是由论坛衍生出来的。它们都是典型的web2.0产品，web2.0中的一个重要特点就是UGC（用户原创内容，有人也称：用户创造内容）。

UGC并不是某一种具体的业务，而是一种用户使用互联网的新方式，即由原来的以下载为主变成下载和上传并重。随着互联网运用的发展，网络用户的交互作用得以体现，用户既是网络内容的浏览者，也是网络内容的创造者。

BBS最早是用来公布股市价格等类信息的。早期的BBS与一般街头和校园内的公告板性质相同，只不过是过来传播或获得消息而已。一直到开始普及之后，有些人尝试将苹果计算机上的BBS转移到个人计算机上，BBS才开始渐渐普及开来。

国内最大的免费开源的BBS(discuz)，在网站中的安装量高达300万。

很多企业、个人站长直接使用开源的BBS进行创业。使用开源的BBS创业，在2000至2013年创建了一大批成功的个人创业者。有些甚至通过论坛创业成为了千万、亿万富翁。

出名的论坛包括：一些地区论坛、一些行业论坛、猫扑社区、凯迪社区、天涯社区、西祠胡同、搜房业主论坛... ..等等。

可以这么讲，会开发论坛了。你将会开发几乎所有的互联网的web2.0的应用。

从像是贴吧、知道、B2C商城、微信微应用、甚至优酷等在线视频网站。

学会了论坛开发，你完全可以在技术上实现一个中小型的上述的站点。

因为他们的业务模式与论坛的业务模式都是一样，都是UGC类业务。

只需要认真学习这章，踏实的写出代码，你很快就能够写出一个论坛。并且在论坛的基础上，扩展出来不同的互联网业务开发，开始自己的互联网之梦！

## 16.2 需求：开发前你要知道他的样子

我们不知道项目需求，我们根本无从下手。

如果不知道需求，就像要造一样女神，只能想象女神的样子。而造出来的女神，又不是用户所想要的。

可以这么讲，将需求弄明白了。就搞明白了，在没完成之前，我们就知道未来开发完后的东西是什么样的样子。

当然需求的采集、产生、策划、定制... ..涉及到太多跨学科的知识，不是我们今天这本书里面涵盖的内容。我们直接跨过需求的采集部分，为大家还原一整套需求，从已确定的需求中开始项目的开发。

### 1.论坛定位

本论坛的定位和开发是根据大家之前所学的知的基础上，将基础知识和内容进行整合而开发出来的教学项目。

大家通过之前的学习的组合，认真学，认真写后会发现自己掌握了很多知识，可是为什么写不出一个中小型的项目呢？

因为，在互联网的业务中需要非常明确自己要做什么。再根据做的东西，每一步每一步把业务需求转换成为代码。

之前的知识学好后，绝大多数人在转换的过程当中也会略有一些阵痛。怎么解决呢？

解决办法：通常不知道需求是什么，不能根据需求设计出来流程和数据表结构和规范。本章会帮你解决好核心的业务流程，提供数据库，并提供全面的代码，你只需要够静下心来一个一个功能写出来。

本章只推荐初学 PHP 且面向过程编程的人学习。

本章需要你付出一定的心智去解决一些或大或小的隐藏任务，以达到真正理解什么是论坛系统(BBS) 和如何运用 PHP独立开发论坛系统的目的。

采用面向过程的方式做完一个项目。在面向对象的过程中，你需要解决的只是了解知识点。

在以后，你学习的更加强大，你会发现面向过程和面向对象没有多大的区别。

你学习了面向对象的知识后，不需要再经历业务转化代码的过程。只需要将代码以面向对象的方式写出来而已。

不用再经历复杂的需求变代码的过程。

### 2.前台、后台；前端、后端

了解了论坛的定位，我们再讲几个基本概念：前台、后台；前端、后端。

前台：就是用户看到的那些东西，提供给所有用户使用。用户包括注册用户和普通游客，二者所拥有的功能不同。系统前台主要包括用户登录、用户注册、发表帖子、论坛浏览、回复内容浏览等功能模块。

后台：有时也称为网站管理后台，通常需要帐号及密码等信息的登陆验证，登陆信息正确则验证然后进入网站后台的管理界面进行相关的一系列操作：可对论坛中相关信息进行管理、维护，普通用户无权使用。后台管理主要包括用户管理、版块管理、帖子管理、回复管理、管理员注销等功能模块。

前端：手机端我们认为是前端，只展现和操作界面，而数据处理通常在远程服务器上存储着。它直接给用户展示内容，我们认为直接给用户展示，我们认为是前端。网页，我们认为是前端。前端主要是指html、css、javascript写出来的页面给用户直接看到的展现。

后端：用户的登陆、用户密码的处理都需要在服务器端处理。用户的银行卡余额等都存在后端服务器中，我们将服务器处理、存储数据和业务逻辑的部份存为后端。

### 3.总体业务流程介绍

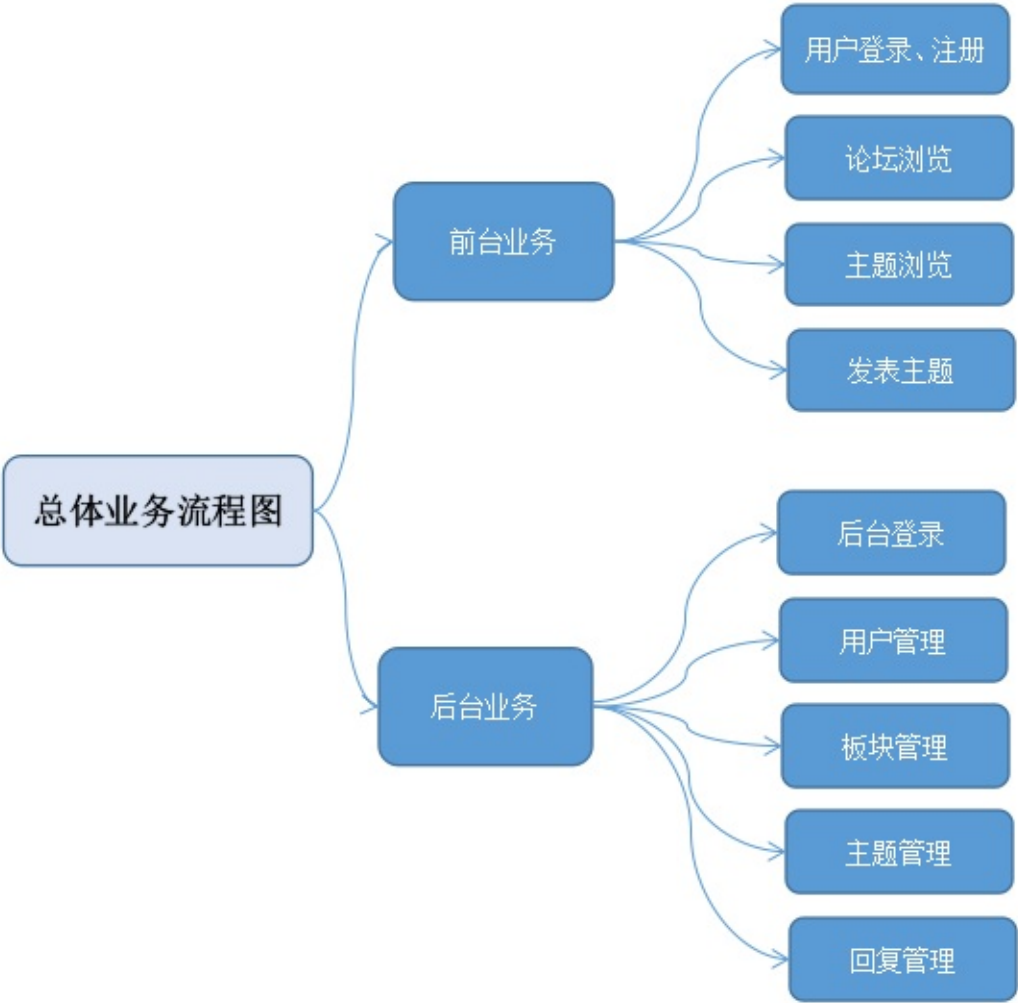
前台业务流程主要包括：用户注册、用户登录；论坛浏览：显示论坛名称、显示论坛创建时间；主题浏览：回复文章作者信息、作者相关信息浏览、原文浏览、回复文章浏览；发表主题：主题回复、发表新主题。

后台业务流程主要包括：后台登录、管理员注销；用户管理：删除用户、查询用户、编辑用户；栏目管理：更换版主、新增论坛、删除栏目；主题管理：主题查询、主题删除；回复管理：回复文章查询、回复文章删除。

前端：手机端我们认为是前端，只展现和操作界面，而数据处理通常在远程服务器上存储着。它直接给用户展示内容，我们认为直接给用户展示，我们认为是前端。网页，我们认为是前端。前端主要是指html、css、javascript写出来的页面给用户直接看到的展现。

后端：用户的登陆、用户密码的处理都需要在服务器端处理。用户的银行卡余额等都存在后端服务器中，我们将服务器处理、存储数据和业务逻辑的部份存为后端。

其处理流程如下图所示：



## 4.内容规划

### 4.1 角色设置

#### 4.1.1 管理员

## 16.2 需求：开发前你要知道他的样子

管理员权限，可进入后台，可进行回帖的管理、版块管理和用户管理等操作；

### 4.1.2 版主

版主权限，在前台管理帖子内容，包括将帖子设置为置顶、精华等操作。管理回帖内容，可屏蔽用户的回帖内容；

### 4.1.3 注册用户

在前台浏览帖子(付费帖需要消费对应的积分才能浏览)，发表帖子、回复；

### 4.1.4 访客

与注册用户的权限类似，可浏览免费帖子和回复内容，无发帖和回帖功能；

## 4.2 功能体现

### 4.2.1 前台功能主要包括：

用户注册、用户登录；论坛浏览：显示论坛名称、显示论坛创建时间；帖子浏览：回复的作者信息、作者相关信息浏览、原文浏览、回复内容浏览；发表帖子：帖子回复、发表新帖子。

### 4.2.2 后台功能主要包括：

后台登录、管理员注销；用户管理：删除用户、查询用户、编辑用户；版块管理：更换版主、新增版块、删除版块；帖子管理：帖子查询、帖子删除；回复管理：回复的查询、回复的删除。

## 4.3 奖励机制

为鼓励用户积极参与，设置如下奖励机制：

注册：赠送50积分；

登陆：每天首次登陆赠送2积分；

发帖：赠送2积分；

回帖：赠送1积分；

## 4.4 论坛版主的权利及义务

版主负责的版面需及时回复网友的问题及为论坛制造气氛，并能及时删除含有非法内容的贴(例如含广告、色情、反动等)

## 16.3 核心业务流程

---

16.3.1 用户注册流程

16.3.2 普通用户和管理员登陆流程

16.3.3 发贴流程

16.3.4 回复流程

16.3.5 版块管理流程

16.3.6 版主业务流程

16.3.7 金币奖励和消耗流程

# 16.3.1 用户注册流程

## 注册功能说明

用户注册是网上最最常见的一个功能，也是我们开始体验一个网站服务的开始。

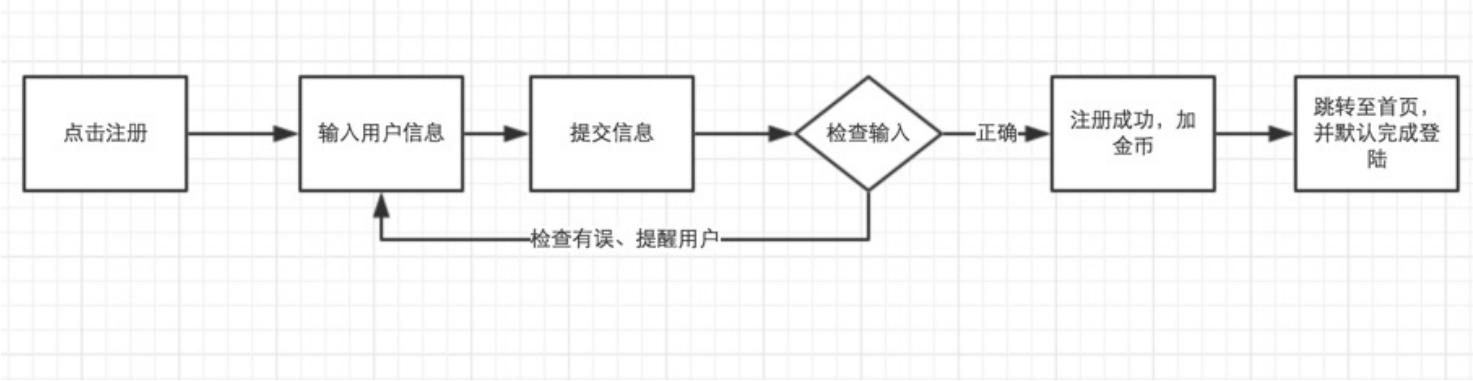
例如：我们在京东、淘宝等网站注册了才能发贴。我们在微信、QQ等社交工具注册了才能够开始加好友开始聊天。

可以这么来讲，如果网站需要我们有个人信息，那么注册是我们体验一个互联网服务的开始的第一步。

而我们在开发过程当中，也会设计按照自己业务的注册流程。

注册流程可以不同，也可以相同。我们现在将本论坛的注册流程用流程图来表示：

## 注册功能流程图



注意：大家可以在《16.6 核心功能说明》中下载安装后，进一步体验每一步流程。

## 用图分解注册流程演示

第一步：点击注册按钮进入注册界面



第二步：在注册页输入注册信息



\*用户名:

---

\*密码:

---

\*确认密码:

---

\*Email:

---

\*验证码:

---


[提交](#)

第三步：判断是否用户输入错误，如果输入错误显示错误信息



用户名长度错误：用户名由 **3 到 12** 个字符组成  
密码长度错误：由 **3 到 12** 个字符组成  
错误：邮箱不合法  
验证码输入错误，请稍候.....  
[如果你的浏览器没有自动跳转，请点击此链接](#)

第四步：如果输入信息无误，提示用户注册成功，并完成跳转。



感谢您的注册，现在将以会员身份登录站点，请稍候.....  
[如果你的浏览器没有自动跳转，请点击此链接 +50](#)

注：

请参考《16.4 数据库表设计》听用户表结构设计。对着流程完成用户注册代码的实现。

## 16.3.2 普通用户和管理员登陆流程

### 普通用户和管理员登陆的区别

我们在玩游戏的时候，会听到一个非常厉害的词:GM(游戏管理员)。游戏管理员，有一个专门的系统可以做到：

- 1. 删除用户
- 2. 封掉一个用户
- 3. 加一个新的游戏地图
- 4. 查看某些用户的聊天违规行为给出警告

... ..

大家会发现，GM在游戏里面有。她与普通用户有一些区别，她有更高的操作权限。

在我们的论坛、办公管理系统、小说网站等系统中也会设计这一类特殊的用户。

这类特殊用户我们统一称之为：管理员。

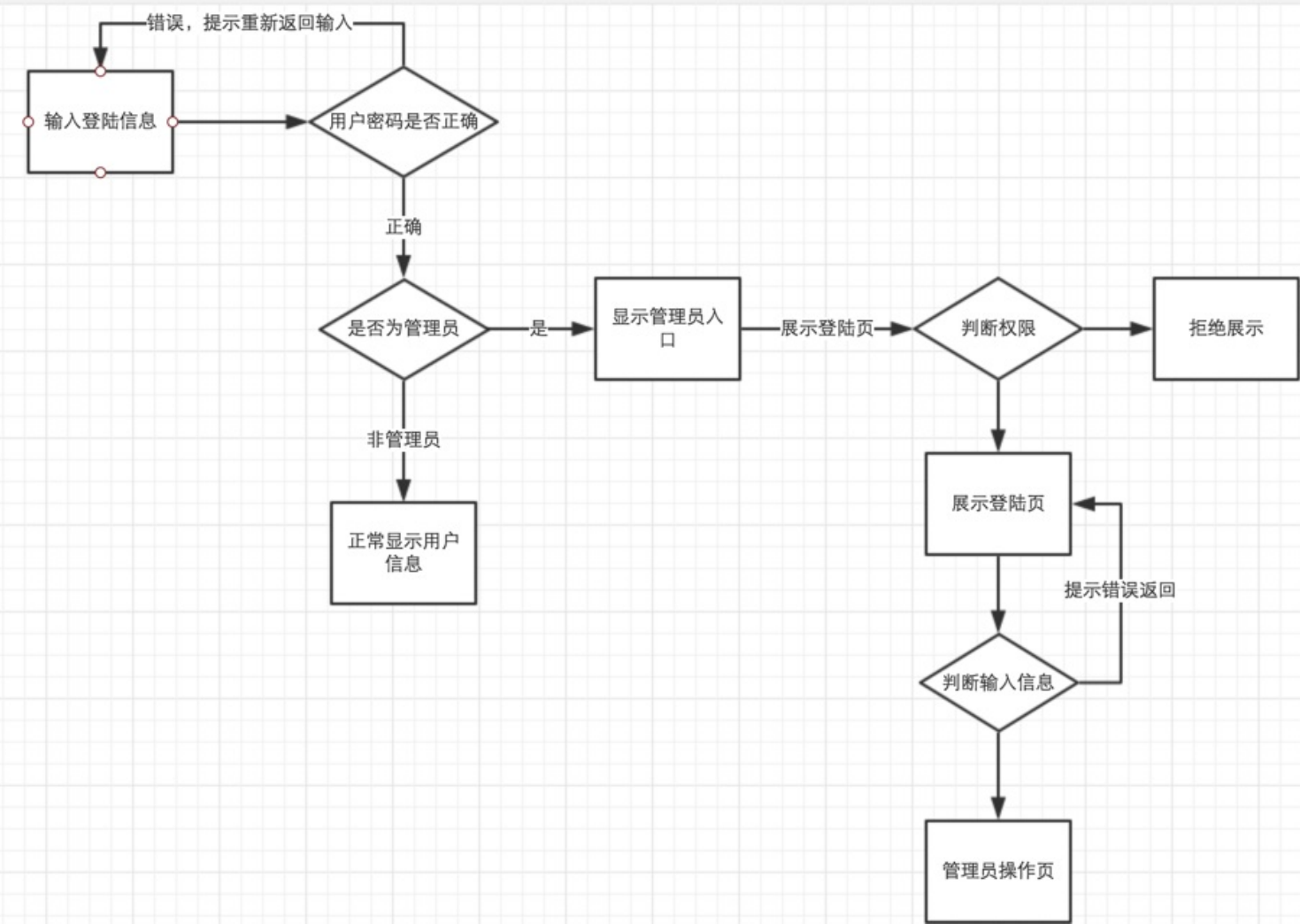
论坛系统中普通用户和管理员的区别：

权限	普通用户	管理员用户
注册	有	有
登陆	有	有
发贴	有	有
回贴	有	有
删自己贴	有	有
删他人贴	无	有
删除用户	版主有，用户没有	有
添加版主	无	有
禁止用户	无	有
添加版本	无	有
修改网站名等基本信息	无	有
添加友情链接	无	有

注意：

所在板块的普通用户中，有一种特殊用户叫版主用户。版主与普通用户的区别是：版主这种普通用户，能够管理自己所在板块的贴子。删除贴子、屏蔽贴子等。

### 普通用户、管理员登陆流程



流程说明：

- 1. 用户输入信息后，显示用户名和密码是否正确，不正确返回重新输入
- 2. 用户名和密码正确则激活用户登陆的cookie
- 3. 用户如果是管理员展示示管理员入口；否则不显示，按普通用户操作
- 4. 管理员用户可进入管理员登陆页，进入前判断是否有进入该页的权限，如果没有则拒绝。
- 5. 输入管理员的帐号密码，进入验证通过则放行，否则拒绝通过。

## 图解登陆流程

### 1.输入用户名和密码界面

用户名

☐ 自动登录

密码

登录

### 2.普通用户登陆成功展示

liwenkai

设置

退出

积分: 50 | 用户权限: 普通用户

3.管理员登陆成功展示



4.管理员点击管理中心进入后台展示页

Discuz! 管理中心

Discuz! 是 腾讯 旗下 Comsenz 公司推出的以社区为基础的专业建站平台，帮助网站实现一站式服务。

用户名: admin

密 码:

提 问: 无安全提问

回 答:

提交

5.后台界面展示

Discuz! Control Panel

你好, 创始人 admin [退出] 站点首页

站点信息 用户管理 版块管理 帖子管理

站点信息

友情链接

站点信息

站点名称: 10分钟学院 站点名称, 将显示在浏览器窗口标题等位置

网站名称: phpxy 网站名称, 将显示在页面底部的联系方式处

网站 URL: http://www.phpxy.com/ 网站 URL, 将作为链接显示在页面底部

网站备案信息代码: 京ICP备 89273号 页面底部可以显示 ICP 备案信息, 如果网站已备案, 在此输入你的授权码, 它将显示在页面底部, 如果没有请留空

关闭站点

关闭站点: 否 暂时将站点关闭, 其他人无法访问, 但不影响管理员访问

提交

Powered by phpxy V2

bbs.com/admin\_index.php



# 16.3.3 发贴流程

## 发贴说明

某个用户登陆成功，就可以使用某个用户发表贴子。

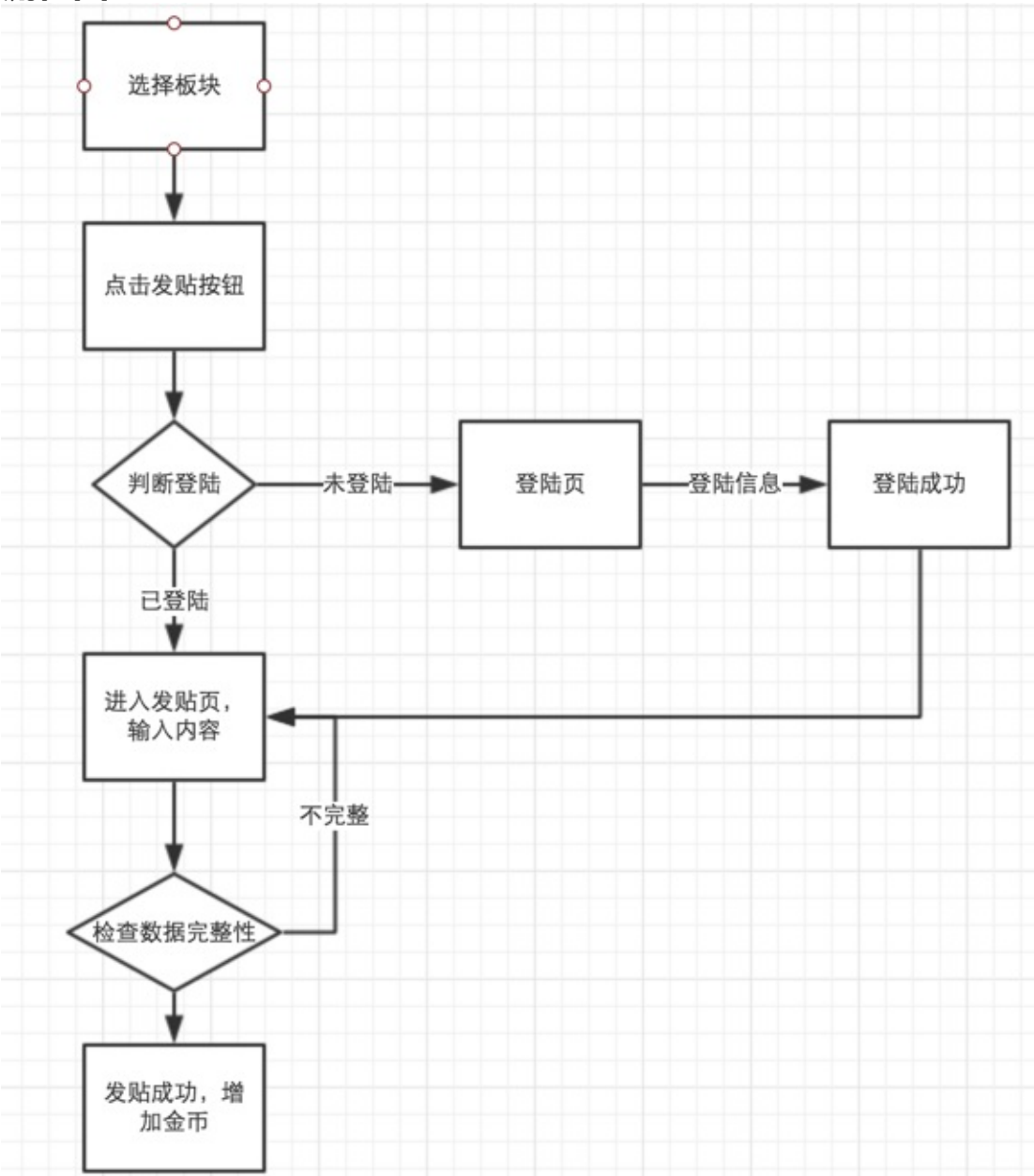
我们将贴子分为了普通贴和金币贴。

它们的区别：

- 1. 普通贴不需要花钱就可以看
- 2. 金币贴必须要使用注册时、每天登陆、回贴时获得的金币购买贴子后才能查看。

在发贴前，我们先要选择对应的版本，在所在贴块中点击发贴，写入对应的内容点击发表，发表成功。

## 流程图



说明：

- 1. 用户选择对应的版块后，点击发帖时传入了版块的ID
- 2. 在用户的发帖界面，将板块的ID放入了隐藏的表单中
- 3. 用户点发帖时，其实是相对应的

图解发帖流程

1.登陆成功后选择论坛中的相应版块，如：选择内核源码

帖子: 9 | 会员: 4 | 欢迎新会员: liwenkai

PHP技术交流		
内核源码 版主: test001, admin	10 / 9	本论坛超管的账号、密码都为admin 2015-10-08 10:49:55 admin
PHP框架 版主: test002, admin	0 / 0	从未
-----		
开源产品 版主: admin	0 / 0	从未
-----		
进阶讨论 版主: admin	0 / 0	从未
-----		
程序人生		
求职招聘 版主: admin	0 / 0	从未
-----		
经验分享 版主: admin	0 / 0	从未
-----		
名人故事 版主: admin	0 / 0	从未

2.点击发帖按钮发表帖子

论坛 > PHP技术交流 > 内核源码

版块导航

PHP技术交流

内核源码

PHP框架

开源产品

进阶讨论

程序人生

求职招聘

经验分享

名人故事

内核源码 今日: 0 | 主题: 9  
版主: test001, admin

发帖

返回

筛选: 全部 | 精华

	作者	回复/查看	最后发表
关于php你可能不知道的—php的事件驱动化设计 - [售价 10 金钱] 精	admin 2015-10-08	0 14	admin 2015-10-08 10:46:58
本论坛超管的账号、密码都为admin	admin 2015-10-08	0 5	admin 2015-10-08 10:49:55
树操作算法，非递归方法 - [售价 5 金钱]	admin 2015-10-08	0 2	admin 2015-10-08 10:46:07
【架构与优化】从facebook中集思广益，相互学习	admin 2015-10-08	10 21	test002 2015-10-08 10:59:19
深入理解zend sapi(zend sapi internals)	admin 2015-10-08	0 1	admin 2015-10-08 10:43:48

3.输入发帖内容、金币等

发表帖子

还可输入 80 个字符

源码

样式普通字体大小A A

body p

◦ 售价

售价: 0 金钱(最高 30 )

发表帖子

4.贴子发表成功给用户虚拟金币奖励

帖子发表成功，请稍候.....

如果你的浏览器没有自动跳转，请点击此链接

发帖赠送 金钱+2



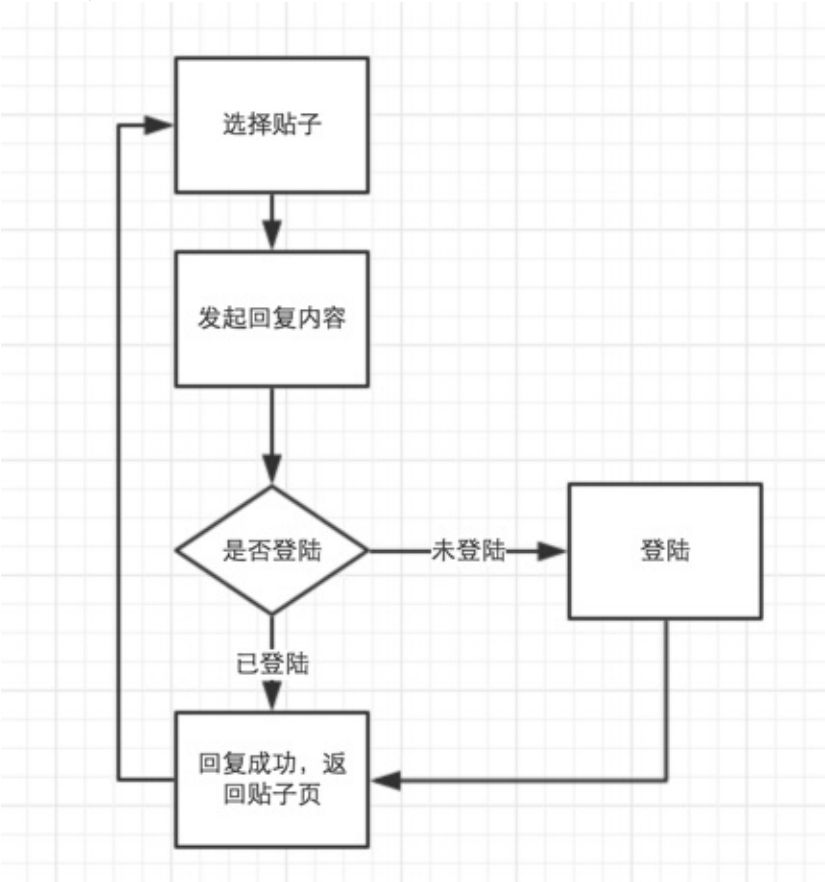
# 16.3.4 回复流程

## 回复流程说明

我们遇到自己感兴趣的贴子，我们在下面可以的输入框中再次发表自己的观点和看法。

回帖成功后，系统可以针对这个用户加上对应的金币。

## 回复流程图



注意：

- 1. 选择贴子发起回复，判断用户是否登陆
- 2. 如果用户已登陆则直接发表成功，在发表的隐藏表单中放入这个贴子的ID
- 3. 用户没登陆进入登陆页，登陆成功后返回至原贴地址重新发起回复

## 图解回复流程

1.选择并查看贴子

[设为首页](#) [收藏本站](#)



[admin](#) | [设置](#) | [管理中心](#) | [退出](#)

积分: 7 | 用户权限: 管理员



[首页](#) | [PHP技术交流](#) | [程序人生](#)

[热搜: 活动 交友 教程](#)

[论坛](#) > [PHP技术交流](#) > [内核源码](#) > [【架构与优化】从facebook中集思广益，相互学习](#)

[删除主题](#) | [置顶](#) | [高亮](#) | [精华](#)

查看: 22 | 回复: 10

[admin](#)



[管理员](#)  
[学士](#)

[【架构与优化】从facebook中集思广益，相互学习](#)

 发表于 2015-10-08 10:45:17

[楼主](#) [电梯直达](#)

本人才学疏浅，提出这个问题只是为了抛砖引玉，希望大家能在自己擅长的领域里对facebook中可能采用的某个技术或者架构方案或优化方法，做个自己的论述。根据2008年9月1日，腾讯网对facebook的技术运营副总裁jonathan heiliger的访问，我们了解到：

- (1) facebook自称全球流量第四的网站，每天9000万活跃用户访问facebook的1万台服务器。
- (2) 25tb数据（08年12月为28tb），40万名外部开发者支持，开发了2.5万套应用软件。
- (3) 使用lamp（[linux](#)、[apache](#)、[mysql](#)、[php](#)）技术构架；数据库使用mysql；使用memcached缓存sql加速（全球最大的分布式memcached缓存，800多台服务器，光缓存在memcached中数据就达20多tb）；使用apc进行opcode编译缓存。关于php的编译执行如下图所示：而facebook的缓存方案如下图所示：
- (4) 一些后台应用是用python、perl和java，以及一些gcc和boost。
- (5) 使用svn和git来进行代码管理，并且全部企业内部的软件部署都采用开源程序。

**2. 可能会采用的mysql架构与优化**

2.在贴子下面输入想要回复的内容





我可以选择一个内容进行回复

body p

发表回复

3.回复成功并跳转回原贴地址



**帖子回复成功**，请稍候.....

如果你的浏览器没有自动跳转，请点击[此链接](#)

回帖赠送 金钱+1

## 16.3.5 版块管理流程

### 版块管理

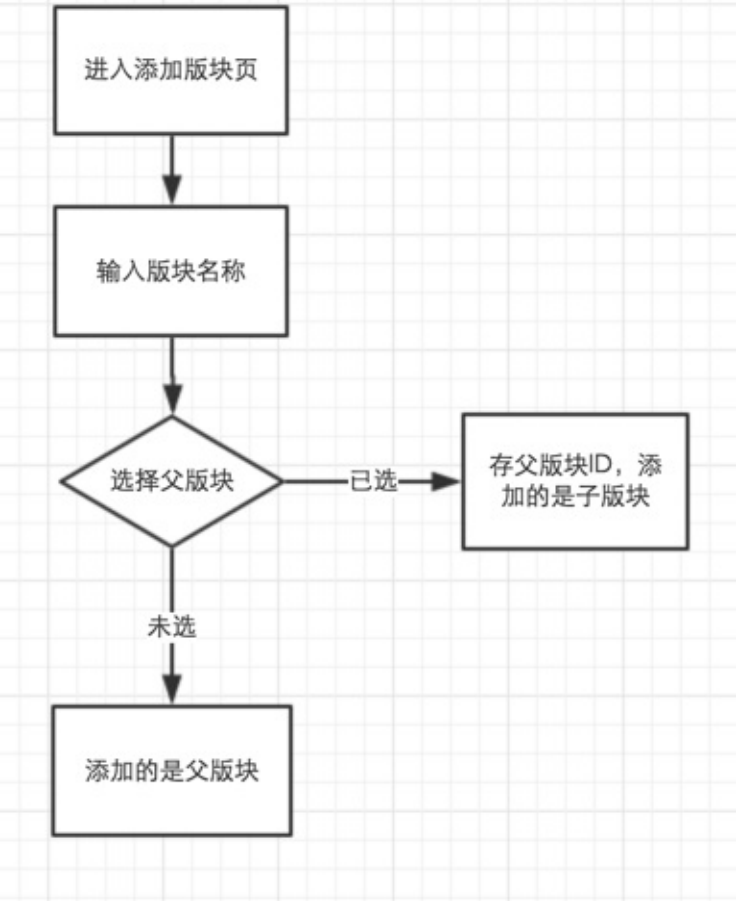
在百度贴吧中，我们可以创建不同的贴吧。不同的贴吧，在论坛中就是版块。

不同的版块（贴吧）讨论不同的内容。

论坛的版本在结构上与贴吧略有一些小的不同：

- 1. 贴吧是单个层级的，没有父板块
- 2. 论坛分为大版块和小版块。而小版本添加时有一个父版块，也就是小版本是属于哪个大版块的。例如电脑版块下，有电脑维修和软件故障。

### 版块管理流程图



### 图解版本管理

- 1.如果是添加大版本在选择大版块是就不填写，这样就产生了一个大版本。如果我们是指定了大版块添加一个新版块，那么新版块就属于这个大版块

Discuz!

Control Panel

站点信息

用户管理

版块管理

帖子管理

管理版块

添加版块

添加版块

技巧提示

添加时不选择大版块即为添加大版块

版块名称

选择大版块

-不选择-

提交

2.展示版块

Discuz!

Control Panel

站点信息

用户管理

版块管理

帖子管理

管理版块

添加版块

版块管理

显示顺序

版块名称

2

PHP技术交流

4

内核源码

3

PHP框架

2

开源产品

1

进阶讨论

1

程序人生

3

求职招聘

2

经验分享

1

名人故事

提交

## 16.3.6 版主业务流程

### 版主业务说明

在真实的项目中一个人很难将所有的工作全部完成。需要其他人来协助它。

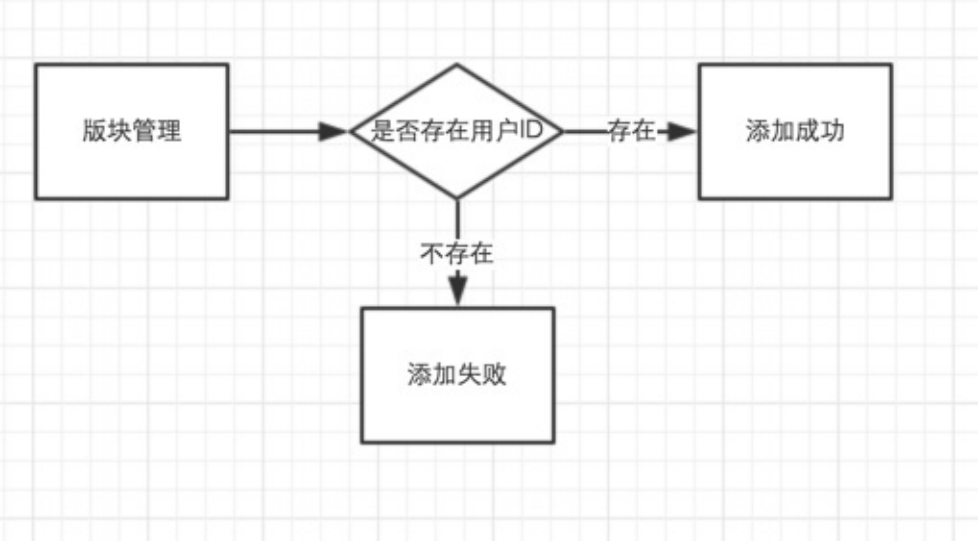
例如在贴吧中可以指定某个热心网友为某个贴吧的小吧主，协助完成某些工作。

论坛中也是，管理员不可能某个版块都能够有足够的精力管理。因此，它可以指定某个用户为版主。

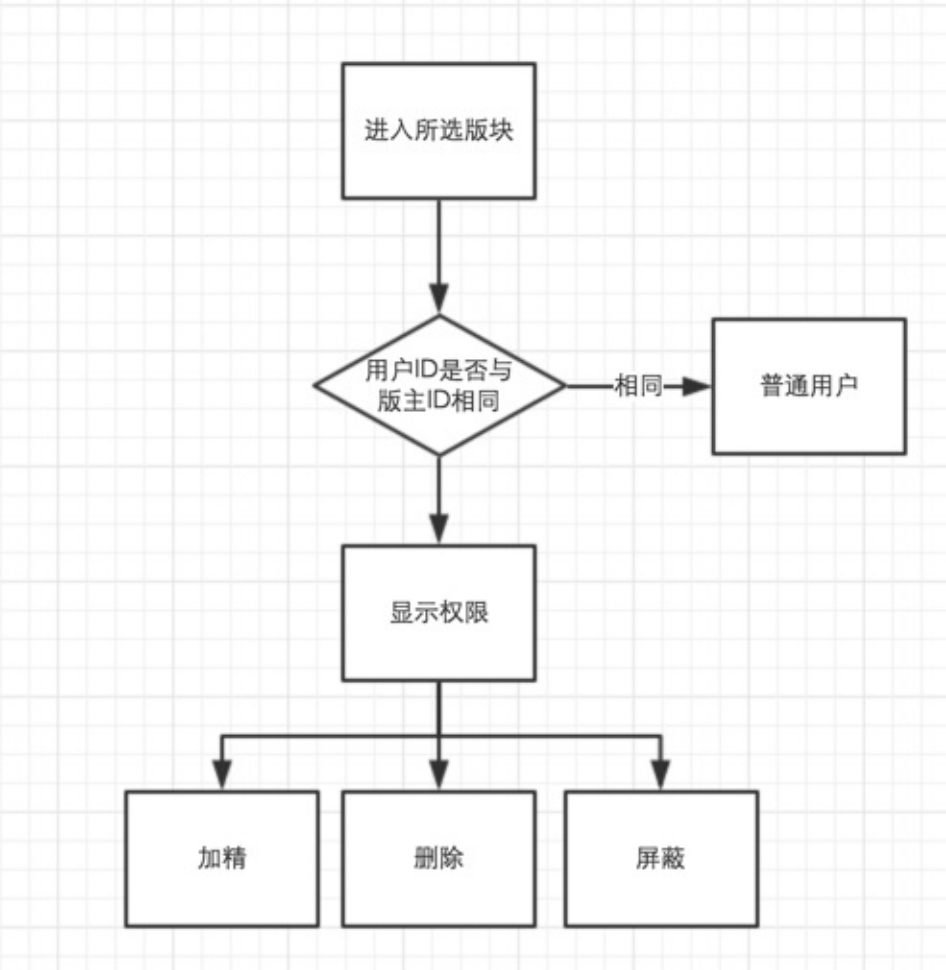
版主能够删掉这个版块的广告、违法信息屏蔽掉等功能。

因此，我们在这里也设计了一个版主功能。

### 添加版主流程



### 版主管理流程



图解版主管理

1.在后台管理界面中展示版本，在版主添写入写上用户的唯一ID

站点信息 | 用户管理 | 版块管理 | 帖子管理

版块管理

显示顺序

版块名称

2

PHP技术交流

4

内核源码

3

PHP框架

2

开源产品

1

进阶讨论

1

程序人生

3

求职招聘

2

经验分享

1

名人故事

提交

版主

1,8

1,9

1

1

1

1

1

1

2.在前台展示这个用户为某个版块的版主

帖子: 10 | 会员: 4 | 欢迎新会员: liwenkai

PHP技术交流

内核源码

版主: test001, admin

11 / 10

已有业务换成pdo遇到的问题  
1分钟前 admin

PHP框架

版主: test002, admin

0 / 0

从未

开源产品

版主: admin

0 / 0

从未

进阶讨论

版主: admin

0 / 0

从未

程序人生

3.版主具有删除贴子等权限，而普通用户没有

论坛 > PHP技术交流 > 内核源码 > 已有业务换成pdo遇到的问题

发帖 回复 返回列表

查看: 4 | 回复: 1

admin

管理员

学士

已有业务换成pdo遇到的问题

发表于 1天前

楼主 电梯直达

公司的老项目从adodb改成pdo，产生了好多问题。升一个版本会遇到很多问题，有个预处理问题最严重。

pdo没有bigint的类型，预处理的时候会出现问题。

当pdo::attr\_emulate\_prepares这个参数 设置为false的时候才会出现这个问题。

pdo::param\_int (integer)  
represents the sql integer data type.

这个类型在mysql里面会超出int 整型所指定的范围。mysql存的是bigint，只能用param\_str去代替。

这个事先没有意识到，紧急的处理是将bigint映射成pdo的param\_str，不至于被转换成int。

回复

admin

发表于 1天前

沙发

我可以选择一个内容进行回复

普通用户此处没有操作功能

4.版主操作功能区展示



已有业务换成pdo遇到的问题

发表于 1天前

楼主 电梯直达

公司的老项目从adodb改成pdo，产生了好多问题。升一个版本会遇到很多问题，有个预处理问题最严重。

pdo没有bigint的类型，预处理的时候会出现问题。

当pdo::attr\_emulate\_prepareds这个参数 设置为false的时候才会出现这个问题。

```
pdo::param_int (integer)  
represents the sql integer data type.
```

这个类型在mysql里面会超出int 整型所指定的范围。mysql存的是bigint，只能用param\_str去代替。

这个事先没有意识到，紧急的处理是将bigint映射成pdo的param\_str，不至于被转换成int。

回复

删除 | 置顶 | 精华

发表于 1天前

沙发

## 16.3.7 金币奖励和消耗流程

### 积分业务说明

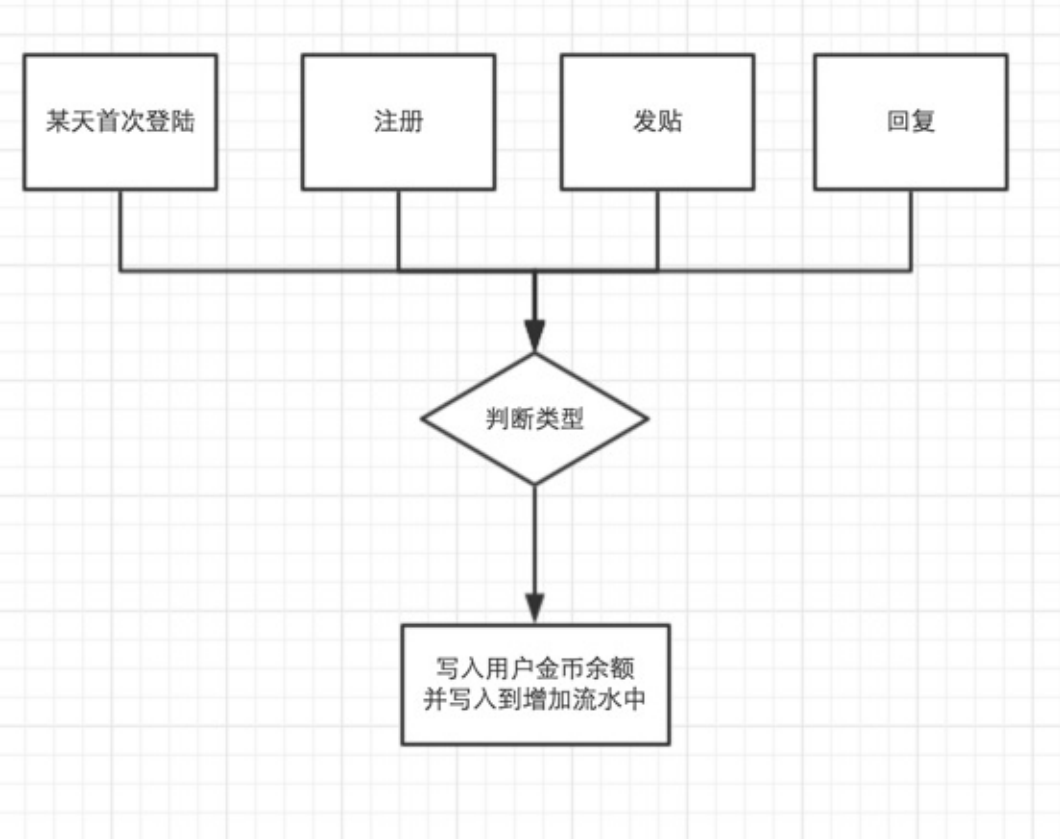
在需求部分我们讲解过积分（金币）的处理机制，机制如下：

- 注册：赠送50积分；
- 登陆：每天首次登陆赠送2积分；
- 发帖：赠送2积分；
- 回帖：赠送1积分；

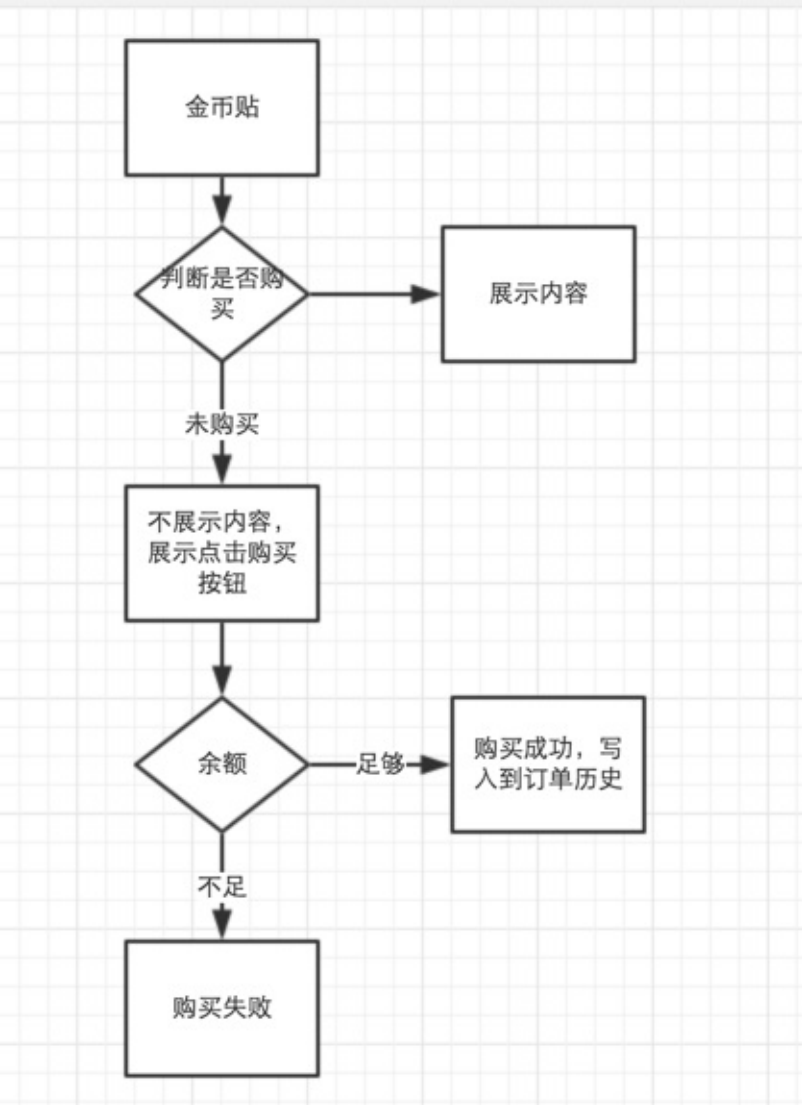
通过这种金币（积分）业务的处理，能够极大的丰富市场运营部门的活动的可用性，增强用户粘性，让用户觉得自己的发帖和回帖是有价值和意义的。

例如：每天首次登陆送2个积分，就像京东商城的京豆一样。每天登陆，我可以送用户金豆，让用户经常访问，增加消息和购物的冲动。

### 增加金币流程图



### 消耗金币流程图



### 增加金币图例

在注册、回复成功和发贴成功等都会显示金币增加的相关图例。



帖子回复成功，请稍候.....

如果你的浏览器没有自动跳转，请点击此链接


回帖赠送 金钱+1



**帖子发表成功**，请稍候.....

如果你的浏览器没有自动跳转，请点击此链接

发帖赠送 金钱+2



**感谢您的注册**，现在将以会员身份登录站点，请稍候.....

如果你的浏览器没有自动跳转，请点击此链接+50

## 展示金币贴图例

### 关于php你可能不知道的一php的事件驱动化设计

 发表于 2015-10-08 10:46:58

楼主 电梯直达 ☐ 

 本主题需向作者支付 **10 金钱** 才能浏览

 购买主题

本主题已加入精华

 回复

# 16.4 数据库表设计

## 数据库设计：E-R图

ER图概念化地构建实体间关系的模型，这使得它们区别于数据库模型图。

ER图的理念是：项目所有参与者能理解ER图。

ER图由不同实体类型、关系、特性和类型构成。

实体是诸如用户的实际对象，有时更抽象，但必须有业务意义。

特性用于描述实体，关系用于实体之间：

- ( 1 ) 实体：现实世界中的事物；

( 2 ) 属性：事物的特性；

( 3 ) 联系：现实世界中事物间的关系。

实体集的关系有一对一、一对多、多对多的联系。

## 数据库表及关系建立

1. 用户基本资料表 (bbs\_user)

用户基本资料表						
bbs_user						
字段名	数据类型	长度	主键	可空	默认值	备注
uid	int	11	是	否	无	AUTO_INCREMENT
username	char	16	否	否	无	账号
password	char	32	否	否	无	密码(md5加密)
email	char	30	否	否	无	邮箱
udertype	tinyint	2	否	否	无	账号类型(1=管理员, 0=用户)
regtime	int	12	否	否	无	注册时间
lasttime	int	12	否	否	无	最后登陆时间
regip	int	12	否	否	无	注册IP
picture	varchar	255	否	否	public/images/avatar_blank.gif	头像
grade	int	10	否	是	0	积分
problem	varchar	200	否	是	NULL	找回密码问题
result	varchar	200	否	是	NULL	答案
realname	char	50	否	是	NULL	真实姓名
sex	tinyint	4	否	是	2	性别(1=男, 2=女)
birthday	varchar	20	否	是	NULL	生日
place	varchar	50	否	是	NULL	所在地
qq	char	13	否	是	NULL	QQ号码
autograph	varchar	500	否	是	NULL	个人签名
allowlogin	tinyint	2	否	否	0	是否允许登陆(0=允许, 1=不允许)

1. IP黑名单表 (bbs\_closeip)

IP黑名单表  
bbs\_closeip

字段名	数据类型	长度	主键	可空	默认值	备注
id	int	10	是	否	无	AUTO_INCREMENT
ip	int	10	否	否	无	IP地址(用到的函数: ip2long)
addtime	int	10	否	否	无	记录添加时间
overtime	int	10	否	是	NULL	IP限制的结束时间

1. 友情链接表 (bbs\_link)

友情链接表  
bbs\_link

字段名	数据类型	长度	主键	可空	默认值	备注
lid	smallint	6	是	否	无	AUTO_INCREMENT
displayorder	tinyint	2	否	否	0	排序(数字大排在前面)
name	varchar	30	否	否	无	名称
url	varchar	255	否	否	无	链接跳转地址
description	mediumtext	0	否	是	NULL	描述
logo	varchar	255	否	是	NULL	LOGO的URL地址
addtime	int	12	否	否	无	添加时间

1. 论坛版块表 (bbs\_category)

论坛版块表 bbs_category						
字段名	数据类型	长度	主键	可空	默认值	备注
cid	int	10	是	否	无	AUTO_INCREMENT
classname	varchar	60	否	否	无	版块名称
parentid	int	10	否	否	无	父级ID
classpath	char	20	否	是	NULL	关系
replycount	int	10	否	否	0	回帖数量
motifcount	int	10	否	否	0	帖子数量
compere	char	10	否	是	NULL	版主
classpic	varchar	255	否	否	public/images/forum.gif	版块ICON
description	mediumtext	0	否	是	NULL	版块描述
orderby	smallint	6	否	否	0	排序
lastpost	varchar	255	否	是	NULL	最后发表
namestyle	char	10	否	是	NULL	
ispass	tinyint	2	否	否	1	审核状态

1. 帖子信息表 (bbs\_details)

帖子信息表  
bbs\_details

字段名	数据类型	长度	主键	可空	默认值	备注
id	init	10	是	否	无	AUTO_INCREMENT
first	tinyint	1	否	否	0	帖子或回复 (1=帖子, 0=回复)
tid	int	10	否	否	无	帖子ID
authorid	int	10	否	否	无	发布人ID
title	varchar	600	否	否	无	帖子标题
content	mediumtext	0	否	否	无	帖子内容/回帖内容
addtime	int	12	否	否	无	发布时间
addip	int	12	否	否	无	发布人IP
classid	int	10	否	否	无	版块ID
replycount	int	12	否	否	0	帖子的回复数量
hits	int	12	否	否	0	浏览次数
istop	tinyint	2	否	否	0	是否置顶贴(1=是, 0=否)
elite	tinyint	2	否	否	0	是否精华贴(1=是, 0=否)
ishot	tinyint	2	否	否	0	是否热门贴(1=是, 0=否)
rate	smallint	3	否	否	0	帖子售价(积分)
attachment	smallint	3	否	是	NULL	帖子附件
isdel	int	2	否	否	0	是否放入回收站
style	char	10	否	是	NULL	帖子标题颜色(css样式)
isdisplay	int	2	否	否	0	是否屏蔽回复内容(1=是, 0=否)

1. 帖子订单表 (bbs\_order)

帖子订单表  
bbs\_order

字段名	数据类型	长度	主键	可空	默认值	备注
oid	int	10	是	否	无	AUTO_INCREMENT
uid	int	11	否	否	无	用户ID
tid	int	11	否	否	无	帖子ID
rate	int	11	否	否	无	价格
addtime	int	11	否	否	无	创建时间
ispay	tinyint	2	否	否	0	是否已支付(0=未支付, 1=已支付)



## 16.5 文件和代码规范

### 通用注释

#### 1. 文件的注释通用样例（普通程序文件，类文件，函数文件，变量定义文件）

```
/**
 * XXXXX的文件
 *
 * 功能1： xxx
 * 功能2： xxx
 *
 * @file      $Source: /home/doc/php开发注释规范.md  $
 * @package   core
 * @author    Joy <anzhengchao@gmail.com>
 * @version   $Id: php开发注释规范.txt,v 1.1 2014/03/04 20:37:46 Joy Exp $
 * @link      http://www.joychao.cc
 */
```

- @package 是团队事先定义好的，在phpdocumentor里同一package的文件可以给出跟踪的链接。项目开发前需要对其进行定义。
- @link 行后面接的地址是程序开发文档的地址，因为我们目前没有在线的程序开发文档库，所以可不加。注意注释的排版，左端保持对齐。

**说明：**以上自动更新版本及文件名需要配置svn，具体请自行google 'SVN自动版本号'

#### 2. 普通函数注释

```
/**
 * 获取头像地址
 *
 * @author Joy <anzhengchao@gmail.com>
 *
 * @param string $imageName 图片文件名
 * @param integer $size     大小
 *
 * @return string
 */
function getAvatarUrl($imageName, $size = 80)
{
    return sprintf(SITE_URL . '/service/images/cropped_%s/'. $imageName, $size);
}
```

顺序按照author、param、return来放，区块间空行。

#### 3. 程序段落注释

段落注释和逻辑注释使用如下方式

```
/**
 * 1 如果$_GET['do']等于buy,则购买条码
 */
if($_GET['do'] == 'buy')
{
    // 1.1 验证用户提交变量是否合法
    if($_POST['strCodeNum'])
    {

    }
}
```

```
// 1.2 验证用户提交的码是否可以购买

// 1.3 .....
} // end if

/**
 * 2 如果$_GET['do']等于list,显示用户选择的条码
 */
if($_GET['do'] == 'list')
{
    // 2.1 验证用户提交变量是否合法
    if($_POST['strCodeNum'])
    {

    }
    // 2.2 验证用户提交的码是否可以购买

    // 2.3 .....
} // end if
```

## 16.6 核心功能说明

---

### 项目实例下载地址

代码你可以登陆：

<http://down.phpxy.com/book/bbs.zip> 下载论坛项目实例。

# 16.6.1 项目目录结构说明

项目目录结构和作用、及主要文件说明：

目录	文件	描述
/		根目录
	install.lock	安装后生成的锁文件
	index.php	前台入口文件
	admin.php	后台入口文件
/theme		模板目录
/theme/default		默认模板样式目录
/public		公共静态资源目录
/public/js		JS文件目录
/public/css		CSS文件目录
/public/images		图片文件目录
/public/ckeditor		Ckeditor 网页编辑器目录
/public/admin		后台静态文件目录
/upload		上传文件目录
/install		安装文件目录
	apple_bbs.sql	网站安装使用的数据库脚本
/install/images		安装程序静态资源目录
/common		公共文件目录
	admin_common.php	后台公共文件，用于验证管理员登陆(SESSION)状态
	common.php	网站公共文件，用于校验是否已经安装、加载数据库配置文件、网站基础配置配置文件、公共函数文件、初始化SESSION等等
/helpers		公共函数目录
	code.php	验证码函数库
	fileupload.php	文件上传函数库
	func.php	公共函数库
	img.php	图片处理函数库
	keywords.php	热门搜索关键词数组
	mysql.php	Mysqli函数库
	mytpl.php	模版引擎函数库
	page.php	分页函数库
	user.php	用户相关函数库
/config		配置文件目录
	config.php	网站基础配置信息

	database.php	数据库配置信息
<b>/compiled</b>		模板缓存文件目录
...	...	...

## 16.6.2 公共文件的使用

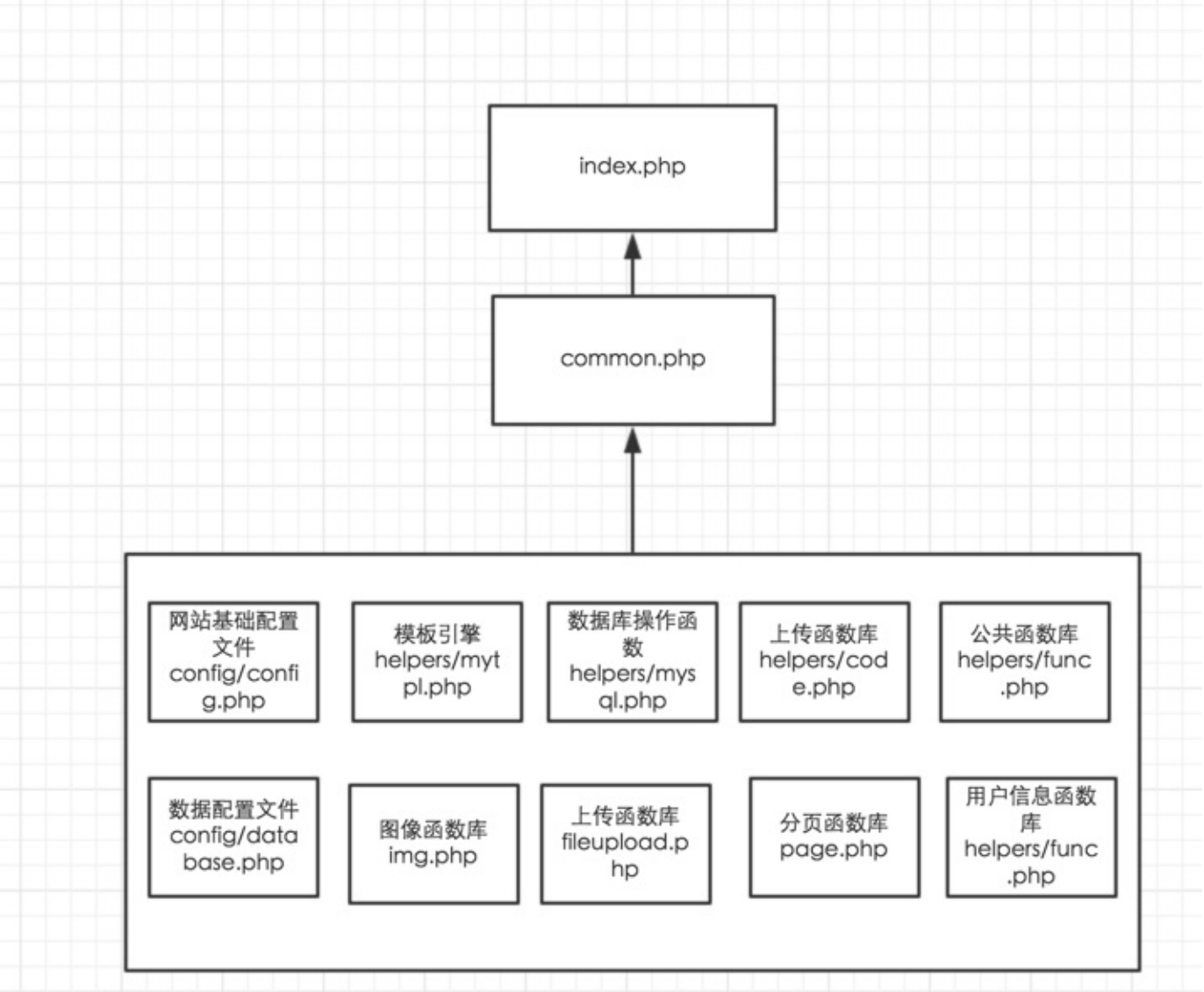
### 网站公共文件 /common/common.php

在 PHP 文件的顶部，使用 `include` 方式加载 `common/common.php` 文件，即可使用配置文件中定义的常量和函数库中的所有函数，并且使其校验过程在当前页面中生效。

网站公共文件的作用：

- 设置 PHP的错误级别、网页字符集；
- 用于校验站点是否已经安装；
- 加载 `config` 中的所有文件、及 `helpers` 中的所有文件；
- 初始化SESSION和网站的公共变量；
- 网站的开启或关闭状态；
- 校验禁止访问的IP地址；
- 获取主导航的相关数据；

### 公共文件包含关系



## 代码详解

```
<?php

error_reporting(E_ALL & ~E_NOTICE);
header('content-type:text/html; charset=utf-8;');

//项目安装
if(!file_exists('install.lock'))
{
    header('location:install/index.php');
    exit;
}

/**
 * 加载文件
 */
include 'config/database.php'; //加载数据库配置文件
include 'config/config.php'; //加载站点基础配置文件
include 'helpers/mytpl.php'; //加载模板函数文件
include 'helpers/mysql.php'; //加载数据库操作函数文件
include 'helpers/user.php'; //加载数据库操作函数文件
include 'helpers/func.php'; //加载公共函数文件
include 'helpers/img.php'; //加载图片处理函数文件
include 'helpers/fileupload.php'; //加载文件上传函数文件
include 'helpers/code.php'; //加载图片验证码函数文件
include 'helpers/page.php'; //加载分页函数文件
$keywords = include 'helpers/keywords.php'; //加载热搜关键词文件

session_start(); //启动 SESSION

//初始化变量
$web_name = WEB_NAME;
$web_btm = WEB_BTM;
$web_url = WEB_URL;
$web_icp = WEB_ICP;
$web_close = WEB_ISCLOSE;
$web_reg = WEB_REG;
$domain_resource = DOMAIN_RESOURCE;

//获取当前页面地址
$fileName = explode('/',$_SERVER["PHP_SELF"]);
$thispage = array_pop($fileName);

//网站关闭, 管理员可以登录
if(empty($_POST['loginsubmit']))
{
    if($web_close && !$_COOKIE['udertype'] && $thispage!='logout.php')
    {
        echo $web_close.'---'. $_COOKIE['udertype'].'---'.$thispage;
        include 'close.php';
        exit;
    }
}

//禁止IP
$Uip = ip2long($_SERVER['REMOTE_ADDR']);
$ip = dbSelect('closeip','*', 'ip='.$Uip.', 'id desc',1);
if ($ip)
{
    exit('本站不欢迎你, 请离开!');
}

if ($_COOKIE['uid'])
{
    $GGUser = dbSelect('user', 'grade,picture', 'uid=' . $_COOKIE['uid'], '', 1);
    if ($GGUser)
    {
        $GGgrade = $GGUser[0]['grade'];
        $GGpicture = $GGUser[0]['picture'];
    }
}
```

```

}

//读取所有大版块信息,做导航
$headMenu = dbSelect('category','cid,classname','parentid=0 and ispass=1','orderby desc,cid desc');

```

## 基础配置 /config/config.php

包含以下配置项：

- 1 基础配置(时区|魔术转译)
- 2 静态资源
- 3 website
- 4 模版引擎
- 5 奖励

```

<?php

// 一
define('TIMEZONE', 'Asia/shanghai'); //时区
define('GPC', get_magic_quotes_gpc() ? 0 : 1); //魔术转译

// 二
define('DOMAIN_RESOURCE', 'http://funcbbs.yhsong.com/public'); //静态资源目录

// 三
define('WEB_NAME', 'PHP学院'); //站点名称, 将显示在浏览器窗口标题等位置
define('WEB_BTm', 'NoAlike'); //网站名称, 将显示在页面底部的联系方式处
define('WEB_URL', 'http://www.phpxy.com/'); //网站 URL, 将作为链接显示在页面底部
define('WEB_ICP', '京ICP备 89273号'); //页面底部可以显示 ICP 备案信息
define('WEB_ISCLOSE', false); //true暂时将站点关闭, 其他人无法访问, 但不影响管理员访问
define('WEB_REG', true); //true开启注册功能

// 四
define('TPL_SKIN', 'theme/default'); //模版文件存放位置
define('TPL_CACHE', 'compiled'); //模版文件缓存位置

// 五
define('REWARD_LOGIN', 2); //每天首次登陆赠送2积分
define('REWARD_REG', 50); //注册赠送50积分
define('REWARD_T', 2); //发帖赠送2积分
define('REWARD_H', 1); //回帖赠送1积分

```

## 数据库连接配置 /config/database.php

网站安装过程中会重写该文件 (确保该文件具有可写权限) 的内容, 无需手动修改配置。

```

<?php

define('DB_HOST','localhost'); //数据库连接地址(IP)
define('DB_USER','root'); //数据库登陆账号
define('DB_PASS','123456'); //数据库登陆账号对应的密码
define('DB_NAME','apple_bbs'); //数据库名称
define('DB_CHARSET','utf8'); //数据库字符集
define('DB_PREFIX','bbs_'); //数据表前缀

```



## 16.6.3 模板引擎讲解

### 什么是模版引擎？

百度百科的解释：

为了使用户界面与业务数据（内容）分离而产生的，它可以生成特定格式的文档，用于网站的模板引擎就会生成一个标准的HTML文档。

是不是听起来很晕？

用一句白话解释：

就是把静态页面中特定的一些标记（字符串）替换成最终需要的PHP标签；

好处：

可以让（网站）程序实现界面与数据分离，这就大大提升了开发效率，好的设计也使得代码重用变得更加容易。

### 编写自己的模板引擎

这里需要注意两点：

1 字符串替换，将 { 和 } 分别替换成 PHP 标签 <?php 和 ?>

2 为提升程序执行效率，使用模板缓存；

```
<?php

function template($tFile)
{
    $tmpFile = TPL_SKIN.'/'.$tFile;
    if(!file_exists($tmpFile)){
        return $tmpFile;
    }

    $comFile = TPL_CACHE.'/'.$basename($tFile, '.html').'.php';
    if (!file_exists($comFile) || filemtime($tmpFile)>filemtime($comFile))
    {
        tplParse($tmpFile,$comFile);
    }

    return $comFile;
}

function tplParse($tFile,$cFile){

    $fileContent=file_get_contents($tFile);

    $fileContent=preg_replace_callback("/^(\xef\xbb\xbf)/", function($r){}, $fileContent);
    $fileContent=preg_replace_callback(
        "/<!\-\-\s*\$\{(.+)\}\s*\-\-\>/is",
        function($r)
        {
            return str_replace('"', "'", '<?php echo ' . $r[1] . ' '; ?>');
        },
        $fileContent
    );

    $fileContent=preg_replace_callback(
        "/\{(\[\a-zA-Z0-9_\[\]\\\ \-\'\,\\%*\V\.\(\)\>\\\"$x7f-\xff\+\})\}/s",
        function($r)
        {
            return str_replace('"', "'", '<?php echo ' . $r[1] . ' '; ?>');
        },
        $fileContent
    );
}
```

```

    $fileContent
);
$fileContent=preg_replace_callback(
    "/\\\$\\{(.+?)\\}/is",
    function($r)
    {
        return str_replace('"', "'", '<?php echo ' . $r[1] . ' ; ?>');
    },
    $fileContent
);
$fileContent=preg_replace_callback(
    "/\<!\-\\-\\s*\\{else\\s*if\\s+(.+?)\\}\\s*\\-\\-\\>/is",
    function($r)
    {
        return str_replace('"', "'", '<?php } else if(' . $r[1] . '){ ?>');
    },
    $fileContent
);
$fileContent=preg_replace_callback(
    "/\<!\-\\-\\s*\\{elif\\s+(.+?)\\}\\s*\\-\\-\\>/is",
    function($r)
    {
        return str_replace('"', "'", '<?php } else if(' . $r[1] . '){ ?>');
    },
    $fileContent
);
$fileContent=preg_replace_callback(
    "/\<!\-\\-\\s*\\{else\\}\\s*\\-\\-\\>/is",
    function($r)
    {
        return "<?php } else { ?>";
    },
    $fileContent
);
for($i=0; $i<5; ++$i){
    $fileContent = preg_replace_callback(
        "/\<!\-\\-\\s*\\{loop\\s+(\S+)\s+(\S+)\s+(\S+)\s*\\}\\s*\\-\\-\\>(.+?)\<!\-\\-\\s*\\{\\loop\\}\\s*\\-\\-\\>/is",
        function($r)
        {
            return str_replace('"', "'", '<?php if(is_array('.$r[1].')){foreach('.$r[1]. ' AS ' . $r[2]. '=>' . $r[3]. '){ ?>' . $r[4]. '<?php }}?>');
        },
        $fileContent
    );
    $fileContent = preg_replace_callback(
        "/\<!\-\\-\\s*\\{loop\\s+(\S+)\s+(\S+)\s*\\}\\s*\\-\\-\\>(.+?)\<!\-\\-\\s*\\{\\loop\\}\\s*\\-\\-\\>/is",
        function($r)
        {
            return str_replace('"', "'", '<?php if(is_array('.$r[1].')){foreach('.$r[1]. ' AS ' . $r[2]. '){ ?>' . $r[3]. '<?php }}?>');
        },
        $fileContent
    );
    $fileContent = preg_replace_callback(
        "/\<!\-\\-\\s*\\{if\\s+(.+?)\\}\\s*\\-\\-\\>(.+?)\<!\-\\-\\s*\\{\\if\\}\\s*\\-\\-\\>/is",
        function($r)
        {
            return str_replace('"', "'", '<?php if('.$r[1].'){?>' . $r[2]. '<?php }}?>');
        },
        $fileContent
    );
}

$fileContent = preg_replace_callback(
    "#<!--\\s*\\{\\s*include\\s+([^\{\\}]+)\\s*\\}\\s*-->#i",
    function($r)
    {
        return '<?php include template("' . $r[1]. '");?>';
    },
    $fileContent
);

```

```

        file_put_contents($cFile,$fileContent);

        return true;

    }

```

## 如何使用自己的模板引擎？

公共文件 `/common/common.php` 中已经载入了模版引擎函数库，并指定了模板风格；

在 `/config/config.php` 中通过修改 `define('TPL_SKIN', 'theme/default');` 的值来，改变应用的模版风格(确保模板风格已经存在)

### 一 创建模板文件时使用的基本语法

#### 1 echo 输出变量：

```
{ $title }
```

#### 2 if 判断：

```
{ if $bigid }
```

```
...
```

```
{ else }
```

```
...
```

```
{ /if }
```

#### 3 foreach 循环：

```
{ loop $LTsMenu $key $val }
```

```
{ $val['title'] }
```

```
{ /loop }
```

#### 4 模版中使用函数

```
{ { getUsername($val['compere']) } }
```

#### 5 include 模板中包含模板：

```
{ include header.html }
```

## 二 使用模板

### 1 分配变量

在包含模版前定义的PHP变量，都可以在模版中使用：

```
$title = '首页 - ' . WEB_NAME;
```

### 2 包含模版文件

首先查找是否已经缓存，若未缓存 则替换标签后生成缓存文件。

```
include template("index.html");
```

## 16.6.4 用户注册、登陆功能讲解

### 注册功能

模板所在位置： `/theme/default/reg.html`

PHP页面 `reg.php`

需要完成的功能点：

- 1 对输入项要进行魔术转义，防止SQL注入；
- 2 验证邮箱格式；
- 3 验证密码：长度，并校验两次输入是否一致；
- 4 校验数据库中是否存在改用户名；
- 5 校验图片验证码输入是否正确；
- 6 若全部校验通过则创建用户，并自动登录，登陆状态使用 `Cookie` 记录；
- 7 注册成功，赠送积分；

```
<?php

include './common/common.php';

$title = '用户注册 - ' . WEB_NAME;

//验证是否为提交注册信息
if (!empty($_POST['regsubmit']))
{
    $uname = strMagic($_POST['username']);
    $upass = trim($_POST['password']);
    $urpass = trim($_POST['repassword']);
    $umail = $_POST['mail'];
    $pyzm = $_POST['yzm'];

    //错误跳转页默认值
    $url = $_SERVER['HTTP_REFERER'];
    $style = 'alert_error';
    $toTime = 3000;

    $alterNotice = false;    //提示页面标记位
    //验证用户名长度
    if(strlen($uname))
    {
        $alterNotice = true;
        $msgArr[] = '<font color=red><b>用户名长度错误：用户名由 3 到 12 个字符组成</b></font>';
    }

    //判断数据库里是否存在这个用户名
    $exists = dbSelect('user', 'uid', 'username="'.$uname.'"', 'uid desc', 1);
    if($exists)
    {
        $alterNotice = true;
        $msgArr[] = '<font color=red><b>用户名已存在</b></font>';
    }

    //验证密码长度
    if(strlen($upass))
    {
        $alterNotice = true;
        $msgArr[] = '<font color=red><b>密码长度错误：由 3 到 12 个字符组成</b></font>';
    }
}
```

```

//验证两次密码是否一致
if(str2Equal($upass, $surpass))
{
    $alterNotice = true;
    $msgArr[] = '<font color=red><b>错误：两次密码不一致</b></font>';
}

//验证email
if(checkEmail($umail))
{
    $alterNotice = true;
    $msgArr[] = '<font color=red><b>错误：邮箱不合法</b></font>';
}

//判断验证码
if(checkVerify($pyzm, $_SESSION['code']))
{
    $alterNotice = true;
    $msgArr[] = '<font color=red><b>验证码输入错误</b></font>';
}

//验证是否需要显示提示信息
if($alterNotice)
{
    $msg = join('<br />', $msgArr);
    include 'notice.php';
    exit;
}

//创建用户
$mone = REWARD_REG;
$n = 'username, password, email, udertype, regtime, lasttime, regip, grade';
$v = '$uname, '.md5($upass)."', '$umail', 0, ".time().", ".time().", ".ip2long($_SERVER['REMOTE_ADDR']).", ".$m
oney;

$result = dbInsert('user', $n, $v);
if(!$result)
{
    $msg = '<font color=red><b>注册失败，请联系管理员</b></font>';
    include 'notice.php';
}else{
    //注册成功后自动登录
    $result = dbSelect('user', 'uid,username,udertype,picture,grade', 'username="'.$uname.'" and password="'.md5($
upass).'", 'uid desc', 1);

    setcookie('uid',$result[0]['uid'],time()+86400);
    setcookie('username',$result[0]['username'],time()+2592000);
    setcookie('udertype',$result[0]['udertype'],time()+86400);
    setcookie('picture',$result[0]['picture'],time()+86400);
    setcookie('grade',$result[0]['grade'],time()+86400);

    $msg = '<font color=green><b>感谢您的注册，现在将以会员身份登录站点</b></font>';
    $url = 'index.php';
    $style = 'alert_right';
    include 'notice.php';

    $msg = '注册赠送';
    include 'layer.php';
}

}else{
    include template("reg.html");
}

?>

```

## 用户登陆

PHP页面 `login.php`

需要完成的功能点：

- 1 自动登陆功能，通过设置 **Cookie** 的过期时间来验证是否使用了自动登陆，有效期为30天。若浏览器 **Cookie** 被清除则自动失效；
- 2 验证登陆账号是否被管理员从后台锁定；
- 3 记录用户最后登陆时间；

```
<?php

include './common/common.php';

$username = strMagic($_POST['username']);
$password = trim($_POST['password']);
$cookietime = $_POST['cookietime'];

$result = dbSelect('user', 'uid, username, udertype, picture, grade, allowlogin, lasttime', 'username="' . $username . '" and password="' . md5($password) . '"');

//判断是否使用了自动登录
if($cookietime)
{
    $longTime = time()+2592000;
}else{
    $longTime = time()+86400;
}

if(!$result)
{
    $msg = '<font color=red><b>登录失败，用户名或密码错误</b></font>';
    $url = $_SERVER['HTTP_REFERER'];
    $style = 'alert_error';
    $toTime = 3000;
    include 'notice.php';
}else{
    if($result[0]['allowlogin'])
    {
        $msg = '<font color=red><b>您的账号已经被锁定，请联系管理员</b></font>';
        $url = $_SERVER['HTTP_REFERER'];
        $style = 'alert_error';
        $toTime = 3000;
        include 'notice.php';
        exit;
    }
    $money = REWARD_LOGIN;
    if(formatTime($result[0]['lasttime'])<date('Y-m-d'))
    {
        //更新最后登录时间,首次登陆还要加积分
        $lasttime = dbUpdate('user', 'lasttime='.time().',grade=grade+'.(int)$money.', 'uid='.$result[0]['uid'].');
        $first = true;
        $grade = $result[0]['grade']+(int)$money;
    }else{
        //更新最后登录时间
        $lasttime = dbUpdate('user', 'lasttime='.time().', 'uid='.$result[0]['uid'].');
        $grade = $result[0]['grade'];
    }
    setcookie('uid', $result[0]['uid'], $longTime);
    setcookie('username', $result[0]['username'], time()+2592000);
    setcookie('udertype', $result[0]['udertype'], $longTime);
    setcookie('picture', $result[0]['picture'], $longTime);
    setcookie('grade', $grade, $longTime);

    $msg = '<font color=green><b>登录成功</b></font>';
    $url = $_SERVER['HTTP_REFERER'];
    $style = 'alert_right';
    $toTime = 3000;

    include 'notice.php';
```

```
        if($first)
        {
            $msg = '每天登陆';
            include 'layer.php';
        }

    }
```

## 退出登陆状态

PHP页面 `logout.php`

Cookie 时间设置为当前时间-1，视为立即失效；

```
<?php

include './common/common.php';

setcookie('uid','',time()-1);
setcookie('udertype','',time()-1);
setcookie('picture','',time()-1);
setcookie('grade','',time()-1);

$msg = '<font color=green><b>您已退出站点，现在将以游客身份转入退出前页面</b></font>';
$url = 'index.php';
$style = 'alert_right';
$toTime = 3000;
include 'notice.php';
```

退出成功后，跳转到首页。

## 16.6.5 发帖功能讲解

### 发帖功能

模板所在位置： `/theme/default/addc.html`

PHP页面 `addc.php`

#### 操作流程

1. 用户点击进入发贴页的时候，将所选板块的ID带上。URL示例如下：<http://bbs.com/addc.php?classid=5>。我们就可以知道用户是向哪个版块发贴。
2. 判断用户是否存在
3. 判断版块是否存在，如果不存在的话，插入的版块即非法。我们让用户发贴不能成功。
4. 展示导航，版块名等基本信息，渲染进入模板
5. 准备好要插入的基本信息，拼接SQL语句，并插入库
6. 判断是否写入成功
7. 成功给用户金币（积分），跳转至贴子页

#### 1. 判断用户是否登陆，未登陆不能发贴

```
//包含公共组件
include './common/common.php';

//判断用户是否登录
if(!$ _COOKIE['uid'])
{
    $notice = '抱歉，您尚未登录，没有权限在该版块发帖';
    include 'close.php';
    exit;
}
```

#### 2. 判断ID是否存在

```
if(empty($_REQUEST['classid']) || !is_numeric($_REQUEST['classid']))
{
    $msg = '<font color=red><b>禁止非法操作</b></font>';
    $url = $_SERVER['HTTP_REFERER'];
    $style = 'alert_error';
    $toTime = 3000;
    include 'notice.php';
}
$classId = $_REQUEST['classid'];
```

#### 3. 读取版块相关的基本信息

```
$category = dbSelect('category','cid,classname,parentid','parentid<>0 and cid='.$_classId.'','','1');
if($category){

    $smallName = $category[0]['classname'];
    $smallId = $category[0]['cid'];
    $parentCategory = dbSelect('category','cid,classname','cid='.$category[0]['parentid'].'','','1');
    if($parentCategory)
    {
        $bigName=$parentCategory[0]['classname'];
        $bigId=$parentCategory[0]['cid'];
    }else{
        $msg = '<font color=red><b>非法操作</b></font>';
    }
}
```



```

        $url = $_SERVER['HTTP_REFERER'];
        $style = 'alert_error';
        $toTime = 3000;
        include 'notice.php';
        exit;
    }

    }else{
        $msg = '<font color=red><b>非法操作</b></font>';
        $url = $_SERVER['HTTP_REFERER'];
        $style = 'alert_error';
        $toTime = 3000;
        include 'notice.php';
        exit;
    }
}

```

## 4.准备发布内容

```

$authorid = $_COOKIE['uid'];          //发布人ID
$title = strMagic($_POST['subject']); //标题
$content = strMagic($_POST['content']); //内容
$addtime = time();                    //发表时间
$addip = ip2long($_SERVER['REMOTE_ADDR']); //发布人IP
$classid = $_POST['classid'];         //类别ID
$rate = $_POST['price'];               //帖子售价

$n = 'first, authorid, title, content, addtime, addip, classid, rate';

$v = '1, '.$authorid.', "'.$title.'", "'.$content.'", '.$addtime.', '.$addip.', '.$classid.', '.$rate.'';

$result = dbInsert('details', $n, $v);

```

## 整体代码展示

```

<?php
/**
 * 发帖子
 */

include './common/common.php';

//判断用户是否登录
if(!$_COOKIE['uid'])
{
    $notice = '抱歉，您尚未登录，没有权限在该版块发帖';
    include 'close.php';
    exit;
}

//判断ID是否存在
if(empty($_REQUEST['classid']) || !is_numeric($_REQUEST['classid']))
{
    $msg = '<font color=red><b>禁止非法操作</b></font>';
    $url = $_SERVER['HTTP_REFERER'];
    $style = 'alert_error';
    $toTime = 3000;
    include 'notice.php';
}
$classid = $_REQUEST['classid'];

//读取导航索引
$category = dbSelect('category', 'cid,classname,parentid', 'parentid<>0 and cid='.$classid.',', '1');
if($category){

    $smallName = $category[0]['classname'];
    $smallId = $category[0]['cid'];
    $parentCategory = dbSelect('category', 'cid,classname', 'cid='.$category[0]['parentid'].'', '1');
}

```

```

    if($parentCategory)
    {
        $bigName=$parentCategory[0]['classname'];
        $bigId=$parentCategory[0]['cid'];
    }else{
        $msg = '<font color=red><b>非法操作</b></font>';
        $url = $_SERVER['HTTP_REFERER'];
        $style = 'alert_error';
        $toTime = 3000;
        include 'notice.php';
        exit;
    }

}

}else{
    $msg = '<font color=red><b>非法操作</b></font>';
    $url = $_SERVER['HTTP_REFERER'];
    $style = 'alert_error';
    $toTime = 3000;
    include 'notice.php';
    exit;
}

//发布帖子
if($_POST['topicsubmit'])
{
    $authorid = $_COOKIE['uid'];          //发布人ID
    $title = strMagic($_POST['subject']); //标题
    $content = strMagic($_POST['content']); //内容
    $addtime = time();                    //发表时间
    $addip = ip2long($_SERVER['REMOTE_ADDR']); //发布人IP
    $classId = $_POST['classid'];         //类别ID
    $rate = $_POST['price'];              //帖子售价

    $n = 'first, authorid, title, content, addtime, addip, classid, rate';
    $v = '1, '.$authorid.', "'.$title.'", "'.$content.', '.$addtime.', '.$addip.', '.$classId.', '.$rate.'';
    $result = dbInsert('details', $n, $v);

    $insert_id = dbSelect('details', 'id', 'title="'.$title.'", id desc', 1);
    $insertId = $insert_id[0]['id'];
    if(!$result){

        $msg = '<font color=red><b>帖子发表失败, 请联系管理员</b></font>';
        $url = $_SERVER['HTTP_REFERER'];
        $style = 'alert_error';
        $toTime = 3000;
        include 'notice.php';
        exit;

    }

    }else{

        $money = REWARD_T; //发帖赠送积分
        $result = dbUpdate('user', "grade=grade+{$money}", 'uid='.$_COOKIE['uid'].'');

        //更新版块表的主题数量[Motifcount](跟帖是回复数量[eplycount])和最后发表[lastpost]
        $last = $insertId.'+|+ '.$title.'+|+ '.$addtime.'+|+ '.$_COOKIE['username'];
        $result = dbUpdate('category', 'motifcount=motifcount+1, lastpost="'.$last.'", cid='.$classId.'');

        $msg = '<font color=red><b>帖子发表成功</b></font>';
        $url = 'detail.php?id='.$insertId;
        $style = 'alert_right';
        $toTime = 3000;
        include 'notice.php';

        $msg = '发帖赠送';
        include 'layer.php';
        exit;

    }

}

}

$onMenu = dbSelect('category', 'cid,classname', 'cid='.$classId.' and ispass=1', 'orderby desc, cid desc');

```

```
if(!$OnMenu)
{
    $msg = '<font color=red><b>没有找到该版块</b></font>';
    $url = $_SERVER['HTTP_REFERER'];
    $style = 'alert_error';
    $toTime = 3000;
    include 'notice.php';
}else{
    $OnCid = $OnMenu[0]['cid'];
    $OnCname = $OnMenu[0]['classname'];
}

$title = '发表帖子' . $OnCname . ' - ' . $WEB_NAME;
$menu = $WEB_NAME;
include template("addc.html");
```

## 16.6.6 回帖功能讲解

---

### 回帖功能

模板所在位置： `/theme/default/detail.html`

PHP页面 `detail.php`

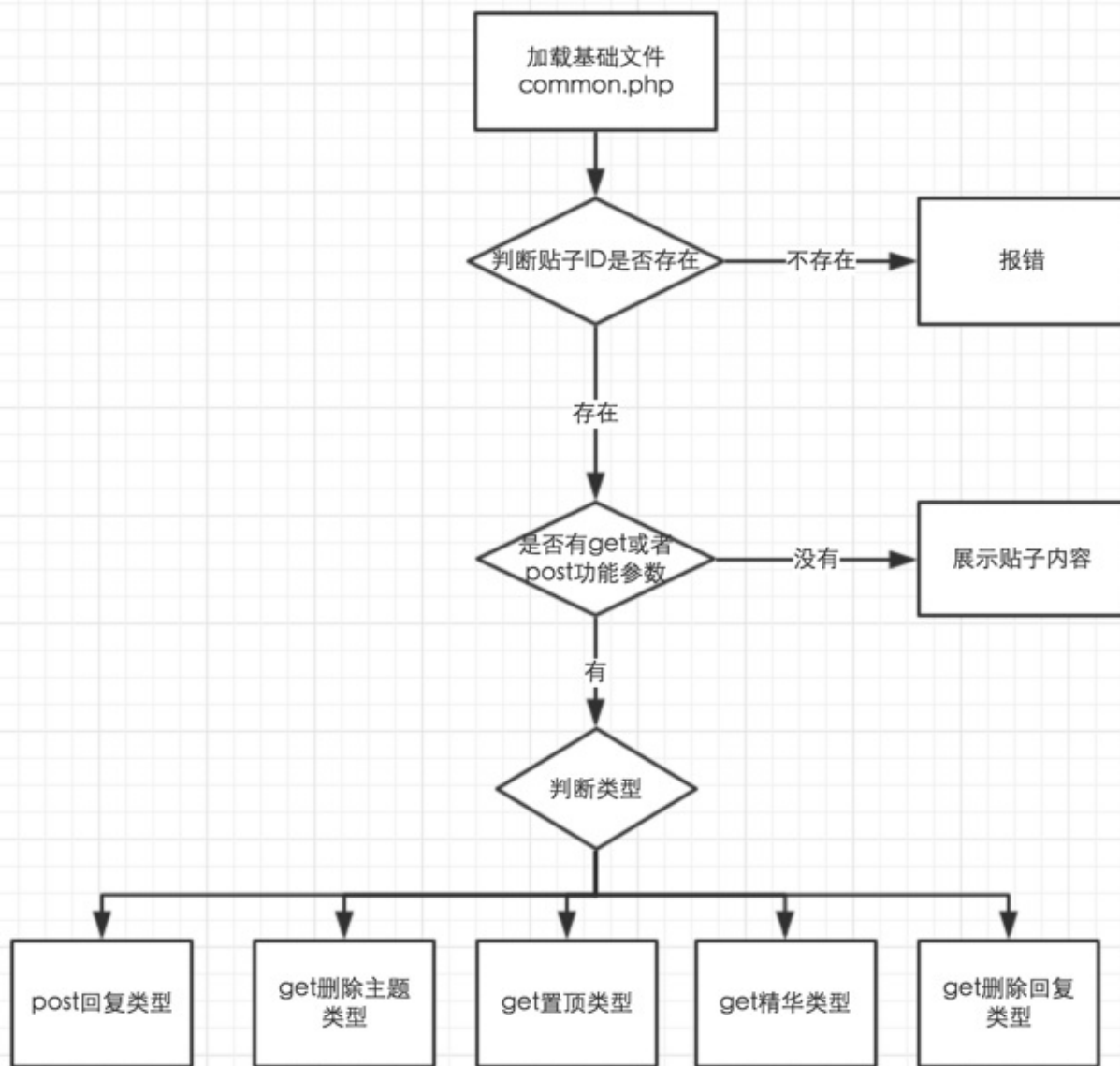
### 页面功能说明

detail.php页面可以展示、也可以回复，还能够实现删贴、高亮、精华、屏蔽等操作。

#### 操作流程

1. 加载基础
2. 判断是否
3. 判断get或者post传入的类型是什么
4. 若是回复则走回复流程，若是删除、精华、屏蔽则走对应的流程
5. 将回复内容和SQL语句准备完成，并写入数据库
6. 失败报错
7. 成功则提示成功，送金币跳转回原贴地址

### 回帖、删除、置顶功能流程图



## 整体代码演示

```

<?php
/**
 * 帖子详情
 */

include './common/common.php';

//判断帖子ID是否存在
if(empty($_REQUEST['id']) || !is_numeric($_REQUEST['id']))
{
    $msg = '<font color=red><b>禁止非法操作</b></font>';
    $url = $_SERVER['HTTP_REFERER'];
    $style = 'alert_error';
    $toTime = 3000;
    include 'notice.php';
}
$id=$_REQUEST['id'];

//保存帖子回复
if($_POST['replysubmit'])
{
    //判断用户是否登录
    if(!$_COOKIE['uid']){

```

```

        $notice='抱歉，您尚未登录';
        include 'close.php';
        exit;
    }

    $tid = $Id;                //跟帖时记录贴子ID
    $authorid = $_COOKIE['uid']; //发布人ID
    $content = strMagic($_POST['message']); //内容
    $addtime = time();          //发表时间
    $addip = ip2long($_SERVER['REMOTE_ADDR']); //发布人IP
    $classId = $_POST['classid']; //类别ID

    $n='first, tid, authorid, content, addtime, addip, classid';
    $v='0, '.$tid.', '.$authorid.', "'.$content.'" , '.$addtime.', '.$addip.', '.$classId.'';
    $result = dbInsert('details', $n, $v);

    if(!$result)
    {
        $msg = '<font color=red><b>回复失败，请联系管理员</b></font>';
        $url = $_SERVER['HTTP_REFERER'];
        $style = 'alert_error';
        $toTime = 3000;
        include 'notice.php';
        exit;
    }else{
        $money = REWARD_H; //回帖赠送积分
        $result = dbUpdate('user', "grade=grade+{$money}", 'uid'=>$_COOKIE['uid'].');

        //更新帖子的回复数量[replycount]
        $result = dbUpdate('details', 'replycount=replycount+1', 'id='.$tid.'');

        //更新版块表的回复数量[replycount]
        $result = dbUpdate('category', 'replycount=replycount+1', 'cid='.$classId.'');
        //header('location:detail.php?id='.$Id);
        $msg = '<font color=red><b>帖子回复成功</b></font>';
        $url = 'detail.php?id='.$Id;
        $style = 'alert_right';
        $toTime = 3000;
        include 'notice.php';

        $msg = '回帖赠送';
        include 'layer.php';

        exit;
    }
}

//点击帖子时访问次数加1
$result = dbUpdate('details', 'hits=hits+1', 'id='.$Id.' and isdel=0 and first=1');
if(!$result)
{
    $msg = '<font color=red><b>您浏览的帖子不存在或已被删除</b></font>';
    $url = $_SERVER['HTTP_REFERER'];
    $style = 'alert_error';
    $toTime = 3000;
    include 'notice.php';
}

//读取帖子信息
$TiZi = dbSelect('details','*', 'id='.$Id.' and isdel=0 and first=1','1');
$authorid = $TiZi[0]['authorid']; //作者ID
$title = $TiZi[0]['title']; //标题
$content = $TiZi[0]['content']; //内容
$addtime = getFormatTime($TiZi[0]['addtime']); //发布时间
$classId = $TiZi[0]['classid']; //版块ID
$replycount = $TiZi[0]['replycount']; //回复数量
$hits = $TiZi[0]['hits']; //点击数量
$elite = $TiZi[0]['elite']; //精华
$rate = $TiZi[0]['rate']; //所需积分数量

```

```

//读取上一条
$stop = dbSelect('details','id','id>'.$Id.' and isdel=0 and first=1','id desc',1);
if($stop)
{
    $stopid=$stop[0]['id'];
}else{
    $stopid=false;
}
//读取下一条
$down = dbSelect('details','id','id<'.$Id.' and isdel=0 and first=1','id desc',1);
if($down){
    $downid = $down[0]['id'];
}else{
    $downid = false;
}

//读取导航索引
$category = dbSelect('category','cid,classname,parentid,compere','parentid<>0 and cid='.$classId.'','',1);
if($category)
{
    $smallName = $category[0]['classname'];
    $smallId = $category[0]['cid'];
    $BanZhu = $category[0]['compere'];
    $parentCategory = dbSelect('category','cid,classname','cid='.$category[0]['parentid'].'','',1);
    if($parentCategory)
    {
        $bigName=$parentCategory[0]['classname'];
        $bigId=$parentCategory[0]['cid'];
    }else{
        $msg = '<font color=red><b>非法操作</b></font>';
        $url = $_SERVER['HTTP_REFERER'];
        $style = 'alert_error';
        $toTime = 3000;
        include 'notice.php';
        exit;
    }
}

}else{

    $msg = '<font color=red><b>非法操作</b></font>';
    $url = $_SERVER['HTTP_REFERER'];
    $style = 'alert_error';
    $toTime = 3000;
    include 'notice.php';
    exit;
}

//读取会员信息
$user = dbSelect('user','username,email,udertype,regtime,lasttime,picture,autograph,grade','uid='.$authorid.'','',1);
if($user)
{
    $U_sername = $user[0]['username'];
    $E_mail = $user[0]['email'];
    $U_dertype = $user[0]['udertype'];
    $R_egtime = formatTime($user[0]['regtime'],false);
    $L_asttime = formatTime($user[0]['lasttime'],false);
    $P_icture = $user[0]['picture'];
    $A_utograph = $user[0]['autograph'];
    $G_rade = $user[0]['grade'];
}

//该主题下的所有回复数量
$TZCount = dbFuncSelect('details','count(id)','tid='.$Id.' and isdel=0 and first=0');
$zCount = $TZCount['count(id)'];
$linum = 10;
$Lpage = empty($_GET['page'])?1:$_GET['page'];
//循环帖子回复信息
$select = 't.id as id,t.isdisplay as isdisplay,t.authorid as authorid,t.content as content,t.addtime as addtime,t.addip as addip,t.isdel as isdel,t.elite as elite,u.username as username,u.email as email,u.udertype as udertype,u.regtime as regtime,u.lasttime as lasttime,u.picture as picture,u.autograph as autograph,u.grade as grade';
$HTiZi = dbDuoSelect('details as t','user as u',' on t.authorid=u.uid',null,null,$select,'t.tid='.$Id.' and t.isdel=0

```

```

and t.first=0', 't.id asc', setLimit($linum));

$title = $Title.' - '.WEB_NAME;
$ggg = 'iPhone 游戏软件分享区';

//查找版主或管理员
$NBanZhu = explode(',', $BanZhu);
if(in_array($_COOKIE['uid'], $NBanZhu))
{
    $GuanLi=true;
}else{
    if($_COOKIE['udertype'])
    {
        $GuanLi=true;
    }
}

//给帖子付款
if(!empty($_POST['paysubmit']))
{
    //判断用户是否登录
    if(!$_COOKIE['uid'])
    {
        $notice='抱歉，您尚未登录';
        include 'close.php';
        exit;
    }

    foreach($_POST['oidarr'] as $key=>$val)
    {
        $nval=explode(',', $val);
        //将order表中的ispay更新为1
        $res = dbUpdate('order', 'ispay=1', 'oid='.$key.'');
        //扣钱
        $res = dbUpdate('user', 'grade=grade-'. $nval[1].', 'uid'=>$_COOKIE['uid'].'');
        //给作者加钱
        $res = dbUpdate('user', 'grade=grade+'. $nval[1].', 'uid='.$nval[0].');
    }
    header('location:detail.php?id='.$Id);
    exit;
}

//删除未购买的帖子
if(!empty($_POST['delsubmit']))
{
    //判断用户是否登录
    if(!$_COOKIE['uid'])
    {
        $notice='抱歉，您尚未登录';
        include 'close.php';
        exit;
    }
    $arrOid = array_keys($_POST['oidarr']);
    $NarrOid = join(',', $arrOid);
    $result = dbDel('order', 'oid in('.$NarrOid.')');
    header('location:detail.php?id='.$Id);
    exit;
}

//购买帖子,点击及加入订单表
if(!empty($_GET['pay']))
{
    //判断用户是否登录
    if(!$_COOKIE['uid'])
    {
        $notice='抱歉，您尚未登录';
        include 'close.php';
        exit;
    }
}

```



```

        //查询订单表中是否有这个购买记录
        $select = 't.title as title,t.authorid as authorid,o.oid as oid,o.tid as tid,o.uid as uid,o.rate as rate';
        $IsOrder = dbDuoSelect('order as o','details as t',' on o.tid=t.id',null,null,$select,'o.uid='.$_COOKIE['uid'].' and t.id='.$Id.','o.oid asc',1);
        if(!$IsOrder)
        {
            //如果没有购买记录, 加入订单表
            $Oresult = dbInsert('order', 'uid,tid,rate,addtime,ispay', $_COOKIE['uid'].'.'.$Id.','.$Rate.','time().',0');
        }

        //读取这个用户还没有付款的记录
        $OrderList = dbDuoSelect('order as o','details as t',' on o.tid=t.id',null,null,$select,'o.uid='.$_COOKIE['uid'].' and o.ispay=0','o.oid asc');
        $allpay = dbFuncSelect('order','sum(rate ) as zpay','uid='.$_COOKIE['uid'].' and ispay=0');

    }

    //检查当前浏览用户是否已付费
    $MyOrder = dbSelect('order','*','uid='.$_COOKIE['uid'].' and ispay=1 and tid='.$Id.','oid asc',1);

    if($GuanLi){
        //删除, 放入回收站
        if(!empty($_GET['del'])){

            $result = dbUpdate('details', "isdel=1", 'id='.$Id.'');
            header('location:index.php');

        }
        //置顶
        if(!empty($_GET['istop'])){
            $result = dbUpdate('details', "istop=1", 'id='.$Id.'');
            header('location:detail.php?id='.$Id);
        }
        //高亮
        if(!empty($_GET['style'])){
            $result = dbUpdate('details', "style='red'", 'id='.$Id.'');
            header('location:detail.php?id='.$Id);
        }
        //精华
        if(!empty($_GET['elite'])){
            $result = dbUpdate('details', "elite=1", 'id='.$Id.'');
            header('location:detail.php?id='.$Id);
        }
        //删除回帖, 放入回收站
        if(!empty($_GET['delht'])){

            $result = dbUpdate('details', "isdel=1", 'id='.$_GET['hid'].'');
            header('location:detail.php?id='.$Id);

        }
        //回帖置顶
        if(!empty($_GET['istopht'])){
            $result = dbUpdate('details', "istop=1", 'id='.$_GET['hid'].'');
            header('location:detail.php?id='.$Id);
        }
        //回帖屏蔽
        if(!empty($_GET['isdislpay'])){
            $result = dbUpdate('details', "isdisplay=1", 'id='.$_GET['hid'].'');
            header('location:detail.php?id='.$Id);
        }
    }

    include template("detail.html");

```

## 16.6.7 项目安装模块讲解

### 安装功能说明

我们在电脑上安装软件的时候，点击下一步调整一些配置选项软件就安装成功了。

不需要非常高深的计算机技术即可让普通人完成安装。

我们在真正的商业型产品中也需要做完整的安装过程：

- 1. 极为方便普通用户
- 2. 看起来特别专业
- 3. 操作起来简单提高效率

我们的PHP的论坛、商城、OA、微信平台都可以做这样的安装包，普通用户下载我们的安装包后。几步就可以开始使用我们的服务。

这样对用户和对开发人员都好。

### 安装相关基本概念

概念	说明
安装目录	通常在install目录下
安装锁	通常为install.lock，有这个文件就是安装过，无这个文件就是未安装
PHP扩展模块判断	mysqli模块没有就无法操作数据库，我们则拒绝用户继续安装
目录权限判断	我们会写入install.lock或者一些临时文件，若某些目录没有写入修改权限则无法安装成功
版本权限判断	如果是php4.x版本想安装我们的应用，我们会提醒用户PHP版本太低无法安装
超级管理员	初使的最高管理员，可以管理后台，方便用户管理，安装时即设置
初始数据库结构	初始的数据库我们的是将开发的数据库清空后，打开.sql文件，分行执行，向数据库写入创建表的语句和初始信息

注：所有的代码文件实例请下载我们的安装包。查看install目录。

### 安装核心步骤说明

- 1. 打开网站。打开网站若不存在install.lock文件则识为未安装
- 2. 存在安装锁文件install.lock则禁止用户执行安装
- 3. 展示安装协议
- 4. 检测操作系统和PHP版本，是否版本准许。若版本不准许则不展示下一步。
- 5. 判断权限是否具备相关目录的写入权限和PHP的图像、数据库模块权限，若不具备相关权限则不显示下一步。
- 6. 输入服务器地址、数据库信息等配置文件，按照输入的信息连接数据库
- 7. 输入管理员信息
- 8. 将基本信息准备好，导入指定数据库数据表内容
- 9. 导入成功，创建一个空文件install.lock

### install.lock文件判断

本文档使用 [看云](#) 构建

在/common/common.php文件中有这么一段：

```
//项目安装
if(!file_exists('install.lock'))
{
    header('location:install/index.php');
    exit;
}
```

若文件不存在则跳转至header目录。停止继续向下执行

/install/top.php文件中有这一段：

```
if(file_exists('../install.lock')){
    header('content-type:text/html; charset=utf-8;');
    exit('网站已经被安装过了，如果需要重新安装网站，请删除 /install.lock 文件');
}
```

若存在install.lock文件禁止执行安装步骤。

## 判断版本

获得操作系统版本

```
function userOS(){

    //$user_OSagent = $_SERVER['HTTP_USER_AGENT'];
    $user_OSagent = PHP_OS;

    if($user_OSagent)
    {
        $visitor_os = $user_OSagent;

    } else {

        $visitor_os = '其它';

    }

    return $visitor_os;

}
```

获得PHP的版本号：

```
echo PHP_VERSION
```

## 判断目录权限

判断目录写入权限的自定义函数：

```
function iswriteable($file){
    if(is_dir($file)){
        $dir=$file;
        if($fp = fopen("$dir/test.txt", 'w')) {
            fclose($fp);
        }
    }
}
```

```

        unlink("$dir/test.txt");
        $writeable = 1;
    }else{
        $writeable = 0;
    }
}else{
    if($fp = fopen($file, 'a+')) {
        fclose($fp);
        $writeable = 1;
    }else {
        $writeable = 0;
    }
}
return $writeable;
}

```

判断模块权限：

```
function_exists('mysqli_connect')
```

如果存在相关函数，则存在相关模块。

## 修改配置文件

相关代码参考本书：《8.11 修改置文件的实验》

## 数据库导入代码

将创建库的SQL语句准备好，创建数据库发送创建数据库的相关命令即可。

```

//执行数据库导入
include '../config/database.php';

//新建数据库
$link = mysqli_connect(DB_HOST, DB_USER, DB_PASS);
if(mysqli_get_server_info($link) > '4.1') {
    mysqli_query($link, "CREATE DATABASE IF NOT EXISTS `".DB_NAME."` DEFAULT CHARACTER SET ".DB_CHARSET);
} else {
    mysqli_query($link, "CREATE DATABASE IF NOT EXISTS `".DB_NAME."`");
}
if(mysqli_connect_errno($link)){
    exit('数据库不存在');
}
mysqli_close($link);

```

导入数据库打开apple\_bbs.sql准备好的SQL文件，这个SQL文件中每一行的行尾以;NoAlike结尾。

我们使用explode将sql文件切割成一个数组，循环数组的每一行完成数据的导入。

```

$sql=file_get_contents('apple_bbs.sql');
$conn=mysqli_connect(DB_HOST, DB_USER, DB_PASS, DB_NAME);
if(mysqli_errno($conn)){

    exit(mysqli_error($conn));
}
mysqli_set_charset($conn, DB_CHARSET);

$arr=explode(';NoAlike;',$sql);

foreach($arr as $val){
    if(!empty($val))

```

```
{
    $Nval = str_replace('bbs_', DB_PREFIX, $val);
    $result = mysqli_query($conn, $Nval);

    if($result){
        $sql = '<font color="green">数据库导入成功</font>';
    }else{
        $sql = '<font color="red">数据库导入失败</font>';
    }
}

mysqli_close($conn);
```

注：所有的代码文件实例请下载我们的安装包。查看install目录。

# 附录1. 版权声明

---

## 版权声明

本书的版权归本书的贡献者所有。

您可以随意转载和传播，但必须注明来源。来源请声明：《7天学会PHP》。

我们保留最终追究的权力。

## 免责声明

由于我们是一本开源开放，共同参与写作的书，我们没法做到100%的审核。书中的部份代码和说明，部份贡献者（开发者）如果侵犯了您的权利。

请邮件给我们：kaiwenli@phpxy.com

我们将在24小时进行删除

## 本书贡献者

主要作者：李文凯

参与作者：聂武 宋艳辉 程龙 李旭光 安正超 荣友元 唐如程

欢迎加入我们共同创作本书、提出意见，让更多的初学者受益，让大学和工作迷茫的人，不再迷茫！

## 附录2 . 学习PHP常用的英文单词

---

Hypertext  
Preprocessor  
Java  
html  
script  
API  
office  
word  
html  
web  
server  
windows  
back  
next  
cancel  
folder  
choose  
setup  
install  
administrator/root/admin  
finish  
stop  
start  
config  
log/logs  
help  
quit  
module  
service  
port  
Explorer  
linux  
mac  
os  
studio  
zend studio  
eclipse  
notepad  
note  
pad

vim  
gvim  
down  
download  
code  
info  
phpinfo  
dollar  
var/variable  
echo  
int  
integer  
bool  
boolearn  
string  
title  
float  
double  
if  
else  
null  
result  
dump  
set  
unset  
object  
array  
resource  
call  
back  
callback  
type  
is  
get  
numeric  
mixed  
auto  
check  
define  
line  
method  
class  
version  
dir



name  
space  
include  
user  
my  
test  
demo  
password  
text  
get  
post  
submit  
value  
input  
body  
address  
file  
request  
fire  
fox  
bug  
action  
software  
content  
home  
role  
length  
protocol  
interface  
status  
time  
connection  
remote  
switch  
case  
default  
break  
date  
while  
go  
to  
goto  
count  
table

continue  
declare  
function  
plus  
cookie  
session  
static  
match  
max  
min  
rand  
year  
unix  
timezone  
seconds  
minutes  
hours  
day  
weekday  
month  
mirco  
first  
end  
tags  
replace  
encoding  
pop  
push  
list  
each  
key  
prev  
reset  
current  
sort  
regex  
read  
create  
write  
move  
copy  
data  
exists  
clear

cache  
able  
lock  
seek  
close  
group  
own  
owner  
path  
base  
build  
parse  
discuz  
upload  
size  
limit  
memory  
enabled  
disabled  
progress  
temp  
done  
error  
field  
style  
png  
jpeg/jpg  
gif  
header  
width  
height  
ascii  
display  
report  
level  
notice  
warning  
all  
core  
STRICT  
DEPRECATED  
trigger  
mysql  
command

monitor  
or  
oracle  
Copyright  
engine  
index  
charset  
execute  
fetch  
row  
assoc  
db  
database  
edit  
delete  
update  
alter  
modify  
change  
add  
unsigned  
ZEROFILL  
enum  
stamp  
union  
order  
goods  
left  
right  
join  
from  
inner  
outer  
shop  
cms  
system  
manger  
money  
access  
agent  
token  
thread  
thread-safe  
throw

video