

Linux上的一个小程序--进度条

- 回车换行：
 - 回车：回到当前行的最开始
 - 换行：列不变，新起一行
- printf 如果不带\n 数据不会被立即刷新，而是被暂存在C语言级别的缓冲区当中(显示器设备刷新策略是行刷新，遇到\n即进行刷新)

字符串是先存在了缓存区中，然后刷新到显示器中

字符串 → c语言开的缓存区 → 显示器

```
-rw-rw-r-- 1 xifeng xifeng 75 Oct 20 16:43 Makefile
-rwxrwxr-x 1 xifeng xifeng 8608 Oct 20 16:44 proc
-rw-rw-r-- 1 xifeng xifeng 343 Oct 20 16:31 proc.c
-rwxrwxr-x 1 xifeng xifeng 8408 Oct 20 16:44 test
-rw-rw-r-- 1 xifeng xifeng 279 Oct 20 16:43 test.c
[xifeng@VM-16-14-centos lesson7]$ ./test
Hello word[xifeng@VM-16-14-centos lesson7]$ cat test.c
#include <stdio.h>
#include<unistd.h>
int main()
{
    //printf("Hello word\n");
    printf("Hello word");
    sleep(5);
    return 0;
}
```

带\n时，显示器先出现Hello word 而不带则是先睡眠在刷新

我们知道执行语句是顺序执行，所以 printf一定先于sleep执行，可为什么 Hello word会在睡眠之后再显示呢？

- fflush(就是立即刷新)：即不带\n也会刷新

fflush(stdout);

FFLUSH(3) Linux Programmer's Manual FFLUSH(3)

NAME

fflush - flush a stream

SYNOPSIS

```
#include <stdio.h>

int fflush(FILE *stream);
```

DESCRIPTION

C程序，默认会打开三个输入输出流：stdin(键盘) stdout(显示器) stderr(显示器)

NAME

stdin, stdout, stderr - standard I/O streams

SYNOPSIS

```
#include <stdio.h>

extern FILE *stdin;
extern FILE *stdout;
extern FILE *stderr;
```

- \r 只回车不换行
- 这样就可以写一个简单的倒计时

```
1: test.c
1 #include <stdio.h>
2 #include <unistd.h>
3 int main()
4 {
5     int count = 5;
6     while(count)
7     {
8         printf("%d\r", count--);
9         fflush(stdout);
10        sleep(1);
11    }
12    return 0;
13 }
```

只回车, 不换行

立即显示

隔一秒

凡是显示到显示器上面的内容都是字符, 凡事从键盘读取的内容都是字符(键盘, 显示器是字符设备); 如果把=5 换成10会出现90 80 70 60。。。因为当成字符的话9只是一个字符而10是两个字符, 所以只会覆盖掉第一个字符; 解决方法就是%2d即可

- o usleep 睡眠毫秒

```
USLEEP(3)          Linux Programmer's Manual          USLEEP(3)

NAME
    usleep - suspend execution for microsecond intervals

SYNOPSIS
    #include <unistd.h>

    int usleep(useconds_t usec);  毫秒

Feature Test Macro Requirements for glibc (see feature_test_macros(7)):

usleep():
    Since glibc 2.12:
        _BSD_SOURCE ||
        (_XOPEN_SOURCE >= 500 ||
         _XOPEN_SOURCE && _XOPEN_SOURCE_EXTENDED) &&
        !(_POSIX_C_SOURCE >= 200809L || _XOPEN_SOURCE >= 700)
    Before glibc 2.12:
        _BSD_SOURCE || _XOPEN_SOURCE >= 500 ||
        _XOPEN_SOURCE && _XOPEN_SOURCE_EXTENDED

DESCRIPTION
    The usleep() function suspends execution of the calling thread for (at least) usec microseconds. The sleep may be lengthened slightly by any system activity or by the time spent processing the call or by the granularity of system timers.

RETURN VALUE
    The usleep() function returns 0 on success. On error, -1 is returned, with errno set to indicate the cause of the error.
```

- 这就是进度条的代码

```
1 #include<stdio.h>
2 #include<string.h>
3 #include<unistd.h>
4 int main()
5 {
6     #define MAX 100
7     char bar[MAX+1];
8     memset(bar, '\0', sizeof(bar));
9     int i=0;
10    const char arr[]="/-\\\";
11    while(i<=100)
12    {
13        printf("\033[31m][%-100s][%3d%%]%c\r ", bar, i, arr[i%4]);
14        fflush(stdout);
15        bar[i]='=';
16        i++;
17        usleep(60000);
18    }
19    printf("\n");
20 }
```

- [%-100s]--->默认是右对齐，加个-是左对齐
- [%3d%%]---> %%是打印出来%
- \r 只回车不换行
- \033[31m]是printf的配色方案