切換導航 Farseerfc的小窩

- 文A
  - 繁體
  - 简体
  - English
  - 日本語
  - 文A繁體
  - 文A简体
  - 文AEnglish
  - 文A日本語
-
  - About
  - Links
  - About
  - Links
-
  - Import
  - Life
  - Tech

C++ Tricks 2.7 I386平臺的其它函數調用模型

從 farseerfc.wordpress.com 導入

2.7 I386平臺的其它函數調用模型

上文介紹的只是I386平臺上C函數調用的標準模型，被稱作__cdecl。事實上，Microsoft Visual C++編譯器還支持其它一些函數調用模型，所有調用模型名稱皆以雙下劃線開頭，下面列出所有函數調用模型的異同：

1 __cdecl

參數壓棧順序：逆序(從右至左)

參數堆棧恢復者：主調函數(caller)

__cdecl明確地指出函數使用C函數調用模型，這是默認的調用模型。

2 __stdcall

參數壓棧順序：逆序(從右至左)

參數堆棧恢復者：被調函數(callee)

__stdcall是微軟所謂的標準調用模型。可惜的是它與__cdecl不兼容。幾乎所有的Win32API函數使用這種函數調用模型，希望在DLL之間，或者在程序和WinNT操作系統之間傳遞函數指針的函數也應該使用這種模型。與__cdecl模型的不同之處在於，__stdcall模型下由被調函數恢復堆棧。主調函數在call語句之後，不需要再加上add語句。而被調函數的ret語句則被添加一個參數，代表函數參數堆棧的長度。因此，被調函數需要明確的知曉函數參數的數量和類型，所以在__stdcall模型下不支持可變參數表，所有參數必須寫明。

3 __thiscall

參數壓棧順序：逆序(從右至左)，this用ecx傳遞。

參數堆棧恢復者：被調函數(callee)

__thiscall是VC編譯器中類的非靜態成員函數(non-static member functon)的默認調用模型。但是如果此成員函數有可變參數表，VC編譯器會使用__cdecl。和__stdcall一樣，__thiscall由被調函數

恢復堆棧。比較獨特的是__thiscall會通過ecx寄存器傳遞成員函數的this指針，而__cdecl下this指針是通過在參數表最前面增加一個函數參數來傳遞的。__thiscall是VC編譯器對this指針的使用的一種優化，大大提高了面向對象程序的效率。在VC2003及之前的編譯器上__thiscall不是一個關鍵字，不能被顯式指定。但可以給成員函數顯式指定__cdecl來避免使用__thiscall。

4 __fastcall

參數壓棧順序：逆序(從右至左)，前兩個32位函數參數放入ecx和edx中

參數堆棧恢復者：被調函數(callee)

快速函數調用模型，將前兩個32位函數參數放入ecx和edx中，其餘參數再逆序壓棧。使用的是和__thiscall類似的優化技術，加快函數調用，適合運用在小型inline函數上。同樣使用__stdcall形式的被調函數恢復堆棧，所以不支持可變參數表。

5 __pascal

參數壓棧順序：正序(從左至右)

參數堆棧恢復者：被調函數(callee)

過程式編程語言Pascal所使用的函數調用模型，由此得名。也是16位版本的Windows使用的API模型，過時的模型，現在已經廢棄且禁止使用。你會看到有些書本仍會不時提到它，所以需要注意。__pascal是正序壓棧，這與大部分I386函數模型都不相同。與__stdcall一樣，由被調者恢復堆棧，不支持可變參數表。歷史上曾有過的別名PASCAL、pascal、_pascal(單下劃線)，現在都改成了__stdcall的別名，與__pascal(雙下劃線)不同。

6 其它函數調用模型，以及模型別名。

__syscall：操作系統內部使用的函數調用模型，由用戶模式向核心模式跳轉時使用的模型。由於用戶模式和核心模式使用不同的棧，所以沒辦法使用棧來傳遞參數，所有參數通過寄存器傳遞，這限制了參數的數量。用戶模式編程中不允許使用。

__fortran：數學運算語言fortran使用的函數模型，由此得名。在C中調用由fortran編譯的函數時使

用。

__clrcall：微軟.Net框架使用的函數模型，託管(Managed)C++默認使用，也可以從非託管代碼調用託管函數時使用。參數在託管棧上正序(從左至右)壓棧，不使用普通棧。

CALLBACK、PASCAL、WINAPI、APIENTRY、APIPRIVATE：I386平臺上是__stdcall的別名

WINAPIV：I386平臺上是__cdecl的別名

7 函數調用模型的指定

函數調用模型的指定方式和inline關鍵字的指定方式相同，事實上，inline可以被看作是C++語言內建的一種函數調用模型。唯一不同的是，聲明函數指針時，也要指明函數調用模型，而inline的指針是不能指明的，根本不存在指向inline函數的指針。比如：

```
int CALLBACK GetVersion();

int (CALLBACK * pf)()=GetVersion;
```