# إتفاقيات الإستدعاء
# (calling conventions)
# C / C++ / VC++

الـ calling convention تصف الطريقة التي ستُستدعى فيها الدوال وكيف ستمرر المعاملات للدالة المُستدعاة ومن سيتولى عملية التنظيف بعد الإنتهاء.

**هناك عدّة أساليب لاستدعاء الدوال وأشهرها :**

- __cdecl
- __stdcall
- __fastcall
- Thiscall

## C calling convention (__cdecl)

هذة الاتفاقية أو الاسلوب هو الافتراضي لبرامج الـ C/C++ ، أو يمكن أختيارها بشكل أفتراضى من خلال إعدادات المترجم(compiler) بأسم /Gd ويمكننا الإعلان على الدالة لكى تستخدم هذة الاتفاقية بأستخدام __cdecl كما فى المثال التالى :

```
int __cdecl sumExample (int a, int b);
```

**مميزات هذة الاتفاقية أو الاسلوب :**

سوف تمرر وسائط الدالة من اليسار إلى اليمين وسوف تخزن فى المكدس(stack) .

سوف يتم تنظيف المكدس(stack) بواسطة المستدعى(caller) .

سوف يتم إضافة علامة _ قبل أسم الدالة .

**مثال: (شكل كود الاسملبى)**

```
; // push arguments to the stack, from right to left
push    3
push    2

; // call the function
call    _sumExample

; // cleanup the stack by adding the size of the arguments to ESP register
add     esp,8

; // copy the return value from EAX to a local variable (int c)
mov     dword ptr [c],eax
```

والدالة التى يتم أستدعائها يكون شكلها كالاتى:

```
; // function prolog
  push        ebp
  mov         ebp,esp
  sub         esp,0C0h
  push        ebx
  push        esi
  push        edi
  lea         edi,[ebp-0C0h]
  mov         ecx,30h
  mov         eax,0CCCCCCCCh
  rep stos    dword ptr [edi]

; //    return a + b;
  mov         eax,dword ptr [a]
  add         eax,dword ptr [b]

; // function epilog
  pop         edi
  pop         esi
  pop         ebx
  mov         esp,ebp
  pop         ebp
  ret
```

## Standard calling convention (__stdcall)

هذة الاتفاقية أو الاسلوب عادة ما تستخدم عند استدعاء دوال الـ Win32 API ، وكما نرى كثيرا الكلمة WINAPI قبل أسم الدالة ، فهى فى الحقيقة هو ثابت قيمة هذا الثابت هى الاتفاقية __stdcall أى أن كأنك تستخدم هذة الاتفاقية ولكن تم تغيير أسمها إلى WINAPI ، وهو معرف فى ملفات المصدر بهذا الشكل .

```
#define WINAPI __stdcall
```

ويمكننا أستخدام هذة الاتفاقية بهذا الشكل مع الدالة :

```
int __stdcall sumExample (int a, int b);
```

ويمكن أيضا أن نجعلها الافتراضية وذلك من خلال إعدادات المترجم(compiler) بأسم /Gz .

### مميزات هذة الاتفاقية أو الاسلوب :

سوف تمرر وسائط الدالة من اليسار إلى اليمين وسوف تخزن فى المكدس(stack) .

سوف يتم تنظيف المكدس(stack) بواسطة الدالة التى سيتم استدعائها(called function) .

سيتم إضافة الرمز _ قبل أسم الدالة ثم الرمز @ وعدد البايتات المطلوبة فى مساحة المكدس(stack) بعد أسم الدالة .

### مثال: (شكل كود الاسملبى)

```
; // push arguments to the stack, from right to left
  push      3
  push      2

; // call the function
  call      _sumExample@8

; // copy the return value from EAX to a local variable (int c)
  mov       dword ptr [c],eax
```

والدالة التى يتم أستدعائها يكون شكلها كالاتى:

```
; // function prolog goes here (the same code as in the __cdecl example)

; //    return a + b;
  mov       eax,dword ptr [a]
  add       eax,dword ptr [b]

; // function epilog goes here (the same code as in the __cdecl example)

; // cleanup the stack and return
  ret       8
```

## Fast calling convention (__fastcall)

هذة الاتفاقية أو الاسلوب يدل على أن الوسائط سيتم تخزينها فى مسجلات المعالج بدلا من تخزينها فى المكدس(stack) ، إذا كان ذلك ممكننا . وهذا الاسلوب سوف يكون أسرع لان التعامل مع مسجلات المعالج يكون أسرع من التعامل مع المكدس .
ويمكننا أستخدام هذا الاسلوب مع الدوال بهذا الشكل:

```
int __fastcall sumExample (int a, int b);
```

ويمكن أيضا أن نجعلها الافتراضية وذلك من خلال إعدادات المترجم(compiler) بأسم /Gr .

### مميزات هذة الاتفاقية أو الاسلوب :

أول وسيطين فى الدالة تحتاج 32 بت أو أقل ، ويتم تخزينهم فى مسجلات المعالج ECX و EDX وبقية الوسائط الاخرى يتم تخزينها فى المكدس(stack) ، وسيتم تمرير وسائط الدالة من اليمين إلى اليسار .

سوف يتم تنظيف المكدس(stack) وذلك بواسطة الدالة التى سيتم استدعائها .

سيتم إضافة الرمز @ ثم أسم الدالة ثم الرمز @ مرة أخرى ثم عدد البايتات المطلوبة فى مساحة المكدس(stack) .

```
; // put the arguments in the registers EDX and ECX
  mov       edx,3
  mov       ecx,2

; // call the function
  call      @fastcallSum@8

; // copy the return value from EAX to a local variable (int c)
  mov       dword ptr [c],eax
```

## مثال: (شكل كود الاسملبى)

والدالة التى يتم أستدعائها يكون شكلها كالاتى:

```
; // function prolog

  push      ebp
  mov       ebp,esp
  sub       esp,0D8h
  push      ebx
  push      esi
  push      edi
  push      ecx
  lea       edi,[ebp-0D8h]
  mov       ecx,36h
  mov       eax,0CCCCCCCCh
  rep stos  dword ptr [edi]
  pop       ecx
  mov       dword ptr [ebp-14h],edx
  mov       dword ptr [ebp-8],ecx
; // return a + b;
  mov       eax,dword ptr [a]
  add       eax,dword ptr [b]
;// function epilog
  pop       edi
  pop       esi
  pop       ebx
  mov       esp,ebp
  pop       ebp
  ret
```

## Thiscall

هذة الاتفاقية أو الاسلوب هو الافتراضى عند التعامل مع الدوال التى تكون ضمن كلاس أو تابعة لكلاس .

### مميزات هذة الاتفاقية أو الاسلوب :

وسائط الدوال الاعضاء يتم تمريرها من اليمين إلى اليسار، ويتم تخزينها فى المكدس(stack)، أما الكلمة المحجوزة this تخزن فى مسجل المعالج ECX .

سوف يتم تنظيف المكدس(stack) بواسطة الدالة التى سيتم استدعائها(called function) .

```
struct CSum
{
   int sum ( int a, int b) {return a+b;}
};
```

ويتم أستخدام هذة الاتفاقية أو الاسلوب عند أستخدام وبشكل أفتراضى مع الدوال الاعضاء كما فى المثال التالى:

## مثال: (شكل كود الاسملبى)

```
push      3
push      2
lea       ecx,[sumObj]
call      ?sum@CSum@@QAEHHH@Z          ; CSum::sum
mov       dword ptr [s4],eax
```

والدالة التى يتم أستدعائها يكون شكلها كالاتى:

```
push      ebp
mov       ebp,esp
sub       esp,0CCh
push      ebx
push      esi
push      edi
push      ecx
lea       edi,[ebp-0CCh]
mov       ecx,33h
mov       eax,0CCCCCCCCh
rep stos  dword ptr [edi]
pop       ecx
mov       dword ptr [ebp-8],ecx
mov       eax,dword ptr [a]
add       eax,dword ptr [b]
pop       edi
pop       esi
pop       ebx
mov       esp,ebp
pop       ebp
ret       8
```

لكن ماذا يحدث إلى كان هناك دالة عضو فى كلاس وهذة الدالة تحتوى على عدة وسائط ، فى هذة الحالة سيتم أستخدام الاتفاقية أو الاسلوب __cdecl ، و this سيتم دفعها إلى المكدس(stack) فى الاخر .

# أحمد فتحى


# جمهورية مصر العربية


البريد الالكترونى

ahmed_fa@hotmail.com


برعاية

موقع زيروجيت

0gate.com



طبعة

2012