<https://www.cnblogs.com/carekee/articles/3252268.html>

# [linux c获得时间和设置时间](http://www.cnblogs.com/carekee/articles/3252268.html)

**[c-sharp]** [view plaincopyprint?](http://blog.csdn.net/mmbl007/article/details/6058048)

1. #include<time.h> //C语言的头文件
2. #include<stdio.h> //C语言的I/O
4. **void** main()
5. {
6. time\_t now; //实例化time\_t结构
7. **struct** tm \*timenow; //实例化tm结构指针
8. time(&now);
9. //time函数读取现在的时间(国际标准时间非北京时间)，然后传值给now
11. timenow = localtime(&now);
12. //localtime函数把从time取得的时间now换算成你电脑中的时间(就是你设置的地区)
14. printf("Local time is %s/n",asctime(timenow));
15. //上句中asctime函数把时间转换成字符，通过printf()函数输出
16. }

注释：time\_t是一个在time.h中定义好的结构体。而tm结构体的原形如下：

**[c-sharp]** [view plaincopyprint?](http://blog.csdn.net/mmbl007/article/details/6058048)

1. **struct** tm
2. {
3. **int** tm\_sec;//seconds 0-61
4. **int** tm\_min;//minutes 1-59
5. **int** tm\_hour;//hours 0-23
6. **int** tm\_mday;//day of the month 1-31
7. **int** tm\_mon;//months since jan 0-11
8. **int** tm\_year;//years from 1900
9. **int** tm\_wday;//days since Sunday, 0-6
10. **int** tm\_yday;//days since Jan 1, 0-365
11. **int** tm\_isdst;//Daylight Saving time indicator
12. };

#include "time.h"

time\_t time(time\_t \*timer);   
调用后将当前系统时间与1900年1月1日相差的秒数存入到timer中，timer可看成是一个长整型数   
struct tm\* localtime(const time\_t \*timer)   
将time()函数调用的结果做为参数传入到localtime()函数中就能得到当前时间和日期，注意得到的年是和1970的差值，月份是和1月的差值

struct tm是一个结构体，定义如下

**[c-sharp]** [view plaincopyprint?](http://blog.csdn.net/mmbl007/article/details/6058048)

1. **struct** tm
2. {
3. **int** tm\_sec; //当前秒
4. **int** tm\_min; //当前分钟
5. **int** tm\_hour; //当前小时
6. **int** tm\_mday; //当前在本月中的天，如11月1日，则为1
7. **int** tm\_mon; //当前月，范围是0~11
8. **int** tm\_year; //当前年和1900的差值，如2006年则为36
9. **int** tm\_wday; //当前在本星期中的天，范围0~6
10. **int** tm\_yday; //当前在本年中的天，范围0~365
11. **int** tm\_isdst; //这个我也不清楚
12. }

求当前时间的示例

**[c-sharp]** [view plaincopyprint?](http://blog.csdn.net/mmbl007/article/details/6058048)

1. **int** getSystemTime()
2. {
3. time\_t timer;
4. **struct** tm\* t\_tm;
5. time(&timer);
6. t\_tm = localtime(&timer);
7. printf("today is %4d%02d%02d%02d%02d%02d/n", t\_tm.tm\_year+1900,
8. t\_tm.tm\_mon+1, t\_tm.tm\_mday, t\_tm.tm\_hour, t\_tm.tm\_min, t\_tm.tm\_sec);
9. **return** 0;
10. }
12. /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*
13. 设置操作系统时间
14. 参数:\*dt数据格式为"2006-4-20 20:30:30"
15. 调用方法:
16. char \*pt="2006-4-20 20:30:30";
17. SetSystemTime(pt);
18. \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/
19. **int** SetSystemTime(**char** \*dt)
20. {
21. **struct** rtc\_time tm;
22. **struct** tm \_tm;
23. **struct** timeval tv;
24. time\_t timep;
25. sscanf(dt, "%d-%d-%d %d:%d:%d", &tm.tm\_year,
26. &tm.tm\_mon, &tm.tm\_mday,&tm.tm\_hour,
27. &tm.tm\_min, &tm.tm\_sec);
28. \_tm.tm\_sec = tm.tm\_sec;
29. \_tm.tm\_min = tm.tm\_min;
30. \_tm.tm\_hour = tm.tm\_hour;
31. \_tm.tm\_mday = tm.tm\_mday;
32. \_tm.tm\_mon = tm.tm\_mon - 1;
33. \_tm.tm\_year = tm.tm\_year - 1900;
35. timep = mktime(&\_tm);
36. tv.tv\_sec = timep;
37. tv.tv\_usec = 0;
38. **if**(settimeofday (&tv, (**struct** timezone \*) 0) < 0)
39. {
40. printf("Set system datatime error!/n");
41. **return** -1;
42. }
43. **return** 0;
44. }

其他时间的函数和结构还有：

timeval结构

#include <include/linux/time.h>

struct timeval   
{   
time\_t tv\_sec;   
susecond\_t tv\_usec; //当前妙内的微妙数   
};

tms结构   
保存着一个进程及其子进程使用的cpu时间

struct tms   
{   
clock\_t tms\_utime;   
clock\_t tms\_stime;   
clock\_t tms\_cutime;   
clock\_t tms\_cstime;   
}

timer\_struct结构

#include <include/linux/timer.h>

struct timer\_struct   
{   
unsigned long expires; //定时器被激活的时刻   
void (\*fn)(void); //定时器激活后的处理函数   
}

utime函数   
更改文件的存取和修改时间

int utime(const char pathname, const struct utimbuf \*times) // return value 0 or -1

times 为空指针，存取和修改时间设置为当前时间

struct utimbuf   
{   
time\_t actime;   
time\_t modtime;

}

分类: [Linux/Unix](http://www.cnblogs.com/carekee/category/416605.html)