**HTTP协议—会话技术**

1. **什么是HTTP协议？**
2. **HTTP协议包括哪几种？每种协议的组成有哪些？**
3. **常见的状态码有哪些？**
4. **常见的响应头有哪些？**

指服务器和客户端间进行通信时的约束和规范，客户端与服务端的数据交互并不是杂乱无章的，需要遵照（基于）一定的规范进行。

# 会话技术介绍

## 为什么需要会话技术

HTTP协议特点：

无状态、无记忆的。多次请求之间，无任何的联系，即同一用户请求同一网站的不同页面，服务器无法识别是否是同一用户发起的请求。因此，用户无法进行连续的业务逻辑。

如：登录，在A页面登录，请求B页面，提示登录。

学生进教学楼： 每次开证明，证明是传智学生 or 带着胸牌

## 什么是会话技术

会话：浏览器与服务器之间的数据交流。

在同一台浏览器对服务器的多次请求中，将数据持久化存储的技术，以实现连续的业务逻辑

## 会话技术分类

按照数据持久化的位置进行区分，

数据持久化存储在浏览器，为**cookie技术**。

数据持久化存储在服务器，为**session技术**。

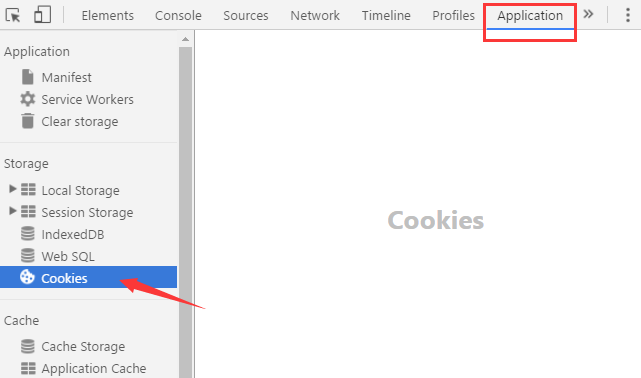
# COOKIE技术初步

## 概念

允许服务器脚本（PHP脚本）在浏览器端存储数据，并以此跟踪及识别用户的技术。

数据设置后，浏览器再次请求服务器指定页面时，将相关的数据(身份信息)发送给服务器脚本，供脚本使用。

## 查看COOKIE数据



## js操作cookie

cookie本质为简单数据，基本操作逻辑为增删改查。

Cookie的属性 有效期（expries） 路径(path) 域名(domain)

var date=new Date(); 东八区！

date.setTime(date.getTime()+7\*3600\*1000); // 设置过期时间

document.cookie='name=zs;expires='+date.toString();

*// 设置cookie*

**function** **setCookie**(key,value,param){

document.cookie = key + '=' + value + '; expires=' + param.expires；

}

*// 获取cookie***function** **getCookie**(key){

**var** cookies = document.cookie;

**var** arr = cookies.split('; ');

**if**(arr){

**for** (**var** i = 0; i < arr.length; i++) {

**var** kv = arr[i].split('=');

**if**(kv[0] == key){

**return** kv[1];

}

}

}

}

**Jquery.cookie.js操作**

$.cookie('weight',100,{expires:7});//设置

$.cookie('name');//获取

$.cookie('name',nll);//删除

$.cookie(键,值,{expires:过期天数})

$.cookie(键) //获取

$.cookie(键,null); //删除

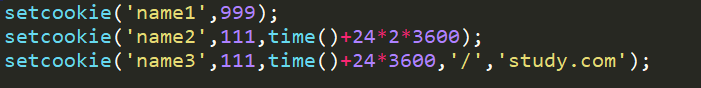
## php操作cookie

### 设置语法

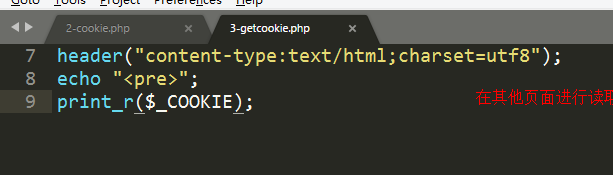
setcookie(数据名，数据)；

setcookie(“名称”,”值”，“有效期”)；

执行以上代码，PHP脚本通知浏览器在其COOKIE数据区存储数据。数据的类型为字符串。



### 读取COOKIE数据

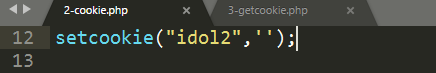


* 设置好的COOKIE数据，下次请求同一网站的页面时，浏览器将对应数据传输给PHP脚本。PHP脚本将数据存储在全局变量$\_COOKIE中。可以通过$\_COOKIE进行读取。
* 同一网站的不同页面之间可以进行COOKIE数据的共享：对于同一个浏览器，在网站的A页面设置的COOKIE数据，可以在网站的B脚本中读取到。
* COOKIE数据是基于浏览器的：在A浏览器请求页面设置的COOKIE数据，存储在A浏览器的COOKIE数据区，在B浏览器请求同样的页面，无法访问。
* 无法通过$\_COOKIE添加数组元素的方式设置数据，实现数据在客户端的长久保存（setcookie本质为通过HTTP响应协议，来通知浏览器对数据进行操作）。

## 删除cookie

没有显式的删除函数。可以通过设置COOKIE数据为空字符串来实现。COOKIE数据为空后，浏览器自动消除该数据空间。

setcookie(“变量名”，“”)；



## 从HTTP协议理解COOKIE数据的传输

### 第一次响应

第一次响应，PHP在响应头中告诉浏览器如何设置数据。

### 第二次请求

浏览器请求同一网站的其他页面时，携带该网站对应的全部COOKIE数据。

### 其他次响应

如果修改COOKIE数据，在响应头中会有setcookie协议项。

# COOKIE数据的属性

## 有效期（生命周期）

COOKIE数据的默认的生命周期是会话周期，即在浏览器窗口关闭之前，一直存在。关闭后立即消失。

关闭浏览器：相当于结束默认的会话，COOKIE数据自动消失。再次打开浏览器，相同页面读取数据，无法读到。

好处：减少空间占用，减少HTTP传输的数据量，提高数据安全。

### 有效期

设置COOKIE的第三个参数，为设置其有效期。

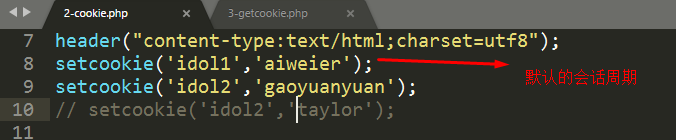
setcookie(“名称”,”值”，“有效期”);

即COOKIE数据的有效时间，使用时间戳（秒数）来表示。COOKIE数据过期后，会被浏览器自动清除掉。HTTP请求中，仅携带在有效期内的COOKIE数据，可以减少携带的数据量。

### 常见的COOKIE生命周期

* 默认会话周期：窗口周期

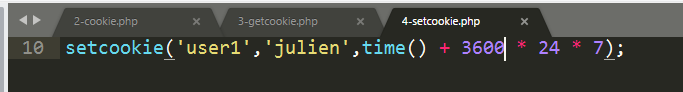
窗口关闭，COOKIE数据自动被清除。



* 删除单个COOKIE数据的另外一种方法：

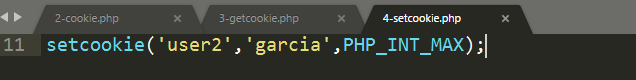


* 一周：



* 永久有效：

使用常量PHP\_INT\_MAX



### 默认值

有效期：默认会话周期。浏览器不关闭，数据一直存在。浏览器关闭数据消失。

time() -1 清除COOKIE数据。

有效路径：当前目录及子目录有效。设置整站有效:/

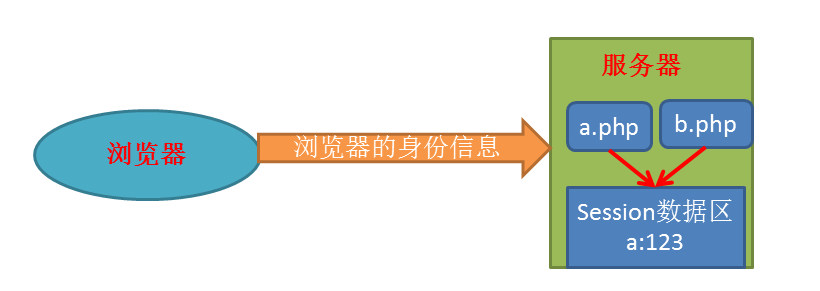
# session技术初步

思考：如何实现网站登录功能？

## 什么是session

为了实现业务逻辑的连续性，将部分信息持久化存储在服务器端，可以保证数据的安全性及各个脚本之间的共享。

## 原理



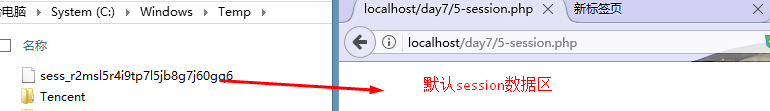
* 将客户端的状态信息（区分不同用户）存储在服务器端的SESSION数据区。
* 可以在网站的不同页面之间传递数据（页面数据共享）
* 可以识别不同的浏览器（基于浏览器）。

## 基本操作

### 开启session机制

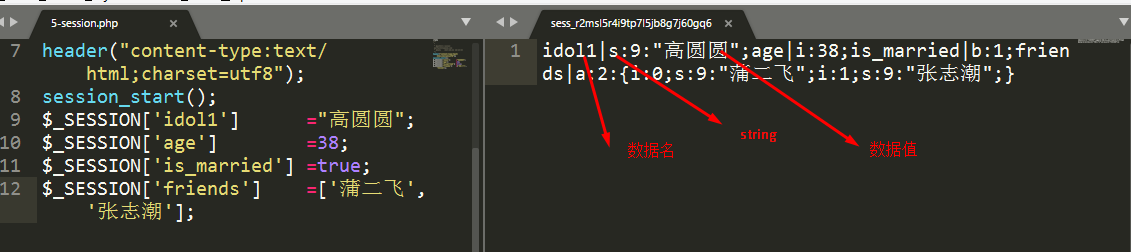
session\_start(); 使用session前一定要先开启

作用：开启session会话或者重用已经创建的会话。

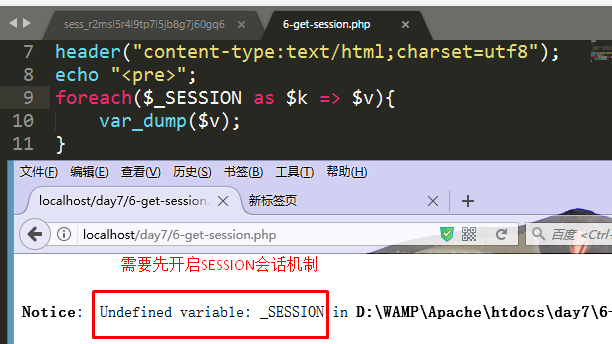


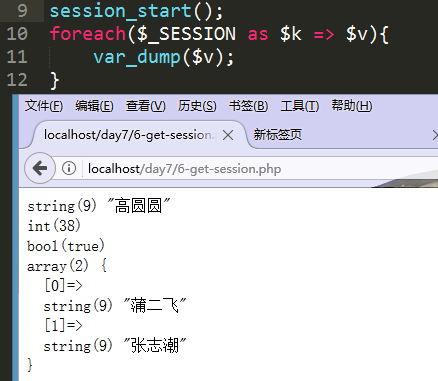
### session数据的读写操作

* SESSION数据的增删改查操作都是通过数组$\_SESSION进行的。数组元素的下标不能使用数字。



* 存储的数据类型几乎可以为PHP的全部数据类型。
* 一个session文件中可以存放多个数据（多个键值对）。
* 数据存储在session文件中，实现同一网站多个页面共享session数据。
* session数据是基于COOKIE数据的，浏览器不同，cookie数据也不同。因此，在A浏览器请求页面，设置的SESSION数据，在B浏览器请求网站的其他页面，无法读取。

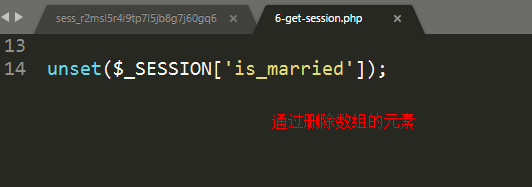




### 删除一个session数据

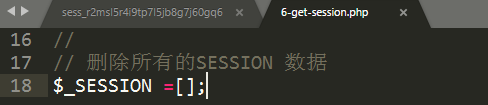
unset($\_SESSION[名称]);

可以使用unset语法删除一个session数据（数组的一个元素）。删除的是session文件中的指定数据，不删除session文件。



### 删除所有session信息

$\_SESSION设置为空数组。本质为使用空数组覆盖session文件中对应的数据。session文件不被删除。  
 $\_SESSION=[];



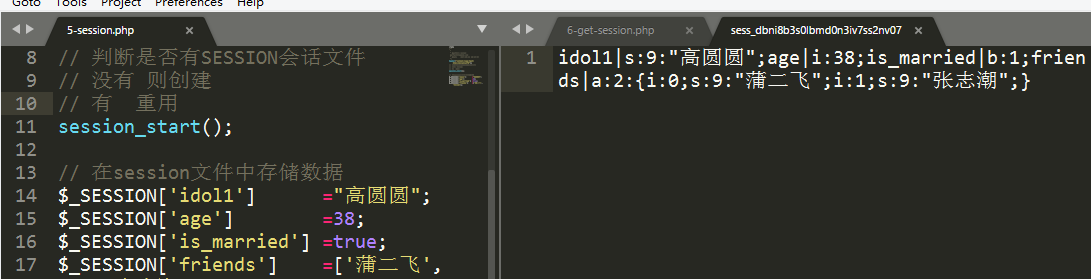
### 销毁session信息

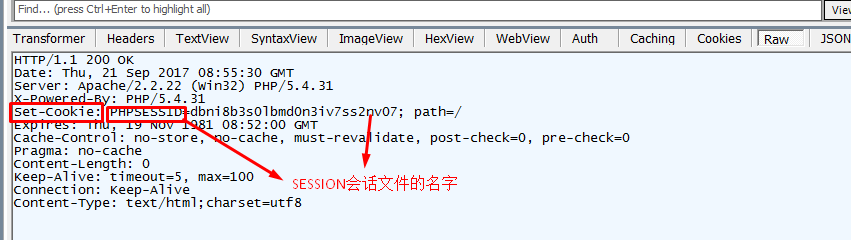
session\_destroy();

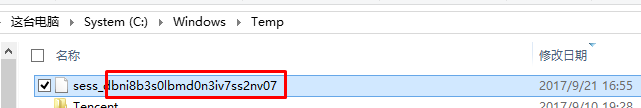
直接删除session会话文件，PHP脚本将无法读取session数据。

# SESSION 数据高级

## 从HTTP协议，SESSION是基于COOKIE的







浏览器请求a.php时，PHP脚本中，创建或者重用对应的session文件，设置SESSION数据后。响应时，PHP脚本会将当前SESSION会话的会话名称（文件名一部分），以COOKIE数据的形式返回给浏览器，并且COOKIE数据的数据名，默认为PHPSESSID。

浏览器再次请求本网站的任意页面时，自动携带名为PHPSESSID的COOKIE数据，PHP按照会话名称，将对应的数据读取到$\_SESSION数组中。

## session与COOKIE之间的区别与联系

### 区别

* **从存储位置**：

COOKIE存储在浏览器，SESSION存储的服务器端。

* **数据安全性：**

session存储在服务器端，不易获取，更加安全。

* **传输的数据量：**

每次请求，COOKIE传输量较大。传递该网站相关的全部的COOKIE数据。

* **支持的数据类型：**

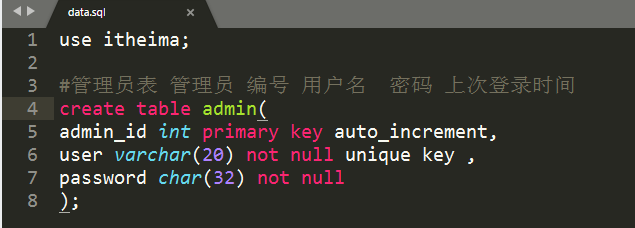
COOKIE支持字符串，SESSION几乎所有的数据类型。

### 联系

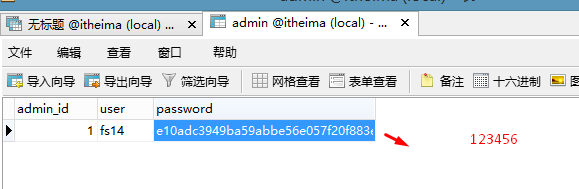
* 都是用来解决HTTP协议的无状态、无记忆的不足之处，以实现业务的连续性，如登录及购物车模块的实现。
* 二者都是会话技术的范畴 。
* SESSION需要依赖COOKIE（浏览器的身份信息）进行数据的传递。禁用COOKIE后，session无法正常使用（确保类似于使用同一个购物车：将来可以在不同的页面之间传递SESSION会话的名字，session\_id()）。

# 会话技术应用--登录、退出模块

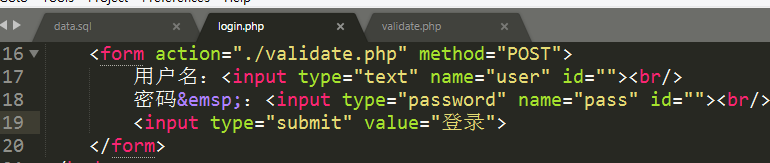
## 数据表设计

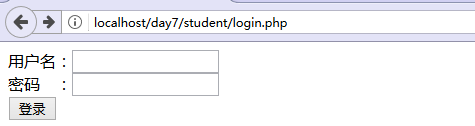


注册一个管理员账号：



## 登录表单设计





## 账号信息验证

* 接收账号信息
* 账号信息验证：
* 登录成功，设置成功标志

使用session技术来实现。

## 其他页面的登录检查

* 添加登录入口

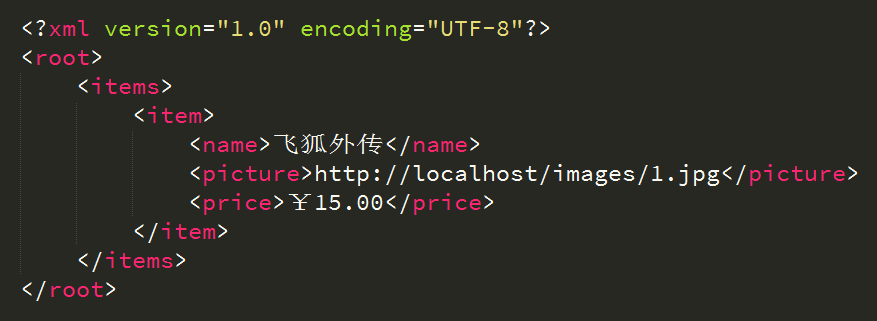
## 退出模块

删除登录标志即可。

# 常见的数据传输格式

## XML

XML是一种标记语言，很类似HTML，其宗旨是用来传输数据，具有自我描述性（固定的格式的数据）。



### 语法规则

1、必须有一个根元素

2、不可有空格、不可以数字或.开头、大小写敏感

3、不可交叉嵌套

4、属性双引号（浏览器自动修正成双引号了）

5、特殊符号要使用实体

6、注释和HTML一样

虽然可以描述和传输复杂数据，但是其解析过于复杂并且体积较大，所以实现开发已经很少使用了。

### XML示例

详见代码

## JSON

即 JavaScript Object Notation，另一种轻量级的文本数据交换格式，独立于语言。

### 语法规则

1、数据在名称/值对中

2、数据由逗号分隔(最后一个健/值对不能带逗号)

3、花括号保存对象方括号保存数组

4、使用双引号

### JSON解析

JSON数据在不同语言进行传输时，类型为字符串，不同的语言各自也都对应有解析方法，需要解析完成后才能读取

**1、Javascript 解析方法**

JSON对象 JSON.parse()、JSON.stringify()；

JSON兼容处理json2.js

**2、PHP解析方法**

json\_encode()、json\_decode()

总结：JSON体积小、解析方便且高效，在实际开发成为首选。