PHP 基础教程

简明教程

目录

序

编写一本有关 PHP 的简明教程，对 PHP 选修课程而言，是很必要的。

PHP 语言是开放源代码语言，由 PHP 开发小组及全世界的 PHP 爱好者时刻进行着维护和更新，不断增强其功能，所以在网络上不断地会涌现大量的 PHP 的电子参考手册，在书店和图书馆里，有关 PHP 的图书也是汗牛充栋，日新月异，这些参考手册和图书，其内容越来越丰富，在描述上不可谓不详尽，但是这些“详尽”的细节描写，对初学者而言，都显得“大而全”，看起来犹如走进了迷宫，总感觉不甚明了。笔者在教学实践中，曾拿它们直接作为教材使用，效果不是很好，感觉它们不便于直接作为教材使用，尤其是对课时 40 左右的 PHP 选修课程而言。

本讲义是作者根据自己在实际 Web 软件开发工作中，对使用 PHP 进行 Web 软件开发，及实际教学经验的一个简明总结，结合相关参考文献，整理，编写而成，力求简明扼要，以适合选修课教学实际的需要，所以，本讲义只包含了初学者入门所需的必要的知识和实践内容。

由于本讲义描述中，含有较多的个人见解和体会，不足之处，请各位专家、老师和读者不吝指正。

参考文献：

①PHP 手册(官方最新版：http://www.php.net/download-docs.php)：由 PHP Documentation Group 编写，全面、权威，不断更新，适合高级编程人员参考。

②PHP 手册，台湾星空浪子翻译版：主要针对 PHP4 而编写，内容简易，适合初学者使用，只是有些台湾用语。

③PHP 程序设计，Rasmus Lerdorf，中国电力出版社，2003：PHP 创始人、PHP 开发小组领军人物 Rasmus Lerdorf 的经典著作，本书是一本全面、详尽、权威的国外经典译著。

④Web 数据库基础教程，魏善沛编著，中国铁道出版社，2003。

⑤PHP 经典实例，Sterling Hughes 等，中国电力出版社，2003：由 PHP 开发小组核心人物 Sterling Hughes 等著，本书也是国外经典译著。

⑥PHP 高级开发技术与应用，曹轶群等， 清华大学出版，2002

1.课程的性质和任务
课程名称：PHP 程序设计语言，英文名称：The PHP Programming language
课程编号：101122120(Web 开发技术)
课程性质：专业选修课、专业技术课
课程任务：是对高年级大学生进行专业技术教育
2.课程的学习目标
对 Web 软件的开发：理解基本原理，树立正确理念，掌握基本技能
3.预备知识：计算机应用基础、C 语言程序设计
4.如何学习，如何考核
化繁为简,精讲精学，循序渐进,登堂入门
实验为主,注重实践，网上自助学习
开卷考试，独立完成
5.学习进度安排
周学时：6，总学时：42，学分：4

学习进度表

（按 1 班上课顺序排布，2 班与此内容同，顺序根据场地作相应调整）

6.重点和难点
Web 开发（Web 程序工作原理，相关概念）
PHP 开发环境搭建(Apache 的配置)
PHP 基本语法
PHP 操纵 HTML、JavaScript（嵌入、互相嵌入）
PHP 操纵数据库（相应的函数）
基于 PHP 的 Web 应用系统设计（规划和分析）
7.习题、作业、课堂讨论：均围绕实验进行
8.学习参考材料
(1)入门学习
本教程

(2)高级参考
Rasmus Lerdorf, PHP 程序设计,中国电力出版社,2003,定价:68 元(国外经典)
Sterling Hughes 等, PHP 经典实例, 中国电力出版社,2003,定价:39 元(国外经典)
曹轶群等,PHP 高级开发技术与应用,2002,清华大学出版社,定价:32 元

第一章 PHP 简介

第一节 PHP 简介

1.Web 程序工作原理

（1）Web 一词的含义

network：【计算机】电脑网络，网

Web：【计算机】万维网(World Wide Web)，互联网(Internet)

Web 程序，顾名思义，即可工作在 Web 上的程序。实际上，它也可工作于企业内网(内联网：Intranet)、企业间网(外联网：Extranet)，只不过它在 Web 上更具应用优势，更为常见，故人们称它为 Web 程序。

（2）单机程序工作原理

单机，即不连接到其他计算机的计算机，不在网络中。两单机 A、B，只在 A 上安装有程序 X，若要在 B 上得到 X 的运行结果，必须在 B 上安装一遍 C，然后运行之，若 B 类的计算机比较多，则需要逐一安装运行，非常麻烦；它们之间不能直接进行通信和协作。如图 1 所示。

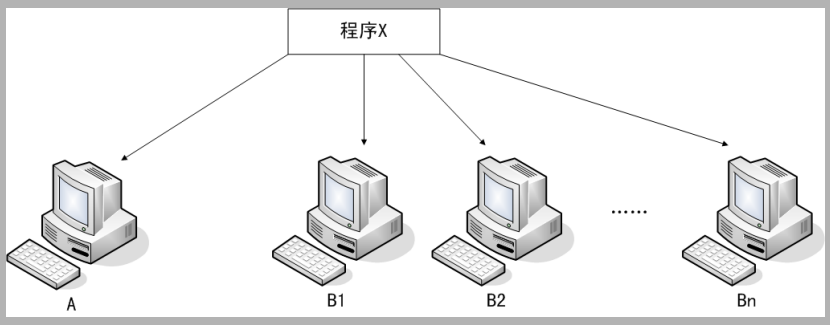


图 1 单机程序工作示意图

（3）客户机/服务器程序工作原理

将单机连成网络，如将 A 与 B 连成网络，则可以使它们之间提供服务，如 A 向 B 提供服务。常见的服务是文件共享，FTP 文件下载等。我们把提供（响应）服务的计算机称作服务器(Server)，接受（请求）服务的计算机称作客户机(Client)，也叫工作站(Workstation)。服务器一般用性能较高的计算机担当。客户机/服务器程序的工作原理如图 2 所示。

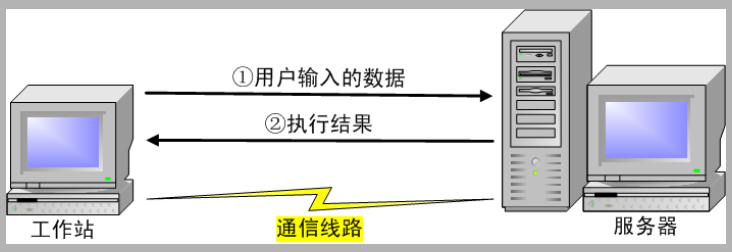


图 2 客户机/服务器程序的工作原理

服务器和客户机的角色可以转换。一台计算机，可以对自己提供服务，这时，它既是服务器，又是客户机。如，计算机 A 把自己的文件夹 a 共享，然后在网络上找到 A，则可以下载 a，即自己对自己提供了服务，自己请求并响应了服务。

客户机/服务器的这种计算机间的协作方式，称作 C/S 方式，或 C/S 架构。

C/S 程序分为两部分：服务器端部分和客户机端（以后简称客户端）部分，分别称为服务器端程序（或服务程序）和客户端程序（或客户程序）。对于客户端程序，对每一个客户机，也都需要分别安装，这一点与单机程序的分发相同，也很麻烦。但是，安装好了客户端程序后，就可以通过通信线路与服务器交互，或通过服务器，与其他客户机通信。典型的例子是大家常用的聊天程序 QQ，如图 3 所示。

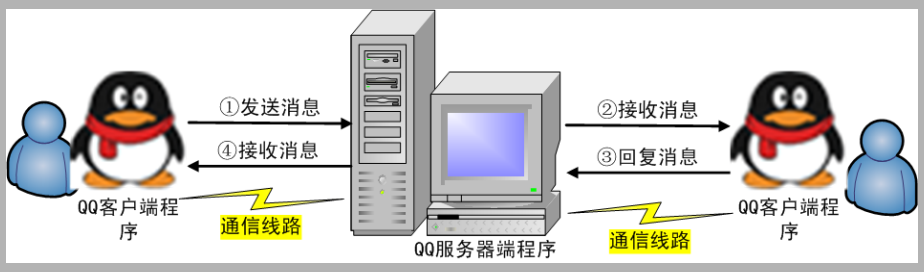


图 3 C/S 程序 QQ 的工作原理

（4）浏览器/服务器程序工作原理

若通过客户机中的浏览器（Browser），向服务器发出请求，接收其响应的结果，那么，这时，我们称这样的协作方式为 B/S 方式，或 B/S 架构，其工作原理如图 3 所示：

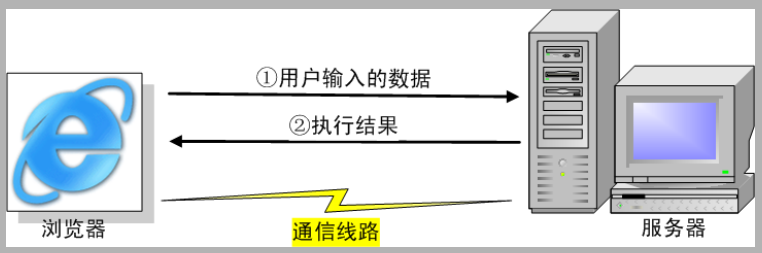


图 4 浏览器/服务器程序的工作原理

这时，客户端程序就是浏览器，而浏览器的安装是随着操作系统的安装完成的，不需要用户额外安装。大多数情况下，大多数人用的操作系统是 Windows 操作系统，对他们而言，使用 B/S 程序（如上网看新闻，收发电子邮件），可不需要安装专门的客户端程序，直接在浏览器中操作即可。这使得 B/S 程序的维护十分方便，因为不用管客户端程序，只要维护好服务器端程序即可。

人们说当今是一个网络时代，实际上着重指得是互联网时代，也就是 Web 的时代。人们使用这种 B/S 程序比较多，是造成这种叫法的一个重要缘故。

本课程中的 Web 程序，就是指这种 B/S 程序。

（5）C/S、B/S 中服务器的组成

服务器，是担负服务任务的机器。这些服务任务一般专门的软件来完成。一般地，把具有某种服务功能的服务器软件及其所在的机器，都统称 XX 服务器（XX 表示某种具体服务）。这些软件可以集中于一台机器中（如图 5），这样的机器可以称为集中式服务器；也可以单独存在于某台机器中（如图 6），这样的机器可以称为独立式服务器，多个独立式服务器可组成服务器群或矩阵(台湾词为“阵列”)。

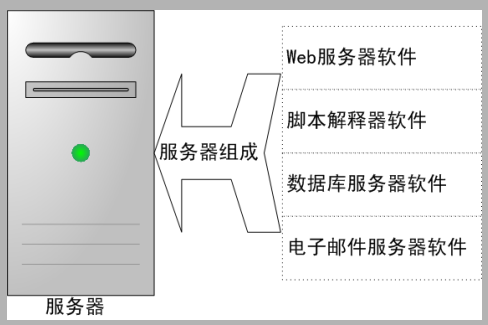


图 5 集中式服务器

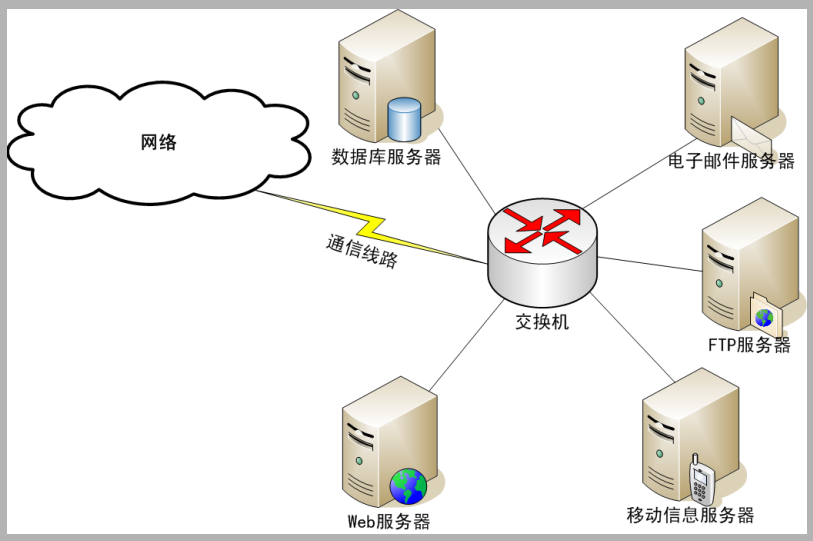


图 6 由独立服务器组成的服务器群

按照服务任务，常见的几种服务器软件如表 1 所示。

表 1 几种常见服务器软件

服务器软件类别	服务器软件举例	功能
Web 服务器软件	Apache 、 IIS 、 PWS 等	接收来自浏览器的任务请求，分派任务给其他服务器软件，接收其他服务器软件对任务的处理的结果，将该结果返回给浏览器
服务器端脚本解释软件(一般与 Web 服务器软件同处于同一台机器上)	PHP、ASP 等	接收来自 Web 服务器软件分派给自己的服务器端脚本执行任务；进行脚本的语法分析，若语法有错误，则向 Web 服务器返回出错信息，否则，执行脚本，将解析结果/执行结果返回给 Web 服务器软件
数据库服务器软件	MySQL、Oracle、MS SQL Server 等	接收来自其他服务器软件的数据处理任务请求，执行该任务，将执行结果返回给请求者
电子邮件服务器软件	MS Exchange、Sendmail 等	接收来自其他服务器软件的邮件处理任务请求，执行该任务，将执行结果返回给请求者

服务器端脚本：用服务器端编程语言编写的程序。

服务器端编程语言：只运行在服务器端，被服务器所解释和执行的编程语言，如 PHP 语言。

（6）B/S 程序工作的具体过程

说明：在以后的 B/S 程序图示中，通信线路不再特别表示。

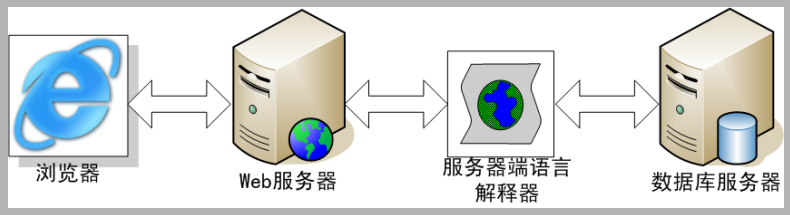


图 7 B/S 程序工作过程示意图

（7）PHP 程序工作的具体过程

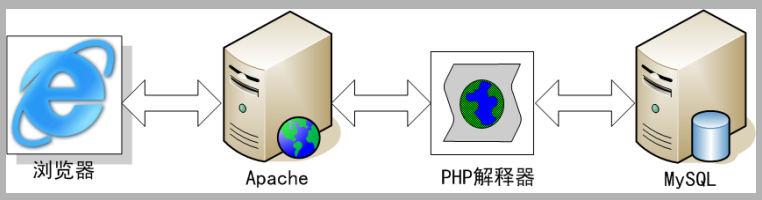


图 8 PHP 程序工作过程示意图

具体过程是 Apache、PHP、浏览器之间的协作过程：

用户通过浏览器向服务器请求 PHP 文件（如在地址栏输入：http://211.64.40.187/index.php），Apache 在服务器上的用户文档发布目录下查找浏览器所请求的 PHP 文件，若找不到，则向浏览器返回错误信息，否则，将其提交给 PHP 解释器解释执行，PHP 解释器对该 PHP 文件进行语法分析，若发现语法错误，则经 Apache 返回

错误信息(以浏览器能识别的代码表示)到浏览器，否则执行该 PHP 程序(可能包含对数据库 MySQL 的操作)，将 PHP 程序执行结果(以浏览器能识别的代码表示)经 Apache 返回到浏览器，浏览器对返回的结果进行解释、执行，执行的结果显示在浏览器窗口中。
如果浏览器所请求的文件不是 PHP 文件，而是 HTML 文件（.htm 文件）或 JavaScript 文件（.js 文件），该过程将简化：Apache 在服务器上的用户文档发布目录下查找浏览器所请求的 HTML 文件或 JavaScript 文件，若找不到，则向浏览器返回错误信息，否则，将其返回给浏览器，浏览器对返回的结果进行解释、执行，执行的结果显示在浏览器窗口中。

2.PHP 简介

PHP，即“PHP: Hypertext Preprocessor”，是一种被广泛使用的开放源代码多用途脚本语言，尤其适用于 Web 开发并可以嵌入到 HTML 中去。其语法利用了 C，Java 和 Perl，非常容易学习。该语言的主要目标是让 Web 开发人员可以很快写出动态生成的网页，但 PHP 的功能远不止如此。
一个简单的 PHP 程序：

```
<?
echo “hello,world”;
?>
```

该程序运行的结果是在网页上输出了一个内容为 hello,world 的字符串。
PHP 是一种语法简单、功能强大的网络编程语言。在语法格式上，PHP 借鉴了广泛流行的 C、Java 和 Perl 等编程语言的特点，非常类似于 C 语言，但比 C 语言更简单，易学和易用，因此特别适合于学习过 C 语言，有志于网络编程的人学习和使用。
在功能上，它提供了丰富的函数库，支持复杂字符串处理，正规表达式和图形处理，可根据用户请求将服务器的数据轻松地产生动态网页，支持目前几乎所有流行的数据库系统，如 Oracle、SQL Server、MySQL、mSQL、Informix、SyBase、ODBC、PostgreSQL 以及 dBase 等。在在可移植性上，PHP 程序可在 Windows 下快速开发，代码甚至不加修改即可运行在 Unix、Linux 上。
本课程中的 PHP 程序，就是指使用这种 PHP 语言编写的程序。它只能被服务器所解释执行，只能运行于服务器端，用户在浏览器中看到的，只是经服务器解释后，返回给浏览器的 HTML 等客户端代码（可从浏览器的“查看->源文件”菜单中看到），这些代码是由浏览器解释执行的，执行的结果，显示在浏览器窗口中，形成网页。所以，在客户端，看不到 PHP 代码，这也是 PHP 语言写成的代码比较安全的一个原因。

第二节 PHP 的开发环境的搭建

从图 8 中可以看出，要使 PHP 程序能够正常工作，必须：
（1） 选用合适的程序编辑软件，以便进行程序的编写；
（2） 为其在服务器上配置好一个运行环境：安装和配置好以下软件，使它们能够正常协作：Apache、PHP、MySQL，以便进行程序的测试和运行。
以上两点构成一个完整的 PHP 开发、测试和运行环境。其中，（1）的 3 个软件部署在服务器上，（2）的软件部署在客户机上。为方便大家学习和理解整个环境的搭建过程，本课程的实验条件是人手一机，每台机器既作服务器又作客户机，整个环境的搭建工作可在一台机器上完成。
以上两点构成一个完整的 PHP 开发、测试和运行环境，其根本目的是为了学习使用 PHP 语言进行 Web 程序编写和软件开发，所以，我们把这个环境的搭建工作称之为 PHP 开发环境的搭建。

1.程序编写，软件开发工具软件的选用

（1）有关工具软件
这些工具软件用于编写，修改源程序文件。我们的源程序文件都是 ASCII 码文件（纯文本文件），所以，只要

是能够处理纯文本的工具软件，都可选用。
常用的编辑软件有：记事本。为了提高效率，常使用一些专业工具软件，如 UtraEdit，DreamWeaver，不建议使用 FrontPage（它对 PHP 的支持不好）。

- 记事本：常用于简单的代码编辑。
- UltraEdit10 简体中文版：短小精悍，代码编辑功能强大，编程高手的利器
- Dreamweaver MX 中文版：图形化编辑环境，速度慢，功能多，常用于复杂网页设计和网站管理。注意该软件提供了“代码”、“代码/设计”、“设计”三种视图：“代码”视图以全部是源代码的方式供用户编辑使用，常用于代码（服务器端代码或客户端代码）的编写，修改；“设计”视图以可视化的方式供用户使用，常用于设计网页界面，以减少手工撰写客户端代码的工作量；“代码/设计”视图是上述两种方式的结合。

（2）源文件扩展名
无论用哪种工具软件编写源程序文件，若文件中含有 PHP 代码，必须确保文件的扩展名，与在 Apache 配置中的 PHP 文件的扩展名的设定一致；若不含 PHP 代码：①若仅含有 HTML 代码，一般以.htm 为扩展名；②若仅含有 JavaScript 代码，这种文件常作为.htm 文件的包含文件（引用文件，类似于 C 程序文件中的头文件）一般以.js 为扩展名；③若含 HTML 代码和 JavaScript 代码，一般以.htm 为扩展名。
（3）开发工具软件的使用
这些开发工具软件，简单易用，用户稍加学习即可掌握。对于 Dreamweaver，操作类似 WORD，另外，它自带学习教程，可以参考。

2.服务器软件的安装和配置

对于 Windows 下多数软件的安装，我们是通过执行该软件的安装程序（setup.exe、install.exe、软件文件名.exe），由安装程序负责具体的安装过程。这个过程中，安装程序做的工作一是系统注册：把该软件的一些系统文件，复制或移动到操作系统的有关系统目录中，实现与操作系统的接口，便于操作系统使用；二是将自身的程序文件、数据文件等复制到本软件的安装目录下，用于本软件的运行。
（1）PHP 语言解释器软件的安装和配置
该软件没有安装程序，所以具体的安装过程需要用户来完成。不过过程也十分简单：解压缩该软件包至安装目的目录，将系统文件 php4ts.dll（PHP 系统扩展函数库）和 php.ini（PHP 配置文件）分别放到操作系统目录下的相应的位置即可。
（2）Apache 的安装和配置
大体经过安装->配置两大步骤完成。配置大体经过编辑配置文件->启动服务完成。

- 安装：其安装过程由其安装程序进行，启动安装程序后，按提示操作即可。
- 配置目的：向 Apache 说明清楚 PHP 的有关情况，以便实现二者的协作。
- 配置方法：主要通过一个配置文件中有关参数的修改或添加进行。所谓“修改”，即对该文件中已经存在的参数，修改其参数值；所谓“添加”，即将该文件中缺少的参数和参数值增加到该文件中相应位置，为了省事，也可以放到该文件的最后。该文件名为 httpd.conf，存在于 Apache 安装目录下的 conf 目录下，是一个纯文本文件，可用记事本打开修改。也可以通过开始菜单中 Apache 程序组中的“Edit Configuration”打开该文件，进行配置操作。注意每次配置变动后一定要保存，并启动或重新启动 Apache 的服务，所做的最新配置才会起作用。
- 有关参数和参数值的说明

  1）格式多是这样的单独的行：

  #参数名　参数值

  行首的#表示注释，应去掉才会使配置起作用。

  2）参数和参数值的含义、作用

表 2　Apache 配置参数设定

示例路径：PHP 安装路径——d:/php；发布文档（即文件）主目录——d:/www

	参数名和参数值	操作	作用
①	BindAddress 服务器机器 IP	修改	进行地址绑定(指定服务器地址)
②	LoadModule php4_module d:/php/sapi/php4apache.dll	添加	指名将 PHP 配置为 Apache 的模块（Apache module）方

			式进行工作时的 PHP 语言解释器
③	Port 80	修改	指定 Apache 对外提供 Web 服务的通信端口
④	ServerAdmin 服务器管理员邮箱(如 abc@abc.com)	修改	当发生错误时送回客户端浏览器的管理员信箱
⑤	ServerName 服务器的计算机名称	修改	指明主机名称
⑥	DocumentRoot "d:\www"	修改	指明向客户端提供 Web 服务的发布文档主目录
⑦	<Directory "d:\www">	修改	发布文档主目录定义,该处目录的值应与⑥中的一致
⑧	ScriptAlias /php/ "d:/php/" AddType application/x-httpd-php .php Action application/x-httpd-php "/php/php.exe"	添加	指明 PHP 脚本语言名称和 PHP 脚本语言解释器的路径 指明 PHP 脚本扩展名 指明 PHP 脚本解释器(PHP 以非模块工作时起作用)
⑨	DirectoryIndex index.php	修改	指定默认文档(主页文档,只请求服务器地址就响应的文档)

- 有关服务的操作：服务，是一直运行，监听来自其他程序（客户端）的请求，接收请求，处理请求（自己处理或提交其他程序处理），返回处理结果的一种运行着的程序，一种进程。一般运行于操作系统后台。有关 Apache 的服务操作，通过开始菜单中 Apache 程序组中的有关命令进行：

★ Install Service：在操作系统中注册服务
★ Uninstall Service：在操作系统中删除服务
★ Start Service：启动服务
★ Restart Service：重新启动服务
★ Stop Service：停止服务

注意：

服务的启动、重新启动、停止，必须以在操作系统中注册了服务为前提；
服务的启动、重新启动、停止过程中有 Apache 的消息反馈。

（3）Apache 与 PHP 的协同测试

测试目的是检查二者是否能够正常协作。具体来说，是检验配置后的 Apache、PHP、浏览器是否具备了这样的协作能力：

用户通过浏览器向服务器请求 PHP 文件（如在地址栏输入：http://211.64.40.187/index.php），Apache 在服务器上的用户文档发布目录下查找浏览器所请求的 PHP 文件，若找不到，则向浏览器返回错误信息，否则，将其提交给 PHP 解释器解释执行，PHP 解释器对该 PHP 文件进行语法分析，若发现语法错误，则经 Apache 返回错误信息(以浏览器能识别的代码表示)到浏览器，否则执行该 PHP 程序(可能包含对数据库 MySQL 的操作)，将 PHP 程序执行结果(以浏览器能识别的代码表示)经 Apache 返回到浏览器，浏览器对返回的结果进行解释、执行，执行的结果显示在浏览器窗口中。

做法是，在服务器上用户的发布文档目录下，放置一个含有 PHP 代码的文件，即 PHP 程序，通过客户端浏览器，向服务器请求这个文件，若浏览器能得到正确的来自服务器的结果，则表明二者安装和配置成功，否则，应根据出错信息，修改配置。

第三节 实验一 PHP 开发环境的搭建

本实验对 Windows95 及其以上版本的 Windows 操作系统机器通用。

0.实验目的

（1）能够快速部署 Windows 下的开发环境，满足学习，使用 PHP 对于开发和服务环境的需要。
（2）加深对 B/S 程序工作原理的理解

1.实验准备

（0）知识准备：Web 程序工作原理（本章第一节）
（1）PHP 服务器环境需要以下 3 个服务器端系统软件
- Web 服务器软件：Apache 1.3.14

- PHP 语言解释器软件：PHP 4.0.4
- 数据库服务器软件：MySQL 3.23.43

（2）开发工具软件：

记事本：Windows 自带。

UltraEdit10 简体中文版（压缩包文件：uedit10.zip）

Macromedia Dreamweaver MX 中文版：已安装好

注： PHP 相关软件、开发工具的获得：在我的网站上的有关网页上下载。

我的网站：http://www.sunshoulong.cn 或 http://211.64.32.2/dsks

（3）操作系统软件：服务器和客户机上均为 Windows

具体到德州学院计算机系的机房环境，请在 Windows 2000 Professional 环境下做

（Win98 下已经搭建好，是为以后章节的学习使用的，无实验意义）

（4）实验中的路径说明

为说明问题的简单起见，路径为比较简单的示例路径，但已经过测试。实际运用时，路径完全可根据自己需要设定。

2.PHP 的安装和配置

（1）安装

①将 PHP-4_0_4-Win32.rar 解压缩到 d:\php 下

②将 d:\php 下的 php4ts.dll 移动到 c:\操作系统安装目录\system 下

③将 d:\php 下的 php.ini-dist 复制到 c:\操作系统安装目录 下，更名为 php.ini

注意：操作系统安装目录，具体位置见表 3；php.ini 是 PHP 的配置文件。

表 3：操作系统安装目录

操作系统	操作系统安装目录
Windows95	windows
Windows98	windows
Windows me	windows
Windows XP	windows
Windows NT 系列	winnt
Windows 2000 系列	winnt
Windows 2003	windows

（2）配置：通过修改 php.ini 中的参数来实现。对 MySQL 而言，若无特殊要求，一般无须配置。因为 PHP 在 php.ini 中已经做好了对 MySQL 的配置，所以一般无须修改。

3.Apache 的安装和配置

（1）安装软件：双击 Apache_1_3_14_win32.exe，按照提示，安装到 d:\apache 下，即完成安装。

（2）配置服务：单击[开始]->[程序]->[Apache Web Server]->[Management]->[Edit configuration]，打开 Apache 的配置文件 httpd.conf，按表 4 提示进行配置，完毕后，保存。

注意：

- 所谓“修改”，即对该文件中已经存在的参数，修改其参数值；所谓“添加”，即将该文件中缺少的参数和参数值增加到该文件中相应位置，为了省事，也可以放到该文件的最后
- 每处的配置要想起作用，必须将行首的#号（注释符号）去掉
- 描述以 httpd.conf 文件的行文顺序进行
- 表 4 中路径、地址、主机名、信箱等，实际配置时请根据自己情况进行
- 每次配置变动后一定要保存，并启动或重新启动 Apache 的服务，所做的最新配置才会起作用

表 4 Apache 配置参数设定

示例路径：PHP 安装路径——d:/php；发布文档（即文件）主目录——d:/www

序号	参数名和参数值	操作	作用
①	BindAddress 服务器机器 IP 或 localhost(127.0.0.1)	修改	进行地址绑定(指定服务器地址)
②	LoadModule php4_module d:/php/sapi/php4apache.dll	添加	指明将 PHP 配置为 Apache 的模块（Apache module）方式进行工作时的 PHP 语言解释器
③	Port 80	修改	指定 Apache 对外提供 Web 服务的通信端口
④	ServerAdmin 服务器管理员邮箱(如 abc@abc.com)	修改	当发生错误时送回客户端浏览器的管理员信箱
⑤	ServerName 服务器的计算机名称	修改	指明主机名称
⑥	DocumentRoot "d:/www"	修改	指明向客户端提供 Web 服务的发布文档主目录
⑦	<Directory "d:/www">	修改	发布文档主目录定义,该处目录的值应与⑥中的一致
⑧	ScriptAlias /php/ "d:/php/" AddType application/x-httpd-php .php Action application/x-httpd-php "/php/php.exe"	添加	指明 PHP 脚本语言名称和 PHP 脚本语言解释器的路径 指明 PHP 脚本扩展名 指明 PHP 脚本解释器(PHP 以非模块工作时起作用)
⑨	DirectoryIndex index.php	修改	指定默认文档(主页文档,只请求服务器地址就响应的文档)

说明：

①处：localhost 指服务器机器本机。如果机器没有插网线，请选择 localhost，否则有可能会提示说找不到地址。

⑦处上面有说明：This should be changed to whatever you set DocumentRoot to

教学网站上有 Apache 配置文件样本可供参考。

（3）Web 服务的安装和启动

• 安装 Apache 服务

将 Apache 在 Windows 操作系统服务中注册。单击：开始→程序→Apache Web Server→Apache as a service→Install service。服务只需要注册一次。

• 启动 Apache 服务

单击：开始→程序→Apache Web Server→Apache as a service→ Start Service，启动服务。还可以停止，重新启动服务。

说明：

在 Windows2000 Professional 或 Windows XP 中，也可通过开始→控制面板→管理工具→服务，根据需要，对 Apache 服务进行启动、停止、重新启动等操作。

4.测试 Apache 对 PHP 的支持

（1）测试目标：检查二者是否能够正常协作

（2）测试方法

①用记事本或 uedit 编写测试脚本，存为 d:\www\index.php，内容为：

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

注意：index.php 被配置成了 Apache 的默认文档

phpinfo()是 PHP 内置函数，用来显示 PHP 和 Apache 配置信息。

②在浏览器中敲入 http://你的机器的 IP 地址或 localhost(127.0.0.1)，回车后若显示类似图 9 的 PHP 配置页面（以 Win98 下为例），则说明你的配置达到了上述的测试目标，Apache 与 PHP 能够正常协作；若不显示类似画面，则配置有误，此时你的 Apache 不能够识别 PHP 脚本，需更改配置。

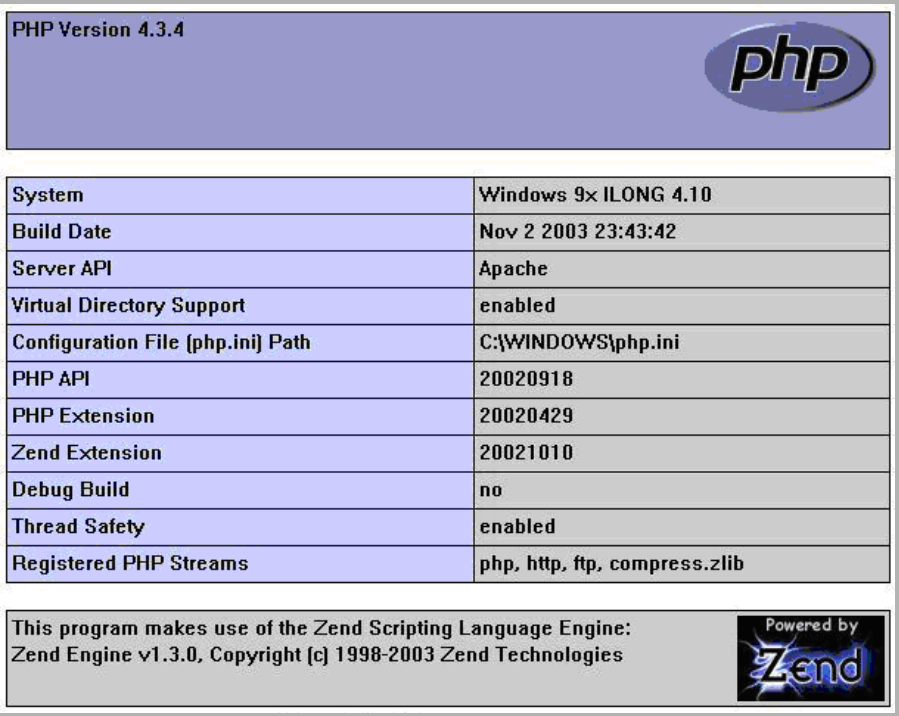


图 9 Apache 配置成功（图中 PHP 以 4.3.4 版本为例）

5.MySQL 软件的安装和服务的启动

（1）安装

将 mysql-3.23.43-win.zip 解压缩后，双击 setup.exe，按照提示，进行安装。

根据指定安装路径的不同，安装可采取的方式有：

①默认安装：安装过程中不指定安装路径，采用默的 c:\mysql。采用该种方式安装，一路按“下一步”按钮即可完成。

②定制安装：安装过程中指定自己的安装路径（如：d:/mysql），而不是采用默的 c:\mysql。采用该种方式安装完成后，需要做一个配置文件，将 MySQL 的有关信息向操作系统特别声明一下。具体方法是：用记事本或 uedit 创建一个文件 my.ini 保存在你的操作系统安装目录下，该文件内容如下

```
[mysqld]
basedir=d:/mysql/
datadir=d:/mysql/data/
```

注意：本实验采用定制安装方式

（2）启动服务程序

双击 d:\mysql\bin 下的 winmysqladmin.exe（MySQL 服务程序），这样就会在操作系统的后台服务中注册并启动 MySQL 服务程序（默认情况下，每次操作系统启动时自动启动该服务，在 Windows2000 的服务中可更改其启动方式）

（3）MySQL 数据库连接测试：通过 MySQL 的客户端程序 mysql.exe，测试其与服务程序的连接是否正常，服务程序是否能够正常工作。

启动客户端程序 mysql.exe：在命令提示符或 MS-DOS 下，进入 d:\mysql\bin，键入命令 mysql 回车。

若出现类似如下结果:

```
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 26 to server version: 3.23.43
Type 'help;' or '\h' for help. Type '\c' to clear the buffer.
mysql>
```

则表明本测试成功。
键入 exit 回车，可退出与 MySQL 服务器的连接

至此，我们已经成功完成了 PHP 脚本解释器软件、Web 服务器软件 Apache、数据库服务器软件 MySQL 的部署。对于操作系统软件和开发工具软件，读者可自行完成部署。

本实验到此结束。

第二章 Web 客户端技术

第一节 Web 客户端工作原理

1.浏览器工作原理

（1）Web 客户端工作原理

Web 客户端，在本课程中，显然就是指浏览器（Browser）端。任何应用系统都必须有一个供用户操作的界面，即用户界面。浏览器的工作，从整个 B/S 程序来看，是用户与整个 B/S 程序打交道的一个界面（接口），即人机界面（接口）、用户界面，它的任务是：

- 收集用户输入的数据（如用户数据：http://211.64.32.2/dsks/index.php）
- 将用户数据发送到服务器（向服务器请求该用户对服务器的请求）
- 接收服务器返回的响应（用浏览器能认识和执行的代码即客户端代码表示，如 HTML 代码，JavaScript 代码等）
- 解释，执行这些代码，将结果显示在浏览器窗口中

可见，浏览器扮演的，是（多数情况下是远程的）服务器在用户那里的一个代理（Agent）的角色。这个代理，具有收集消息，请求响应和解释其领导（服务器）发回的指示的作用。

（2）Web 客户端代码

无论是组织用于收集用户数据的界面，还是解释来自服务器的指示形成结果界面，都是用 Web 客户端代码表示的。常用的 Web 客户端代码使用 HTML 语言或 JavaScript 语言编写的，分别称为 HTML 代码或 JavaScript 代码。另外还有 CSS、XML 等语言。本课程仅介绍 HTML 语言或 JavaScript 语言。

2.HTML 工作原理

（1）HTML 简介

HTML：Hyper Text Markup Language，超文本标记语言。

20 世纪 80 年代末，在欧洲粒子物理实验室（CERN：the European Laboratory for Particle Physics）工作的 Tim Berners-Lee（人称 WWW 之父）通过研究发现：人们的视觉处理是以页为基础的。于是他得出了一个结论：电子资料应以页的方式呈现。以此为出发点，他使用超文本为中心的管理方式来组织网络上的资料，并提出了建立、存取与浏览网页的方法；建立了超文本标记语言；设计了超文本传输协议（HTTP：Hypertext Transport Protocol），用于获取超链接文件；使用统一资源定位器（URL：Uniform Resource Locator）来定位网络文件、站点或服务器。

（2）HTML 工作原理

（1）HTML 本质

HTML 不是程序语言，而是一种标记语言。所谓标记，在有的书上也叫标签，从面向对象的角度而言，就是对浏览器对象的标识的意思。它用来控制文字、图片等浏览器的子对象在浏览器中的表现，以及如何建立文件之间链接的标记，这些标记均放在文本格式的文件中。而程序与之最大的不同，就是可用来控制操作系统或应用程序执行并完成某项作业。超文本标记语言的文档应该尽量做到，从形式上看，无论在任何操作系统的任何浏览器上打开都具有相同的效果。

（2）HTML 的基本结构

其基本结构分为三部分：

- 版本声明，即序（Prologue）
- 头部（Head）
- 主体（Body）

其中，主要部分是头部和主体。

【例 2-1】Web 文档基本 HTML 结构标记

2-1.htm 源代码：

```
<!--
代码内容:Web 文档基本 HTML 结构标记
作　　者:孙寿龙
日　　期:20050907
-->
<HTML>
    <HEAD>
        <TITLE>这里写网页的标题</TITLE>
    </HEAD>
    <BODY>
        这里是网页的主体（显示在浏览器窗口中的部分）
    </BODY>
</HTML>
```

说明：

①标记一般成对出现：<开始标记></结束标记>，为了防止忘记写结束标记符，可采用成对书写，然后在中间插入的写法。

②放在 HEAD 标记内的信息一般不显示在浏览器的窗口中，通常这里面用来定义 JavaScript 函数，包含 JavaScript 代码文件，包含层叠样式表（CSS：Cascading Style Sheets）文件等一些预处理工作。

③BODY 标记内通常放上需要表示或展示内容的标记格式。

④HTML 中的注释：

第一种格式：<!注释内容>。其中，注释内容中不可出现“>”，常用于说明标记里的内容；

第二种格式：<!--注释内容-->。其中，注释内容中可包括“>”在内的任何符号，常用于注释大段的内容。

⑤HTML 编辑软件：只要是文本编辑器或自带 HTML 编辑器的软件，就可以用来编写 HTML 文件，也可以使用 HTML 专用编辑器如 Dreamweaver 等来编辑 HTML。可通过使用 HTML 专用编辑器快速生成一个 HTML 的基本结构；快速学习并掌握 HTML 语言。

⑥标记符中的字母，如<HTML>中的 HTML，大小写不敏感，建议统一大些或小写使用。

⑦文件的扩展名：若仅含有 HTML 代码，一般以.htm 为扩展名；若仅含有 JavaScript 代码，这种文件常作为.htm 文件的包含文件（引用文件，类似于 C 程序文件中的头文件）一般以.js 为扩展名；若含 HTML 代码和 JavaScript 代码，一般以.htm 为扩展名。

（3）HTML 标记简介

1）基本标记（Basic Tags）

- <HTML></HTML>　定义整个超文本文档（网页）对象，描述 Web 页面的起始与终止。
- <HEAD></HEAD>　设置页面的头部分，用来包含当前文档的一些相关信息。如定义样式、网页的标题、网页中使用的脚本语言以及对搜索引擎有帮助的关键字。
- <TITLE></TITLE>　用来指明文件的标题，其内容将显示在浏览器的标题栏内，设置它的好处：可为下载时提供默认的文件名；可为搜索引擎提供搜索关键字。
- <BODY></BODY>　放置Web页面的正文内容，包含文件内的文字、超链接文字的颜色、背景色彩、图片、动画、影像、音效等几乎所有对网页的展示功能。
- <META>　用来介绍与文件内容相关的信息。每一个<META>标记用于指明一个名称或数值对，常常放在头部标记中。

2）文本、字符格式（Text & Char Format）

- <Hn></Hn>　标题文字（n=1~6）
-
　换行标记
- <P></P>　段落标记
- <HR>　水平线标记

- <FONT FACE=”字体名称” SIZE=”大小” COLOR=”颜色名或颜色值”>字符串</FONT> 设置字符串的字体、大小、颜色

  颜色名：red,green,blue,yellow,black,white 等。

  颜色值：格式为#rrggbb，其中，r,g,b 分别用十六进制数表示的红、绿、蓝三种颜色，如#FF0000 表示红色，而#6CB0A6 表示一种青色。
- 字符格式标记

<B></B> 粗体

<I></I> 斜体

<U></U> 加下划线

<STRONG></STRONG> 着重强调

 定义上标

 定义下标

3）超链接（Hyperlink）

标记<A href=”URL” … ></A>表示一个超链接元素。超链接的属性主要有超链接地址、超链接文件打开的窗口位置，都在其开始标记中定义。

【例 2-2】超链接标记

2-2.htm 源代码：

```
<A  href="http://nc.dzu.edu.cn/article/show.php?id=139" target="_blank" >ASP,PHP,JSP 之比较</A>
```

其中，

href 即超链接地址，其值为 http://211.64.32.2/dsks/index.php

target 即窗口位置，对其值_blank 而言，浏览器接收到服务器 211.64.32.2 发来的文件 ndex.php，将在一个新浏览器的窗口中显示。

超链接一般简称链接。

4）表格（Table）

常用表格来精确定义页面文本或图片等的排版格式、排版布局，以使整齐美观。

<TABLE></TABLE> 定义一个表格

<TR></TR> 定义表格内的一行

<TD></TD> 定义一行内的一个单元格

5）表单（Form）

表单的概念同 VB、VF、VC 等程序设计语言，它是浏览器收集、发送用户所填数据的一种浏览器对象(控件)，就像一部货车，一艘轮船，它本身不承载数据，而是通过包含表单对象（就像轮船上的集装箱）这些可以盛放数据的数据容器来承载数据，传送数据，从这个角度来看，它实际上是一个盛放数据容器的容器。

表单是 B/S 程序中人机交互界面的主要形式。从服务器的角度来看，或者说从服务器程序编写人员来看，表单及表单对象的名称（即其 NAME 属性名）被服务器看作变量来接收，称作表单变量；表单变量的值即用户在客户端表单对象中填写的数据。

表单的一般标记有：

- 表单本身：<FORM NAME=”form1” ACTION=”chuli.php” METHOD=”POST” ></FORM> 定义表单，其中：

  属性 ACTION 的值起指明将表单中数据提交（发送的意思）的方向，即服务器上的某个处理程序。

  属性 METHOD 指明提交数据的方法，常用 POST 和 GET。
- <INPUT> 输入型表单对象

  ①文本字段，类似于 VB、VF、VC 里的文本框控件，基本标记形式如下：

  <input name="textfield" type="text" value="这里是文本字段的值">

  ②隐藏域，设计时可见，运行时不可见的文本字段，程序员常用它向 FORM 的 ACTION 指向的文件传送变量。

  <input type="hidden" name="hiddenField">

  ③文本区域，类似于 VB、VF、VC 里的文本框控件，基本标记形式如下：

  <textarea name="textarea" cols="25" rows="5">这里是文本区域的值</textarea>

④单选按钮，类似于 VB、VF、VC 里的单选按钮控件（有人也称之为无线按钮），作用是在同名的多个单选按钮中提供单项选择。
⑤复选框，作用是在同名的多个复选框中提供多项选择。
⑥列表/菜单域，概念等同于 VB、VF、VC 里的下拉列表框。<SELECT></SELECT>
⑦提交表单型表单按钮，标记形式：
<INPUT TYPE=”submit” VALUE=”提交”> 作用是将表单中的数据提交到表单属性 ACTION 的值所指向的服务器端程序，由服务器端程序处理
⑧重置型表单按钮，标记形式：
<INPUT TYPE=”reset” VALUE=”重新填写”> 作用是清空表单中每个输入域中的数据，等待用户重新输入。
⑨定制型表单按钮，标记形式：
<INPUT TYPE=”button” VALUE=”转到教学网站” ONCLICK=”window.location=’ http://211.64.32.2/dsks’ ”> 作用是执行用户指定的函数、过程。这里，用户通过指定该按钮的单击事件处理过程为：将当前页面跳转到教学网站主页。
【例 2-3】FORM 标签示例
2-3.htm 源代码：

```
<form name="form1" method="post" action="2-3.php">
    姓名：<input name="xm" type="text"><br>
    简介：<textarea name="jj" cols="25" rows="5"></textarea><br>
    性别：
    <input type="radio" name="xb" value="1">男
    <input type="radio" name="xb" value="0">女
    <br>
    爱好：
    <input type="checkbox" name="ah" value="1">运动
    <input type="checkbox" name="ah" value="2">音乐
    <input type="checkbox" name="ah" value="3">旅游<br>
    今天要去哪里逛一逛：
    <select name="where">
        <option value="你未选择任何地方！" selected>请选择</option>
        <option value="http://211.64.32.2/dsks">孙寿龙教学网站</option>
        <option value="http://www.dzu.edu.cn">德州学院网站</option>
        <option value="http://www.163.com">网易</option>
    </select>
    <br>
    <input type="hidden" name="hiddenField">
    <br>
    <input type="submit" name="Submit" value="提交">
    <input type="reset" name="Submit2" value="重置">
    <input type="button" name="Submit3" value="定制">
</form>
```

向服务器请求 2-3.htm 的结果见图 10。

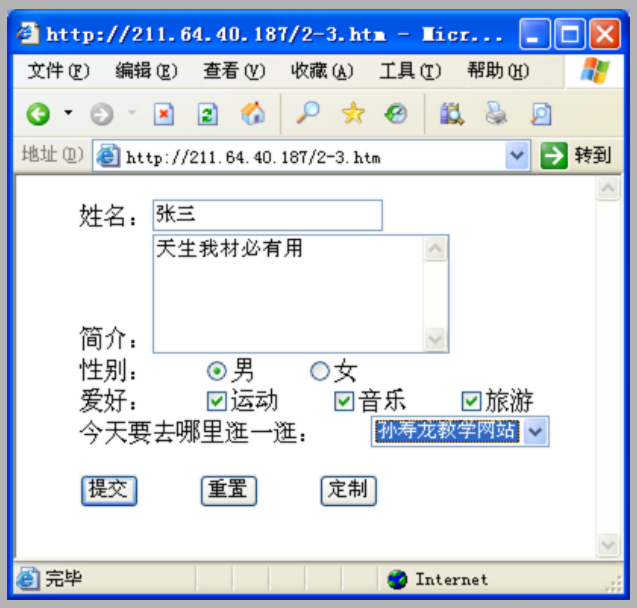


图 10 FORM 标签示例

以上介绍了 HTML 的常用标记，美国麻省理工学院（MIT：Massachusetts Institute of Technology）计算机科学实验室的 WWW 标准化组织 W3C（又称万维网联盟 W3C），是 HTML 的先锋，其互联网地址为：http://www.w3.org。有关 HTML、CSS、XML 语言的各种最新的、权威的、官方的资料，在该网站均可查到。

3.JavaScript 工作原理

HTML 代码所表示的文档是一种静态文档，几乎没有交互功能，很难使页面成为动态页面。增加脚本语言，可使数据发送到服务器之前先进行处理和校验，动态地创建新的 Web 内容，更重要的是，引入脚本语言使我们有了事件驱动的软件开发环境。

（1）JavaScript 简介

JavaScript 的原名叫 LiveScript，是 NetScape 公司在引入 Sun 公司有关 Java 的程序设计概念后，重新设计而更名的。JavaScript 是一种可以嵌入 HTML 文档的，基于对象并具有某些面向对象特征的脚本语言。

说明：浏览器端脚本语言除了 JavaScript，还有 VBScript 和 Jscript，后两个是 Microsoft 公司设计的，欲了解相关内容，可到 Microsoft 公司网站查询。本课程仅介绍 JavaScript。

（2）JavaScript 的特点

JavaScript 是一种基于对象(Object-Based)和事件驱动(Event Driven)，由浏览器解释执行的，具有安全性能的客户端脚本语言。使用它的目的是与 HTML、Java Applet（Java 小程序）一起实现在一个 Web 页面中链接多个对象，与 Web 客户交互作用，从而可以开发客户端的应用程序等。它是通过嵌入在 HTML 语言中实现的。它的出现弥补了 HTML 语言的缺陷，它是 Java 与 HTML 折衷的选择，具有以下几个基本特点：

- 是一种脚本语言

  采用小程序段的方式实现编程，以嵌入的方式，与 HTML 标识结合在一起，方便用户的使用操作。

- 基于对象的语言。

这里的对象，是指客户机、浏览器、网页文档。也就是说，JavaScript 以类似 C、Java 的语法，以客户机、浏览器、网页文档、文档内部各种以标记表示的 HTML 元素为对象，以控制这些对象为目标，进而控制整个客户端的一种客户端脚本编程语言。

- 简单

首先它是一种基于 Java 基本语句和控制流之上的简单而紧凑的设计，从而对于学习 Java 是一种非常好的过渡。其次它的变量类型是采用弱类型，并未使用严格的数据类型。

- 安全

它不允许访问服务器本地的硬盘，因此不能将数据存入到服务器上；不允许对网络文档进行修改和删除，只能

通过浏览器实现信息浏览或动态交互。从而有效地防止数据的丢失。

- 动态

它可以直接对用户的输入做出响应，无须经过 Web 服务程序。它对用户的响应，是采用以事件驱动的方式进行的。事件(Event)可分为两类，一是用户对浏览器进行的某种操作，比如按下鼠标、移动窗口、选择菜单等，可以视为用户事件；二是系统事件，如时间的时刻变化等。当事件发生后，会向浏览器发送相应的消息(用户消息或系统消息)，根据消息，浏览器可能会做出相应的响应，这种响应称为事件驱动，也叫消息驱动。

- 跨平台

JavaScript 代码由浏览器解释执行，与操作环境无关，只要能运行浏览器的计算机，并支持 JavaScript 的浏览器就可正确执行，从而实现了“编写一次，走遍天下”的梦想。

实际上 JavaScript 最杰出之处在于可以用很小的程序做大量的事。无须有高性能的电脑，软件仅需一个字处理软件及一浏览器，无须 WEB 服务器通道，通过自己的电脑即可完成所有的事情。

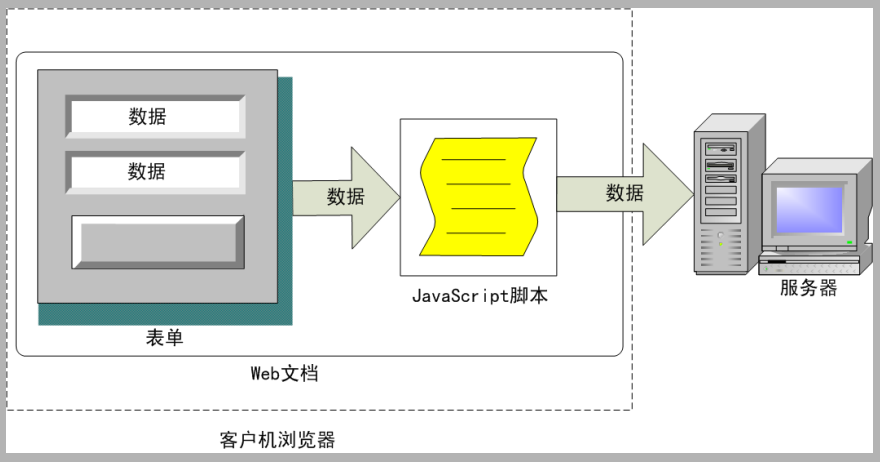
JavaScript 和 Java 很类似，但并不一样。Java 是一种比 JavaScript 更复杂许多的程序语言，而 JavaScript 则是相当容易了解的语言。许多 Java 的特性在 Java Script 中并不支持。

（3）JavaScript 的工作原理

1）JavaScript 的工作原理

JavaScript 编程可以完成诸如构造动画，动态菜单等使页面更加生动、活泼的任务（实例见洪恩在线：http://www.hongen.com/pc/homepage/javascript/index1.htm），还可以对客户机文件系统、注册表等进行操作，如对文件夹、文件的建立，复制，删除，修改注册表，锁定注册表，锁定浏览器等等，有许多随着网页打开而运行的病毒就是含在网页中的 JavaScript 程序在作怪。由此可见，JavaScript 是控制客户机的精灵。

在 B/S 程序中，为了均衡负载，减轻服务器的计算负担，凡是不需要服务器程序做的工作，可尽量交给客户端程序（如 JavaScript 程序）去做。我们用 HTML 标记构造出用户界面，用户通过界面输入数据，向浏览器请求数据等操作。在用户输入数据，或者是输入完毕，将数据向服务器提交的时候，对数据的检验等任务完全可交给 JavaScript 程序来完成。本课程主要介绍此类任务的 JavaScript 编程技术。



通过 JavaScript 脚本检验 FORM 数据

JavaScript 的工作原理，就是以基于对象和一些面向对象的特征：

- JavaScript 通过控制客户机上各种对象的方式，控制客户机，对客户机进行操作。
- 根据用户或系统事件，做出相应的响应。

第二节 Web 客户端基本技术

1.数据传递

（1）客户端向服务器传递数据的方法：POST 和 GET

浏览器向服务器进行数据传送，若使用 FORM，常用的传送数据的方法是 GET 和 POST。

①GET 方法通过 URL 请求来传递用户的输入，形式：URL?var_name1=value1&var_name2=value2，即将表单内各字段名称与其内容，以成对的字符串连接，置于表单 ACTION 属性所指的 URL 后，如 http://211.64.32.2/login.php?name=abc&password=123，数据都会直接显示在 URL 上，就像用户点击一个链

接一样；POST 方法通过 HTTP POST 机制，将表单内各字段名称与其内容放置在 HTML 表头(header)内一起传送给服务器端交由 ACTION 属性能所指的程序处理，该程序会通过标准输入(stdin)方式，将表单的数据读出并加以处理。

②通过 GET 方法提交数据，可能会带来安全性的问题。比如一个登陆页面，当通过 GET 方法提交数据时，用户名和密码将出现在 URL 上。如果登陆页面可以被浏览器缓存或其他人可以访问客户的这台机器。那么，别人就可以从浏览器的历史记录中，读取到此客户的账号和密码。所以，在某些情况下，GET 方法会带来严重的安全性问题。

③GET 方式传输的数据量非常小，一般限制在 2 KB 左右，但是执行效率却比 POST 方法好；而 POST 方式传递的数据量相对较大，它是等待服务器来读取数据，不过也有字节限制，这是为了避免对服务器用大量数据进行恶意攻击。使用 PHP，默认的 POST_MAX_SIZE 是 2M（通过配置 php.ini 实现），如果你想利用 POST 方式上传软件，就需要更改这个值了（我设置为 20M 仍然能够正确上传文件），但是倘若试图使用 GET 方式，就没有可能实现这种功能。

建议在 FORM 中，使用 POST 方法。

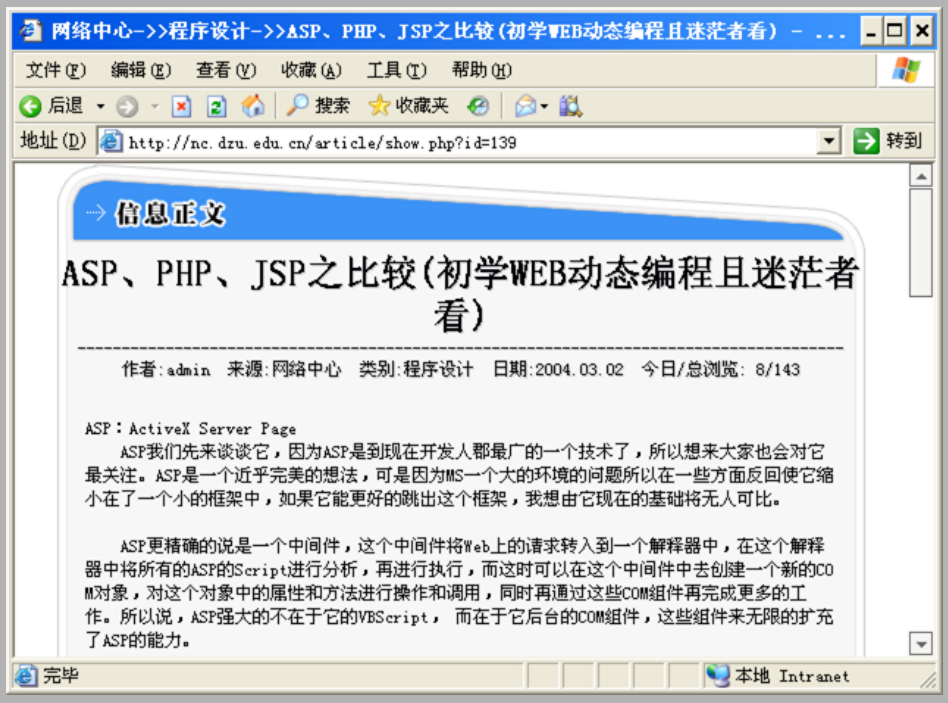
下列情况表明传递数据使用的是 GET 方法：

- 在地址栏中进行 GET 请求

  在地址栏中使用请求形式：URL?var_name1=value1&var_name2=value2……。

  如 http://nc.dzu.edu.cn/article/show.php?id=139（其中 URL 为 http://nc.dzu.edu.cn/article/show.php），

  作用：向 URL 所指向的程序文件 show.php 传递一个变量，变量名为 id，变量值为 139，该程序将从数据库中取出有关 id 为 139 的那条新闻的信息，表示成 HTML 代码后返回给请求者的浏览器。



在地址栏中进行 GET 请求的情况

- 单击了一个链接。

  【例 2-4】通过链接传递数据的 HTML 标记

2-4.htm 源代码：

```
<A  href="http://nc.dzu.edu.cn/article/show.php?id=139" >ASP,PHP,JSP 之比较</A>
```

- 对 FORM 的 METHOD 属性，使用 GET 属性值

（2）小结

从 PHP 的角度看，浏览器向服务器传递数据：

使用表单对象时，$表单对象名=表单变量。

使用 URL?参数名=参数值时，$参数名=查询字符串（querystring）变量。

如【例 2-3】中，用户在文本字段 textfield 中输入的数据，发送到 FORM 的 ACTION 指向的服务器程序 2-3.php 后，2-3.php 将接收到表单变量$textfield 和该变量内的值。

当在超链接地址或地址栏采用 URL?var_name1=value1&var_name2=value2……形式时，var_name1、var_name2 等是查询字符串变量。

另外，一个服务器程序，也称页，对接收到的表单变量或查询字符串变量，都作为页内局部变量处理，本页程序执行完毕，这些变量统统自动释放，所以不能将这些变量传递到另外一个 PHP 程序中去，即不能跨页传递。就像接力棒比赛中，不能跨人传棒。

既然默认情况下，不能将 x.php 中的$a 传到 y.php，要想将 x.php 中的局部变量$a 传到程序 y.php，可借助于隐藏域：在 x.php 中用表单中的隐藏域，临时存储$a 的值，然后提交表单到 y.php，即可实现向 y.php 发送数据的目的。

具体做法一般是这样：

x.php:

```
…
<form action=”y.php” method=”post”>
   <input type=”hidden” name=”a” value=”<? echo $a;?>”>
   <input type=”submit” value=”提交”>
   <input type=”reset” value=”重置”>
</form>
…
```

这时，从请求服务的角度来看，x.php 就是客户端程序，y.php 就是服务器程序了。

2.JavaScript 嵌入 HTML 的方式

与 C 语言非常相似，但去掉了 C 语言中有关指针等容易产生的错误，并提供了功能强大的类库。对于已经具备 C 语言的人来说，学习 JavaScript 脚本语言是一件非常轻松愉快的事。

JavaScript 的脚本包括在 HTML 中，它成为 HTML 文档的一部分。与 HTML 标识相结合，构成了一个功能强大的 Internet 网上编程语言。

（1）JavaScript 嵌入 HTML 的方法

1）块嵌入：显式的 JavaScript 脚本块嵌入的方法

JavaScript 块：

```
<Script Language ="JavaScript">
   JavaScript 语句 1；
   JavaScript 语句 2；
   ……
</Script>
```

说明： 通过脚本语言开始标记<Script>和脚本语言结束标记</Script>指明 JavaScript 脚本源代码块。

通过属性 Language ="JavaScript"说明标记中是使用的何种语言，这里是 JavaScript 语言，表示在 JavaScript 中使用的语言。

嵌入的地方：

- 嵌入<head>…</Head>中：在主页和其余部分代码之前装载，从而可使代码的功能更强大；
- 嵌入<Body>…</Body>中：以实现某些部分动态地创建文档。

下面是将 JavaScript 脚本块加入 Web 文档中的例子:

【例 2-5】将 JavaScript 脚本块加入 Web 文档

2-5.htm 源代码：

```
<HTML>
   <Head>
      <Script Language ="JavaScript">
         document.write("hello,world");
         //document.close();注释方式同 C 和 PHP
      </Script>
   </Head>
</HTML>
```

在浏览器的窗口中调用 2-5.htm，则显示“hello,world”字串。见图所示。

用 JavaScript 脚本输出的 hello,world

说明:

①document.write()是文档对象的输出函数，其功能是将括号中的字符或变量值输出到窗口；

②document.close()是将输出关闭。

在实际应用中，常常将自定义的 JavaScript 函数放在<head>…</Head>中，JavaScript 脚本块形成这个样子：

```
<Script Language ="JavaScript">
  function   fun1(参数表){
    JavaScript 语句集
  }
  function   fun2(参数表){
    JavaScript 语句集
  }

  ……
</Script>
```

2)包含文件：为了避免<head>…</Head>中 JavaScript 脚本块过大导致的网页文档代码过长，还可采取一种形式类似 C 程序，在头部包含 JavaScript 代码的做法：

```
<HEAD>
<script   language=”JavaScript”   src=”abc/xyz.js”></script>
</HEAD>
```

xyz.js 中的内容即具体的 JavaScript 脚本块。

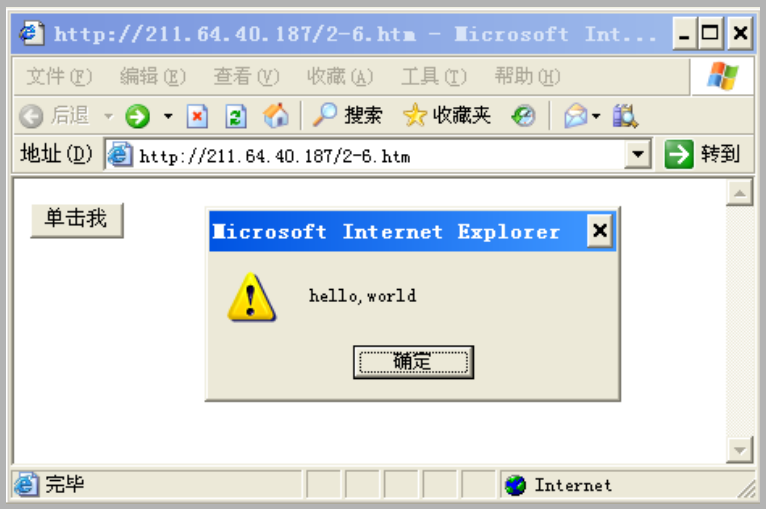
3）隐式的嵌入方式

不进行声明或仅进行简短声明，直接用于事件驱动的处理程序中。

【例 2-6】直接用于事件处理代码中的 JavaScript 脚本

2-6.htm 源代码：

```
<input type="button" name="Submit3" value="单击我" onClick="javascript:alert('hello,world');">
<!--或：
<input type="button" name="Submit3" value="单击我" onClick="alert('hello,world');">
-->
```



直接用于事件驱动中的 JavaScript 脚本

显然这种方式对于较短的事件处理 JavaScript 代码很适用，若这种代码较长，应采取块嵌入或包含文件的方法。

3.客户机对象技术

从 JavaScript 的工作原理可以看出，为了更好地完成控制操作和做出响应动作，JavaScript 编程者必须清楚地了解常用的客户机对象。

（1）对象的基础知识

1)使用对象的什么：使用对象的属性、事件、方法。在 JavaScript 中，属性，表示对象的性质的值，往往用“对象名．属性名”的形式引用；事件往往用“on 事件名”来侦测、标识，表示“当……的时候”；方法是对象发出的动作，往往用“对象名. 方法名()”的形式使用。

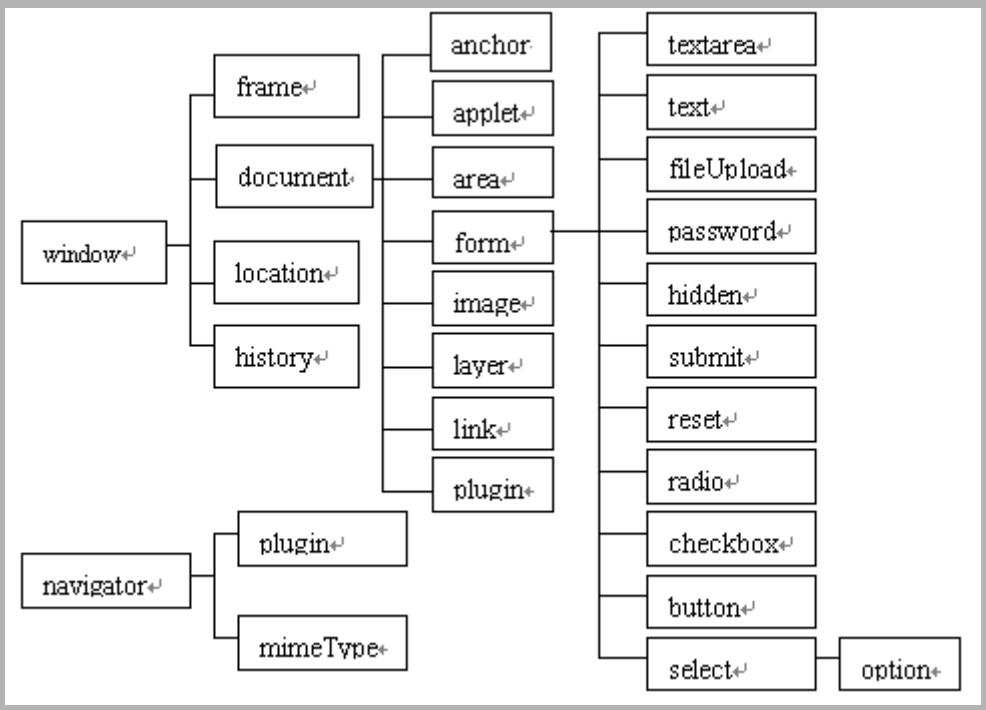
2）如何获得（引用）对象：一个对象要真正地被使用，可采用以下几种方式获得该对象：

- ★ 引用 **JavaScript 内部对象**（常用）
- ★ 引用**浏览器对象**（常用）
- ★ 创建**新对象**，然后引用之

即对象使用之前必须存在，要么创建新的对象，要么利用现存的对象。

JavaScript 内部对象：JavaScript built-in Object，即 JavaScript 语言本身的对象，如 eval(字符串)——返回字符串表达式中的值。

浏览器对象：如窗口 WINDOW，文档 DOCUMENT、表单 FORM 等，它们之间是分层次的树状关系，反映这种关系的模型，称作文档对象模型（DOM：Document Object Model）。



创建新对象：

格式：新对象名=new　已存在对象名(参数表);

如：

```
var now = new Date();
var year = now.getYear();
alert('现在是'+now);
alert('今年是'+year);
```

（2）对象的引用方式

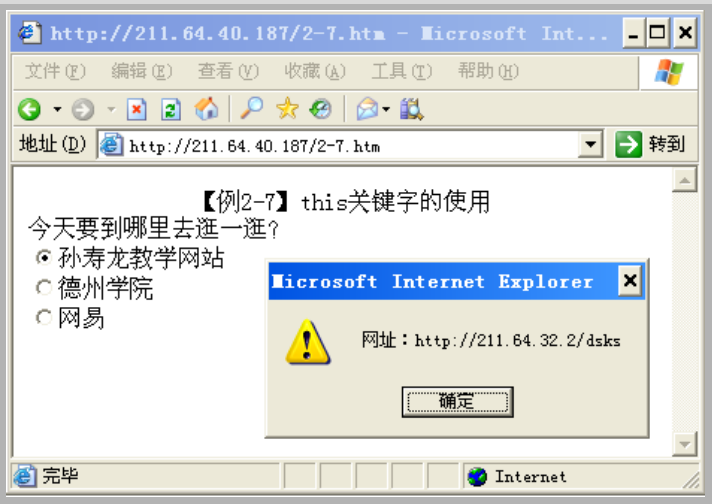
■ 自引用

指对象的自我引用，用关键字 this 代指自己。

【例 2-7】使用 this 关键字进行自引用

2-7.htm 源代码：

```
<CENTER>【例 2-7】this 关键字的使用</CENTER>
今天要到哪里去逛一逛？<br>
<input type="radio" name="where" value="http://211.64.32.2/dsks" onClick="alert('网址：'+this.value);">孙寿龙教学网站<br>
<input type="radio" name="where" value="http://www.dzu.edu.cn" onClick="alert('网址：'+this.value);">德州学院<br>
<input type="radio" name="where" value="http://www.163.com" onClick="alert('网址：'+this.value);">网易
```



this 关键字的使用

■ 按层次引用：按文档对象模型层次进行引用

按层次引用引用的基本模式是：父对象．子对象，如：

window.alert(window.document.form1.textfield1.value);

其中，当前窗口可不用指明 window 对象，所以上一行代码常常写成这个样子：

alert(document.form1.textfield1.value);

按 DOM 层次引用对象的适用条件：应明确知晓父子关系和各自名称。

■ 按下标引用。适用于对象集合的处理，同名对象的集合按该名字命名的数组，通过数组下标的访问引用每个对象，下标从 0 开始。使用场合：同名单选按钮组、复选框组等，组内各成员的引用。

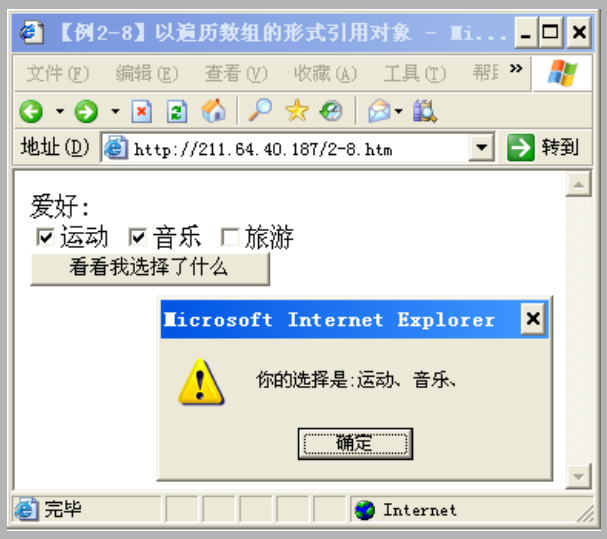
【例 2-8】以遍历数组的形式引用对象

2-8.htm 源代码：

```
<html>
    <head>
        <meta http-equiv="Content-Type" content="text/html; charset=gb2312">
        <title>【例 2-8】以遍历数组的形式引用对象</title>
        <script language="javascript">
```

```
            function func2_8(){
            var mychoice="";
            for(var i=0;i<document.form1.ah.length;i++){
                if(document.form1.ah[i].checked)
                    mychoice=mychoice+document.form1.ah[i].value+"、";
            }
            alert("你的选择是:"+mychoice);
        }
        </script>
    </head>
    <body>
        <form name="form1" method="post" action="">
            爱好:<br>
            <input type="checkbox" name="ah" value="运动">运动
            <input type="checkbox" name="ah" value="音乐">音乐
            <input type="checkbox" name="ah" value="旅游">旅游<br>
            <input type="button" name="Submit" value="看看我选择了什么" onClick="func2_8();">
        </form>
    </body>
</html>
```

说明：数组元素的个数，即数组的长度，用“数组名．length”引用数组的长度属性值 length 得到。



以遍历数组的形式引用对象

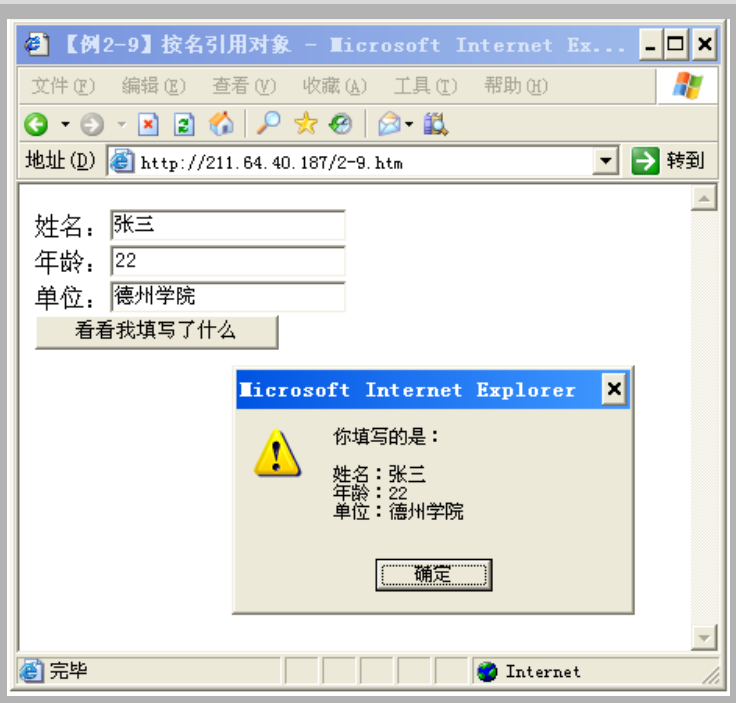
■ 按名引用。同类型但不同名的对象集合内各成员，可通过这种方式引用。

【例 2-9】按名引用对象

2-9.htm 源代码：

```
<html>
    <head>
        <meta http-equiv="Content-Type" content="text/html; charset=gb2312">
        <title>【例 2-9】按名引用对象</title>
        <script language="javascript">
            function func2_9(){
            var xm="";
            var nl="";
            var dw="";
            xm=document.form1.elements["xm"].value;
```

```
            nl=document.form1.elements["nl"].value;
            dw=document.form1.elements["dw"].value;
            alert("你填写的是：\n\n 姓名："+xm+"\n 年龄："+nl+"\n 单位："+dw);
        }
        </script>
    </head>
    <body>
        <form name="form1" method="post" action="">
            姓名：<input type="text" name="xm"><br>
            年龄：<input type="text" name="nl"><br>
            单位：<input type="text" name="dw"><br>
            <input type="button" name="Submit" value="看看我填写了什么" onClick="func2_9();">
        </form>
    </body>
</html>
```



按名引用对象

类似地，也可通过 document. Forms[]数组来引用在同一个页面上多个表单中的某个表单，例如，将第一个表单中名为 textfield1 的文本框赋值 123：

```
document.forms[0].textfield1.value=123;
```

（2）事件及事件处理

①基本概念

通常鼠标或热键的动作我们称之为事件（Event），而由鼠标或热键引发的一连串程序的动作，称之为事件驱动（Event Driver）。驱动，在这里就是引发，触发的意思。这些动作，就是对事件进行处理的程序或函数，我们称之为事件处理程序（Event Handler）。其基本格式与函数全部一样，可以将前面所介绍的所有函数作为事件处理程序。格式如下：

```
Function 事件处理名（参数表）{
  事件处理语句集;
}
```

可以使用自己编写的函数作为事件处理程序，也可以使用 JavaScript 中内部的函数，还可以直接使用 JavaScript 的代码等，如例 2-6。

②主要事件

■ onClick——单击

当用户单击鼠标按钮时，产生 onClick 事件，同时 onClick 指定的事件处理程序将被调用执行。在很多对象中产生：

- button（按钮对象）：包括
  - ★ reset button（重置型按钮）
  - ★ submit button（提交型按钮）
  - ★ radio （单选按钮或无线按钮）
- checkbox（复选框或检查框）
- 超链接

……

使用方法大同小异，参考【例 2-6】即可。

■ onChange——数据被更改

下列情况将触发相关对象的 onChange 事件：

text 元素输入的字符值改变时

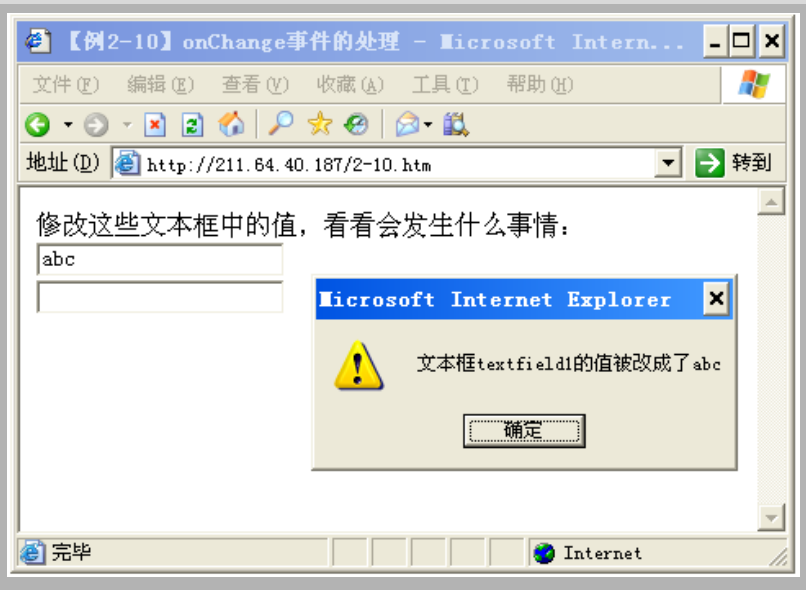
texturea 元素输入的字符值改变时

select 元素选项改变后

【例 2-10】onChange 事件的处理

2-10.htm 源代码：

```
<html>
    <head>
        <meta http-equiv="Content-Type" content="text/html; charset=gb2312">
        <title>【例 2-10】onChange 事件的处理</title>
        <script language="javascript">
            function func2_10(textname,textvalue){
                alert("文本框"+textname+"的值被改成了"+textvalue)
            }
        </script>
    </head>
    <body>
        修改这些文本框中的值，看看会发生什么事情：<br>
        <input name="textfield1" type="text" onChange="func2_10(this.name,this.value);"><br>
        <input name="textfield2" type="text" onChange="func2_10(this.name,this.value);">
    </body>
</html>
```



更多事件的详细介绍，请参考相关资料，在此不再赘述。
（3）常用 JavaScript 内置对象的使用
按数据类型可分为：字符串(string)对象、算术函数（math）对象、日期(Date)对象
按使用时是否需要创建实例可分为：静态对象和动态对象。
静态对象：在引用其属性或方法时不需要为它创建实例，如 string（字符串）；
动态对象：在引用其属性或方法时必须为它创建一个实例，如 Date（日期）。
基本使用格式：对象名．属性名
对象名．方法名()
1）串对象
①属性。只有一个属性，即 length。它表明了字符串中的字符个数，包括所有符号。例：
②方法。共有 19 个，主要用于串在 Web 页面中的显示、字体大小、字体颜色、字符的搜索以及字符的大小写转换等。
其中，常用方法如下：
• 显示的控制方法
（a）fontsize(size)字体大小：作用同 HTML 字体标签。
语法：fontsize(size)　其中，1≤size≤7
（b）bold()粗体字
（c）Italics()斜体字
• 字体颜色；fontcolor(color)
• 大小写转换
toLowerCase()小写转换，toUpperCase()大写转换。
• 取指定位置的字符：charAt(index)，，0≤index≤串．长度-1。
• 定位字符首次出现位置：indexOf(character，fromIndex)
从指定 formIndtx 位置开始，在串中搜索 character 首出现的位置，0≤formIndex≤串．长度-1。
• 定位字符末次出现位置：lastIndexOf(character，fromIndex)
从指定 formIndtx 位置开始，在串中搜索 character 末次出现的位置，0≤formIndex≤串．长度-1。
• 取子串：substring(start,end)
取下标为[start，end）的子串。
若 start>end，返回下标为[start，end）的子串；
若 start=end，返回空串；
若 start>end，返回下标为[start，end）的子串
• 上标：sup()，作用同 HTML 上标标签
• 下标：sub()，作用同 HTML 下标标签
【例 2-11】JavaScript 串对象的使用
2-11.htm 源代码：

```
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=gb2312">
<title>【例 2-11】JavaScript 串对象的使用</title>
</head>
<script language="javascript">
    document.write("<center>【例 2-11】JavaScript 串对象的使用</center>");
    document.write("<br>");

    sa="hello";
    sb="world";
    document.write("串 sa："+sa);
```

```
    document.write("<br>");
    document.write("串 sb：  "+sb);
    document.write("<br>");

    document.write("sa 的长度："+sa.length);
    document.write("<br>");

    document.write("sa 设成 7 号字体："+sa.fontsize(7));
    document.write("<br>");

    document.write("sa 变红色："+sa.fontcolor("red"));
    document.write("<br>");

    document.write("sa 变大写："+sa.toUpperCase());
    document.write("<br>");

    document.write("sa 中首次出现字母 l 的下标位置="+ sa.indexOf("l",0));
    document.write("<br>");

    document.write("sa 中末次出现字母 l 的下标位置="+ sa.lastIndexOf("l",0));
    document.write("<br>");

    document.write("sa.substring(0,2)="+ sa.substring(0,2));
    document.write("<br>");

    document.write("sa.substring(2,0)="+ sa.substring(2,0));
    document.write("<br>");

    document.write("sa.substring(2,2)="+ sa.substring(2,2));
    document.write("<br>");

    document.write("sa.charAt(2)="+ sa.charAt(2));
    document.write("<br>");

    document.write("sb 输出为上标："+sb.sup());
    document.write("<br>");

    document.write("sb 输出为 sa 的上标："+sa+sb.sup());
    document.write("<br>");

    document.close();
</script>
<body>
</body>
</html>
```

2）系统函数

JavaScript 中的系统函数又称内部方法。它提供了与任何对象无关的系统函数，使用这些函数不需创建任何实例,可直接用。

方法名：eval（字串表达式）

作用：返回字符串表达式中的值

例：

test=eval("8+9+5/2");//test=19.5

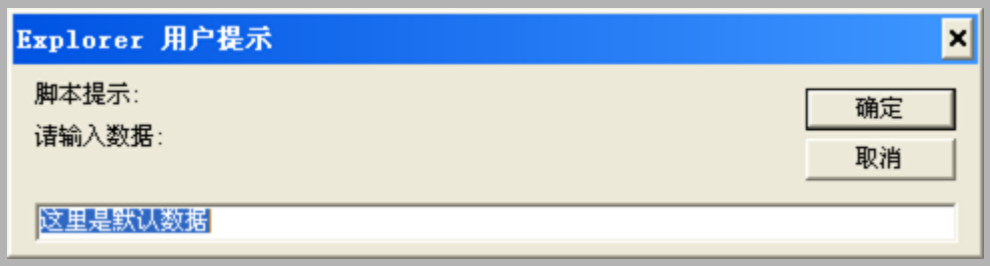
（4）窗口对象的使用

有关输入可通过窗口（Window）对象来完成，而输出可通过文档（document）对象的方法来实现。

【例 2-14】窗口的简单例子

2-14.htm 源代码：

```
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=gb2312">
<title>【例 2-14】窗口的简单例子</title>
</head>
<body>
<script languaga="JavaScript">
    var yourinput=window.prompt("请输入数据:","这里是默认数据");
    document.clear();
    document.write("你刚才输入的是："+yourinput);//输出流
    document.close();//关闭输出流
</script>
</body>
</html>
```



窗口的简单例子

其中 window.prompt()是窗口对象的一个方法，作用是，当装入 Web 页面时，在屏幕上显示一个具有“确定”和“取消”的对话框，让你输入数据。document.write 是文档对象的一个方法，作用是实现 Web 页面的输出显示。

窗口对象在 DOM 中的层次：顶层

1）属性：主要用来对浏览器中存在的各种窗口和框架的引用

- parent：指明当前窗口或框架的父窗口。
- defaultStatus：窗口的状态栏的默认显示值。
- status：窗口状态栏信息。
- top：引用顶层窗口。
- window：指的是当前窗口
- self：当前窗口自引用。
- frames 框架集合
- location 窗口中文档地址

框架，窗口的窗口，可以理解为子窗口，实现一个窗口的分隔，注意以下几点：

①frames 中各 frame 是通过 HTML 标识<frame>的顺序来引用的，它包含了一个窗口中的全部帧数。

②框架本身是窗口的子窗口，继承了窗口对象所有的全部属性和方法。

2）方法：主要用来提供信息或输入数据以及创建一个新的窗口。

- open()：将文档输出到一个新窗口中去。

语法格式：window.open(新窗口的文档来源，新窗口名称，新窗口特征参数);

可以创建一个新的窗口。其中参数表提供有窗口的主要特性和文档及窗口的命名。

表 新窗口特征参数

新窗口特征参数名	设定值	含 义
toolbar	yes/no	窗口中是否含有标准工具栏
location	yes/no	窗口中是否含有地址栏
directions	yes/no	窗口中是否含有文件夹按钮
status	yes/no	窗口中是否含有状态栏
menubar	yes/no	窗口中是否含有菜单栏
scrollbar	yes/no	窗口中是否含有滚动条
revisable	yes/no	能否调整窗口
width	像素值	确定窗口的宽度
height	像素值	确定窗口的

说明：yes/no 也可以换成 1/0；参数之间用逗号分隔。

例如：window.open(“abc/xyz.htm”,xyz,”toolbar=no,status=no,resizable=no”);以 xyz 为窗口名称，在其中输出文档 abc/xyz.htm，窗口 xyz 不带有工具栏、状态栏，不允许调整大小，其他特征采用浏览器对窗口的默认设定值。

- alert()：输出一个具有“确定”按钮的消息框，用来向用户显示消息，一旦按“确定”钮后，方可继续执行其他脚本程序，因此也可作为调试程序的手段。

  语法格式：window.alert(消息);

  常常采用简化格式：alert(消息);
- confirm()：输出一个具有“确定”和“取消”按钮的 确认框，获取用户确认信息。

语法格式：window.comfirm(消息);

- prompt()：输出一个允许用户输入信息的输入框。
  语法格式：window.prompt（“提示信息”，默认值）
- close()：关闭当前窗口
  语法格式：window.close();

3）事件

窗口对象对应于 HTML 文档中的<body>和<frame>两种标识；

- onload 事件：装入 Web 文档
- onunload 事件：卸载 Web 文档

如下代码标识的文档，当打开时和关闭时，都有相应的消息显示。

```
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=gb2312">
<title></title>
</head>
<body onload="alert('欢迎光临！');"    onunload="alert('欢迎再来！');">
</body>
</html>
```

（5）文档(document)对象

属性：

- bgColor 网页背景颜色
- fgColor 网页前景颜色，即网页中文字的颜色
- forms 网页中表单的集合
- lastModified 网页最后修改日期
- linkColor 网页中超链接文字颜色
- links 网页中超链接的集合
- location 网页文档地址
- title 网页文档的标题，即<title></title>中的内容

方法：

- clear () 清空文档内容
- write(字符串) 输出字符串
- writeln(字符串) 输出字符串，然后换行

事件：

- onload 事件：装入 Web 文档
- onunload 事件：卸载 Web 文档

第三节 实验二 Web 客户端技术

实验目的：掌握基本的 Web 客户端技术

实验要求：掌握基本 HTML 设计和 JavaScript 编程技术

实验环境：

实验室已经安装好如下实验环境

操作系统：Windows98

Web 服务器：Apache 1.3.14 安装路径：c:\Apache 默认发布文档路径：c:\Apache\htdocs

PHP：PHP 4.0.4 安装路径：c:\php

脚本编辑器：Macromedia Dreamweaver MX 中文版、记事本或 UltraEdit10 简体中文版
注意：使用以上脚本编辑器，若要保存为.php 文件，必须在保存时指明扩展名为.php
尤其是记事本，必须在保存时，单击“保存类型”下拉框，选择“所有文件(*.*)”类型，然后指定文件扩展名是.php，单击“保存”按钮后，保存的文件才能保证是.php 文件。常见错误是直接保存，导致保存的文件其实是文本文件，如 abc.php.txt 等以.txt 为扩展名的文本文件。

实验内容：

- 本章实例程序（代码可从教学网站下载）：【例 2-1】~【例 2-14】

实验方法：
（1）编写程序：录入或下载本章实例程序或附加试验程序，保存到发布文档目录
（2）阅读程序：结合讲义等资料，阅读理解这些程序，也可根据自己需要修改这些程序
（3）观摩效果：在浏览器中通过 HTTP 协议方式请求这些程序文件（而不是在我的电脑或资源管理器中打开）
做法：http://localhost/要访问的文件，回车

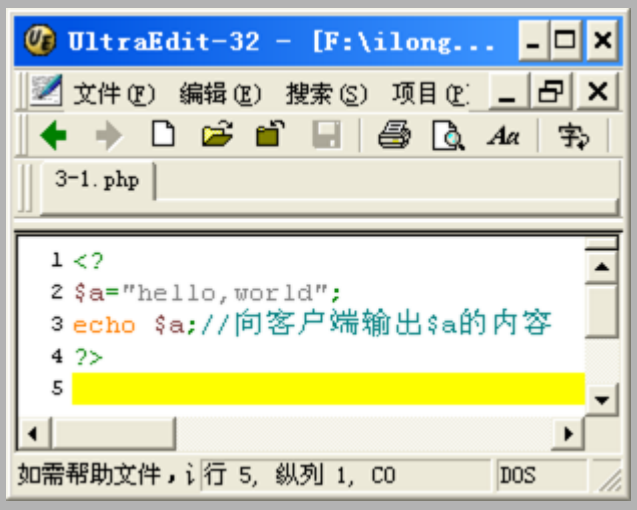
第三章 PHP 语法

本章主要介绍 PHP 基本语法，为进一步控制客户端表示，和操纵数据库做准备。

第一节 基本语法

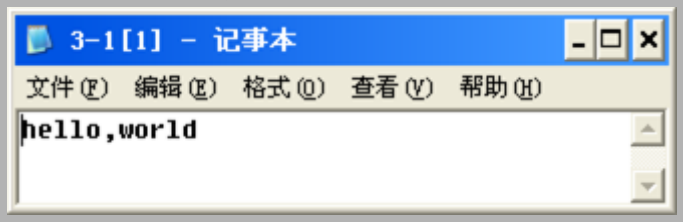
1.最简单的 PHP 程序

【例 3-1】最简单的 PHP 程序（3-1.php）



例 3-1 源程序

该例作用是 PHP 向 Apache 送出一个字符串常量，内容是 hello,world，再由 Apache 将其送到给向客户端的浏览器，由浏览器来解释执行。浏览器接收到的内容如图所示，可通过浏览器的查看菜单单击源文件来看到。



例 3-1 浏览器收到的代码

由于 hello,world 不是 HTML 或 JavaScript 等浏览器能解释的客户端代码，所以它只能被浏览器原样输出到浏览器窗口中。在浏览器窗口中看到的最终结果如图所示。

例 3-1 浏览器执行结果

2.剖析最简单的 PHP 程序

例 3-1 源程序 3-1.php 中：

（1）PHP 语言标记符：<?是 PHP 语言开始标记符，?>是结束标记符，二者之间的部分是 PHP 代码。

注意：

①PHP 语言标记符有多种，<? … ?>是最为精简的一种，为简便起见，本课程采用这种标记符。

②<? … ?>中的 PHP 代码与<? 以及 ?>至少应有一个空格或回车，以便 PHP 语言解释器能正确区分它们。

（2）变量：$a 是 PHP 变量。变量名区分大小写。

变量的表示：一个美元符号跟一个变量名称。

有效的变量名由字母或者下划线开头，后面跟上任意数量的字母，数字，或者下划线。

例如，$a 是一个变量，$A 是不同于$a 的另一个变量。$a、$_a、$_a123_123_a 是合法变量，$123a 不合法。

变量的声明：使用前不需要声明数据类型，其类型由所存储的数据决定，即所用即所声明。

PHP 在变量定义中不需要明示的类型定义；变量类型是根据使用该变量的上下文所决定的。也就是说，如果你把一个字符串值赋给变量 var，var 就成了一个字符串。如果你又把一个整型值赋给 var，那它就成了一个整数。

（3）字符串常量："hello,world"是 PHP 字符串常量。

（4）赋值运算：一个等号“=”表示赋值运算。

（5）语句结束符：分号“；”是语句结束符，这一点也同 C 语言。

（6）关键字：echo 是 PHP 关键字，是 PHP 的内置函数名，可换成 print。许多 PHP 编程者常使用只有 4 个字母的 echo。

注意：

①关键字 echo 等不区分大小写；

②同 C 语言，PHP 是函数化语言，echo 关键字就是 echo()函数的函数名，echo $a 也可写为 echo($a)或 print($a)。

（7）注释：本例采用的是 C/C++语言形式的注释方法，常用于简短的行注释。行注释也可采用 Unix Shell 语言的注释方法，即把“//”换成“#”即可。多行注释的方法与 C 语言同。

总结：例 3-1 的程序，虽然简单，但基本浓缩了 PHP 的语法要素，其形式与 C 语言类似甚至更简单。

3.常用变量处理函数

（1）gettype——获取变量类型

语法格式：string gettype (mixed var)

返回 PHP 变量 var 的类型。

常见的返回的字符串的可能值为：

“boolean”

“integer”

“double”

“string”

“array”

“object”

“resource”

“NULL”

（2）判断变量是否为某种类型

is_int -- 检测变量是否是整数

语法格式：

bool is_int (mixed var)

描述：
如果 var 是 integer 则返回 TRUE，否则返回 FALSE。
注: 若想测试一个变量是否是数字或数字字符串（如表单输入，它们通常为字符串），必须使用 is_numeric()。
其他常用的，判断变量是否为某种类型的函数：
is_bool()
is_float()
is_integer()
is_numeric()
is_string()
is_array()
（3）unset -- 释放给定的变量
语法格式：
void unset (mixed var [, mixed var [, ...]])
描述：
unset() 销毁指定的变量。

```
<?
// 销毁单个变量
unset ($foo);

// 销毁单个数组元素
unset ($bar['quux']);

// 销毁一个以上的变量
unset ($foo1, $foo2, $foo3);
?>
```

unset()：删除数组或数组元素
注: unset() 函数允许取消一个数组中的键名。要注意数组将不会重建索引。

```
<?
$a = array( 1 => 'one', 2 => 'two', 3 => 'three' );
unset( $a[2] );
/* 将产生一个数组，定义为
    $a = array( 1=>'one', 3=>'three');
    而不是
    $a = array( 1 => 'one', 2 => 'three');
*/
$b = array_values($a);
// 现在数组 $b 是 array(0 => 'one', 1 =>'three')
?>
```

（4）empty -- 检查一个变量是否为空
描述
bool empty (mixed var)
如果 var 是非空或非零的值，则 empty() 返回 FALSE。换句话说，""、0、"0"、NULL、FALSE、array()、var $var; 以及没有任何属性的对象都将被认为是空的，如果 var 为空，则返回 TRUE。
除了当变量没有置值时不产生警告之外，empty() 是 (boolean) var 的反义词。参见转换为布尔值获取更多信息。

例子 empty() 与 isset()（检测变量是否设置）的一个简单比较。

```
<?
$var = 0;

// 结果为 true，因为 $var 为空
if (empty($var)) {
    echo '$var is either 0 or not set at all';
}

// 结果为 false，因为 $var 已设置
if (!isset($var)) {
    echo '$var is not set at all';
}
?>
```

（5）var_dump -- 打印变量的相关信息

语法格式：

void var_dump (mixed expression [, mixed expression [, ...]])

描述：

此函数显示关于一个或多个表达式的结构信息，包括表达式的类型与值。数组将递归展开值，通过缩进显示其结构。

var_dump() 与 print_r()常用于程序调适，前者会显示更多的信息。

例子 var_dump() 示例

```
<?
$a = array (1, 2, array ("a", "b", "c"));
var_dump ($a);

/* 输出：
array(3) {
  [0]=>
  int(1)
  [1]=>
  int(2)
  [2]=>
  array(3) {
    [0]=>
    string(1) "a"
    [1]=>
    string(1) "b"
    [2]=>
    string(1) "c"
  }
}

*/

$b = 3.1;
$c = TRUE;
```

```
var_dump($b,$c);

/* 输出：
float(3.1)
bool(true)

*/
?>
```

（6）print_r ——打印关于变量的易于理解的信息。
语法格式：
bool print_r (mixed expression)
描述：
print_r() 显示关于一个变量的易于理解的信息。如果给出的是 string、integer 或 float，将打印变量值本身。如果给出的是 array，将会按照一定格式显示键和元素。object 与数组类似。
记住，print_r() 将把数组的指针移到最后边。使用 reset() 可让指针回到开始处。

```
<?
    $a = array ('a' => 'apple', 'b' => 'banana', 'c' => array ('x','y','z'));
    print_r ($a);
?>
```

上边的代码将输出：

```
Array
(
    [a] => apple
    [b] => banana
    [c] => Array
        (
            [0] => x
            [1] => y
            [2] => z
        )
)
```

4.访问客户端变量的方法

注意：在 B/S 程序中，客户端指浏览器 Browser 端。
从第二章第二节可以了解到：
从 PHP 的角度看，浏览器向服务器传递数据：
    使用表单对象时，$表单对象名=表单变量。
    使用 URL?参数名=参数值时，$参数名=查询字符串（querystring）变量。
一个简单的 HTML 表单
x.htm

```
<form action="x.php" method="POST">
    姓名：<input type="text" name="username"><br>
    电子邮件：<input type="text" name="email"><br/>
```

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

根据特定的设置和个人的喜好，有很多种方法访问客户端变量，这里仅介绍常用的两种方法：

例：访问 POST 表单变量（简称 POST 变量）

x.php

```
<?
//①直接使用客户端变量（学习使用）
//PHP 配置文件中 指令 register_globals = on 时可用。不过
//为提高服务器安全性和提升性能
//自 PHP 4.2.0 起默认为 register_globals = off。
// 在实际应用中，不提倡使用/依赖此种方法。
    echo $username;

// ②通过超全局变量数组引用表单变量的方式（实际应用）
//自 PHP 4.1.0 起可用
    echo $_POST['username'];//若 username 是通过 GET 方法传递的，这里应将$_POST 换成$_GET
    echo $_REQUEST['username'];
?>
```

通过 GET 方法传递：当表单采用 GET 方法，或 username 为查询字符串变量时。

关于第二种，首先要弄清楚变量的传递方法（POST/GET），然后通过相应的超全局变量数组（$_POST/$_GET）来引用，为方便页内使用，可先转成简单变量的形式，如$username=$_POST['username']，然后再使用。

第一种方法不用管客户端传递数据的方法，按名访问即可，较为简单。因实验条件原因（实验室 Win95 下安装的 PHP 版本为 4.0.4），同时考虑到方便学习，本课程采用第一种：直接使用客户端变量的方法。

5.PHP 变量的作用域

按照 PHP 变量的定义方式，PHP 变量的分类和相应的作用域分别为：

（1）客户端变量：主要是表单变量、查询字符串变量等，由客户端编程人员设计、定义、提交的变量。
作用域是一个 PHP 页。

（2）服务器端程序员变量：在 PHP 程序中程序员定义的变量，如例 3-1 中的$a。
作用域是一个 PHP 页。

（3）预定义变量：由 PHP 自己定义好的变量，变量名是固定的，存储在$_SERVER、$_ENV 等部分超全局数组中。作用域是全局。

6.超全局变量数组

自 PHP 4.1.0 起，取得客户端变量的首选方法是通过引用超全局变量数组中的元素。超全局变量数组元素包含了来自 Web 服务器（如果可用），运行环境，和用户输入的数据（客户端变量）。

其中，

（1）存有客户端变量的数组：

$_POST：通过 HTTP POST 方法传递的变量组成的数组。

$_GET：通过 HTTP GET 方法传递的变量组成的数组。

$_COOKIE：通过 HTTP cookies 传递的变量组成的数组。

$_REQUEST：此数组包含 $_GET，$_POST 和 $_COOKIE 中的全部内容。

$_FILES：通过 HTTP POST 方法传递的已上传文件项目组成的数组。
$_SESSION：包含当前脚本中 session 变量的数组。

（2）$GLOBALS：由所有已定义的全局变量组成的数组。变量名就是该数组的索引。
（3）$_SERVER：存储来自 Web 服务器信息的数组，是一个包含诸如头信息（header）、路径（path）和脚本位置（script locations）的数组。数组由 Web 服务器创建。不能保证所有的服务器都能产生所有的信息；服务器可能忽略了一些信息，或者产生了一些新的信息。

常用的$_SERVER 数组元素：
PHP_SELF
当前正在执行脚本的文件名，与 document root 相关。
例如，在 URL 地址为 http://www.sunshoulong.cn/abc/xyz.php 的脚本中使用 $_SERVER['PHP_SELF'] 将会得到/abc/xyz.php 这个结果。__FILE__ 常量包含当前（例如包含）文件的绝对路径和文件名。

使用格式：（注意大小写敏感）

```
echo $PHP_SELF;//当 php.ini 中 register_globals=On 时
echo $_SERVER[“PHP_SELF”]; //当 php.ini 中 register_globals=Off 时
```

以下使用形式同上，仅解释意义：

SERVER_NAME：服务器名字
SERVER_SOFTWARE：Web 服务器软件名称
DOCUMENT_ROOT：发布文档主目录
HTTP_HOST：服务器主机名
SERVER_PORT：服务器 Web 服务端口
以上五个对应 Apache 配置文件中的相应参数

HTTP_USER_AGENT：客户端浏览器信息
REMOTE_ADDR：客户机地址
REMOTE_HOST：客户机主机名
以上三个存有服务器获取的正在访问客户机的信息

相关 PHP 函数：
gethostbyaddr -- 根据客户 IP 得到客户主机名
语法格式：
string gethostbyaddr (string ip_address)
例：

```
<?
$hostname=gethostbyaddr($_SERVER['REMOTE_ADDR']);
echo “你的计算机主机名是：”.$hostname;
?>
```

gethostbyname -- 根据给定的主机名（域名）得到 IP
语法格式：
string gethostbyname (string hostname)
例：

```
<?
$ip = gethostbyname('ilong');
```

```
echo “主机名为 ilong 的计算机的 IP 地址是：”.$ip;     //211.64.40.187
echo “<br>”;
$ip = gethostbyname('www.dzu.edu.cn');
echo   “域名为 www.dzu.edu.cn 的计算机的 IP 地址是：”.$ip;     // 211.64.32.1
?>
```

7.数据类型

仅选择介绍如下几种：

■ 原始类型

★4 种标量类型：

布尔型（boolean）

整型（integer）

浮点型（float）（浮点数，也作“double”）

字符串（string）

★1 种复合类型：

数组（array）

★2 种特殊类型：

资源（resource）

NULL

■ 2 种伪类型：

mixed　mixed 说明一个参数可以接受多种不同的（但并不必须是所有的）类型

number　number 说明一个参数可以是 integer 或者 float。

（1）布尔型（boolean）

这是最简单的类型。boolean 表达了真值，可以为 TRUE 或 FALSE。

语法

指定布尔值：使用关键字 TRUE 或 FALSE。两个都是大小写不敏感的。

```
<?
$a = TRUE; // 指定 TRUE 到 $a
?>
```

布尔值的检测：

```
<?
// 这样检测无必要
if ($a == TRUE) {
   echo "to do something";
}
//只需简单地这样检测
if ($a) {
   echo "to do something";
}
```

真值和假值的情况：

FALSE

布尔值 FALSE

整型值 0（零）

浮点型值 0.0（零）

空白字符串和字符串 "0"

没有成员变量的数组
没有单元的对象
特殊类型 NULL（包括尚未设定的变量）
TRUE
所有其它值都被认为是 TRUE（包括任何资源）
包括-1 和其它非零值（不论正负）
（2）整型（integer）
整型值可以用十进制，十六进制或八进制符号指定，前面可以加上可选的符号（- 或者 +）。
如果用八进制符号，数字前必须加上 0（零），用十六进制符号数字前必须加上 0x。

```
<?
$a = 1234; // 十进制数
$a = -123; // 一个负数
$a = 0123; // 八进制数（等于十进制的 83）
$a = 0x1A; // 十六进制数（等于十进制的 26）
?>
```

（3）浮点型（float）
浮点数（也叫“floats”，“doubles”或“real numbers”）可以用以下任何语法定义：

```
<?
$a = 1.234;
$a = 1.2e3;
$a = 7E-10;
?>
```

（4）字符串型（string）
string 是一系列字符。
常用两种方法定义：单引号、双引号。
单引号
指定一个简单字符串的最简单的方法是用单引号（字符 '）括起来。
要表示一个单引号，需要用反斜线（\）转义；要表示一个反斜线，需要用两个反斜线表示。

```
<?
// 输出结果：Tom said: "I'll be back"
echo 'Tom said: "I\'ll be back"';

//输出结果：Will you want to delete C:\*.*?
echo 'You deleted C:\\*.*?';

//输出结果：将不会把 \n 转义为换行
echo '将不会把 \n 转义为换行';

$a=”abc”;
//输出结果： 变量名$a 也不转义为变量$a 的值 abc
echo '变量名$a 也不转义为变量$a 的值 abc';
?>
```

双引号
如果用双引号（"）括起字符串，PHP 支持更多特殊字符的转义序列：
转义字符

序列	含义
\n	换行

\r	回车
\t	制表符
\\	反斜线
\$	美元符号
\"	双引号
\[0-7]{1,3}	此正则表达式序列匹配一个用八进制符号表示的字符
\x[0-9A-Fa-f]{1,2}	此正则表达式序列匹配一个用十六进制符号表示的字符

注意：试图转义任何其它字符，反斜线本身也会被显示出来。

双引号字符串最重要的一点是其中的变量名会被变量值替代。

```
<?
$a=”abc”;
//输出结果： 变量名 abc 转义为变量 abc 的值 abc
echo '变量名$a 转义为变量$a 的值 abc';
?>
```

变量解析

当用双引号指定字符串时，其中的变量会被解析。

如果遇到$，PHP 会尽可能多地取得后面的字符以组成一个合法的变量名。若要显示地指明变量名，用花括号把变量名括起来。 应明确双引号中变量名，以免发生变量解析错误。

```
<?php
$beer = 'TsingTao';
echo " $beer：中国啤酒名牌"; //出错，这里用的是中文冒号
echo " $beer ： 中国啤酒名牌"; //正常工作，这里用的是中文冒号，但在冒号前加了个空格
echo " $beer: 中国啤酒名牌"; //正常工作，英文冒号: 是无效变量标识符
echo "$beer's taste is great"; //正常工作, 单引号’是无效变量标识符
echo "He drank some $beers";  //出错, 's' 是有效的变量标识符
echo "He drank some ${beer}s"; //正常工作，使用花括号强制指明变量名
echo "He drank some {$beer}s"; //正常工作，使用花括号强制指明变量名
?>
```

（5）数组型(array)

定义数组： array()

可以用 array() 语言结构来新建一个 array。它接受一定数量用逗号分隔的 key => value 参数/值对。

```
array(key_1=>value_1，key_2=>value_2，……)
// key_n 可以是 integer 或者 string
// value_n 可以是任何值
```

例如：

```
<?
$a = array("foo" => "bar", 12 => true);

echo $arr["foo"]; // bar
echo $arr[12];    // 1
?>
```

键名：key，即数组元素的下标

键值：value，即数组元素的值

如果键名是一个 integer 的标准表达方法，则被解释为整数（例如 "8" 将被解释为 8，而 "08" 将被解释为 "08"）。key 中的浮点数被取整为 integer。PHP 中没有不同的数字下标和和关联下标数组，数组的类型只有一种，它可以同时包含整型和字符串型的下标。

注意：如果方括号内没指定键名，则取当前最大整数索引值，新的键名将是该值 + 1。如果当前还没有整数索引，则键名将为 0。如果制定的键名已经有值了，该值将被覆盖。

如：

```
<?
// 有些键没有指定健名的数组
array(5 => 43, 32, 56, "b" => 12);

// 上数组等同于这个数组
array(5 => 43, 6 => 32, 7 => 56, "b" => 12);
?>
```

value 值可以是任何值。

数组元素的修改：

语法格式：

```
$arr[key] = value;
$arr[] = value;
// key 可以是 integer 或者 string
// value 可以为任何值。
```

如果 $arr 还不存在，将会新建一个。这也是一种定义数组的替换方法。要改变一个值，只要给它赋一个新值。

数组元素和数组的删除：

unset(数组名[键名]) 删除数组元素

unset(数组名) 删除整个数组

例：

```
<?
$arr = array(5 => 1, 12 => 2);
$arr[] = 56; // 本行等价于 $arr[13] = 56;
$arr["x"] = 42; // 增加一个新的元素，键名为"x"
unset($arr[5]); // 将键名为 5 的元素删除
unset($arr);      // 将整个数组删除
?>
```

常用数组实用函数、语句结构

有相当多的实用函数作用于数组，更多资料，可参见有关资料中的数组函数库部分。

- count -- 统计数组中的元素的个数

语法格式：

int count (数组名)

例 count() 例子

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

$b[0] = 7;
$b[5] = 9;
$b[10] = 11;
$result = count ($b);
// $result == 3;
```

```
?>
```

• foreach ——控制结构：遍历数组

两种语法格式：

第一种格式

```
foreach (array_expression as $value)
    statement
```

遍历给定的 array_expression 数组。每次循环中，当前单元的值被赋给 $value 并且数组内部的指针向前移一步（因此下一次循环中将会得到下一个单元）。

第二种格式

```
foreach (array_expression as $key => $value)
    statement
```

做同样的事，只除了当前单元的键值也会在每次循环中被赋给变量 $key。

• list——把数组中的值赋给一些变量

语法格式：void list (mixed ...)

list() 用一步操作给一组变量进行赋值。 list() 仅能用于数字索引的数组并假定数字索引从 0 开始。

例： list() 例子

```
<?
$info = array('张三', '男', '25 岁');

// 取出数组$info 中所有元素值，分别赋值到三个变量中
list($name, $sex, $age) = $info;
print "$name 是$sex 性，今年$age 岁\n";

//取出数组$info 中部分元素值，分别赋值到相应变量中
list( , , $age) = $info;
print "张三今年$age 岁\n";
?>
```

（6）资源型

一个资源是一个特殊变量，保存了到外部资源的一个引用。资源是通过专门的函数来建立和使用的。

资源类型变量用于保存打开文件、数据库连接、图形画布区域等的特殊句柄。

以数据库应用为例，在同时有众多的数据库连接存在时，要进行查询和关闭连接等操作，必须指明这些操作是对哪个连接的，所以有必要给每个连接赋予一个标识值，一般是整数。这种标识值的数据类型称为资源型。

具体应用见第四章。

（7）NULL 型

特殊的 NULL 值表示一个变量没有值。NULL 类型唯一可能的值就是 NULL。

在下列情况下一个变量被认为是 NULL：

被赋值为 NULL。

尚未被赋值。

被 unset()。

语法

NULL 类型只有一个值，就是大小写敏感的关键字 NULL。

8.运算符

（1）算术运算符

加（+）、减（-）、乘（*）、除（/）、取模（求余：%）

$a % $b：取模运算，求$a 除以 $b 的余数。

除号（“/”）总是返回浮点数，即使两个运算数是整数（或由字符串转换成的整数）也是这样。
注: 取模 $a % $b 在 $a 为负值时的结果也是负值。
（2）赋值运算符
基本的赋值运算符是“=”。
适合于所有二元算术和字符串运算符的“组和运算符”例如：

```
<?
$a = 3;
$a += 5; // sets $a to 8, as if we had said: $a = $a + 5;
$b = "Hello ";
$b .= "There!"; // sets $b to "Hello There!", just like $b = $b . "There!";
?>
```

（3）错误控制运算符
错误控制运算符：@。当将其放置在一个 PHP 表达式之前，该表达式可能产生的任何错误信息都被忽略掉。
（4）比较运算符
为避免出错，$a 若与$b 类型不同，请先转换成同类型，再比较。
比较运算符：相等（==）、不等（!=或<>）、小于（<）、小于等于（<=）、大于（>）、大于等于（>=）
（5）逻辑运算符
与（and，&&）、或（or，||）、非（！）、异或（xor）
（6）字符串连接操作符
连接运算符（“.”）：
　　如$c=$a.$b，它将$a 和$b 拼接成一个新的字符串$c。
连接赋值运算符（“.=”）：
　　如$a.=$b，它将字符串$b 的内容附加在字符串$a 的后面。

```
<?
$a = "Hello ";
$a.= "World!";        // now $a contains "Hello World!"
?>
```

9.函数

主要分为系统函数、用户自定义函数。
（1）系统函数。PHP 定义的系统函数十分丰富，多达 162 个函数库，用于 162 方面的处理。如上述用于变量检测、数组循环等，分别属于变量处理函数库和数组函数库。用户按照说明使用即可。
常用的是：数组函数库、变量函数库、字符串处理函数库、MySQL 函数库（将在第四章介绍）、时间日期函数库、HTTP 相关函数库、数学函数库。
数组函数库、变量函数库中的常用函数已经在前面介绍，下面介绍其余函数库中常用的函数。
• 字符串处理函数库
①int strlen (字符串名) – 得到字符串的长度
②substr()——截取子串
string substr (string string, int start [, int length])
例. substr()基本用法

```
<?
echo substr('abcdef', 1);        // bcdef
echo substr('abcdef', 1, 3);    // bcd
echo substr('abcdef', 0, 4);    // abcd
echo substr('abcdef', 0, 8);    // abcdef
echo substr('abcdef', -1, 1); // f
```

```
?>
```
如果 start 是负数，将从母串的末尾开始反向截取
```
<?
$rest = substr("abcdef", -1);    // returns "f"
$rest = substr("abcdef", -2);    // returns "ef"
$rest = substr("abcdef", -3, 1); // returns "d"
?>
```
③ord()——取字符的 ASCII 码
int ord (string string)
④str()——取 ASCII 码对应的字符
string chr (int ascii)
⑤trim()——去掉串首串尾的空格
string trim (string str)
⑥ltrim()——去掉串首的空格
string ltrim (string str)
⑦rtrim()——去掉串尾的空格
string rtrim (string str)
⑧explode()——将字符串拆分成数组
array explode (string separator, string string)
此函数返回由字符串组成的数组，每个元素都是 string 的一个子串，它们被字符串 separator 作为边界点分割出来。
如果 separator 为空字符串（""），explode() 将返回 FALSE。如果 separator 所包含的值在 string 中找不到，那么 explode() 将返回包含 string 单个元素的数组。
例. explode() 示例
```
<?
// 示例 1
$pizza  = "piece1 piece2 piece3 piece4 piece5 piece6";
$pieces = explode(" ", $pizza);//注意这里用空格作为分隔符，而不是空字符串
echo $pieces[0]; // piece1
echo $pieces[1]; // piece2
?>
```

⑨implode()——将数组元素联成字符串
string implode (string glue, array pieces)
```
<?
$array = array('lastname', 'email', 'phone');
$comma_separated = implode(",", $array);
echo $comma_separated; // lastname,email,phone
?>
```

- 时间日期函数库

①date()——格式化一个本地时间 / 日期
string date (string format)

表 常用格式字符串(format)

format	说明	返回值例子
Y	4 位数字年份	例如：1999 或 2003
m	2 位数字月份	01 到 12
d	2 位数字，月份中的第几天	01 到 31

format	说明	返回值例子
H	2 位数字小时，24 小时格式	00 到 23
i	2 位数字*分钟*	00 *到* 59
s	2 位数字*秒*	00 *到* 59

```
<?
// 假设现在的服务器时间是: 2001 年 3 月 10 日，5:16:18 pm
$today = date("Ymd");   // 20010310
$time = date("H:i:s"); // 17:16:18
$todaytime1=date(“Ymd，H:i:s”);// 20010310，17:16:18
$todaytime2=date(“Y-m-d，H:i:s”);// 2001-03-10，17:16:18
$todaytime3=date(“Y 年 m 月 d 日，H 时:i 分:s 秒”);// 2001 年 03 月 10 日，17 时:16 分:18 秒
?>
```

与 date()具有类似功能的函数是 getdate()，可供参考使用。

- HTTP 相关函数库

header(string)函数

向浏览器发出头信息。

头信息 (header) 是服务器以 HTTP 协议输出 HTML 到浏览器前所送出的字串，在头信息与 HTML 文件之间尚需空一行分隔。

函数 header()函数需要在输出流中增加头信息，但是头信息只能在其它任何输出内容之前发送。在使用这些函数前不能有任何（如 HTML）的输出。

如果你的 PHP 程序中需要输出 HTML（如要使用 echo 等输出什么东西时 ）前，也需要使用 header()函数，那么，要先用 header()函数输出所有的头信息，否则会出错。

可能会返回的错误消息：

“Warning: Cannot send session cookie - headers already sent...”或者“Cannot add header information - headers already sent...”。

头信息参数 string 的形式：常见的头信息有下面三种之一，并只能出现一次。

Location: URL（掌握）

Content-Type: xxxx/yyyy

Status: nnn xxxxxx

header(“Location:URL”)

作用：服务器直接向浏览器发送一个网络地址为 URL 的页面。

举例：

```
<?
Header("Location: http://www.dzu.edu.cn");
exit;
?>
```

作用类似于 JavaScript 的 window.location=URL，但后者是浏览器向 URL 中的服务器请求这个 URL，该服务器受到这个请求后，将该服务器上地址为 URL 的页面返回给浏览器，整个过程是请求-响应（两段），前者仅响应（一段）。

- 数学函数库

floor -- 向下取整

语法格式：float floor (float value)

返回不大于 value 的下一个整数，将 value 的小数部分舍去取整。floor() 返回的类型仍然是 float，因为 float 值的范围通常比 integer 要大。

例 floor() 例子

```
<?
echo floor(4.3);    // 4
echo floor(9.999); // 9
?>
```

ceil -- 向上取整

语法格式：float ceil (float value)

返回不小于 value 的下一个整数，value 如果有小数部分则进一位。ceil() 返回的类型仍然是 float，因为 float 值的范围通常比 integer 要大。

例 ceil() 例子

```
<?
echo ceil(4.3);      // 5
echo ceil(9.999);   // 10
?>
```

（2）用户自定义函数

定义的语法格式

```
<?
function function_name($arg_1, $arg_2, ..., $arg_n)
{
    echo "Example function.\n";
    return $retval;
}
?>
```

尽量在被调用之前定义

可放在包含文件中，用包含文件即可实现定义。

10.session 的应用

（1）session 的概念

1）使用 session 的意义：

HTTP 是一种无状态会话（请求/响应），没记性的协议。每次会话结束，所有的数据都将不复存在。例如你又从这页转到了别的页，HTTP 也就忘记了你刚才的状态。如何使它记住个别客户（浏览器）的状态，PHP 提供了一种 session 变量，可实现持续状态的会话。

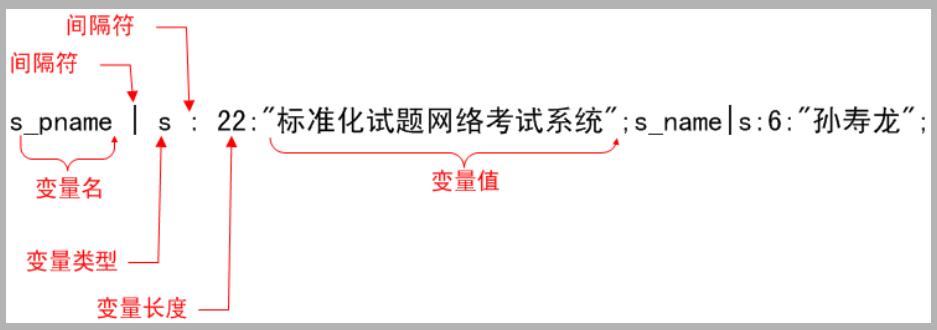
session 变量可以让我们继续使用以前的页面数据，好像服务器已经记住了或者说跟踪了我。因此，可以在 PHP 程序文件之间传递数据（数值、字符串、数组和对象）。当用户在应用程序的页间进行跳转时，session 变量不会释放（在设定的 session 存活期时间内，一般为 180 分钟，可自行在 php.ini 设定 session.cache_expire 的值）。

由此可见，session 变量存储的是个别浏览器端专用的数据。当用户浏览 Web 站点时，使用 session 变量可以为每一个用户保存指定的数据。任何存储在用户 session 变量中的数据可以在用户调用下一个页面时取得。

实际应用中，在身份认证、操作监控、客户消费偏好跟踪等许多需要持续会话的场合，应用广泛。

2）基本原理：

服务器使用唯一的 session 标志码字符串命名的小文件，来存储跟踪客户数据，每个客户一个文件，每个文件名均不相同，每个客户的数据均集中存储在他那个 session 变量文件中。每个客户的 session 变量存储在一个单独的以标志码命名的文本文件中，如 sess_d8c46f13d7d346e53a621bb5e82eeb13（无扩展名，可用记事本打开）。



session 文件的内容：四维数组

从上图可见，有关某一个客户的 session 变量都被集中到了一起，形成了一个四维数组。该数组中每个元素的下标（键名）就是变量名，其余三位分别是该元素的类型、长度、具体的值。

这些文件在服务器设定的目录下。

（2）session 变量的使用

准备工作：

①必须建立或指定一个文件夹供 PHP 存放 session 变量文件。

该文件夹路径、名字任意（一般为 tmp）。

如：d:\tmp 或 d:\abc\pqr\xzy

②要告诉 PHP 这个存放 session 变量文件的路径：在 PHP 配置文件 php.ini 中

设置 session.save_path= d:/tmp 或其他文件夹，注意这里用正斜线/，是 Unix 系统中目录的写法。

在程序中使用：通过专门的函数进行。

一般的顺序和使用的函数如下：

①session_start——初始化 session。

语法: boolean session_start();

本函数初始化一个新的 session，若该客户数据已在 session 之中，则连上原 session。本函数返回值均为 true。

注意：在程序中，该函数的使用必须在任何向浏览器输出之前。

②session_register——注册新的 session 变量

语法: boolean session_register(string name);

本函数增加一个变量到当前客户的 session 变量数组中。参数 name 即为欲加入的变数名。成功则返回 true 值。

③象使用一般 PHP 变量一样使用 session 变量，

如 echo $s_name.",你好!"等等。

④不使用时，基于安全的原因，应及时释放，最简洁的方法顺序如下：

首先，unset 掉每个 session 变量元素

unset($s_name)或 unset ($_SESSION['varname'])

删除名为 s_name 的那个 session 变量。

或说，删除了键名为 s_name 的那个 session 数组中的元素。

本质：销毁变量的内存空间

注意：不要 unset($_SESSION)，这样将会使 $_SESSION 不能发挥作用（）.

然后，destroy 掉 session 文件

session_destroy()——删除 session 文件

语法： session_unset()

本质：删除变量的外部存储空间

例：x.php

```
<?
……
session_start();//初始化 session，在这之前，不要向浏览器输出任何东西
session_register("s_name");//注册新的 session 变量 s_name
$s_name=”孙寿龙”;//给 s_name 赋值
……
Header("Location:y.php");//转到 y.php 页，注意：在此之间不能向浏览器输出任何东西
?>

y.php
<?
……
session_start();//初始化 session，若当前客户 session 已存在，则读取其中的各个变量，以便下文使用
echo “当前用户是：”.$s_name;;//当前用户是：孙寿龙
……
unset($s_name);//销毁该元素所占内存（从内存中删除该元素）
session_destroy();//删除 session 文件
echo “当前用户是：”.$s_name;;//当前用户是：（空）
……
?>
```

第二节 PHP 编程要点

1.流程控制结构

（1）基本结构 if，switch，while，for，break，continue 同 C 语言，详细用法参见有关资料，此处略。
（2）exit 和 die
exit([string message])：输出消息 message 后终止脚本的执行。若省略 message，则什么也不输出就终止了脚本的执行，这时，可以不带括号。
例如：用于调试，查看$a 中的内容是否为空串，是则中断执行，同时输出消息

```
if($a=””)
   exit(‘$a 是空串!’);
```

die 是 exit 的别名，作用同 exit。
注意：exit 和 die 都是用于控制语句流程的关键字，不是函数，因此不具备返回值的能力。
（3）包含文件

```
require(“要包含的文件”)
include(“要包含的文件”)
```

两者具有相同的目的：包括并运行指定文件，类似于 C 程序中包含头文件的作用。
若要包含的文件找不到，include() 产生警告后继续执行，而 require() 则会报错并停止。因此，要想在丢失文件时停止处理页面，那就需要用 require()。
通常使用 require()。
注意
①require()和 include()都是语言结构，用于流程控制的，不是函数。
②由于它们包含进来的是文件（一般是多条代码的集合），为保证主程序的逻辑流程正确，在条件语句中使用必须将其放在语句组中（花括号中）。
例：基本的 require() 例子

```
<?
require(“database_open.php”);//database_open.php 专门用于连接，打开数据库
……
require(“database_close.php”);//database_close.php 专门用于关闭与数据库的连接
?>
```

2. PHP 嵌入 HTML 或 JavaScript 中

使用四种不同的 PHP 语言标记符，嵌入 PHP 代码。
其中两种：<?php …?> 和 <script language="php"> …</script> 总是可用的。
另两种是
短标记：<? …?> 在 php.ini 配置文件的指令 short_open_tag=on 时可用。
ASP 风格标记<% …%>，在 php.ini 配置文件的指令 asp_tags =on 时可用。
例子：
（1）<?php echo '若要服务于 XML 或 XHTML 应用，请采用这种标记符'; ?>
（2）<script language="php">
echo '有些编辑器如 frontpage 不支持这种标记格式';
</script>
（3）<? echo '这是最简练的标记格式'; ?>
（4）<% echo '这是 ASP 风格的标记格式'; %>
注意：

- PHP 代码与 PHP 语言开始标记符和结束标记符之间，至少应有一个空格或回车，以便 PHP 语言解释器能正确区分它们。
- 如果将 PHP 嵌入到 XML 或 XHTML 中则需要使用 <?php …?> 以保持符合标准。
- 在以下情况应避免使用短标记：开发需要发行的程序或者库，或者在用户不能控制的服务器上开发。因为目标服务器可能不支持短标记。为了代码的移植及发行，确保不要使用短标记。
- 凡嵌有 PHP 代码的程序文件，应具有 Apache 能正确识别的 PHP 文件扩展名(在 Apache 配置文件中可查)，以便 Apache 能将其提交给 PHP 去解释执行。
- 本课程采用短标记：<? …?>。

例子：PHP 嵌入 HTML 或 JavaScript 中
用 PHP 程序通过数据库查询成绩，查询结果存放到一个数组$cj 中，这个数组的情况如下：
2 维，即 2 列，分别是：学号，成绩
5 个元素，即 5 行
整个数组表示成二维表格形式是：

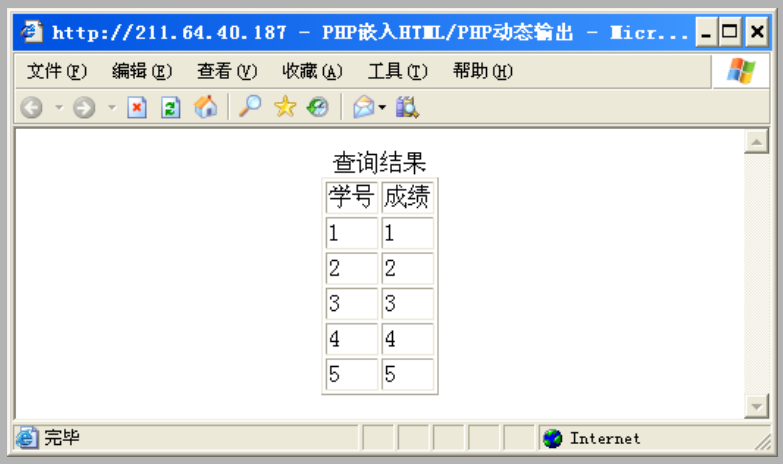
下标	学号	成绩
0	1	1
1	2	2
2	3	3
3	4	4
4	5	5

现在要求输出这个查询结果为网页。
解：采用 PHP 嵌入 HTML 的方法

```
<?
//查询数据库，将查询结果保存到数组$cj
//以上过程代码略
//为演示方便，用下面直接赋值的方式代替以上过程
$cj=array(0=>array("1","1"),
```

```
            1=>array("2","2"),
            2=>array("3","3"),
            3=>array("4","4"),
            4=>array("5","5"),
);
?>
<html>
<head>
<title>hello,world</title>
</head>
<body>
<center>查询结果</center>
<table border=1 align=center>
  <tr><td>学号</td><td>成绩</td></tr>
  <? for($i=0;$i<count($cj);$i++){?>
    <tr><td><? echo $cj[$i][0];?></td><td><? echo $cj[$i][0];?></td></tr>
  <? }?>
</table>
</body>
</html>
```

保存成 PHP 文件，从浏览器请求的结果为：



思考：

（1）数据量的问题。假如从数据库查询的结果不止 5 个呢？例如是 1000 个，若采用手工编写 HTML 代码生成表格的方法，显然很费事。

（2）数据处理频度问题。假如每天有 500 用户查询，且每次查询时间不确定，若采用手工编写 HTML 代码生成表格的方法，显然更费事，费神。

而采用 PHP 程序输出表格，无上述 2 点限制，只要服务器在运行。

3.用 PHP 输出 HTML 或 JavaScript

用 HTML 或 JavaScript 书写的客户端代码都是文本符号，可用 PHP 输出他们。

上例可改成：

```
<?
//查询数据库，将查询结果保存到数组$cj
//以上过程代码略
//为演示方便，用下面直接赋值的方式代替
```

```
$cj=array(0=>array("1","1"),
                1=>array("2","2"),
                2=>array("3","3"),
                3=>array("4","4"),
                4=>array("5","5"),
);
echo "<html>";
echo "<head>";
echo "<title>PHP 嵌入 HTML/PHP 动态输出</title>";
echo "</head>";
echo "<body>";
echo "<center>查询结果</center>";
echo "<table border=1 align=center>";
echo "<tr><td>学号</td><td>成绩</td></tr>";
for($i=0;$i<count($cj);$i++)
   echo "<tr><td>".$cj[$i][0]."</td><td>".$cj[$i][0]."</td></tr>";
echo "</table>";
echo "</body>";
echo "</html>";
?>
```

效果同上例。

再例：输出 JavaScript 的例子

```
<?
//用户验证程序
//将用户提交的帐号和密码
//与数据库中的用户帐号和密码比对
//一致，则令$logon_flag=1
//否则，令$logon=0
//以上过程代码略

if($logon_flag==0)
   {   echo "<script language='javascript'>";
       echo "alert('对不起,帐号或密码错误!');";
       echo "history.back();";
       echo "</script>";
   }
if($logon_flag==1)
   {
       session_start();
       session_register("s_name");
       $s_name=$username;// $username
       //进入系统
      Header("Location:index1.htm");
   }
?>
```

4.自服务程序

B/S 程序多采取客户端程序和服务器程序分离的形式，也可以将二者程序文件合成一个程序。将请求和响应合成的程序，可称自服务程序。

例：

```
<?
if($a=="")
   echo '$a 是空的';
if($a!="")
   echo '$a='.$a;
?>
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=gb2312">
<title>无标题文档</title>
</head>

<body>
<form name="form1" method="post" action="<? echo $PHP_SELF;?>">
   请输入$a：<input name="a" type="text">
   <input type="submit" name="Submit" value="提交">
</form>
</body>
</html>
```

说明：

（1）上例中，实现合成的关键就是通过定义表单的 action 指向本文件来实现。

（2）上例中，头 6 行用于接收处理客户端数据。这些代码可放置到后面的<body></body>中，也可放到该程序的末尾部分。

（3）这种客户端程序和服务器程序二合一的写法，仅适合代码量不大的情况。若合成后的程序代码冗长，还是分开的好，以提高可读性。

第三节 实验二 PHP 语法实验

实验目的：锻炼使用 PHP 编程的基本能力

实验要求：掌握 PHP 基本语法；会进行客户端与服务器的混合编程

实验环境：

实验室已经安装好如下实验环境

操作系统：Windows98

Web 服务器：Apache 1.3.14 安装路径：c:\Apache 默认发布文档路径：c:\Apache\htdocs

PHP：PHP 4.0.4 安装路径：c:\php

脚本编辑器：Macromedia Dreamweaver MX 中文版、记事本或 UltraEdit10 简体中文版

实验内容：

PHP 语法：本章实例程序（见讲义）

实验方法：

（1）编写程序：录入或下载本章实例程序或附加试验程序，保存到发布文档目录

（2）阅读程序：结合讲义等资料，阅读理解这些程序，也可根据自己需要修改这些程序
（3）观摩效果：在浏览器中通过 HTTP 协议方式请求这些程序文件（而不是在我的电脑或资源管理器中打开）
做法：http://localhost/要访问的文件，回车

第四章 MySQL 数据库

第一节 MySQL 的基本使用

1.数据库基础知识

数据库（DataBase）：是现代数据处理的主要技术。
单个数据库可理解为多个表的集合。
数据库的类型：按数据间的关系，数据库可分为关系型、层次型、树状型。最常用的是关系型数据库。
数据库管理系统（DBMS：DataBase Management System）：是种软件，操作数据库的人机接口，对维护数据的安全性、完整性起重要作用。

2.MySQL 简介

MySQL 是一个精巧、快速、多线程、多用户、安全和强壮的 SQL 数据库管理系统。
MySQL 主要目标是快速、健壮和易用。
MySQL 的开发者，T.C.X 公司：自 1996 年以来，我们一直都在使用 MySQL，其中包含超过 40 个数据库，10,000 个表，其中 500 多个表超过 7 百万行，这大约有 100 个吉字节(GB)的关键应用数据。
MySQL 是一个真正的多用户、多线程 SQL 数据库服务器。SQL（结构化查询语言）是世界上最流行的和标准化的数据库语言。MySQL 是以一个客户机/服务器结构的实现，它由一个服务器守护程序 mysqld 和很多不同的客户程序和库组成。
由于它的强大功能、灵活性、丰富的应用编程接口（API）以及精巧的系统结构，受到了广大自由软件爱好者甚至是商业软件用户的青睐，特别是与 Apache 和 PHP 结合，为建立基于数据库的动态网站提供了强大动力。
对 Unix 和 OS/2 平台，MySQL 是免费的；但对微软平台，你在 30 天的试用期后必须获得一个 MySQL 许可证。
对初学者而言，它的易用性更是显而易见。
MySQL 主页提供有关 MySQL 的最新信息。http://www.mysql.com/

3.MySQL 的命令操作

MySQL 数据库管理系统，采用 C/S 结构，由一个服务器程序 mysqld.exe（NT 平台下为 mysqld-nt.exe）和很多不同的客户程序和库组成。
mysql.exe：客户命令监控/解释程序（环境）。
在这些客户程序中，mysql.exe 是最常用的一个。它相当于 MS-DOS 操作系统的命令解释程序 command.com，作用是对用户发出的命令进行语法检查，检查无误，向服务器提交这些命令请求，接受和向用户返回服务器执行的结果反馈信息；检查中若发现用户发出的命令不合法，则拒绝向服务器提交命令，同时返回出错信息。
mysql.exe 是在 MS-DOS 或命令提示符下使用的一种客户端工具，通过命令的方式操纵服务器。

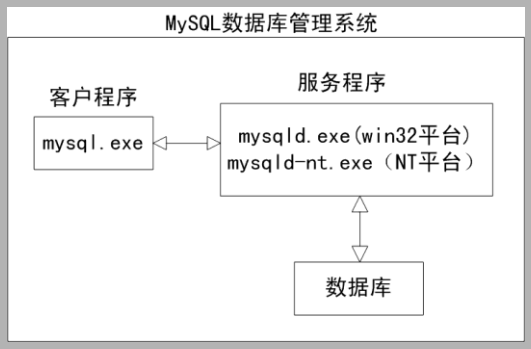


图 MySQL数据库管理系统中的C/S结构

注意：mysql.exe 可以安装在远程主机上，不一定必须和服务程序在一起。

以下说明的前提：安装了 MySQL，启动了服务程序。若这部分工作没有做，参考第一章第三节实验中有关内容现行完成准备工作。

（1）登录和注销

登录：通过客户端程序 mysql.exe 与服务器程序建立信任连接。

建立连接时，需要提供客户机名，用户名，密码等参数，经服务器验证通过后，会返回成功建立连接的信息，表明连接成功。

启动客户端程序 mysql.exe：在命令提示符或 MS-DOS 下，进入 d:\mysql\bin，键入命令 mysql 回车。

若出现类似如下结果:

```
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 26 to server version: 3.23.43
Type 'help;' or '\h' for help. Type '\c' to clear the buffer.
mysql>
```

则表明连接成功。

mysql 命令的完整形式是：

mysql –h servername –u username –p

参数说明：

-h　指明主机，省略此参数，则默认为 localhost

localhost，MySQL 服务器，在服务器看来，它所在的机器就是 localhost，即本地机。

-u　指明用户名（账号）

-p　使用密码

上面键入的命令 mysql 后面不带任何参数，这是因为 MySQL 安装完毕后，系统数据库 mysql 中的权限表 user 中，存在默认的空帐号、无密码的超级用户，因此可以从远程主机使用 mysql.exe 匿名访问服务器主机。

注销：即断开当前与服务器的连接，键入 exit 回车即可。

注意：

“命令+参数；”是命令语句行的一般形式。

在 mysql.exe 环境下，使用的命令不区分大小写。

命令的末尾必须带语句结束符——分号，以便让 mysql.exe 正确区分、解释、执行一个完整的语句。

（2）数据库操作

- 创建数据库

  基本句法：CREATE DATABASE *db_name;*

  db_name 是要创建的数据库的名字。

  该语句执行成功后，会在服务器的 MySQL 数据目录（即 MySQL 安装目录下的 data 目录）下面创建一个名为 db_name 的目录。

- 查看有哪些数据库

  基本句法：SHOW DATABASES;
- 打开数据库（选定数据库）

  基本句法：USE db_name;

  若要对表进行查询，修改，删除等操作，必须先打开数据库。
- 删除数据库

  基本句法：DROP DATABASE [IF EXISTS] db_name;

  DROP DATABASE 删除数据库中的所有表和数据库。要小心地使用这个命令。

  DROP DATABASE 返回从数据库目录被删除的文件的数目。通常，这 3 倍于表的数量，因为每张表对应于一个“.myd”文件、一个“.myi”文件和一个“.frm”文件。

文件	作用
tbl_name.frm	表定义(表格)文件
tbl_name.MYD	数据文件
tbl_name.MYI	索引文件

（3）表操作
- 创建表

  基本句法：CREATE TABLE *table_name*(列 1 定义，列 2 定义，…，列 n 定义);
- 查看有哪些表

  基本句法：SHOW TABLES;
- 查看表结构

  DESCRIBE *table_name*;
- 查看表中数据

  SELECT 列 FROM table_name [WhERE 条件子句] [GROUP 分组子句] [ORDER 条件子句]；
- 修改表中数据

  UPDATE *table_name* SET 列=新值 [WhERE 条件子句
- 删除表中数据

  DELETE FROM *table_name* [WhERE 条件子句

（3）权限操作

进行权限操作的帐户必须有进行此类操作的权限。

GRANT 权限列表 ON *db_name. table_name* TO "*username*"@ "*host*" [IDENTIFIED BY "*password*"] [WITH GRANT OPTION];

MySQL 默认的超级用户帐号有 root、空，且密码都为空。这使得别有用心的人很容易从网络上用 mysql.exe 连接进入，进行破坏活动。

为提高安全性，应在安装完后，迅速更改帐号密码或权限等。

每次更改完毕，必须使用 flush privileges；语句通知服务器启用最新更改的帐号的权限验证用户。

给用户 username（若不存在就新创建一个）从主机 host 以密码 password 访问数据库 db_name 中的表 table_name 的权限（ALL PRIVILEGES、 SELECT、UPDATE、DELETE、DROP、CREATE、ALTER、FILE、GRANT 等）。

赋予 ALL PRIVILEGES 权限的用户，在其作用域（如某数据库之某表：db_name.table_name）内，是超级用户。

权限列表中，权限之间用逗号分隔。

可用符号“*”通配 db_name 或 table_name，表示所有数据库或所有表。

可用符号“%”通配 host，表示除本地机（localhost，服务器所在机器）外的所有主机。

如：

```
GRANT SELECT , INSERT , UPDATE , DELETE ON `test`. * TO "aaa"@"%" IDENTIFIED BY "aaa";
FLUSH PRIVILEGES ;
```

上两句的意思：授予用户 aaa 对数据库 test 内所有表的 SELECT , INSERT , UPDATE , DELETE 权限，允许他使用密码 aaa，能从本地机 localhost 以外的所有主机登录服务器。
GRANT SELECT , INSERT , UPDATE , DELETE ON `test`. * TO "aaa"@"localhost" IDENTIFIED BY "aaa";
FLUSH PRIVILEGES ;
上两句的意思：授予用户 aaa 对数据库 test 内所有表的 SELECT , INSERT , UPDATE , DELETE 权限，允许他使用密码 aaa，只能从本地机 localhost 登录服务器。

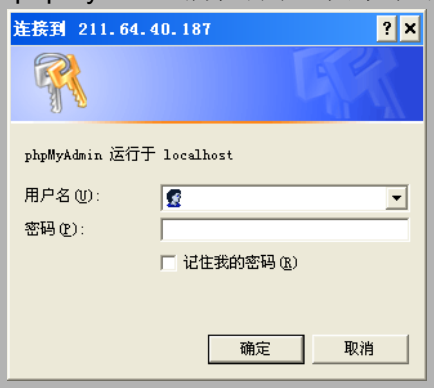
SET PASSWORD FOR "username"@ "host" = PASSWORD("*password*")
以加密形式，更改用户 username 的密码

REVOKE ALL PRIVILEGES ON * . * FROM "username"@ "host";
收回用户 username 从主机 host 访问服务器上所有数据库中所有表的所有权限。

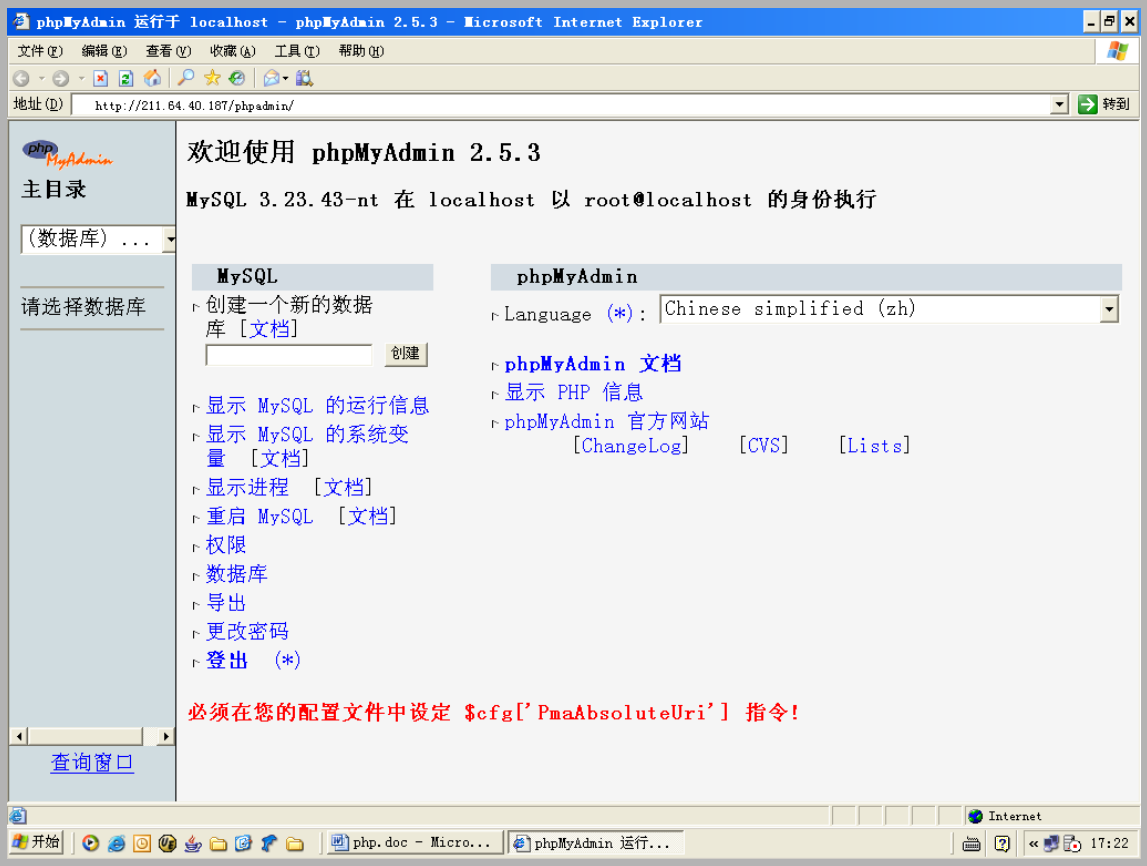
FLUSH PRIVILEGES;
刷新权限列表，通知服务器，启用最新权限，达到使更改后的权限起作用的目的。

4.MySQL 的图形化客户端工具软件

为方便程序员或数据库系统管理员（DBA）对 MySQL 的操作，可采用一些图形化客户端工具软件。常用的有 phpMyAdmin、EMSMySQLManager 等。二者使用方法大同小异，这里只介绍 phpMyAdmin。
phpMyAdmin 是一款使用 PHP 语言编制的基于 Web 使用的 MySQL 客户端工具软件。它功能比较丰富，在广大 MySQL 爱好者中得到了普遍赞誉。
安装
该软件安装包在教学网站提供下载。
将软件包解压，放到服务器的 documentroot 下，如果你设置的默认文档为 index.php，则键入 http://服务器地址/phpMyAdmin 所在目录，回车即可出现 phpMyAdmin 的用户登录界面。



如果初次安装好 MySQL，可采用默认超级用户的帐号和密码登录：
在用户名中输入帐号（如 root），在密码框中输入密码（空，什么也不输入），回车或单击确定按钮，就会进入系统。
进入系统后，在 Language 下拉框中选择适合中国人的系统显示所用的字符集：Chinese Simplified(zh)，然后就会出现中文界面：



单击左边的主目录，界面同上，可完成一些系统级操作，如权限管理，数据库管理，导出数据，更改密码，重新登录（登出）等。

左边主目录下边的数据库下拉框，提供了当前用户所能使用的全部数据库，可选择后进行相关操作。

具体操作，根据中文界面提示，自行学习掌握，这里不再描述。

第二节 用 PHP 操纵 MySQL

前面我们介绍了通过客户端工具，如 mysql.exe 或 phpMyAdmin，这些都是系统管理员或者程序员使用的，方便他们管理数据库或开发工作。但是，在应用系统中，对于普通用户，基于安全的考虑，我们不能允许他们这么做。一般的做法是通过编制程序，让用户通过程序，来操纵数据库，这样可以根据需要灵活地控制用户对数据库的操作：既能满足实现他们存取数据的应用需求，又能最大限度地。本课程中，将介绍使用 PHP 语言编制程序，让用户在网络上，通过这种程序来将他们的数据保存到数据库中，或实现修改、删除等对数据库的操作。

1.PHP 数据类型：资源型

概念：一个资源是一个特殊变量，保存了到外部资源的一个引用。资源类型变量保存有为打开文件、数据库连接、图形画布区域等的特殊句柄，一般用整数来标识。 这个句柄，好像一根管道，将编程语言与其引用的资源如数据库连接起来，二者的交互，在管道中进行，管道或句柄的代号，即一个正整数标识，就是资源型数据。

资源的释放：

在一个 PHP 程序的末尾，资源使用的所有外部资源都会被系统自动释放。如有需要，也可以使用某些释放资源的函数来显式地释放资源所占内存。

资源的建立：

同资源的释放类似，资源是通过专门的函数来建立和使用的。

2.PHP 中用于操纵 MySQL 数据库的函数

PHP 将对 MySQL 数据库的操作，如连接，断开，查询，修改，删除等，都封装成了函数，有些教材中也叫接口。

这些函数属于 PHP 手册中的“MySQL 函数库”，常用的有以下几个：

- **mysql_error**

返回上一个 MySQL 操作产生的文本错误信息。

语法格式：

string mysql_error ([resource link_identifier])

返回上一个 MySQL 函数的错误文本，如果没有出错则返回 ''（空字符串）。如果没有指定连接资源号，则使用上一个成功打开的连接从 MySQL 服务器提取错误信息。

- **mysql_errno**

返回上一个 MySQL 操作中的错误信息的数字代码。

语法格式：

int mysql_errno ([resource link_identifier])

返回上一个 MySQL 函数的错误代码，如果没有出错则返回 0（零）。

注意以上两个函数仅返回最近一次 MySQL 函数的执行（不包括 mysql_error() 和 mysql_errno()）的错误文本或代码，因此如果要使用它们，确保在调用另一个 MySQL 函数之前检查它们的值。

- **mysql_connect**

打开一个到 MySQL 服务器的连接

语法格式：

mysql_connect($servername,$username,$password)

如果成功则返回一个 MySQL 连接标识，一般为资源型数据，失败则返回 FALSE。

函数中参数的解释：

$servername：指明 MySQL 数据库所在的服务器主机名称，可用 IP 表示

$username：访问该服务器主机的帐号名称

$password：访问该服务器的密码

注意：

①不提供参数时使用以下默认值：

$servername=””（相当于$servername= 'localhost'）

$username =””

$password =””

$servername 参数可以包括端口号，如 "servername:port"。

②可以在函数名前加上 @ 来抑制失败时产生的错误信息。

③一旦脚本结束，到服务器的连接就会被关闭，这点与 PHP 每到页末就释放简单变量和客户端变量相同。若要显式（强制）地释放该资源，可用 mysql_close()函数。应养成用完连接，及时释放连接的好习惯。

- **mysql_close**

语法格式：

bool mysql_close ([resource link_identifier])

如果成功则返回 TRUE，失败则返回 FALSE。

mysql_close() 关闭指定的连接标识所关联的到 MySQL 服务器的连接。如果没有指定 link_identifier，则关闭上一个打开的连接。

通常不需要使用 mysql_close()，因为由mysql_connect打开的连接会在脚本执行完毕后自动关闭。但若在脚本中间用完后，提倡使用此函数及时连接资源，以提高效率。

例子 建立和关闭 MySQL 连接例子

```
<?
    $server_link = mysql_connect("localhost", "root", "")//$server_link 是资源型变量
```

```
                    or die("Can not connect: to server" . mysql_error());
    print ("Connected successfully");
    mysql_close($server_link);
?>
```

- **mysql_select_db("test",$server_link)**

选择一个 MySQL 数据库，使其成为当前数据库。一个数据库成为当前数据库，那么当前所有的操作都是针对它的。

语法格式：

bool mysql_select_db (string database_name [, resource link_identifier])

如果成功则返回 TRUE，失败则返回 FALSE。

mysql_select_db() 设定与指定的连接标识符所关联的服务器上的当前数据库。如果没有指定连接标识符，则使用上一个打开的连接。如果没有打开的连接，本函数将无参数调用 mysql_connect() 来尝试打开一个并使用之。

例子 mysql_select_db() 例子

```
<?
$lnk = mysql_connect('localhost', 'mysql_user', 'mysql_password')
        or die("Can not connect: to server");

// 将数据库 abc 设置为当前数据库
mysql_select_db('abc', $server_link) or die ('Can\'t use abc : ' . mysql_error());

?>
```

- **mysql_query**

发送一个 MySQL 查询到当前数据库，由当前数据库执行之。

语法格式：

resource mysql_query (string query [, resource link_identifier])

mysql_query() 向与指定的连接标识符关联的服务器中的当前数据库发送一条查询，由当前数据库执行之。

如果没有指定 link_identifier，则使用上一个打开的连接。如果没有打开的连接，本函数会尝试无参数调用 mysql_connect() 函数来建立一个连接并使用之。

注: 查询字符串不应以分号结束。

查询结果会被缓存。

查询结果：

mysql_query() 仅对 SELECT，SHOW，EXPLAIN 或 DESCRIBE 语句返回一个资源标识符，如果查询执行不正确则返回 FALSE。对于其它类型的 SQL 语句，mysql_query() 在执行成功时返回 TRUE，出错时返回 FALSE。非 FALSE 的返回值意味着查询是合法的并能够被服务器执行。

以下查询语法上有错，因此 mysql_query() 失败并返回 FALSE：

```
<?
$result = mysql_query("SELECT * WHERE 1=1")
    or die("Invalid query: " . mysql_error());
?>
```

以下查询当 my_col 并不是表 my_tbl 中的列时语义上有错，因此 mysql_query() 失败并返回 FALSE：

```
<?
$result = mysql_query("SELECT my_col FROM my_tbl")
    or die("Invalid query: " . mysql_error());
?>
```

如果没有权限访问查询语句中引用的表时，mysql_query() 也会返回 FALSE。

查询结果的查看：
查看 SELECT 语句的查询结果的行数，调用 mysql_num_rows()；
查看 DELETE，INSERT，REPLACE 或 UPDATE 语句影响的行数，调用 mysql_affected_rows()。

仅对 SELECT，SHOW，DESCRIBE 或 EXPLAIN 语句 mysql_query() 才会返回一个新的结果标识符，可以将其传递给 mysql_fetch_array() 和其它处理结果表的函数。

处理完结果集后可以通过调用 mysql_free_result() 来释放与之关联的资源，尽管脚本执行完毕后会自动释放内存。

- **mysql_num_rows**

取得结果集中行的数目
语法格式：
int mysql_num_rows (resource result)
mysql_num_rows() 返回结果集中行的数目。此命令仅对 SELECT 语句有效。要取得被 INSERT，UPDATE 或者 DELETE 查询所影响到的行的数目，用 mysql_affected_rows()。
例子 mysql_num_rows() 例子

```
<?
$server_link = mysql_connect("localhost", "mysql_user", "mysql_password");
mysql_select_db("db1", $link);

$result = mysql_query("SELECT * FROM table1", $server_link);
$num_rows = mysql_num_rows($result);

echo "$num_rows Rows\n";
?>
```

- **mysql_affected_rows**

取得前一次 MySQL 操作所影响的记录行数
语法格式：
int mysql_affected_rows ([resource link_identifier])
取得最近一次与 link_identifier 关联的 INSERT，UPDATE 或 DELETE 查询所影响的记录行数。
例子 mysql_affected_rows() 例子

```
<?
$link = mysql_connect('localhost', 'mysql_user', 'mysql_password');
if (!$link) {
    die('Could not connect: ' . mysql_error());
}
mysql_select_db('mydb');

/* 本例返回被删除记录的准确数目 */
mysql_query('DELETE FROM mytable WHERE id < 10');
```

```
printf("Records deleted: %d\n", mysql_affected_rows());

/* 对于非真值的 WHERE 子句，应返回 0 */
mysql_query('DELETE FROM mytable WHERE 0');
printf("Records deleted: %d\n", mysql_affected_rows());
?>
```

上例的输出类似于：

```
Records deleted: 10
Records deleted: 0
```

例子 使用事务处理的 mysql_affected_rows() 例子

```
<?
$link = mysql_connect('localhost', 'mysql_user', 'mysql_password');
if (!$link) {
    die('Could not connect: ' . mysql_error());
}
mysql_select_db('mydb');

/* Update records */
mysql_query("UPDATE mytable SET used=1 WHERE id < 10");
printf ("Updated records: %d\n", mysql_affected_rows());
mysql_query("COMMIT");
?>
```

上例的输出类似于：

```
Updated Records: 10
```

事务处理：如果使用事务处理（transactions），需要在 INSERT，UPDATE 或 DELETE 查询后调用 mysql_affected_rows() 函数，而不是在 COMMIT 命令之后。

- **mysql_fetch_array**

从结果集中取得一行作为关联数组，或数字数组，或二者兼有。

语法格式：

array mysql_fetch_array (resource result [, int result_type])

返回根据从结果集取得的行生成的数组，如果没有更多行则返回 FALSE。

例子 2. mysql_fetch_array 使用 MYSQL_NUM

```
<?
  mysql_connect("localhost", "mysql_user", "mysql_password") or
       die("Could not connect: to server" . mysql_error());
  mysql_select_db("mydb");
  $result = mysql_query("SELECT id, name FROM mytable");
  while ($row = mysql_fetch_array($result)) {
       printf ("ID: %s   Name: %s", $row[0], $row[1]);
       //或者：printf ("ID: %s   Name: %s", $row[“id”], $row[“name”]);
    }
  mysql_free_result($result);
?>
```

- **mysql_free_result**

释放资源变量所占内存。

语法格式：

bool mysql_free_result (resource result)

mysql_free_result() 将释放所有与结果标识符 result 所关联的内存。

mysql_free_result() 仅需要在考虑到返回很大的结果集时会占用多少内存时调用。在脚本结束后所有关联的内存都会被自动释放。

如果成功则返回 TRUE，失败则返回 FALSE。

3. PHP 操纵 MySQL 实例

（1）准备示例数据

使用 mysql.exe 或 phpMyAdmin：

在 test 数据库中建立表 abc，

表 abc 的结构如下：

列名	类型
a	varchar(10)
b	varchar(10)
c	varchar(10)

在表 abc 中插入示例数据：

列 a 的数据	列 b 中的数据	列 c 的数据
a1	b1	c1
a2	b2	c2
a3	b3	c3

（2）准备示例程序 abc.php，存放在发布文档目录 d:\www 下，其内容为：

```
<?
 //连接数据库
$hostname="";
$username="";
$password="";

//连接服务器
//$server_link 为资源型变量
$server_link=@mysql_connect($hostname,$username,$password) or die ("连接服务器失败!程序中断执行!");
if($server_link) echo "与服务器的连接成功!<br>";

//打开数据库
//$db_link 为资源型变量
$db_link=@mysql_select_db("test",$server_link) or die ("连接数据库失败!程序中断执行!");
//die 是 exit 的别名,它们的作用是强制中断程序执行
//若程序在这里终止执行，则上面的资源型变量所占内存资源将自动回收（释放资源型变量）

//查询表，并将查询结果存入数组（遍历记录集，将记录集中的数据转到数组）
$sql="select * from abc";
//$result 为资源型变量
$result=mysql_query($sql,$server_link);
$i=0;
while($temp_array =mysql_fetch_array($result)) {
   $abc_array[$i][0]=$temp_array ["a"];
```

```
    $abc_array [$i][1]=$temp_array ["b"];
    $abc_array [$i][2]=$temp_array ["c"];
    $i++;
}

//释放记录集,这是显式地关闭资源，若程序终止执行，该关闭将自动进行
mysql_free_result($result);
//释放服务器连接，这也是显式地使用关闭函数回收资源变量
mysql_close($server_link) or die("关闭服务器连接失败");

//将数组中的数据输出到浏览器(以表格形式)
?>
<table border=1>
<tr><td>a 列数据</td><td>b 列数据</td><td>c 列数据</td></tr>
<?
for($i=0;$i<count($abc_array);$i++){
?>
    <tr>
      <td><? echo $abc_array[$i][0];?></td>
      <td><? echo $abc_array[$i][1];?></td>
      <td><? echo $abc_array[$i][2];?></td>
    </tr>
<?
}
?>
```

测试步骤

在浏览器地址栏输入：http://你的机器 IP 地址/mysqltest.php，回车,显示如下图所示，则使用 PHP 从 MySQL 数据库中成功取出了数据，表明 PHP 与 MySQL 能够协同工作了。

4.MySQL 的常用列类型

在设计数据库表时，必须考虑数据类型。MySQL 支持大量的列类型，常用的可分为 3 类：数字类型、日期和时间类型以及字符串(字符)类型。这里简明介绍常用的列类型，更多列类型的详细说明请参考手册。

约定：

M ：指出最大的显示尺寸。最大的合法的显示尺寸是 255 。

D ：适用于浮点类型并且指出跟随在十进制小数点后的数码的数量。最大可能的值是 30，但是应该不大于 M-2。

方括号(“[”和“]”)指明其中的参数可选。

注意，如果带上参数 ZEROFILL，MySQL 将为该列自动地增加 UNSIGNED 属性。

（1）数字类型

TINYINT[(M)] [UNSIGNED] [ZEROFILL]

微整型，一个很小的整数。有符号的范围是-128 到 127，无符号的范围是 0 到 255。 存储时占用一个字节。

INT[(M)] [UNSIGNED] [ZEROFILL]

整型，一个正常大小整数。有符号的范围是-2147483648 到 2147483647，无符号的范围是 0 到 4294967295。

存储时占用 4 个字节。

FLOAT[(M,D)] [ZEROFILL]

实型，一个小(单精密)浮点数字。不能无符号。允许的值是-3.402823466E+38 到-1.175494351E-38，0 和 1.175494351E-38 到 3.402823466E+38。M 是显示宽度而 D 是小数的位数。

存储时占用 4 个字节。

（2）日期时间类型

DATE

一个日期。支持的范围是'1000-01-01'到'9999-12-31'。MySQL 以'YYYY-MM-DD'格式来显示 DATE 值，但是允许你使用字符串或数字把值赋给 DATE 列。

存储时占用 3 个字节。

DATETIME

一个日期和时间组合。支持的范围是'1000-01-01 00:00:00'到'9999-12-31 23:59:59'。MySQL 以'YYYY-MM-DD HH:MM:SS'格式来显示 DATETIME 值，但是允许你使用字符串或数字把值赋给 DATETIME 的列。

存储时占用 8 个字节。

（3）字符类型

CHAR(M)

一个定长字符串，当存储时，总是是用空格填满右边到指定的长度。M 的范围是 1 ～ 255 个字符。当值被检索时，空格尾部被删除。CHAR 值根据缺省字符集以大小写不区分的方式排序和比较，除非给出 BINARY 关键词。CHAR 是 CHARACTER 的一个缩写。

存储时占用 M 个字节。

VARCHAR(M)

一个变长字符串。注意：当值被存储时，尾部的空格被删除。M 的范围是 1 ～ 255 个字符。 VARCHAR 值以大小写不区分的方式排序和比较，除非给出 BINARY 关键词值。VARCHAR 是 CHARACTER VARYING 一个缩写。

存储时占用 L+1 字节, 在此 L <= M 和 1 <= M <= 255

CHAR 或 VARCHAR 列只接受它定义长度内的字符串，超过部分不接收。

CHAR 和 VARCHAR 类型的存储方式不同：

CHAR 列存储所定义的固定长度的字符串。当 CHAR 值被存储时，用空格在右边填补到指定的长度。当 CHAR 值被检索时，这些空格被删去。

与 CHAR 相反，VARCHAR 列只存储所需的字符，外加一个字节记录长度，长度不足的地方不用空格填补。

下表显示了两种类型的列的不同，通过演示存储变长字符串值到 CHAR(4)和 VARCHAR(4)列：

值	CHAR(4)	存储需求	VARCHAR(4)	存储需求
''	' '	4 个字节	''	1 字节
'ab'	'ab '	4 个字节	'ab'	3 个字节
'abcd'	'abcd'	4 个字节	'abcd'	5 个字节
'abcdefgh'	'abcd'	4 个字节	'abcd'	5 个字节

长度小于 4 的 VARCHAR 被改变为 CHAR。

如果在一个表中的任何列有可变长度，结果是整个行是变长的。因此，如果一张表包含任何变长的列

(VARCHAR、TEXT 或 BLOB)，所有大于 3 个字符的 CHAR 列被改变为 VARCHAR 列。这在任何方面都不影响列的使用；在 MySQL 中，VARCHAR 只是存储字符的一个不同方法。MySQL 实施这种改变，是因为它节省空间并且使表操作更快捷。

TINYTEXT

一个 TEXT 列，最大长度为 255(2^8-1)个字符。

TEXT

一个 TEXT 列，最大长度为 65535(2^16-1)个字符。

列类型	需要的存储量
CHAR(M)	M 字节，1 <= M <= 255
VARCHAR(M)	L+1 字节，在此 L <= M 和 1 <= M <= 255
TINYTEXT	L+1 字节，在此 L< 2 ^ 8
TEXT	L+2 字节，在此 L< 2 ^ 16

VARCHAR 和 TEXT 类型是变长类型，对于其存储需求取决于列值的实际长度(在前面的表格中用 L 表示)，而不是取决于类型的最大可能尺寸。例如，一个 VARCHAR(10)列能保存最大长度为 10 个字符的一个字符串，实际的存储需要是字符串的长度(L)，加上 1 个字节以记录字符串的长度。对于字符串'abcd'，L 是 4 而存储要求是 5 个字节。

TEXT 类型需要 1，2，3 或 4 个字节来记录列值的长度，这取决于类型的最大可能长度。

5.用在查询中的运算符和函数

同 PHP 等程序设计语言一样，MySQL 等许多 DBMS 都有自己的数据类型（即列类型），运算符，语句结构，关键字，以及函数。其中广泛用于 SELECT 和 WHERE 子句中的函数，对程序员简化查询语句的构造，提高查询计算的效率，起着非常重要的作用。这些函数将一些复杂的查询计算操作用函数封装起来，由 MySQL 自己执行计算，仅将结果返回给 PHP。

这些函数，涵盖了数学运算，字符串处理，逻辑运算，日期时间运算，比较运算，语句流程控制，分组汇总，排序等许多方面的处理，下面介绍常用到的一些函数。

为简便起见，用->表示执行查询后 MySQL 返回的结果。

（1）强制运算（）

括号。使用它们来强制在一个表达式的计算顺序。

```
mysql> select 1+2*3;
        -> 7
mysql> select (1+2)*3;
        -> 9
```

（2）算术运算

+ 加法

```
mysql> select 3+5;
        -> 8
```

以此类推：

- 减法

* 乘法

/ 除法

被零除产生一个 NULL 结果：

```
mysql> select 102/(1-1);
        -> NULL
```

（3）逻辑运算

所有的逻辑函数返回 1（TRUE）或 0（FALSE）。 NULL 被认为是假值。

NOT （！）逻辑非

OR （||）逻辑或
AND（&&）逻辑与
（4）比较运算符
比较操作得出值 1（TRUE）、0（FALSE）或 NULL 等结果。这些函数工作运用在数字和字符串上。
= 等于
<>不等于
!= 不等于
<= 小于或等于
< 小于
>= 大于或等于
> 大于

```
mysql> select 2 > 2;
        -> 0
```

IS NULL 是否为空
IS NOT NULL 是否不为空

```
mysql> select 1 IS NULL, 0 IS NULL, NULL IS NULL:
        -> 0 0 1
```

expr BETWEEN min AND max
如果 expr 对大于或等于 min 且 expr 是小于或等于 max，BETWEEN 返回 1，否则它返回 0。

```
mysql> select 1 BETWEEN 2 AND 3;
        -> 0
mysql> select 2 BETWEEN 2 AND '3';
        -> 1
```

expr IN (value,...)
如果 expr 是在 IN 表中的任何值，返回 1，否则返回 0。如果所有的值是常数，那么所有的值根据 expr 类型被计算和排序，然后项目的搜索是用二进制的搜索完成。这意味着如果 IN 值表全部由常数组成，IN 是很快的。如果 expr 是一个大小写敏感的字符串表达式，字符串比较以大小写敏感方式执行。

```
mysql> select 2 IN (0,3,5,'wefwf');
        -> 0
mysql> select 'wefwf' IN (0,3,5,'wefwf');
        -> 1
```

expr NOT IN (value,...)
与 NOT (expr IN (value,...))相同。

ISNULL(expr)
如果 expr 是 NULL，ISNULL()返回 1，否则它返回 0。

```
mysql> select ISNULL(1+1);
        -> 0
mysql> select ISNULL(1/0);
        -> 1
```

注意，使用=的 NULL 的值比较总为假！

（5）字符串比较函数
通常，如果在字符串比较中的任何表达式是区分大小写的，比较以大小写敏感的方式执行。
expr LIKE pat tern[ESCAPE 'escape-char']
将 expr 与模式字符串 pattern 进行模式匹配。返回 1（TRUE）或 0（FALSE）。用 LIKE，你可以在模式中使用下列 2 个①%　匹配任何数目的字符，甚至零个字符

②_ 精确匹配一个字符

```
mysql> select 'David!' LIKE 'David_';
        -> 1
mysql> select 'David!' LIKE '%D%v%';
        -> 1
```

为了测试一个通配符的文字实例，用转义字符的加在字符前面。如果你不指定 ESCAPE 字符，假定为“\”：

\% 匹配%字符
_ 匹配_字符

```
mysql> select 'David!' LIKE 'David\_';
        -> 0
mysql> select 'David_' LIKE 'David\_';
        -> 1
```

为了指定一个不同的转义字符，使用 ESCAPE 子句：

```
mysql> select 'David_' LIKE 'David|_' ESCAPE '|';
        -> 1
```

LIKE 允许用在数字的表达式上！（这是 MySQL 对 ANSI SQL LIKE 的一个扩充。)

```
mysql> select 10 LIKE '1%';
        -> 1
```

expr NOT LIKE pattern [ESCAPE 'escape-char']
与 NOT (expr LIKE pattern[ESCAPE 'escape-char'])相同。
（6）控制流函数
IF(expr1,expr2,expr3)
如果 expr1 是 TRUE(expr1<>0 且 expr1<>NULL)，那么 IF()返回 expr2，否则它返回 expr3。IF()返回一个数字或字符串值，取决于它被使用的上下文。

```
mysql> select IF(1>2,2,3);
        -> 3
mysql> select IF(1<2,'yes','no');
        -> 'yes'
mysql> select IF(strcmp('test','test1'),'yes','no');
        -> 'no'
```

expr1 作为整数值被计算，它意味着如果你正在测试浮点或字符串值，应该使用一个比较操作来做。

```
mysql> select IF(0.1,1,0);
        -> 0
mysql> select IF(0.1<>0,1,0);
        -> 1
```

在上面的第一种情况中，IF(0.1)返回 0，因为 0.1 被变换到整数值，导致测试 IF(0)。这可能不是你期望的。在第二种情况中，比较测试原来的浮点值看它是否是非零，比较的结果被用作一个整数。
CASE value WHEN [compare-value] THEN result [WHEN [compare-value] THEN result ...] [ELSE result] END
CASE WHEN [condition] THEN result [WHEN [condition] THEN result ...] [ELSE result] END
第一个版本返回 result，其中 value=compare-value。第二个版本中如果第一个条件为真，返回 result。如果没

有匹配的 result 值，那么结果在 ELSE 后的 result 被返回。如果没有 ELSE 部分，那么 NULL 被返回。

```
mysql> SELECT CASE 1 WHEN 1 THEN "one" WHEN 2 THEN "two" ELSE "more" END;
        -> "one"
mysql> SELECT CASE WHEN 1>0 THEN "true" ELSE "false" END;
        -> "true"
mysql> SELECT CASE BINARY "B" when "a" then 1 when "b" then 2 END;
        -> NULL
```

（7）数学函数

所有的数学函数在一个出错的情况下返回 NULL。

ABS(X)

返回 X 的绝对值。

SIGN(X)

返回参数的符号，为-1、0 或 1，取决于 X 是否是负数、零或正数。

MOD(N,M)

%

模 (类似 C 中的%操作符)。返回 N 被 M 除的余数。

```
mysql> select MOD(29,9);
        -> 2
```

FLOOR(X)

对 X 向下取整。

```
mysql> select FLOOR(1.23);
        -> 1
mysql> select FLOOR(-1.23);
        -> -2
```

CEILING(X)

对 X 向上取整。

```
mysql> select CEILING(1.23);
        -> 2
mysql> select CEILING(-1.23);
        -> -1
```

ROUND(X)

返回参数 X 的四舍五入的一个整数。

```
mysql> select ROUND(-1.23);
        -> -1
mysql> select ROUND(-1.58);
        -> -2
mysql> select ROUND(1.58);
        -> 2
```

ROUND(X,D)

返回参数 X 的四舍五入的有 D 位小数的一个数字。如果 D 为 0，结果将没有小数点或小数部分。

```
mysql> select ROUND(1.298, 1);
        -> 1.3
mysql> select ROUND(1.298, 0);
        -> 1
```

RAND()

RAND(N)

返回在范围 0 到 1.0 内的随机浮点值。如果一个整数参数 N 被指定，它被用作种子值。

```
mysql> select RAND();
```

```
        -> 0.5925
mysql> select RAND(20);
        -> 0.1811
mysql> select RAND(20);
        -> 0.1811
mysql> select RAND();
        -> 0.2079
mysql> select RAND();
        -> 0.7888
```

你不能在一个 ORDER BY 子句用 RAND()值使用列，因为 ORDER BY 将重复计算列多次。然而在 MySQL3.23 中，你可以做： SELECT * FROM table_name ORDER BY RAND()，这是有利于得到一个来自 SELECT * FROM table1,table2 WHERE a=b AND c<d ORDER BY RAND() LIMIT 1000 的集合的随机样本。注意在一个 WHERE 子句里的一个 RAND()将在每次 WHERE 被执行时重新评估。

LEAST(X,Y,...)

有 2 和 2 个以上的参数，返回最小(最小值)的参数。参数使用下列规则进行比较：

如果返回值被使用在一个 INT 上下文，或所有的参数都是整数值，他们作为整数比较。

如果返回值被使用在一个 FLOAT 上下文，或所有的参数是实数值，他们作为实数比较。

如果任何参数是一个大小敏感的字符串，参数作为大小写敏感的字符串被比较。

在其他的情况下，参数作为大小写无关的字符串被比较。

```
mysql> select LEAST(2,0);
        -> 0
mysql> select LEAST(34.0,3.0,5.0,767.0);
        -> 3.0
mysql> select LEAST("B","A","C");
        -> "A"
```

在 MySQL 3.22.5 以前的版本，你可以使用 MIN()而不是 LEAST。

GREATEST(X,Y,...)

返回最大(最大值)的参数。参数使用与 LEAST 一样的规则进行比较。

```
mysql> select GREATEST(2,0);
        -> 2
mysql> select GREATEST(34.0,3.0,5.0,767.0);
        -> 767.0
mysql> select GREATEST("B","A","C");
        -> "C"
```

在 MySQL 在 3.22.5 以前的版本，你能使用 MAX()而不是 GREATEST.

（8）字符串函数

对于针对字符串位置的操作，第一个位置被标记为 1。

MySQL 根据上下文自动变换数字为字符串，并且反过来也如此：

```
mysql> SELECT 1+"1";
        -> 2
```

CONCAT(str1,str2,...)

返回来自于参数连结的字符串。如果任何参数是 NULL，返回 NULL。可以有超过 2 个的参数。一个数字参数被变换为等价的字符串形式。

```
mysql> select CONCAT('My', 'S', 'QL');
        -> 'MySQL'
mysql> select CONCAT('My', NULL, 'QL');
        -> NULL
mysql> select CONCAT(14.3);
```

```
        -> '14.3'
mysql> SELECT CONCAT(2,' test');
        -> '2 test'
```

LEFT(str,len)

返回字符串 str 的最左面 len 个字符。

```
mysql> select LEFT('foobarbar', 5);
        -> 'fooba'
```

RIGHT(str,len)

返回字符串 str 的最右面 len 个字符。

```
mysql> select RIGHT('foobarbar', 4);
        -> 'rbar'
```

SUBSTRING(str,pos,len)

MID(str,pos,len)

从字符串 str 返回一个 len 个字符的子串，从位置 pos 开始。

```
mysql> select SUBSTRING('123456789123456',5,6);
        -> '567891'
```

LTRIM(str)

返回删除了其前置空格的字符串 str。

```
mysql> select LTRIM('   barbar');
        -> 'barbar'
```

RTRIM(str)

返回删除了其尾部空格的字符串 str。

```
mysql> select RTRIM('barbar    ');
        -> 'barbar'
```

TRIM(str)

返回去处了首尾空格的字符串 str。

```
mysql> select TRIM('   bar    ');
        -> 'bar'
```

SPACE(N)

返回由 N 个空格字符组成的一个字符串。

```
mysql> select SPACE(6);
        -> '      '
```

REPEAT(str,count)

返回由重复 countTimes 次的字符串 str 组成的一个字符串。如果 count <= 0，返回一个空字符串。如果 str 或 count 是 NULL，返回 NULL。

```
mysql> select REPEAT('MySQL', 3);
        -> 'MySQLMySQLMySQL'
```

LCASE(str)

LOWER(str)

返回字符串 str，根据当前字符集映射(缺省是 ISO-8859-1 Latin1)把所有的字符改变成小写。

```
mysql> select LCASE('ABCDEFG');
        -> 'abcdefg'
```

UCASE(str)

UPPER(str)

返回字符串 str，根据当前字符集映射(缺省是 ISO-8859-1 Latin1)把所有的字符改变成大写。

```
mysql> select UCASE(' abcdefg ');
        -> ' ABCDEFG '
```

LOAD_FILE(file_name)

读入文件并且作为一个字符串返回文件内容。文件必须在服务器上，你必须指定到文件的完整路径名，而且你必须有 file 权限。文件必须所有内容都是可读的并且小于 max_allowed_packet。如果文件不存在或由于上面原因之一不能被读出，函数返回 NULL。

```
mysql> UPDATE table_name
            SET blob_column=LOAD_FILE("/tmp/picture")
            WHERE id=1;
```

（9）日期和时间函数

NOW()

SYSDATE()

CURRENT_TIMESTAMP

以'YYYY-MM-DD HH:MM:SS'或 YYYYMMDDHHMMSS 格式返回当前的日期和时间，取决于函数是在一个字符串还是在数字的上下文被使用。

```
mysql> select NOW();
        -> '1997-12-15 23:50:26'
mysql> select NOW() + 0;
        -> 19971215235026
```

CURDATE()

CURRENT_DATE

以'YYYY-MM-DD'或 YYYYMMDD 格式返回今天日期值，取决于函数是在一个字符串还是数字上下文被使用。

```
mysql> select CURDATE();
        -> '1997-12-15'
mysql> select CURDATE() + 0;
        -> 19971215
```

CURTIME()

CURRENT_TIME

以'HH:MM:SS'或 HHMMSS 格式返回当前时间值，取决于函数是在一个字符串还是在数字的上下文被使用。

```
mysql> select CURTIME();
        -> '23:50:26'
mysql> select CURTIME() + 0;
        -> 235026
        -> 10
```

（10）分组计算函数

这些函数，常常是与 GROUP BY 子句一起使用的函数，作用是对聚合在组内的行，进行计算。

如果在不包含 GROUP BY 子句的一个语句中使用聚合函数，它等价于聚合所有行。

COUNT(expr)

返回由一个 SELECT 语句检索出来的行的非 NULL 值的数目。

```
mysql> select student.student_name,COUNT(*)
            from student,course
            where student.student_id=course.student_id
            GROUP BY student_name;
```

COUNT(*)在它返回的检索出来的行数目上有些不同，不管他们是否包含 NULL 值。如果 SELECT 从一个表检索，或没有检索出其他列并且没有 WHERE 子句，COUNT(*)被优化以便快速地返回。例如：

```
mysql> select COUNT(*) from student;
```

COUNT(DISTINCT expr,[expr...])

返回一个无重复值的数目。

```
mysql> select COUNT(DISTINCT results) from student;
```

在 MySQL 中，你可以通过给出一个表达式列表以得到不同的表达式组合的数目。

AVG(expr)

返回 expr 的平均值。

```
mysql> select student_name, AVG(test_score)
            from student
            GROUP BY student_name;
```

MIN(expr)

MAX(expr)

返回 expr 的最小或最大值。MIN()和 MAX()可以有一个字符串参数；在这种的情况下，他们返回最小或最大的字符串值。

```
mysql> select student_name, MIN(test_score), MAX(test_score)
            from student
            GROUP BY student_name;
```

SUM(expr)

返回 expr 的和。注意，如果返回的集合没有行，它返回 NULL！

第三节 实验四 MySQL 实验

1.MySQL 基本命令练习

(1)准备数据

在 MS-DOS 或命令提示符下，进入 MySQL 安装目录\bin

键入命令 mysql 回车，连接成功后键入以下简单命令，在 test 数据库中建立测试表 abc，并插入示例数据。

具体步骤和相应命令如下：

①use test;（回车）[说明：打开名为 test 的数据库]

Database changed （MySQL 服务器返回的结果）

②create table abc(a varchar(10),b varchar(10),c varchar(10)); （回车）[说明：在 test 数据库中建立测试表 abc]

Query OK, 0 rows affected (0.05 sec) （MySQL 服务器返回的结果）

③desc abc; （回车）[说明：显示表 abc 的结构]

（下面为 MySQL 服务器返回的结果）

```
+-------+------------+------+-----+---------+---------+---------+
| Field | Type       | Null | Key |  Default      | Extra   |
+-------+------------+------+-----+---------+-------+-----------+
| a     | varchar(10) | YES  |     | NULL    |       |
| b     | varchar(10) | YES  |     | NULL    |       |
| c     | varchar(10) | YES  |     | NULL    |       |
+-------+------------+------+-----+---------+---------+---------+
3 rows in set (0.06 sec)
```

④select * from abc; （回车）[说明：查询表 abc 中的数据]

Empty set (0.11 sec) [说明：表 abc 中无数据]

⑤insert into abc values('a1','b1','c1'); （回车）[说明：在表 abc 中插入第一行数据]

Query OK, 1 row affected (0.05 sec)

⑥insert into abc values('a2','b2','c2'); （回车）[说明：在表 abc 中插入第二行数据]

Query OK, 1 row affected (0.05 sec)

⑦insert into abc values('a3','b3','c3'); （回车）[说明：在表 abc 中插入第三行数据]

Query OK, 1 row affected (0.05 sec)

select * from abc;

（下面为 MySQL 服务器返回的结果表明，已经成功地在 abc 表中插入了三行数据）

```
+------+------+------+
| a    | b    | c    |
+------+------+------+
| a1   | b1   | c1   |
| a2   | b2   | c2   |
| a3   | b3   | c3   |
+------+------+------+
3 rows in set (0.00 sec)
```

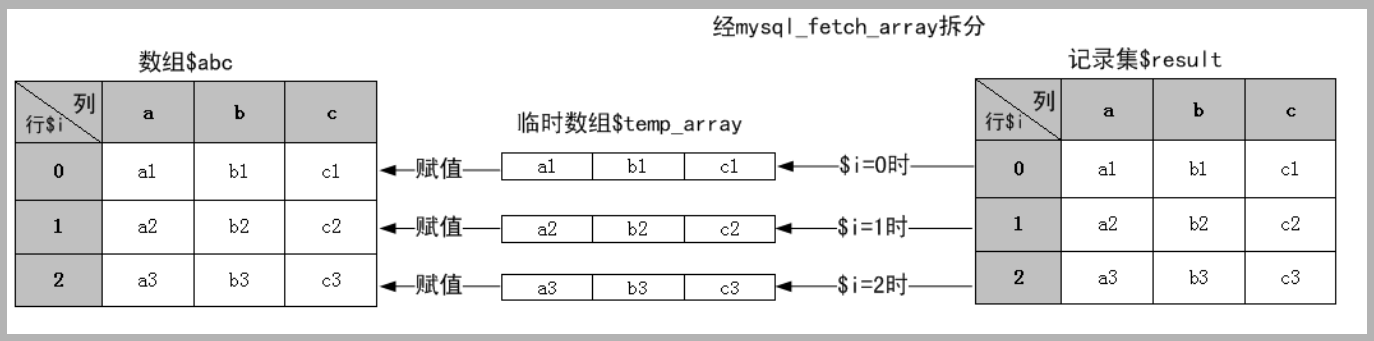
（2）准备示例程序 abc.php，存放在发布文档目录 d:\www 下，其内容为：

```
<?
 //连接数据库
$hostname="";
$username="";
$password="";

//连接服务器
//$server_link 为资源型变量
$server_link=@mysql_connect($hostname,$username,$password) or die ("连接服务器失败!程序中断执行!");
if($server_link) echo "与服务器的连接成功!<br>";

//打开数据库
//$db_link 为资源型变量
$db_link=@mysql_select_db("test",$server_link) or die ("连接数据库失败!程序中断执行!");
//die 是 exit 的别名,它们的作用是强制中断程序执行
//若程序在这里终止执行，则上面的资源型变量所占内存资源将自动回收（释放资源型变量）

//查询表，并将查询结果存入数组（遍历记录集，将记录集中的数据转到数组）
$sql="select * from abc";
//$result 为资源型变量
$result=mysql_query($sql,$server_link);
$i=0;//行计数器
while($temp_array =mysql_fetch_array($result)) {//$temp_array 为临时数组
   $abc_array[$i][0]=$temp_array ["a"];//$abc_array 为记录查询结果的数组
   $abc_array [$i][1]=$temp_array ["b"];
   $abc_array [$i][2]=$temp_array ["c"];
   $i++;
}
//释放记录集,这是显式地关闭资源，若程序终止执行，该关闭将自动进行
mysql_free_result($result);
//释放服务器连接，这也是显式地使用关闭函数回收资源变量
mysql_close($server_link) or die("关闭服务器连接失败");

//将数组$abc_array 中的数据输出到浏览器(以表格形式)
?>
<table border=1>
<tr><td>a 列数据</td><td>b 列数据</td><td>c 列数据</td></tr>
<?
```

```
for($i=0;$i<count($abc_array);$i++){
?>
  <tr>
    <td><? echo $abc_array[$i][0];?></td>
    <td><? echo $abc_array[$i][1];?></td>
    <td><? echo $abc_array[$i][2];?></td>
  </tr>
<?
}
?>
```

程序重点说明

本程序中的 while 循环部分：



测试步骤

在浏览器地址栏输入：http://你的机器 IP 地址/mysqltest.php，回车,显示如下图所示，则使用 PHP 从 MySQL 数据库中成功取出了数据，表明 PHP 与 MySQL 能够协同工作了。

2.使用 phpMyAdmin

安装好 phpMyAdmin（若没安装；具体安装方法见第二节）

通过 phpMyAdmin：

删除数据库 test

建立上面的数据库 test

在数据库 test 中建立上面所述结构的表 abc

在表 abc 中插入上面所述的数据

运行上面的 PHP 程序，查看结果

第五章 Web 软件开发

本章以一个新闻发布系统的实现为实例，介绍使用 PHP，进行 Web 软件开发的一般过程和方法。
本章内容也是前四章内容的综合运用。

第一节 系统分析和系统设计

1.用户需求

经过对用户的调查，并与用户协商，一致确定最终的需求，表述如下：
系统名称：ABC 新闻发布系统
系统功能：
（1）前台功能——最终客户，用于浏览新闻。
1）浏览最新新闻标题：显示 10 条最新新闻的标题；只列出允许发布到前台的新闻标题。
发布：允许让最终客户在前台看到。
2）浏览更多新闻标题：每页显示 25 条新闻标题，总数超过 25 条新闻时，分页显示
3）浏览新闻全文：单击 1）、2）中新闻标题后可查看到新闻的具体内容：标题，作者，发布时间（年月日时分秒），内容
（2）后台管理——新闻管理人员，用于管理新闻
1）人员管理：
人员——使用本后台进行新闻管理的人。
人员信息——帐号，密码，姓名，联系方式，权限
其中，权限：登录系统，增加人员，修改人员，添加新稿，修改稿件，审核稿件，业务查询。
功能：
①人员验证：
任何人员进入后台管理系统，必须经过验证，验证的凭据是帐号和密码，必须与系统中的用户名与密码一致。
②人员信息的增加
③人员信息的修改
2）稿件管理：
编辑稿件：
稿件录入：录入新闻稿件信息。新增稿件录完成，提交至服务器，等待审核通过后发布。
稿件修改：对未发布或暂存的新闻稿件，可修改。修改稿提交前，必须再次检查是否已经被发布，若已发布，则服务器拒绝接受提交。
稿件删除：对未发布的稿件，如暂存稿件，可进行删除操作。
审核稿件：对提交待发的新闻稿件，决定发布还是退回。
说明：稿件的状态，有：提交待发，暂存，发布，退回 4 种。
3）业务记录
8 种业务操作：登录系统，增加人员，修改人员，添加新稿，修改稿件，审核稿件，业务查询，退出后台，这均要自动记录到系统中去，记录的信息包括：
姓名，操作机器，操作时间，业务操作，操作稿件
4）业务查询
根据用户输入的查询信息，查询出符合条件的业务记录。

2.系统分析与设计

分析以上用户需求，设计形成便于技术人员理解的专业技术描述。
本例主要从数据和功能两个方面来分析和设计一个系统。
从技术人员的角度来说，就是从数据库和程序两个方面，来分析和设计一个系统。
（1）数据库的分析与设计
1）分析用户数据，形成概念设计，转化为关系模式，从而完成逻辑设计
方法：采用实体-联系（E-R）方法
两个基本实体：新闻和人员
分别转化为关系
关系一：新闻（标题，时间，作者，状态，内容）
关系二：人员（帐号，密码，姓名，联系方式，权限）
关系三：操作记录（姓名，操作机器，操作时间，操作类型，操作稿件）
以上三个关系，是用户需求的原型表述，还不够规范，不便于直接应用于系统的设计。

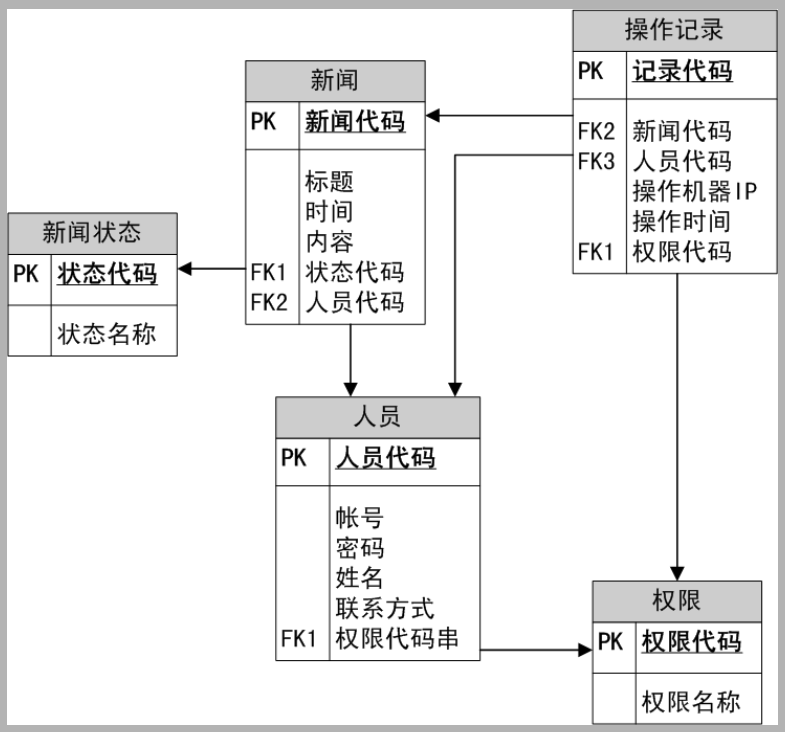
为此，下面对上面的关系，根据规范化理论，采用模式分解方法，进行规范化：
关系一：新闻（标题，时间，作者，状态，内容）
其中，状态属性值∈{提交待发，暂存，发布，退回}
规范化后，分解为两个关系：
①新闻（新闻代码，标题，时间，人员代码，状态代码，内容）
②新闻状态（状态代码，状态名称）

状态代码	状态名称
1	提交待发
2	暂存
3	发布
4	退回

关系二：人员（帐号，密码，姓名，联系方式，权限）
其中，权限属性值∈{登录，增加人员，修改人员，添加稿件，修改稿件，审核稿件}
规范化后，分解为两个关系：
③权限（权限代码，权限名称）

权限代码	权限名称
1	登录系统
2	增加人员
3	修改人员
4	添加稿件
5	修改稿件
6	审核稿件
7	业务查询

④人员（人员代码，帐号，密码，姓名，联系方式，权限代码串）
其中，权限代码串是权限代码的组合，表示权限组合。组合中，每种权限代码只出现一次。
关系三：操作记录（姓名，操作机器，操作时间，操作类型，操作稿件）
其中，每次仅限一种操作，这样，操作类型信息可用权限信息来代替，故关系规范化后，分解为两个关系：
⑤操作记录（记录代码，人员代码，操作机器 IP，操作时间，新闻代码，权限代码）
③权限（权限代码，权限名称）
上述关系①②③④⑤之间的联系，如下图所示：



上图采用 Microsoft Viso 画成。

箭头指向表示外键（FK：Foreign Key）参照（参考解释的意思）。

PK：Primary Key，主键。

2）数据库的物理设计与实现

包括字段，表，数据库名称，用户和权限等内容的方案设计。

命名约定：字段，表，数据库的命名采用汉语拼音首字母。

方案设计的依据：概念设计和逻辑设计的成果：上述 5 个关系模式。

方案设计的步骤：自底向上，具体如下：

第一步：字段设计——确定所有字段的名称、类型和宽度。

第二步：表设计——确定数据表的名称，主键等。

第三步：确定数据库名称。

第四步：确定用户和权限方案。

第五步：实现——在具体的 DBMS 中实现上面四步的设计方案。

下面逐一进行：

■ 第一步：字段设计

方法：

确定哪些属性参加字段设计：把所有关系中的属性集中列表，重复的字段只列一次，表中最终的这些属性需要设计其相应的字段。把它们集中到一起进行设计的目的是：避免重复设计。

字段的命名：将每个属性转化成字段。根据命名约定确定字段的名称，

字段的类型和宽度：根据实际需要确定类型和宽度，在满足应用需求的前提下，尽量减少存储宽度。

字段设计结果：字段名称、类型和宽度

序号	属性名	字段名	类型	宽度（西文字符）
1	新闻代码	xwdm	整型，无符号，自动编号	9
2	标题	bt	变长字符型	50
3	时间	sj	日期时间型	（DBMS 自定）
4	人员代码	rydm	微整型，无符号，自动编号	2
5	状态代码	ztdm	微整型，无符号，自动编号	1
6	内容	nr	文本型	（DBMS 自定）
7	状态名称	ztmc	变长字符型	8
8	帐号	zh	变长字符型	16
9	密码	mm	变长字符型	16

10	姓名	xm	变长字符型	8
11	联系方式	lxfs	变长字符型	20
12	权限代码串	qxdmc	变长字符型	7
13	权限代码	qxdm	微整型，无符号，自动编号	1
14	权限名称	qxmc	变长字符型	8
15	记录代码	jldm	整型，无符号，自动编号	9
16	操作机器 IP	czjq_ip	变长字符型	15
17	操作时间	czsj	日期时间型	（DBMS 自定）

■ 第二步：表设计——确定数据表的名称，表中包括的字段，字段中的主键、外键等

方法：

确定表名：遵照命名约定，将关系名转化为表名。

确定表中的字段：根据原关系中的属性，选择相应的字段。

确定主键和外键。

表的设计

序号	关系名	表名	表中字段	主键	外键
1	新闻	xw	xwdm,bt,sj,rydm,ztdm,nr	xwdm	rydm,ztdm
2	新闻状态	xwzt	ztdm,ztmc	ztdm	
3	人员	ry	rydm,zh,mm,xm,lxfs,qxdmc	rydm	qxdmc
4	权限	qx	qxdm,qxmc	qxdm	
5	操作记录	czjl	jldm,rydm,czjq_ip,czsj,xwdm,qxdm	jldm	rydm,xwdm,qxdm

■ 第三步：确定数据库名称

xwxt（新闻系统）

■ 第四步：确定用户和权限方案

用户和权限设计

序号	用户帐号	用户密码	登录主机	用户权限	说明
1	root	自行设定	自行设定	全局全部权限	超级用户
2	user_xwxt	user_xwxt	localhost %	仅对数据库 xwxt 内所有表具有 SELECT，UPDATE，DELETE，INSERT 权限	受限用户，程序访问数据库用

localhost：服务器所在机器。

%：除 localhost 以外的任何一台机器。

■ 第五步：实现——在具体的 DBMS 中实现上面四步的设计方案

选定的 DBMS：我们选择 MySQL 为实现上述物理方案设计的 DBMS。

以超级用户身份，通过 phpMyAdmin，快速实现上述四步设定的方案。

具体步骤：自上而下

第一步：创建数据库 xwxt

第二步：依次创建 5 个表，完成每个表中的字段设计，主键、外键设计。

注意：凡是外键，不必设计成自动编号型（auto_increment）。

第三步：创建用户 user_xwxt，设定其密码为：user_xwxt，设定其权限为：对数据库 xwxt 内所有表具有全部权限，无全局权限。关于超级用户 root，若已存在，则不必创建。

这三步的实现，用 MySQL 的脚本命令描述如下：

```
CREATE DATABASE `xwxt`;
USE `xwxt`;
CREATE TABLE `xw` (
`xwdm` INT( 9 ) UNSIGNED NOT NULL AUTO_INCREMENT ,
`bt` VARCHAR( 50 ) NOT NULL ,
`sj` DATETIME NOT NULL ,
`rydm` TINYINT( 2 ) UNSIGNED NOT NULL ,
`ztdm` TINYINT( 1 ) UNSIGNED NOT NULL ,
```

```
`nr` TEXT NOT NULL ,
PRIMARY KEY ( `xwdm` )
);
CREATE TABLE `xwzt` (
`ztdm` TINYINT( 1 ) UNSIGNED NOT NULL AUTO_INCREMENT ,
`ztmc` VARCHAR( 8 ) NOT NULL ,
PRIMARY KEY ( `ztdm` )
);
CREATE TABLE `ry` (
`rydm` TINYINT( 2 ) UNSIGNED NOT NULL AUTO_INCREMENT ,
`zh` VARCHAR( 16 ) NOT NULL ,
`mm` VARCHAR( 16 ) NOT NULL ,
`xm` VARCHAR( 8 ) NOT NULL ,
`lxfs` VARCHAR( 20 ) NOT NULL ,
`qxdmc` VARCHAR( 7) NOT NULL ,
PRIMARY KEY ( `rydm` )
);
CREATE TABLE `qx` (
`qxdm` TINYINT( 1 ) UNSIGNED NOT NULL AUTO_INCREMENT ,
`qxmc` VARCHAR( 8 ) NOT NULL ,
PRIMARY KEY ( `qxdm` )
);
CREATE TABLE `czjl` (
`jldm` INT( 9) UNSIGNED NOT NULL AUTO_INCREMENT ,
`rydm` TINYINT( 2 ) UNSIGNED NOT NULL ,
`czjq_ip` VARCHAR( 15 ) NOT NULL ,
`czsj` DATETIME NOT NULL ,
`xwdm` INT( 9 ) UNSIGNED NOT NULL ,
`qxdm` TINYINT(1 ) UNSIGNED NOT NULL ,
PRIMARY KEY ( `jldm` )
);
GRANT SELECT , INSERT , UPDATE , DELETE ON `xwxt`. * TO "user_xwxt"@"%" IDENTIFIED BY "user_xwxt";
FLUSH PRIVILEGES ;
GRANT SELECT , INSERT , UPDATE , DELETE ON `xwxt`. * TO "user_xwxt"@"localhost" IDENTIFIED BY "user_xwxt";
FLUSH PRIVILEGES ;
```

将这些脚本命令存成.脚本文件(默认是.sql 文件，也可以是.txt 文件)，使用客户端工具 mysql.exe 或 phpMyAdmin 执行脚本文件，可在 MySQL 中迅速完成物理设计的实现。

（2）功能的分析与设计

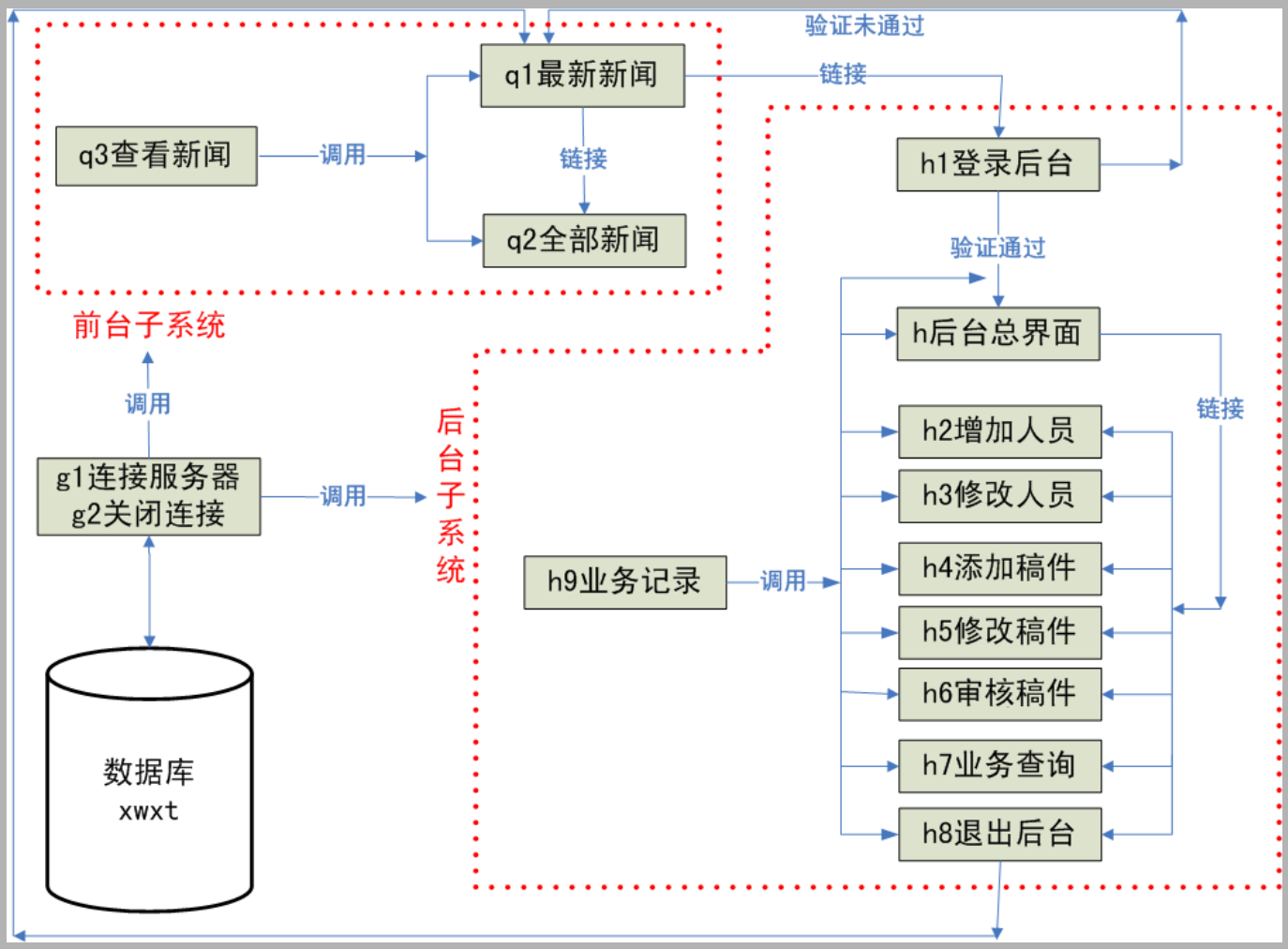
1）功能模块设计

确定功能模块的程序组成，及模块间调用关系。

序号	模块编号	功能模块名称	程序名
1	q1	最新新闻	q1.php 浏览最新新闻标题，前台首页
2	q2	所有新闻	q2.php 浏览所有新闻标题
3	q3	查看新闻	q3.php 浏览某条新闻全文，是 q1 和 q2 的公共调用模块
4	h1	登录后台	h1.php 登录信息输入界面→h1_chuli.php 登录信息验证程序
5	h	后台首页	h.php，链接 h2，h3，h4，h5，h6，

6	h2	增加人员	h2.php 新人信息输入界面→h2_chuli.php 新人信息处理程序
7	h3	修改人员	h3.php 人员修改信息输入界面→h3_chuli.php 人员修改处理程序
8	h4	添加稿件	h4.php 添加稿件界面→h4_chuli.php 添加稿件处理程序
9	h5	修改稿件	h5.php 修改稿件界面→h5_chuli.php 修改稿件处理程序
10	h6	审核稿件	h6.php 审核稿件界面→h6_chuli.php 审核稿件处理程序
11	h7	业务查询	h8.php 查询信息录入界面→h8_chuli.php 查询程序
12	h8	退出后台	h9.php 销毁所有当前操作人员信息，安全退出后台，回到前台
13	h9	业务记录	h9.php 业务记录，是 h1_chuli.php，h2_chuli.php，h3_chuli.php，h4_chuli.php，h5_chuli.php，h6_chuli.php，h7_chuli.php，h8.php 的公共调用模块。该模块无界面。
14	g1	连接服务器	g1.php
15	g2	关闭连接	g2.php

2）模块间关系图示



3）文件部署

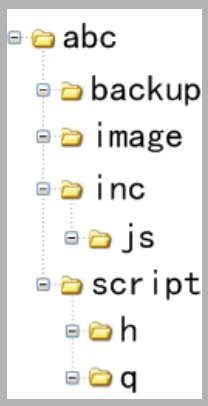
将程序文件、图片文件、SQL 脚本文件等，建立各自的文件夹，分门别类地存放于 Web 发布文档目录下。

文件夹的名称按照见名知义的原则命名。

文件夹命名的一般做法：

文件夹	用途
backup	备份资料
image	存放网页、程序引用的图片文件
inc	存放公用程序、自定义函数等
inc/js	存放公用 JavaScript 程序脚本文件
script	存放程序脚本文件
script/q	存放所有前台程序
script/h	存放所有后台程序

设计结果如图所示：（abc 为发布文档根目录）



4）程序设计

根据程序功能设计的要求，以及程序之间的调用关系，对程序进行具体设计。

■ g1.php 连接服务器

本程序供其它程序调用，无界面。

本程序算法比较简单，直接用编码表示为：

```
<?
$hostname="";//相当于 localhost
$username="user_xwxt";
$password="user_xwxt";
$server_link=@mysql_connect($hostname,$username,$password) or die ("连接服务器失败");
$db_link=@mysql_select_db("xwxt",$server_link) or die ("连接数据库失败");
?>
```

■ g2.php 关闭与服务器的连接

本程序供其它程序调用，无界面。

本程序算法比较简单，直接用编码表示为：

```
<?
$db_close=@mysql_close($server_link) or die("关闭与服务器的连接失败");
?>
```

■ q1.php 最新新闻：

①界面设计

其中：

标题链接指向 q3.php?xwdm=当前标题对应的新闻代码 xwdm 值，打开链接的窗口是新窗口

“全部新闻”链接到 q2.php

“登录后台”链接到 h1.php

②动态构造新闻标题列表的算法

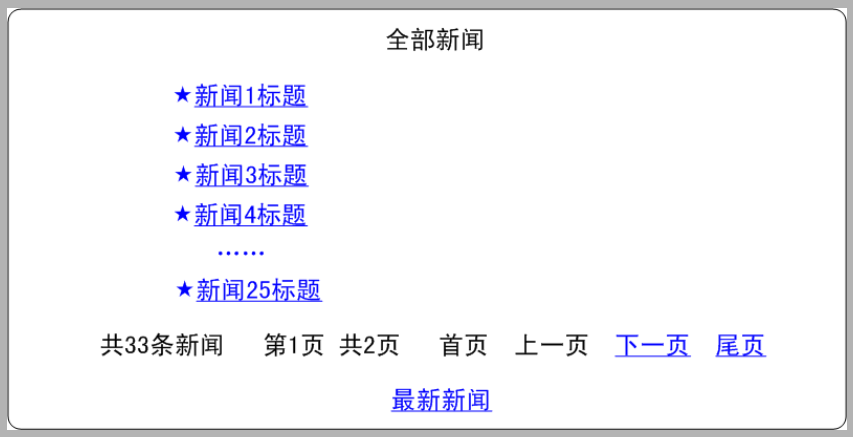
调用 g1.php，连接数据库

查询已发布的 10 条新闻的 xwdm，bt，sj，按 sj 倒序排序

将查询结果装入数组
调用 g2.php，关闭连接
遍历数组，以表格形式输出标题
　　标题链接指向 q3.php?xwdm=数组当前行存有 xwdm 的元素的值，打开链接的窗口是新窗口

■　q2 所有新闻：

①界面设计



标题链接指向 q3.php?xwdm=当前标题对应的新闻代码 xwdm 值，打开链接的窗口是新窗口
“最新新闻”链接到 q1.php
首页，上一页，下一页，尾页——的链接由程序根据当前页位置决定
②算法表述
约定：
用变量$page 存储当前页码：$page 为空则令$page=0，表示当前为第 1 页；$page=n 则表示当前为第(n+1)页。
用变量$pagesize 存储每页显示的记录数：本例中规定$pagesize=25。

$page 为空则令$page=0
调用 g1.php，连接数据库

取得属于当前页中的新闻代码 xwdm，标题 bt：
　　查询已发布新闻的 xwdm，bt，sj，按 sj 倒序排序，从结果集中的第$page*$pagesize 行开始取，最多取$pagesize 行。
　　将取得的这些行中的 xwdm，bt 装入数组$xw。

取得已发布的新闻记录总数$total_records。
调用 g2.php，关闭连接

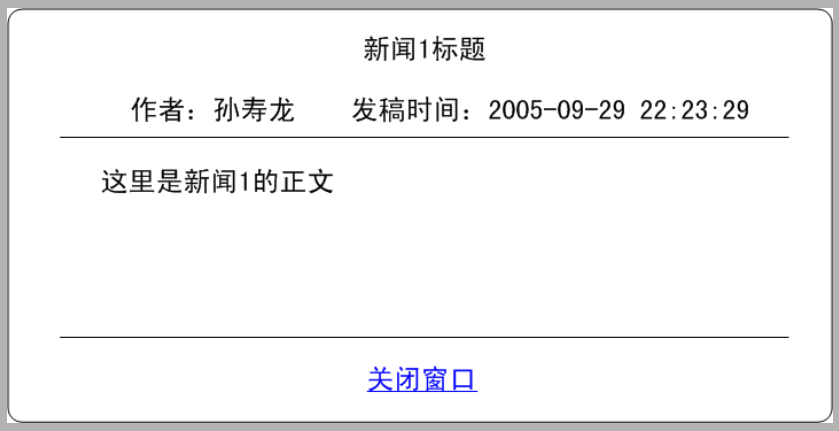
计算总页数$page_count：
　　若$total_records==0，则$page_count 为 0；
　　若$total_records<$pagesize，则$page_count=1；
　　若$total_records%$pagesize>0，则$page_count=向上取整($total_records/$pagesize)；
　　若$total_records%$pagesize==0，则$page_count= $total_records/$pagesize。
构造导航条信息$pageinfo（界面中倒数第二行）：
　$pageinfo=”共”. $total_records.”条新闻　第”.($page+1).”页　共”.$page_count.”页”
　　下面确定：“首页”，“上一页”，“下一页”，“尾页”的链接的情况
　　若有链接，链接到什么地方，同时传递什么样的参数及参数值：
　　链接到本页即$PHP_SELF，同时传递参数 page，其值根据所链接的文字而定：
　　参数 page=0，当链接文字为“首页”（也可省略传递参数）

参数 page=$page-1，当链接文字为“上一页”
参数 page=$page+1，当链接文字为“下一页”
参数 page=$page_count-1，当链接文字为“尾页”
下面确定何时应具有链接：
当$page==0 时，当前页是首页，故“首页”，“上一页”不应具有链接
当$page>0 时，当前页不是首页，故“首页”，“上一页”应具有链接
当$page<$page_count-1 时，当前页未到达尾页，故“下一页”，“尾页”应具有链接
当$page==$page_count-1 时，当前页已是尾页，故“下一页”，“尾页”不应具有链接
$pageinfo=$pageinfo.根据上述规则动态确定链接的首页，上一页，下一页，尾页

上述计算工作完成后，下面将计算结果输出到界面：
遍历数组$xw，以表格形式输出其中的标题 bt
标题链接指向 q3.php?xwdm=数组$xw 当前行存有 xwdm 的元素的值，打开链接的窗口是新窗口
输出导航条信息$pageinfo

■ q3 查看新闻：

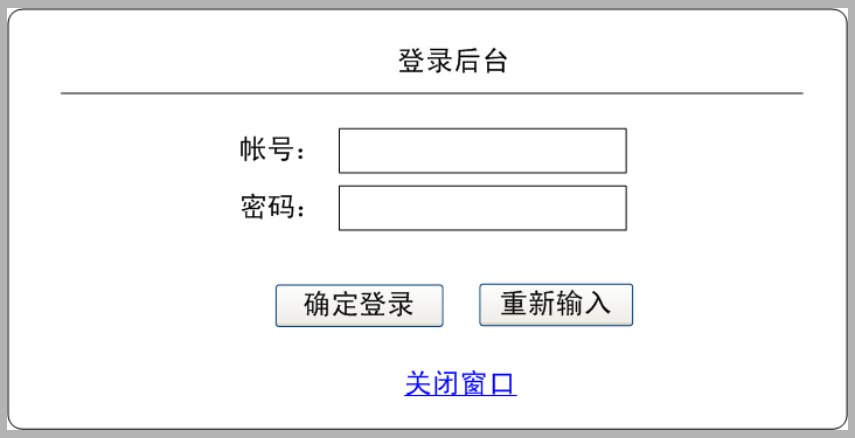
①界面设计



②算法表述
算法的目标是输出界面上的动态信息：
根据接收到的新闻代码，查出该条新闻的标题，作者，时间，内容信息
将查出的这些信息输出到界面的相应位置

■ h1 登录后台：

h1.php 登录信息输入界面设计：



其中，第一个文本框名为 zh，第二个名为 mm；第一个按钮是提交表单型按钮，第二个是重置表单型按钮。
表单动作指向 h1_chuli.php，传递数据的方法是 POST
算法表述：
算法的目标是，提交表单时，检查帐号和密码，若有一个为空则不允许提交。
用客户端语言 JavaScript 实现

h1_chuli.php 登录信息验证程序：
算法表述：
在人员表 ry 中查找与接收到的帐号和密码匹配的记录
找到，则
    检查是否有登录权限，有，则
        跟踪记录该用户的人员代码 rydm，姓名，权限
        调用业务记录模块 h9，记录当前用户的登录操作
        转到台总界面模块 h
    否则
        告诉用户权限不足，退出系统，返回到登录界面
否则，告诉用户帐号或密码错误，返回登录界面

（其他程序的分析和算法设计，读者可按照上述思路，自行完成，此处略）

第二节 系统实施和系统测试

1.编码
略
2.测试
略

第三节 实验

学生在教师指导下，参考第一、二节，自行完成 ABC 新闻系统的分析，设计，实现，测试。

复习指导

复习的思路：本课程的复习，对 HTML、JavaScript、PHP、MySQL，围绕数据处理这个中心目的，理解基本原理，掌握基本的，常用的知识和技能即可。

■ HTML 部分：

HTML 常用标记的使用
重点掌握表单的构造

■ JavaScript 部分：

如何标记 JavaScript 语言
如何编写 JavaScript 函数
在什么地方放置 JavaScript 函数
如果 JavaScript 函数过长，如何放到外部文件，对这种外部文件如何引用
怎样调用 JavaScript 函数——在事件处理中
常用事件处理：onclick、onsubmit（例子见前期试卷）等
常用 DOM 对象及其属性、方法、事件的使用
JavaScript 的经典应用：表单数据检查（例子见前期试卷）

■ PHP 部分：

PHP 的语法：语句格式
变量的使用：形式，声明，大小写
数据类型：字符串型
运算符
流程控制：if、while、for、PHP 语言的嵌入、文件包含
session 变量的使用
常用函数：字串处理、时间日期处理、session 处理、header

■ MySQL 部分：

资源型变量、数据
PHP 中操纵 MySQL 的函数
MySQL 常用的列类型
用在 MySQL 查询语句中的常用函数
用手工操作 MySQL——通过客户端工具 mysql.exe 和 phpMyAdmin
用 PHP 程序实现自动化操作 MySQL——通过 PHP 程序