# ILoveDBA

昵称: ILoveDBA
园龄: 3年7个月
粉丝: 6
关注: 0
+加关注

博客园 首页 博问 闪存 新随笔 联系 订阅XML 管理
随笔-9 评论-9 文章-6 trackbacks-0

< 2010年3月 >

| 日 | 一 | 二 | 三 | 四 | 五 | 六 |
|---|---|---|---|---|---|---|
| 28 | 1 | 2 | 3 | 4 | 5 | 6 |
| 7 | 8 | 9 | 10 | 11 | 12 | 13 |
| 14 | 15 | 16 | 17 | 18 | 19 | 20 |
| 21 | 22 | 23 | 24 | 25 | 26 | 27 |
| 28 | 29 | 30 | 31 | 1 | 2 | 3 |
| 4 | 5 | 6 | 7 | 8 | 9 | 10 |

搜索
找找看
谷歌搜索

常用链接
我的随笔
我的评论
我的参与
最新评论
我的标签

我的标签
oracle(2)
shell(1)
VSFTP配置虚拟用户(1)
DBA(1)
DML(1)
linux(1)

随笔档案
2010年6月 (5)
2010年5月 (1)
2010年3月 (1)
2010年1月 (2)

文章分类
Ajax

## 堆 和 栈的 区别(经典)

### 转载一篇理解堆和栈区别的好文章

此文章虽然是面向C/C++程序员写得，但是对咱们Java程序员还是很有帮助的。

堆和栈的区别

一、预备知识—程序的内存分配
一个由C/C++编译的程序占用的内存分为以下几个部分
1、栈区（stack）— 由编译器自动分配释放 ， 存放函数的参数值，局部变量的值等。其
操作方式类似于数据结构中的栈。
2、堆区（heap） — 一般由程序员分配释放， 若程序员不释放，程序结束时可能由OS回
收 。注意它与数据结构中的堆是两回事，分配方式倒是类似于链表，呵呵。
3、全局区（静态区）（static）—，全局变量和静态变量的存储是放在一块的，初始化的
全局变量和静态变量在一块区域， 未初始化的全局变量和未初始化的静态变量在相邻的另
一块区域。 - 程序结束后由系统释放。
4、文字常量区 —常量字符串就是放在这里的。 程序结束后由系统释放
5、程序代码区—存放函数体的二进制代码。

二、例子程序
这是一个前辈写的，非常详细

```
//main.cpp
int a = 0; 全局初始化区
char *p1; 全局未初始化区
main()
{
int b; 栈
char s[] = "abc"; 栈
char *p2; 栈
```

ASP.net(1)
JaveScript
Linux(2)
MSSQL(2)
Mysql
Oracle(4)
PHP

最新评论

阅读排行榜

1. 堆 和 栈的 区别(经典)(9849)
2. ERP 形像比喻(8747)
3. ORACLE DBA 之路（一）---新手上路(2498)
4. oracle 9i启用Execution Plan的方法(355)
5. 利用rman创建standby(一)(转http://ningoo.itpub.net/post/2149/230943）(270)

评论排行榜

1. ERP 形像比喻(5)
2. ORACLE DBA 之路（一）---新手上路(3)
3. 一道类似SQL题目，接近实际,(0)
4. C#正则表达式整理备忘(0)
5. 堆 和 栈的 区别(经典)(0)

推荐排行榜

1. ERP 形像比喻(5)
2. 堆 和 栈的 区别(经典)(4)
3. ORACLE DBA 之路（一）---新手上路(1)

char *p3 = "123456"; 123456\0在常量区，p3在栈上。
static int c =0;　全局（静态）初始化区
p1 = (char *)malloc(10);
p2 = (char *)malloc(20);
分配得来得10和20字节的区域就在堆区。
strcpy(p1, "123456"); 123456\0放在常量区，编译器可能会将它与p3所指向的"123456"
优化成一个地方。
}

二、堆和栈的理论知识
2.1申请方式
stack:
由系统自动分配。 例如，声明在函数中一个局部变量 int b; 系统自动在栈中为b开辟空
间
heap:
需要程序员自己申请，并指明大小，在c中malloc函数
如p1 = (char *)malloc(10);
在C++中用new运算符
如p2 = new char[10];
但是注意p1、p2本身是在栈中的。

2.2
申请后系统的响应
栈：只要栈的剩余空间大于所申请空间，系统将为程序提供内存，否则将报异常提示栈溢
出。
堆：首先应该知道操作系统有一个记录空闲内存地址的链表，当系统收到程序的申请时，
会遍历该链表，寻找第一个空间大于所申请空间的堆结点，然后将该结点从空闲结点链表
中删除，并将该结点的空间分配给程序，另外，对于大多数系统，会在这块内存空间中的
首地址处记录本次分配的大小，这样，代码中的delete语句才能正确的释放本内存空间。
另外，由于找到的堆结点的大小不一定正好等于申请的大小，系统会自动的将多余的那部
分重新放入空闲链表中。

2.3申请大小的限制
栈：在Windows下,栈是向低地址扩展的数据结构，是一块连续的内存的区域。这句话的意
思是栈顶的地址和栈的最大容量是系统预先规定好的，在WINDOWS下，栈的大小是2M（也有
的说是1M，总之是一个编译时就确定的常数），如果申请的空间超过栈的剩余空间时，将
提示overflow。因此，能从栈获得的空间较小。
堆：堆是向高地址扩展的数据结构，是不连续的内存区域。这是由于系统是用链表来存储
的空闲内存地址的，自然是不连续的，而链表的遍历方向是由低地址

向高地址。堆的大小
受限于计算机系统中有效的虚拟内存。由此可见，堆获得的空间比较灵活，也比较大。

2.4申请效率的比较:
栈由系统自动分配，速度较快。但程序员是无法控制的。
堆是由new分配的内存，一般速度比较慢，而且容易产生内存碎片,不过用起来最方便.
另外，在WINDOWS下，最好的方式是用VirtualAlloc分配内存，他不是在堆，也不是在栈是
直接在进程的地址空间中保留一块内存，虽然用起来最不方便。但是速度快，也最灵活。

2.5堆和栈中的存储内容
栈:　在函数调用时，第一个进栈的是主函数中后的下一条指令（函数调用语句的下一条可
执行语句）的地址，然后是函数的各个参数，在大多数的C编译器中，参数是由右往左入栈
的，然后是函数中的局部变量。注意静态变量是不入栈的。
当本次函数调用结束后，局部变量先出栈，然后是参数，最后栈顶指针指向最开始存的地
址，也就是主函数中的下一条指令，程序由该点继续运行。
堆:　一般是在堆的头部用一个字节存放堆的大小。堆中的具体内容由程序员安排。

2.6存取效率的比较

```
char s1[] = "aaaaaaaaaaaaaa";
char *s2 = "bbbbbbbbbbbbbbbb";
aaaaaaaaaaa是在运行时刻赋值的;
而bbbbbbbbbbb是在编译时就确定的;
但是，在以后的存取中，在栈上的数组比指针所指向的字符串(例如堆)快。
比如:
#include
void main()
{
char a = 1;
char c[] = "1234567890";
char *p ="1234567890";
a = c[1];
a = p[1];
return;
}
对应的汇编代码
10: a = c[1];
00401067 8A 4D F1 mov cl,byte ptr [ebp-0Fh]
0040106A 88 4D FC mov byte ptr [ebp-4],cl
11: a = p[1];
0040106D 8B 55 EC mov edx,dword ptr [ebp-14h]
```

00401070 8A 42 01 mov al,byte ptr [edx+1]
00401073 88 45 FC mov byte ptr [ebp-4],al
第一种在读取时直接就把字符串中的元素读到寄存器cl中，而第二种则要先把指针值读到
edx中，再根据edx读取字符，显然慢了。

2.7小结:
堆和栈的区别可以用如下的比喻来看出:
使用栈就象我们去饭馆里吃饭，只管点菜（发出申请）、付钱、和吃（使用），吃饱了就
走，不必理会切菜、洗菜等准备工作和洗碗、刷锅等扫尾工作，他的好处是快捷，但是自
由度小。
使用堆就象是自己动手做喜欢吃的菜肴，比较麻烦，但是比较符合自己的口味，而且自由
度大。（经典！）



ILoveDBA
关注 - 0
粉丝 - 6
+加关注



« 上一篇: C#正则表达式整理备忘
» 下一篇: ORACLE DBA 之路（一）---新手上路

posted on 2010-03-05 11:45 ILoveDBA 阅读(9849) 评论(0)
编辑 收藏