第21章　日期和时间的管理

在本章中，我们将介绍日期和时间的检查和格式化，以及日期在不同格式之间的转换。

当要在MySQL日期格式和PHP日期格式、UNIX日期格式与PHP日期格式，以及用户输入HTML表单的日期格式之间进行转换时，这是非常重要的。

在本章中，我们将主要介绍以下内容：

- 在PHP中获取日期和时间
- 在PHP日期格式和MySQL日期格式之间进行转换
- 计算日期
- 使用日历函数

21.1　在PHP中获取日期和时间

在第1章“PHP快速入门教程”中，我们讨论了使用date()函数来获取并格式化PHP中的日期和时间。在本章中，我们仍将讨论该函数，并且更详细地介绍PHP中其他的时间和时间函数。

21.1.1　使用date()函数

我们能够回忆起来，date()函数带有两个参数，其中有一个是可选的。第一个是格式字符串，第二个（即可选的一个）是UNIX时间戳。如果没有指定时间戳，在默认的情况下，date()函数将返回当前的日期和时间。该函数可以返回一个格式化后表示适当日期的字符串。

date()函数的常见调用方式如下所示：

```
echo date('jS F Y');
```

以上调用将返回格式为“19th June 2008”的日期。表21-1给出PHP中所支持的日期格式代码。

表21-1　PHP的date()函数所支持的格式代码

代　码	描　述
a	上午或下午，两个小写字符表示，“am”或“pm”
A	上午或下午，两个大写字符表示，“AM”或“PM”
B	Swatch Internet时间，一种统一的时间模式。在http://www.swatch.com/可以找到关于该时间模式的详细信息
c	ISO 8601日期。其日期用YYYY-MM-DD表示。大写T用来间隔日期和时间。时间用HH:MM:SS来表示。最后，时区是用格林威治时间（GMT）的偏差来表示的，例如，2008-06-26T21:04:42+11:00（PHP 5.1.0已经引入这个格式代码）
d	两位数字表示的月份中的日期，带前导0，从“01”到“31”
D	3个缩略字符表示的星期，文本格式，从“Mon”到“Sun”
e	时区识别器（PHP 5.1.0）

（续）

代　码	描　述
F	年中的月份，全写，从“January”到“December”
g	日期中的时间，12小时制，无前导0，从“1”到“12”
G	日期中的时间，24小时制，无前导0，从“0”到“23”
h	日期中的时间，12小时制，带前导0，从“01”到“12”
H	日期中的时间，24小时制，带前导0，从“00”到“23”
i	小时中的分钟，带前导0，从“00”到“59”
I	夏令时制，以布尔值表示，若为夏令时，返回“1”，否则返回“0”
j	月份中的日期，数字型，无前导0，从“1”到“31”
l	星期，全称，从“Sunday”到“Saturday”
L	闰年，以布尔值表示，如果日期所在年是闰年，返回“1”，否则返回“0”
m	以两位数字表示的月份，带前导0，从“01”到“12”
M	以3个缩略字符表示的月份，文本格式，从“Jan”到“Dec”
n	年中的月份，以数字表示，无前导0，从“1”到“12”
o	ISO-8601的年份数。与“Y”具有相同值。不同点在于，如果ISO周数（W）属于上一年或下一年，将使用新的年份数（在PHP 5.1.0版本引入）
O	当前时区与格林威治时间之间小时时差。例如，+1600
r	RFC822格式的日期和时间。例如，Wed, 1 Jul 2008 18:45:30 +1600（这是在PHP 4.0.4中新增加的）
s	带有前导0的秒钟时间数，其取值为“00”到“59”
S	序数，日期后缀，以两个字符表示，包括“st”、“nd”、“rd”或“th”，具体取决于日期数字后面的数字是什么
t	月份的天数，从“28”到“31”
T	服务器的时间区域设置，例如，“EST”
U	从1970年1月1日到某时刻总的秒数；也叫该日期的UNIX时间戳
w	星期，以单个数字表示，从“0”（星期日）到“6”（星期六）
W	一年当中的星期数，与ISO-8601格式相兼容（在PHP 4.1.0中新添加的）
y	两位数字表示的年份，例如，“08”
Y	四位数字表示的年份，例如，“2008”
z	数字表示的日期，从“0”到“365”
Z	与当前时区的时区差，单位为s（秒）。从“-43200”到“43200”

21.1.2　使用UNIX时间戳

date()函数的第二个参数是UNIX时间戳。我们对时间戳的意义做一个简单介绍。大多数UNIX系统保存当前日期和时间的方法是：保存格林威治标准时间从1970年1月1日零点起到当前时刻的秒数，以32位整列表示，其中1970年1月1日零点也叫UNIX纪元。如果对它不熟悉，这看起来有点深奥，但它是一个标准，而且整数适用于计算机处理。

UNIX时间戳是保存时间的一种紧凑简洁的方法，同时，它不会遭遇千年虫（Y2K）问题，该问题可以影响其他的紧凑或缩略的日期格式。然而，它们具有类似的问题，因为时间是通过一个32位的整数来表示的。如果软件需要处理1902年以前或2038年以后的事件，将会遇到一些问题。

在某些系统中（包括Windows），其范围更加有限。时间戳不能为负数，因此1970年以前的时间戳无法使用。要使代码具有可移植性，必须记住这一点。

我们不需要担心软件是否可以在2038年以后使用。时间戳没有固定的大小；它们是C语言的长整型，至少有32位。如果软件在2038年还会继续使用，那你的系统将肯定会使用了更大的类型。

虽然这是标准的UNIX惯例，但是即使在Windows服务器中运行PHP，这个格式仍然被date()函数和许多的PHP其他函数使用。唯一的不同就是，对于Windows来说，时间戳必须是正数。

如果要将一个日期和时间转变成UNIX时间戳，可以使用mktime()函数。该函数原型如下所示：

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

除了最后一个参数*is_dst*外，其他参数的含义都很容易理解。参数is_dst表示该日期所示的时间是否是夏令时。如果是，可以将其设置为1，如果不是，设置为0（默认值），如果不知道，则设置为-1（默认值）。在使用-1的情况下，PHP将根据所运行的系统来确定它。该参数是可选的，很少用到。

使用该函数需要避免的一个主要陷阱是其参数顺序非常不直观。参数的顺序不允许漏掉一个时间参数。如果对具体时刻不在乎，可以将0传给*hour*、*minute*和*second*参数。

可以从参数列的右边开始遗漏参数的值。如果参数为空，将默认为当前时间。因此，如下所示的调用：

```
$timestamp = mktime();
```

将返回当前日期和时间的UNIX时间戳。当然，也可以通过如下所示的调用获取当前的UNIX时间戳：

```
$timestamp = time();
```

time()函数不需要任何参数，而且通常返回当前日期和时间的UNIX时间戳。

另一个选项就是date()函数，正如我们已经讨论的。格式字符串“U”要求一个时间戳。

如下语句等价于上两个语句：

```
$timestamp = date("U");
```

可以将用2位或4位数字表示的年份数传递给mktime()函数。从0～69的2位数字表示的年份可以解释成2000年到2069年，而从70～99的年份解释成1970年到1999年。

如下所示的代码是说明mktime()函数使用的其他例子：

```
$time = mktime(12, 0, 0);
```

该语句将给出今天日期的中午时间。

```
$time = mktime(0,0,0,1,1);
```

该语句将给出当前年的1月1日。请注意，在小时参数中，我们使用了0（而不是24）来表示午夜。

也可以在简单日期算法中使用mktime()函数。如下所示：

```
$time = mktime(12,0,0,$mon,$day+30,$year);
```

虽然（$day+30）通常都会大于一个月的日期，该语句将在指定的日期基础上增加30天。

要消除冬令时和夏令时之间的问题，可以使用12点来代替0点。如果在25小时日中增加(24*60*60) s，将停留在同一天。在中午时间增加相同的秒数，将给出11am时间，但是至少是在正确的一天。

21.1.3 使用getdate()函数

能够确定当前时间的另一个很实用的函数是getdate()函数。该函数原型如下所示：

```
array getdate ([int timestamp])
```

它以时间戳作为可选参数，返回一个相关数组，表示日期和时间的各个部分，如表21-2所示。

表21-2　getdate()函数返回的相关数组中的关键字-值对

关 键 字	值	关 键 字	值
seconds	秒钟，数字	year	年份，数字
minutes	分钟，数字	yday	年份中的日期，数字
hours	小时，数字	weekday	星期，全写
mday	月份中的日期，数字	month	月份，全写
wday	星期，数字	0	时间戳，数字
mon	月份，数字		

在数组中定义了以上数据后，可以很方便地将它们转换成任何所需的格式。数组的0元素（时间戳）可能没有什么用途，但是如果调用没有参数的getdate()函数，它将返回当前的时间戳。

```
<?php
$today = getdate();
print_r($today);
?>
```

produces something similar to the following output:

```
Array (
[seconds] => 45
[minutes] => 6
[hours] => 20
[mday] => 14
[wday] => 3
[mon] => 3
[year] => 2007
[yday] => 72
[weekday] => Wednesday
[month] => March
[0] => 1173917205
```

```
)
```

21.1.4 使用checkdate()函数检验日期有效性

可以调用checkdate()函数来检验日期是否有效。这对检查用户输入的日期来说是非常有用的。checkdate()函数的原型如下所示：

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

它将检查年份数是否为介于0～32 767的一个整数，月份是否介于1～12，以及日期是否存在于特定的月份。当判断一个日期是否有效时，该函数同样会考虑闰年。

例如：

```
checkdate(3, 29, 2008)
```

将返回true，而：

```
checkdate(2, 29, 2007)
```

则返回false。

21.1.5 格式化时间戳

使用strftime()函数，你可以根据系统的locale（地域，Web服务器的本地设置）来格式化一个时间戳。这个函数具有如下所示的原型：

```
string strftime ( string $format [, int $timestamp] )
```

$format参数是定义了如何显示时间戳的格式化代码。$timestamp参数是传递给该函数的时间戳。这个参数是可选的。如果没有传递时间戳参数，本地系统的时间戳（脚本运行时的时间戳）将被返回。如下代码所示：

```
<?php
 echo strftime('%A<br />');
 echo strftime('%x<br />');
 echo strftime('%c<br />');
 echo strftime('%Y<br />');
?>
```

以上代码用四种不同格式显示了系统的当前时间戳。这段代码将产生类似于如下的输出：

Friday

03/16/07

03/16/07 21:17:24

2007

表21-3给出了strftime()函数格式化代码的完整列表。

需要注意的是，表21-3中列出的格式化代码中，如果它包含了标准格式，其值将被Web服务器的本地相关设置所代替。Strftime()函数对于用不同的格式显示日期和时间是非常有用的，而且有助于改进页面的用户友好体验。

表21-3　strftime()函数的格式化代码

代　码	描　述
%a	星期几（英文缩写）
%A	星期几
%b或%h	月份（英文缩写）
%B	月份
%c	标准格式的日期和时间
%C	公元
%d	月份内的日期（从01～31）
%D	缩写格式的日期（mm/dd/yy）
%e	月份内的日期（两个字符组成的字符串，“1”～“31”）
%g	根据周数的年份数，两位数字
%G	根据周数的年份数，四位数字
%H	小时数（从00～23）
%I	小时数（从1～12）
%j	年内的日期（从001～366）
%m	月份（从01～12）
%M	分钟数（从00～59）
%n	换行（\n）
%p	am或pm（或与地域风俗等价）
%r	使用a.m./p.m.风格的时间表示
%R	使用24小时风格的时间表示
%S	秒数（从00～59）
%t	制表符（\t）
%T	%h:%M:%S格式的时间
%u	星期内的日期（从1-Monday到7-Sunday）
%U	周数（一年中第一个周日将作为第一周的第一天）
%V	周数（一年的第一周将至少有4天，这周才会当作一周）
%w	星期内的日期（从0-Sunday到7-Saturday）
%W	周数（一年的第一个周一作为该周的第一天）
%x	标准格式的日期（没有时间）
%X	标准格式的日期（没有日期）
%y	年份数（两位数字）
%Y	年份数（四位数字）
%z或%Z	时区

21.2　在PHP日期格式和MySQL日期格式之间进行转换

MySQL中的日期和时间是以ISO8601标准处理的。从其中获取时间是相对正常的，但是ISO8601期望输入的日期要首先输入年。例如，2008年3月29日应该输入2008-03-29或08-03-29。在默认的情况下，从MySQL获取日期的顺序也是如此。

根据目标用户不同，我们可能会发现这个函数并不是用户友好的。通常，要在PHP和MySQL之间通信，需要进行日期格式的转换。这可以在其中任一端进行。

当从PHP将日期输入到MySQL时，可以调用前面介绍的date()函数轻松地将其转换为适

当的格式。需要注意的一个小问题是，在进行操作时应该使用带有前导0格式的日期和月份，这样可以避免在MySQL中造成混乱。可以使用两位数字的年份，但是使用4位年份是一个不错的想法。如果希望在MySQL端进行转换，可以使用两个有用的函数，它们分别是DATE_FORMAT()和UNIX_TIMESTAMP()。

DATE_FORMAT()函数与PHP的同名函数类似，只是使用不同的格式代码。通常，我们最希望做的事情就是以MM-DD-YYYY的形式格式化日期，而不是采用MySQL中固有的ISO格式，也就是YYYY-MM-DD格式。可以通过如下所示的查询代码来完成它：

```
SELECT DATE_FORMAT(date_column , '%m %d %Y')
FROM tablename;
```

格式代码%m表示2位数字的月份；%d表示2位数字的日期；而%Y表示4位数字的年份。表21-4列出了更多MySQL支持的实用格式代码。

表21-4　MySQL的DATE_FORMAT()函数的格式代码

代　码	描　述
%M	月份，全称
%W	星期，全称
%D	月份中的日期，数字，带文本后缀（例如，1st）
%Y	年份，数字，4位数字
%y	年份，数字，2位数字
%a	星期，3字符
%d	月份中的日期，数字，带前导0
%e	月份中的日期，数字，不带前导0
%m	月份，数字，带前导0
%c	月份，数字，不带前导0
%b	月份，文本，3个字符表示
%j	年中的日，数字
%H	小时，24小时制，带前导0
%k	小时，24小时制，不带前导0
%h或%I	小时，12小时制，带前导0
%l	小时，12小时制，不带前导0
%i	分钟，数字，带前导0
%r	时刻，12小时制（hh:mm:ss[AM[PM]]）
%T	时刻，24小时制（hh:mm:ss）
%S或%s	秒钟，数字，带前导0
%p	AM或PM
%w	星期，数字，从0（星期日）到6（星期六）

UNIX_TIMESTAMP()函数功能与之类似，但是它可以将一列转换为一个UNIX时间戳。例如：

```
SELECT UNIX_TIMESTAMP(date_column)
FROM tablename;
```

将返回已经被格式化成UNIX时间戳的日期。这样，就可以像在PHP中一样处理它。

使用UNIX时间戳，可以很方便执行日期计算和比较操作。但是请记住，时间戳通常可以

表示1902年至2038年之间的日期，而MySQL日期类型具有更大的时间范围。

作为一条重要的规则，当只是保存和显示日期的时候，应该使用UNIX时间戳来计算日期和作为标准日期格式。

21.3 在PHP中计算日期

在PHP中，计算两个日期之间长度的最简单方法就是通过计算两个UNIX时间戳之差来获得。程序清单21-1所示的脚本中就使用了这种方法。

程序清单21-1　calc_age.php——根据某人的生日计算年龄

```
<?php
  // set date for calculation
  $day = 18;
  $month = 9;
  $year = 1972;

  // remember you need bday as day month and year
  $bdayunix = mktime (0, 0, 0, $month, $day, $year); // get ts for then
  $nowunix = time(); // get unix ts for today
  $ageunix = $nowunix - $bdayunix; // work out the difference
  $age = floor($ageunix / (365 * 24 * 60 * 60)); // convert from seconds to years

  echo "Age is $age";
?>
```

在以上脚本中，我们设置了用以计算年龄的日期。在一个实际的应用程序中，该信息很可能来自一个HTML表单。首先，我们调用了mktime()函数分别计算生日的时间戳和当前时间的时间戳：

```
$bdayunix = mktime (0, 0, 0, $month, $day, $year);
$nowunix = time(); // get unix ts for today
```

因为这些日期具有相同的格式，因此，我们可以直接将它们相减。

```
$ageunix = $nowunix - $bdayunix;
```

现在，来处理一个有点棘手的问题——将这个时间段转化为更为友好的时间度量单位。这并不是一个时间戳，而是一个用秒钟量度的人的年龄。通过用一年的秒数来除当前以秒度量的年龄，将其转化为以年来度量。这样，我们就可以使用floor()函数对所得结果进行取整处理，以20岁为例，到他20岁那年为止：

```
$age = floor($ageunix / (365 * 24 * 60 * 60)); // convert from seconds to years
```

但是值得注意的是，该方法是有缺陷的，它受UNIX时间戳（通常是32位整型）范围的限制。生日计算并不是时间戳的很好应用。这个例子只适用于在所有平台下计算1970年以后出生的人的生日。Windows无法管理1970年以前的时间戳。即使这样，这种计算通常也不是非常准确的，因为它不支持闰年，并且如果某人的生日刚好是冬令时和夏令时（或夏令时和冬令时）

进行切换的午夜，这种计算也会出现错误。

21.4　在MySQL中计算日期

PHP没有提供更多内置的日期操作函数。很明显，我们可以编写自己的函数，但是务必考虑闰年和时令切换的时间。另一个选择是下载别人的函数。在PHP手册中，可以找到许多用户编写的函数，但是只有很少的一部分才是考虑全面的。

提示　在PHP 5.3版本中，增加了一些日期计算函数，包括date_add()、date_sub()和date_diff()函数。这些日期操作函数消除了必需使用MySQL来提供PHP以前版本所缺少的日期操作函数。

一个并不是非常好的选择是使用MySQL。MySQL提供了大量的日期操作函数，这些函数适用于UNIX时间戳以外的可供日期范围。必须连接MySQL服务器来运行一个MySQL查询，但是不使用数据库的数据。

如下所示的查询在1700年2月28日的基础上增加了一天，并且返回了结果日期：

```
select adddate('1700-02-28', interval 1 day)
```

1700年不是闰年，因此其结果是1700-03-01。

在MySQL手册中，可以找到大量描述和修改日期和时间的语法，网址为：http: //www.mysql.com/doc/en/Date_and_time_functions.html。

不幸的是，要获得两个日期之间的年数并不是一件容易的事情，因此生日例子还存在一些问题。我们可以很容易获得以天为单位的某人年龄，程序清单21-2所示的代码将年龄转换为年份，这并不是非常准确的。

程序清单21-2　mysql_calc_age.php——使用MySQL来计算某人基于生日的年龄

```
<?php
  // set date for calculation
  $day = 18;
  $month = 9;
  $year = 1972;

  // format birthday as an ISO 8601 date
  $bdayISO = date("c", mktime (0, 0, 0, $month, $day, $year));

  // use mysql query to calculate an age in days
  $db = mysqli_connect( localhost', 'user', 'pass');
  $res = mysqli_query($db, "select datediff(now(), '$bdayISO')");
  $age = mysqli_fetch_array($res);

  // convert age in days to age in years (approximately)
  echo "Age is ".floor($age[0]/365.25);
?>
```

在将生日格式化成一个ISO时间戳后，可以将如下所示的查询提交给MySQL：

```
select datediff(now(), '1972-09-18T00:00:00+10:00')
```

MySQL的now()函数通常将返回当前的日期和时间。MySQL的datediff()函数（在PHP 4.1.1版本引入）将两个日期相减并返回日期的差。

以上脚本的执行并不需要从一个表格选择数据，甚至选择一个数据库，但是必须使用有效的用户名和密码登录到一台MySQL服务器。

由于没有特定的内置函数可以用于这种计算，因此用来计算确切年份的SQL查询就比较复杂。这里，我们采用了捷径，用日期年龄除以365.25来获得年龄。如果对某人的生日进行如此计算，根据这个人一生可能经历的闰年数不同，年份生日可能会出现一年的偏差。

21.5　使用微秒

对于某些应用程序来说，以s（秒）来计量时间不够精确。如果希望以更短的时间段来计量时间，例如运行所有PHP脚本所需的时间，必须使用microtime()函数。

在PHP 5中，调用microtime()并且将参数get_as_float设置为true。当给出了这个可选参数后，这个调用将返回浮点数的时间戳。该时间戳与mktime()函数、time()函数或date()返回值相同，但是还有小数部分。

如下所示的语句：

```
echo number_format(microtime(true), 10, '.', '');
```

将生成类似于1174091854.84的输出。

在早期版本中，无法请求浮点数类型的输出。它是以字符串形式提供的。没有给出参数的microtime()函数调用将返回一个字符串，类似于“0.34380900 1174091816”。第一个数字是小数点部分，而第二个数字是整个秒数，该秒数是1970年1月1日以后的所有秒数。

与处理字符串相比，处理数字更加方便。因此在PHP 5中，最简单的方法就是调用参数为true的microtime()函数。

21.6　使用日历函数

PHP提供了一组日历函数，这些函数可以实现日期在不同的日历系统之间的转换。我们使用的主要日历有Gregorian、Julian和Julian Day Count。

Gregorian日历是大多数西方国家目前所使用的历法。Gregorian中的日期1582年10月15日，1582与Julian日历中的1582年10月5日等效。而在此日期以前，Julian日历是人们更常用的历法。不同的国家将原日历转换为Gregorian日历的时期不同，有些国家甚至在20世纪早期才转换。

除了这两个日历之外，我们可能还没有听说过Julian Day Count日历。该日历与UNIX时间戳有许多相似之处。它是从大约公元前4000年起的某个日期开始计算的日子数，自身并不是特别有意义，但是它对于格式之间的转换却非常有用。要将一个日历格式转换到另一个日历格式，我们首先要转换成Julian Day Count，然后再将其转换成要输出的日历。

要在UNIX下使用这些函数，必须已经在PHP中编译了日历扩展库，通过--enable-calendar选项实现。这些日历扩展库已经内置在Windows系统的安装中。

要体验这些函数，我们首先要了解这些可能用来将日期从Gregorian日历转换到Julian日历的函数原型：

```
int gregoriantojd (int month, int day, int year)
string jdtojulian(int julianday)
```

要转换一个日期，需要调用这两个函数：

```
$jd = gregoriantojd (9, 18, 1582);
echo jdtojulian($jd);
```

以上代码将以MM/DD/YYYY格式显示Julian日期。

这些函数的变体可以实现日期格式在Gregorian、Julian、French以及Jewish日历和UNIX时间戳之间转换。

21.7 进一步学习

如果要了解PHP和MySQL中更多的时间与日期函数，可以参阅PHP在线指南的相关部分，网址如下所示：http: //php.net/manual/en/ref.datetime.php以及http://dev.mysql.com/doc/refman/5.0/en/date-and-timefunctions. html。

如果要在日历之间进行转换，可以查看PHP日历函数的手册页： http: //php.net/manual/en/ref.calendar.php。

21.8 下一章

使用PHP可以实现的独特而有意义的事情之一就是创建动态图像。在第22章“创建图像”中，我们将讨论如何使用图形库函数来获得一些有趣而又有用的效果。