[使用date命令获取各种时间](http://blog.csdn.net/todd911/article/details/9092905)

获得当天的日期

date +%Y-%m-%d

输出： 2011-07-28

将当前日期赋值给DATE变量  
DATE=$(date +%Y%m%d)

有时候我们需要使用今天之前或者往后的日期，这时可以使用date的 -d参数

获取明天的日期

date -d next-day +%Y%m%d

获取昨天的日期

date -d last-day +%Y%m%d

获取上个月的年和月

date -d last-month +%Y%m

获取下个月的年和月  
date -d next-month +%Y%m

获取明年的年份  
date -d next-year +%Y

下面是一些date参数的说明和一些例子

名称 : date   
使用权限 : 所有使用者   
使用方式 : date [-u] [-d datestr] [-s datestr] [--utc] [--universal] [--date=datestr] [--set=datestr] [--help] [--version] [+FORMAT] [MMDDhhmm[[CC]YY][.ss]]   
说明 : date 能用来显示或设定系统的日期和时间，在显示方面，使用者能设定欲显示的格式，格式设定为一个加号后接数个标记，其中可用的标记列表如下 :   
时间方面 :   
% : 印出   
% %n : 下一行   
%t : 跳格   
%H : 小时(00..23)   
%I : 小时(01..12)   
%k : 小时(0..23)   
%l : 小时(1..12)   
%M : 分钟(00..59)   
%p : 显示本地 AM 或 PM   
%r : 直接显示时间 (12 小时制，格式为 hh:mm:ss [AP]M)   
%s : 从 1970 年 1 月 1 日 00:00:00 UTC 到目前为止的秒数 %S : 秒(00..61)   
%T : 直接显示时间 (24 小时制)   
%X : 相当于 %H:%M:%S   
%Z : 显示时区   
日期方面 :   
%a : 星期几 (Sun..Sat)   
%A : 星期几 (Sunday..Saturday)   
%b : 月份 (Jan..Dec)   
%B : 月份 (January..December)   
%c : 直接显示日期和时间   
%d : 日 (01..31)   
%D : 直接显示日期 (mm/dd/yy)   
%h : 同 %b   
%j : 一年中的第几天 (001..366)   
%m : 月份 (01..12)   
%U : 一年中的第几周 (00..53) (以 Sunday 为一周的第一天的情形)   
%w : 一周中的第几天 (0..6)   
%W : 一年中的第几周 (00..53) (以 Monday 为一周的第一天的情形)   
%x : 直接显示日期 (mm/dd/yy)   
%y : 年份的最后两位数字 (00.99)   
%Y : 完整年份 (0000..9999)   
若是不以加号作为开头，则表示要设定时间，而时间格式为 MMDDhhmm[[CC]YY][.ss]，   
其中 MM 为月份，   
DD 为日，   
hh 为小时，   
mm 为分钟，   
CC 为年份前两位数字，   
YY 为年份后两位数字，   
ss 为秒数   
把计 :   
-d datestr : 显示 datestr 中所设定的时间 (非系统时间)   
--help : 显示辅助讯息   
-s datestr : 将系统时间设为 datestr 中所设定的时间   
-u : 显示目前的格林威治时间   
--version : 显示版本编号   
例子 :   
显示时间后跳行，再显示目前日期 : date +%T%n%D   
显示月份和日数 : date +%B%d   
显示日期和设定时间(12:34:56) : date --date 12:34:56   
设置系统当前时间（12:34:56）：date --s 12:34:56   
注意 : 当你不希望出现无意义的 0 时(比如说 1999/03/07)，则能在标记中插入 - 符号，比如说 date +%-H:%-M:%-S 会把时分秒中无意义的 0 给去掉，像是原本的 08:09:04 会变为 8:9:4。另外，只有取得权限者(比如说 root)才能设定系统时间。 当你以 root 身分更改了系统时间之后，请记得以 clock -w 来将系统时间写入 CMOS 中，这样下次重新开机时系统时间才会持续抱持最新的正确值。   
-------------------------------------------------------------------   
扩展功能   
date 工具可以完成更多的工作，不仅仅只是打印出当前的系统日期。您可以使用它来得到给定的日期究竟是星期几，并得到相对于当前日期的相对日期。了解某一天是星期几   
GNU 对 date 命令的另一个扩展是 -d 选项，当您的桌上没有日历表时（UNIX 用户不需要日历表），该选项非常有用。使用这个功能强大的选项，通过将日期作为引号括起来的参数提供，您可以快速地查明一个特定的日期究竟是星期几：   
$ date -d "nov 22"   
Wed Nov 22 00:00:00 EST 2006   
$   
在本示例中，您可以看到今年的 11 月 22 日是星期三。   
所以，假设在 11 月 22 日召开一个重大的会议，您可以立即了解到这一天是星期三，而这一天您将赶到驻地办公室。   
获得相对日期   
d 选项还可以告诉您，相对于 当前日期若干天的究竟是哪一天，从现在开始的若干天或若干星期以后，或者以前（过去）。通过将这个相对偏移使用引号括起来，作为 -d 选项的参数，就可以完成这项任务。   
例如，您需要了解两星期以后的日期。如果您处于 Shell 提示符处，那么可以迅速地得到答案：   
$ date -d "2 week"   
关于使用该命令，还有其他一些重要的方法。使用 next/last指令，您可以得到以后的星期几是哪一天：   
$ date -d "next monday" (下周一的日期)   
$ date -d next-day +%Y%m%d（明天的日期）或者：date -d tomorrow +%Y%m%d   
$ date -d last-day +%Y%m%d(昨天的日期) 或者：date -d yesterday +%Y%m%d   
$ date -d last-month +%Y%m(上个月是几月)   
$ date -d next-month +%Y%m(下个月是几月)   
使用 ago 指令，您可以得到过去的日期：   
$ date -d "30 days ago"（30天前的日期）   
您可以使用负数以得到相反的日期：   
$ date -d ’dec 14 -2 weeks’ （相对:dec 14这个日期的两周前的日期）   
$ date -d ’-100 days’ (100天以前的日期)   
$ date -d ’50 days’(50天后的日期)   
这个技巧非常有用，它可以根据将来的日期为自己设置提醒，可能是在脚本或 Shell 启动文件中，如下所示：   
DAY=`date -d ’2 weeks’ +"%b %d"`   
if test "`echo $DAY`" = "Aug 16"; then echo ’Product launch is now two weeks away!’; fi   
  
##############################   
  
unix shell中的日期格式转换   
$ t\_t="Jul  1 21:29"   
$ date "+%G-%m-%d %H:%M:%S" -d "$t\_t" # $t\_t的格式是比较随意的   
2008-07-01 21:29:00   
$ date +%b/%d/%G -d "2008-07-01"   
Jul/01/2008  
  
unix shell中的日期之间间隔的天数   
$ expr '(' $(date +%s -d "2008-07-02") - $(date +%s -d "2008-05-30") ')' / 86400   
33   
  
shell中日期加减指定间隔单位   
增加36小时：   
$ a=`date +%Y-%m-%d`   
$ b=`date +%Y-%m-%d -d "$a +36 hours"`   
10天前：   
$ date -d "$a -10 days"   
Sun Jun 22 00:00:00 CST 2008   
  
以指定格式显示文件更改后最后日期，如yyyy-mm-dd hh24:mi:ss   
$ date "+%Y-%m-%d %H:%M:%S" -r test.bak   
2008-07-01 21:28:55