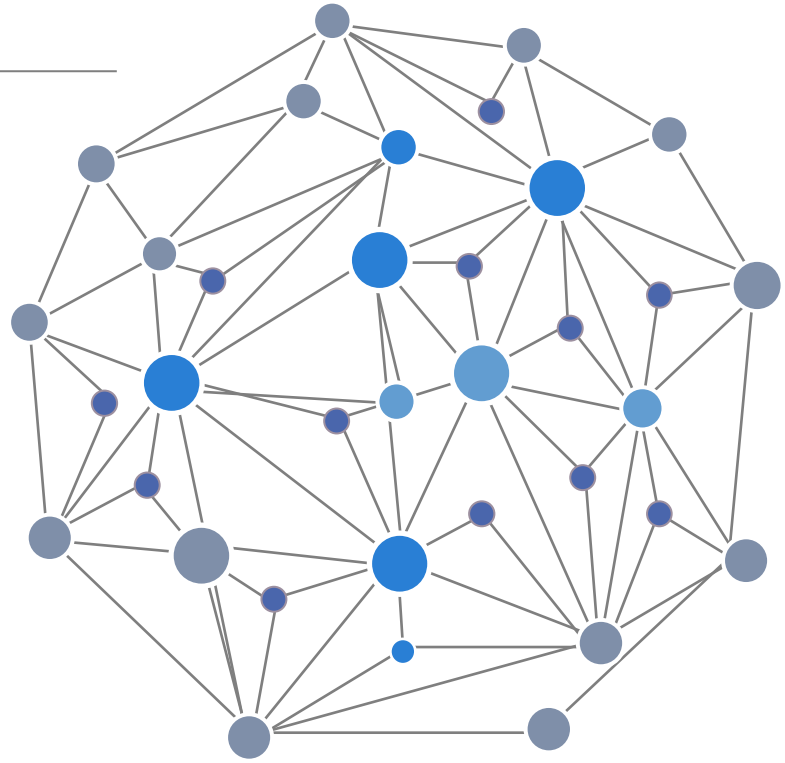
Web 程序设计

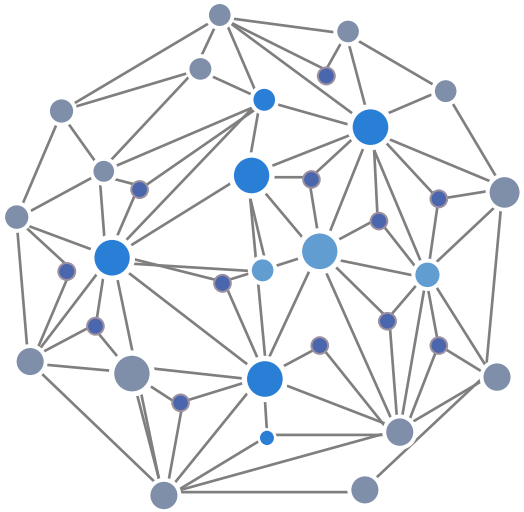


第八讲 PHP 编程 (2): PHP 应用编程

福州大学 计算机与大数据学院
软件工程系 陈昱

内容提要

- 会话跟踪 Cookie 和 Session
- 表单处理
- 文件与目录操作
- PHP 函数库/类库



会话跟踪

会话与会话跟踪

- **会话**指的是某个浏览器与服务器之间的一系列HTTP请求和响应

- 在会话期间，客户端和服务器之间可能有些数据需要保存下来，例如登陆后的用户名，需在多个交互中保持下来

HTTP 协议与保持状态

- 但是，HTTP 是一个 "无状态" 的协议
- 它没有办法存储一个会话的信息使得这些信息对后面的交互也有效

- 当一个顾客在请求一个页面后再请求一个页面时，HTTP 无法告诉我们两个请求是来自同一个顾客

会话跟踪的目的

- 会话跟踪的目的就是在网站中能够跟踪一个用户
- 当从一个页面跳到另一个页面时，服务器端程序需要知道仍是原来的用户在访问网站

会话跟踪的目的

- 例如：
  - 网站上的一些页面是不允许所有用户访问
  - 对这些页面的访问需要进行身份认证
  - 就需要页面能知道用户的登陆状态

- 两种实现方式：
  Cookie 和 Session

会话跟踪的实现

- 两种实现方式：Cookie 和 Session

- Cookie 和 Session 都是用来记录用户相关信息的机制，区别在于
  - Cookie 将信息存储在客户端
  - Session 将信息存储在服务器端

Cookie 方式

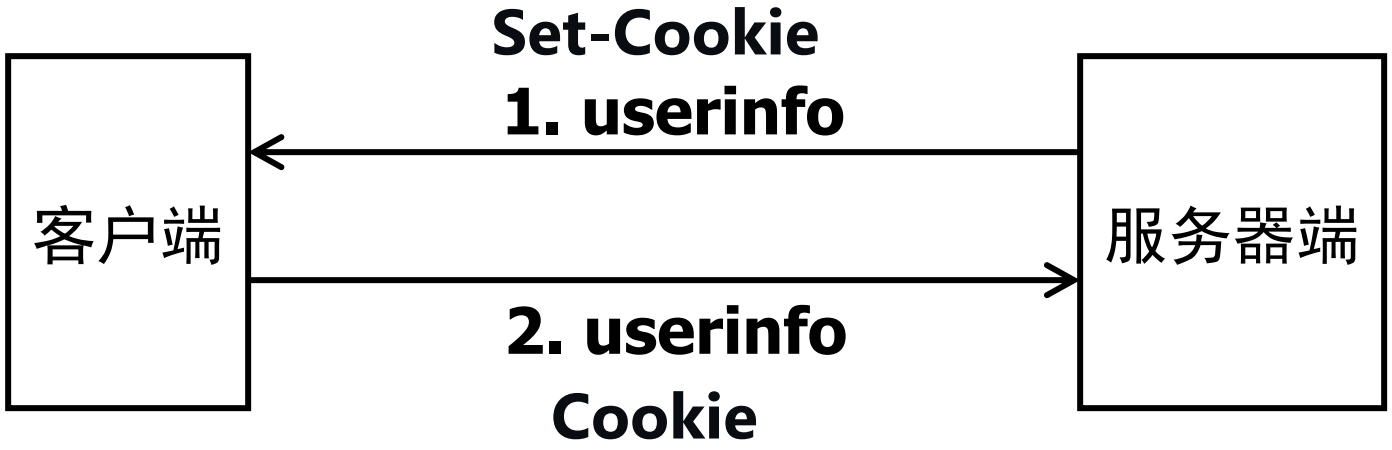
Cookie 方式原理:

- 当用户登陆时，服务器端程序将用户信息保存在用户浏览器的 Cookie 里面

- 当我们访问网站其他页面时，浏览器将 Cookie 中的用户信息发送给服务器端程序，服务器端程序使用这些信息生成页面

Cookie 方式实现原理

- Cookie 的具体实现，使用的是 HTTP 头部报文中的两个字段：
- 响应报文中的 Set-Cookie 字段
- 请求报文中的 Cookie 字段



PHP 中实现 Cookie

- 服务器端用 setcookie 函数创建一个 cookie
- setcookie(name, value, lifetime …);

- 例如：
  - setcookie("user", "Alex Porter", time()+3600);

PHP 中实现 Cookie

- 需要验证身份的页面可以通过超级全局变量 $_COOKIE 得到之前设置的 cookie 值
  - $user = $_COOKIE['user'];

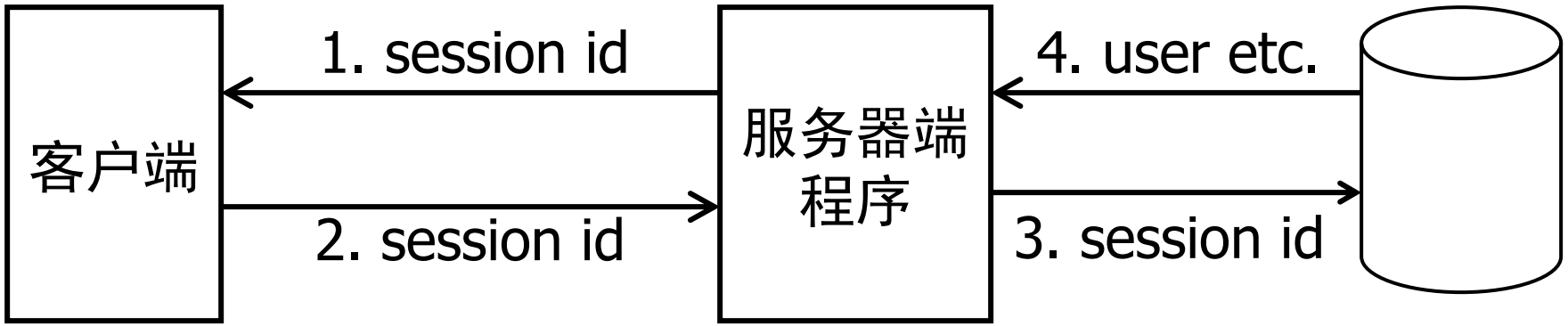
- 注意，cookie 必须在所有程序输出前发送
  - echo 或 <?php … ?> 之外的 html 代码都是输出，因为 Set-Cookie 是响应报文头部，而输出是放到实体主体中

Session 方式实现原理

- 当用户初次登陆网站的时候，服务器为其生成一个随机的字符串，叫做 session id，并将其传递给客户端
  - 例如：eadc453415011bac07c29523b64
- 服务器端程序在服务器的临时文件夹中创建了一个文件保存与该 session id 相关的信息（比如用户名称、购买商品等）
- 客户端再次访问访问网站时，提交 session id，服务器就知道客户又来了，从文件取出相应信息

Session 方式实现原理

- 可以把 session id 想像成钥匙，服务器设置一个保险箱存放客户端数据
- 客户端持有 session id 即可到服务器上读取原来它存放的数据



Session 方式实现原理

- 服务器和客户端交换 session id 的方式主要有两种：

- 一种通过 Cookie:
  - PHPSESSID=eadc453415011bac07c29523b6461f5f
  - JSESSIONID=FB9D15A4DB677AC8039329F89FB5D1EA

Session 方式实现原理

- 另一种通过 URL 改写方式传递
- http://...../xyz.php?PHPSESSID=eadc45341 5011bac07c29523b6461f5f

- PHP 默认使用 Cookie 方式

PHP 中 Session 方式的实现

- 开始会话：

1. 在所有页面的开头执行 **session_start()**;
2. 如果会话 ID 不存在，它会创建一个；如果会话 ID 已存在，则载入存在的会话
3. 创建会话变量：
   - $_SESSION['user'] = 'admin';
4. 使用会话变量：
   - if (isset($_SESSION['user'])) { …… }

PHP 中 Session 方式的实现

销毁一个 Session 变量

- unset($_SESSION['user']);

销毁一个会话（也就是用户退出）

- session_destroy();

Session 的数据放在什么地方?

- 在客户端，会话ID 保存在以 PHPSESSID 为名字的 cookie 中
- 在服务器上，会话数据保存为形如 /tmp/sess_fcc17f071 的临时文件
- 你可以使用 session_save_path 函数找到(改变)会话数据存放的文件夹
- 对于大型的应用，会话数据可以保存在SQL数据库(或其它目标路径)中而不是使用文件

会话超时

- 因为HTTP是无状态的，服务器难以知道用户什么时候完成一个会话
- 理想状态下，用户会明确地登出，但大部分用户不会
- 客户在浏览器关闭时删除会话cookie
- 服务器在一个周期后自动删除旧会话
  - 旧会话会浪费资源并且可能造成安全危险
  - 在 PHP 服务器上设置或者使用 session_cache_expire 函数调节会话
  - 你可以调用 session_destroy 删除一个会话

两种方式的比较

- Cookie 变量是存储在客户端浏览器里的
- Session 变量是存储在服务器端的临时文件里（也可以放到数据库中）

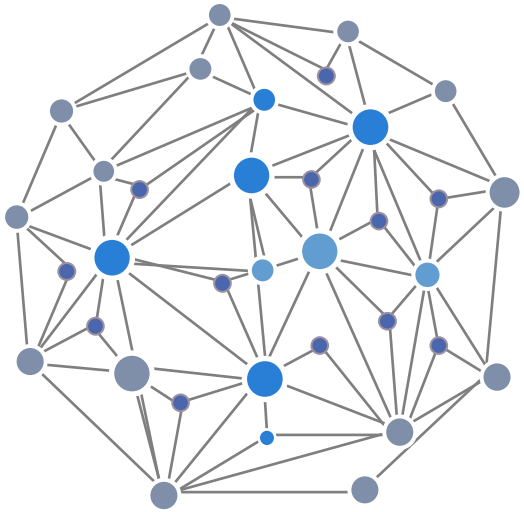
- Cookie 不能存储太多的数据
- Cookie 存在安全隐患，可能被盗用

实例：登录的实现 login.php

```
$pass_hash = hash('sha256', $_POST['pass'];
$user = $_POST["user"];

// 与数据库中保存的用户名密码Hash比较……略

// 当验证通过后，启动 Session
session_start();
//  注册登陆成功的 admin 变量，并赋值 true
$_SESSION["admin"] = true;
```

在需要用户验证的页面启动 Session，然后判断是否登陆

```
//  启动会话，这步必不可少
session_start();

//  判断是否登陆
if (isset($_SESSION["admin"]) &&
    $_SESSION["admin"] === true)  {
    echo "您已经成功登陆";
} else {
    //验证失败，将$_SESSION["admin"]置为 false
    $_SESSION["admin"] = false;
    die("您无权访问");
}
```



表单处理

表单参数传递方式

- Form 中的 method 可以设置为 get 和 post 两种方式
- 在表单数据传输过程中分别对应了 HTTP 协议请求报文中的 GET 和 POST 方法

- get 是 form 的默认方法

```
<form action="…/cgi-bin/hello.cgi" method="get">
```

表单参数传递的方式 — GET

- get 方法将表单数据按照 name=value 的形式，添加到 action 所指向的 URL 后面
  - http://...../Search?search=铁路&go=Go

- 特点
  - 不够安全，表单内容容易被看到
  - 传输的数据量小，受 URL 长度限制
  - 特殊字符需转成 URL 编码 (例如 空格 %20)

表单参数传递的方式 — POST

- POST 方法是将表单中的数据放在 HTTP 报文的数据部分 (而不是 URL)
  - 同样按照 “name=value” 的方式排列
  - POST 可以传输大量的数据，所以在上传文件时只能使用 POST
- 提交信息时，POST 请求比把参数嵌在 URL 中的 GET 请求更合适
  - URL 的长度是有限的(~ 1024 字符)
  - URL 不能包含未经编码的特殊字符
  - URL 中的私密数据会被看到

POST 报文例子

- POST /cgi-bin/login.cgi HTTP/1.1
- Host: mail.sina.com.cn
- Content-Type: application/x-www-form-urlencoded
- Content-Length: 56
  - logintype=uid&u=test&psw=test&btnloginfree=%B5%C7+%C2%BC

POST 上传数据时的内容类型

- Content-Type 有两种：
- application/x-www-form-**urlencoded**
  - 报文中的数据同样需被 URL 编码（浏览器自动完成）

- multipart/form-data
  - 用于传输大量二进制数据或者非 ASCII 字符的文本

POST 报文上传文件

Content-Type: multipart/form-data; boundary=----------------------------26500191691572 4
Content-Length: 20523

----------------------------26500191691572 4
Content-Disposition: form-data; name="files[upload_0][description]"

16350.jpg
----------------------------26500191691572 4
Content-Disposition: form-data; name="files[upload]"; filename="16350.jpg"
Content-Type: image/jpeg

PHP 表单处理

- 将 form 的 **action** 属性指向一个 php 程序，即可用该 php 程序进行表单处理

- 在 PHP 中引入了**超全局变量数组** $_POST 和 $_GET，$_REQEST来获取表单值
- 例如：有个name为 phone 的文本框，表单传递方法是 post，可通过下列方式得到值：

  $_POST["phone"]

PHP 超全局变量数组

- $_GET
  - 经由 URL 请求提交至脚本的变量
- $_POST
  - 经由 HTTP POST 方法提交至脚本的变量
- $_COOKIE
  - 经由 HTTP Cookies 方法提交至脚本的变量
- $_REQUEST
  - 经由 GET，POST 和 COOKIE 机制提交至脚本的变量

表单变量 - 来自 PHP 之外的变量

- 当表单提交时，表单中的字段会自动成为 PHP 脚本中超全局变量数组的成员

```
<form action="foo.php" method="post">
 Name:  <input type="text" name="username" />
 Email:   <input type="text" name="email" /></p>
 <input type="submit" name="submit"
        value="Submit me!" />
</form>
```

表单处理

- 表单提交后，在 PHP 中通过超全局变量数组访问表单元素的数据，数组索引为元素名称

```
<?php
        echo $_POST['username'];
        echo $_REQUEST['email'];
?>
```

PHP 超全局变量

- 之所以称为超全局变量是因为，它们在任何范围内都有效
- 它们并不需要 'global' 声明
- 这是它们与全局变量的区别

URL 编码

- 某些特定字符不允许出现在URL的查询参数中:
  - 例如: " ", "/", "=", "&"
- 当参数被传递时，它是URL编码的
  - “Eric’s cool!?" → “Eric %27s+cool %3F%21"
- 你一般不需要担心以下事项:
  - 在传递参数前，浏览器会自动对它们进行编码
  - PHP的 $_REQUEST 数组会自动对它们解码
  - … 但有时会出现编码后的版本 (例如在 Firebug)

文件上传 form

```
<form enctype="multipart/form-data"
          action="upload.php"
           method="post">

  <input type="hidden"
          name="MAX_FILE_SIZE"
           value="30000" />
   Send this file:
   <input name="userfile" type="file" />
   <input type="submit" value="Send File" />
</form>
```

- 可以使用type属性为file 的input标签, 在表单中上传文件
- 必须设定表单的enctype 属性
- 当你按下提交按钮后，具体发生了什么事情?

在 PHP 里处理一个上传文件

- 上传的文件放在全局数组 $_FILES，而不是 $_REQUEST
- $_FILES 里每一个元素自身是一个关联数组，包含:
  - name：用户所上传的本地文件名
  - type：上传数据的MIME类型，例如image/jpeg
  - size：文件大小，以byte 为单位
  - tmp_name：存储在服务器的临时副本的文件名
    - 为了永久保存这个文件，需要从这个临时位置移动到其它地方

上传细节

- <input type=“file” name=“avatar” />

- 例如：如果你上传参数名为 avatar 的 borat.jpg
  - $_FILES[“avatar”][“name”] 会是"borat.jpg "
  - $_FILES[“avatar”][“type”] 会是"image/jpeg "
  - $_FILES[“avatar”][“tmp_name”] 会是类似 “/var/tmp/phpZtR4TI “ 的东西

PHP 处理上传的文件

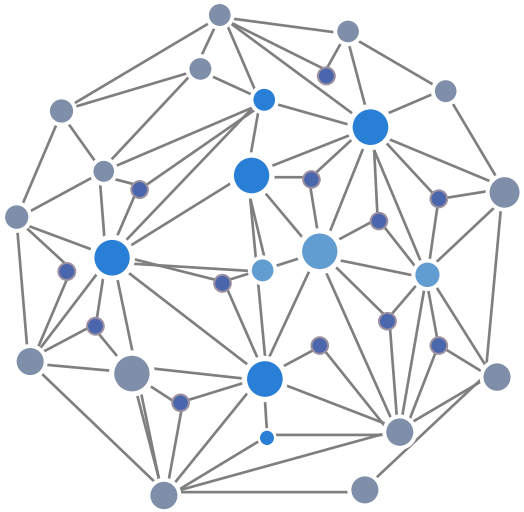
```
$uploaddir = '/var/www/uploads/';
// basename 可以提取路径中的文件名
$uploadfile = $uploaddir . basename($_FILES['userfile']['name']);

echo '<pre>';

if ( is_uploaded_file($_FILES[‘userfile][‘tmp_name’])) ) {
    move_uploaded_file($_FILES['userfile']['tmp_name'], $uploadfile);
    echo "File is valid, and was successfully uploaded.\n";
} else {
    echo "Possible file upload attack!\n";
}

echo 'Here is some more debugging info:';
print_r($_FILES);
print "</pre>";
```

PHP 处理上传的文件

- 用于处理上传文件的参数:
  - is_uploaded_file (filename)
  - 如果给定的文件名是由用户上传的话，返回 TRUE
  - move_uploaded_file(from，to)
  - 把文件从一个临时位置移动到一个永久位置
- 惯用做法：检查 is_uploaded_file，然后使用 move_uploaded_file

常见问题

- php.ini 的一些设定值会影响上传功能的使用
  - file_uploads　　是否开启PHP上传功能
  - upload_tmp_dir　　上传文件临时存放目录
  - upload_max_filesize 上传的最大文件大小
  - memory_limit　　PHP 脚本的内存使用限制也要足够大



文件与目录操作

文件操作函数

- 打开文件 — fopen()
- 关闭文件 — fclose()
- 读取文件 — fread()、fgets()、file_get_contents()
- 写入文件 — fwrite()、fputs()、file_put_contents()
- 复制文件 — copy()
- 删除文件 — unlink()
- 重命名（移动）文件 — rename()

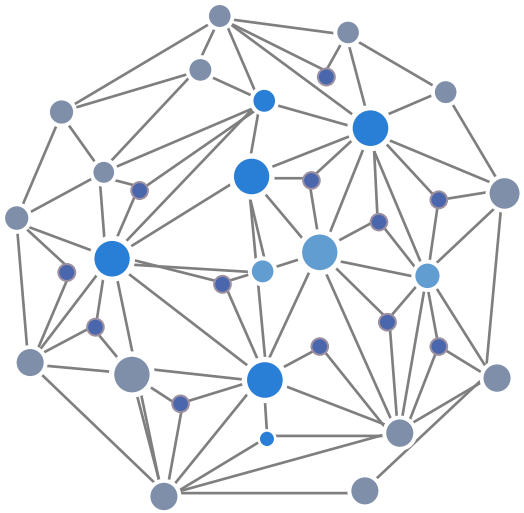
目录操作函数

- 打开一个目录 – opendir()
- 读取目录文件列表 – readdir(),rewinddir()
- 创建，删除目录 – mkdir(), rmdir()
- 获取目录权限掩码 – umask()

常见问题

- 注意文件在系统中的权限问题
- 像Linux系统，一般文件的权限可分为所有者、组、其他三种不同的使用者权限
- 每种又分"读、写、执行"三种属性权限

- 只有正确设置了权限，PHP才能对文件进行操作



PHP 函数库/类库

PHP 函数库/类库

- PHP 提供了一个庞大的函数库，可以大大节省开发时间
- 共分成了 100 多类
- 以扩展模块的形式加载
- 存放在 php 目录下的 ext 子目录里
- 分成大类：核心模块 和 PECL 模块

- 除此之外，还有由社区提供的 PEAR 类库

PEAR - PHP 扩展与应用类库

- PEAR
  - PHP Extension and Application Repository
  - http://pear.php.net/

- PEAR 将PHP程序开发过程中常用的功能编写成类库，涵盖页面呈面、数据库访问、文件操作、数据结构、缓存操作、网络协议等许多方面
  - 让 PHP 社区“不要重复发明轮子！”

PECL 和 PEAR 的区别

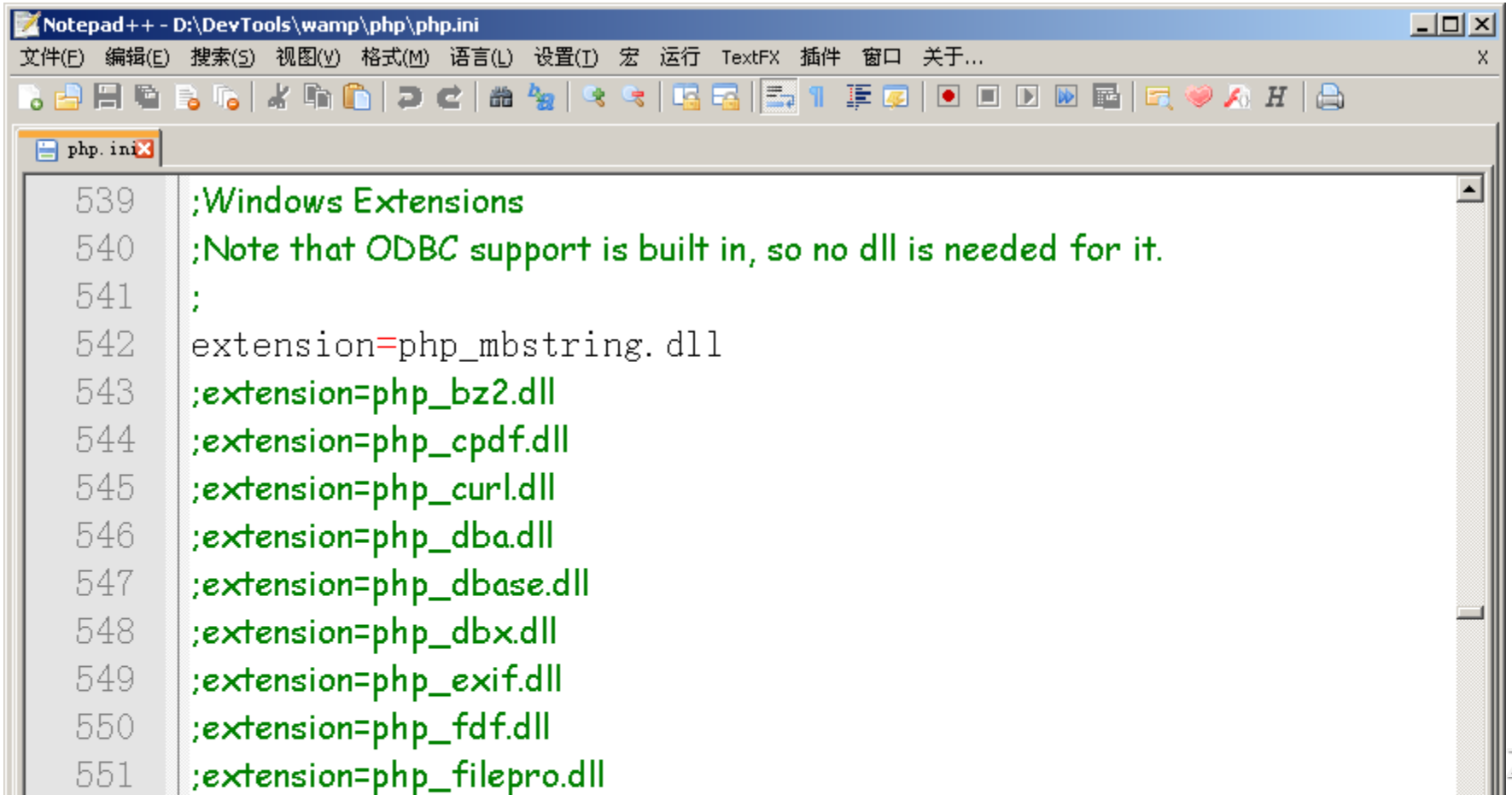
- PECL :: The PHP Extension Community Library

- PECL 是用 C 编写的函数库
- PEAR 是用 PHP 编写的类库

- 因此 PECL 需要编译后才能加载使用
- PECL 下载：http://pecl.php.net/

在 php.ini 中修改扩展加载

- 去掉前面的分号“;”，并重启Apache服务器



获取 PHP 配置信息

- 模块需要加载后才能使用其中的函数

- 通过下面两个函数：
- get_loaded_extensions();
- get_extension_funcs();
- 可以知道服务器上PHP的模块和函数是否可用

获取 PHP 配置信息

- <?php
  print_r(get_loaded_extensions());
  ?>

```
Array
(
    [0] => bcmath
    [1] => calendar
    [2] => com_dotnet
    [3] => ctype
    [4] => session
    [5] => filter
        ……
```

第八讲(2) 课后练习

- 学习 w3school 相关内容
  - https://www.w3school.com.cn/php/php_sessions.asp
  - https://www.w3school.com.cn/php/php_cookies.asp
  - https://www.w3school.com.cn/php/php_forms.asp
  - https://www.w3school.com.cn/php/php_file.asp
  - https://www.w3school.com.cn/php/php_file_upload.asp
- 通过实验和后期编程实践熟悉 cookie, session, 表单处理等功能

THANKS

本章结束

陈昱

福州大学 计算机与大数据学院 软件工程系