

المنتدىء في البرمجة على نظام الويندوز

التأليف : خليل أونيس

من منتدى: عروة.نت

www.orwah.net

الطبعة الأولى

مقدمة

بسم الله الرحمن الرحيم

و الصلاة و السلام على خاتم الأنبياء و المرسلين

أخي الكريم , أقدم إليك هذا الكتاب المجاني و المتواضع الذي سيكون به الأساسيات في البرمجة بلغة السي++ على نظام الويندوز. سأحدث في هذا الكتاب عن بعض الأساسيات في السي++ باستخدام دوال الـ Application Programming Interface(API) .

أتمنى أن لا تكون أخطاء في هذا الكتاب و إن كانت أرجو تصحيحها لأن هذا الكتاب ملك لأي شخص للعربي مسلم و إن كانت لديك أي اقتراحات أو ملاحظات فأذهب إلى منتدى عروقة.نت في قسم السي++ . هناك يمكنك أخي الكريم إضافة أي تعاليق أو إقتراحات.

أساسيات

* لن تأخذ وقتا كبيرا في الانتهاء من هذا الكتاب الذي يقدم
إليك أساسيات في البرمجة باستخدام دوال API على نظام
الويندوز باستعمال لغة السي و بأبسط الطرق .
* كل ما تحتاجه الآن هو مصرفه للسي ++
أنا أستعمل فيجوال السي ++ 6.0 و المقدم من شركة ميكروسوفت.
آخر نسخ هي

Visual C++ Express Edition Beta 2

و التي يمكنك تحميلها من

<http://go.microsoft.com/?linkid=2676892>

الفصل الأول:

* في هذا الفصل سنشرح كيفية إنشاء نافذة بسيطة و إظهارها على الويندوز .

* يجب عليك الآن تتبع الخطوات التالية :

- 1- قم بتشغيل الفيجوال السي++
- 2- من القائمة اختر `File`
- 3- ثم اختر `New` أي يمكنك الضغط على الأزرار `Ctrl+N`
- 4- من `Project` اختر `Win32 Application`
- 5- من `Project Name` و الذي هو اسم المشروع اكتب `Chapter 1`
- 6- ثم اضغط `Ok`
- 7- الآن تأتي نافذة بعنوان `Win32 application - step 1`
of 1 اختر `An Empty Project`
- 8- ثم اضغط على `Finish`
- 9- تأتي نافذة جديدة ... اضغط على الزر `Ok`
- 10- ثم اختر مرة أخرى `File -> New`
- 11- من `Files` اختر `c++ source file` و من `file` اكتب `Name`
- 12- ثم اضغط على `Ok`
- 13- ثم اكتب هذا الكود :

```

#include<windows.h>

LRESULT CALLBACK WndProc(HWND          hWnd,
                                UINT     Msg,
                                WPARAM    wParam,
                                LPARAM    lParam)
{
    switch(Msg)
    {
        case WM_DESTROY:
            PostQuitMessage(0);
            break;

    }

    return DefWindowProc(hWnd, Msg, wParam, lParam);
}

INT WINAPI WinMain(HINSTANCE          hInstance,
                   HINSTANCE          hPrevInstance,
                   LPSTR               lpCmdLine,
                   INT                nShowCmd)
{
    WNDCLASSEX wc = {sizeof(WNDCLASSEX),
                     0,
                     WndProc,
                     0, 0,
                     hInstance,
                     LoadIcon(NULL, IDI_WINLOGO),
                     LoadCursor(NULL, IDC_ARROW),
                     (HBRUSH)(COLOR_WINDOW+2),
                     NULL,
                     "ClassName",
                     NULL};

    RegisterClassEx(&wc);

    HWND hWnd = CreateWindow("ClassName",
                             "Chapter 1",
                             WS_OVERLAPPEDWINDOW,
                             CW_USEDEFAULT,CW_USEDEFAULT,
                             640, 480,
                             NULL,
                             NULL,
                             hInstance,
                             NULL);

    ShowWindow(hWnd, nShowCmd);
    UpdateWindow(hWnd);

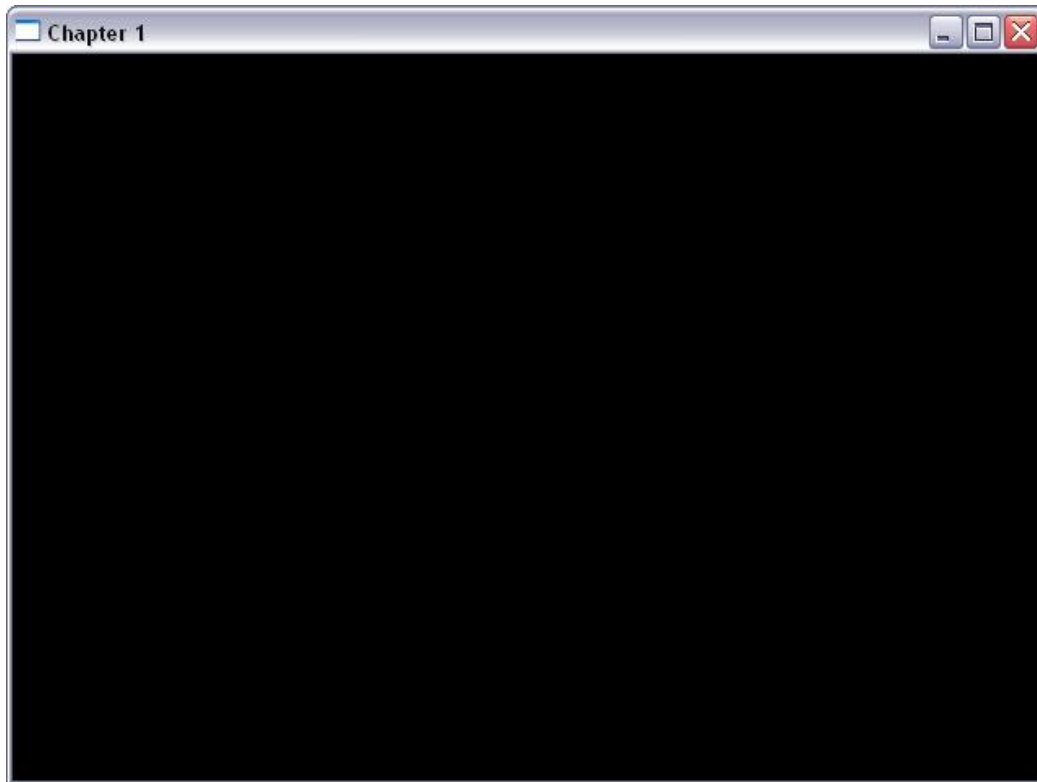
    MSG msg;
    while(GetMessage(&msg, NULL, 0, 0))
    {
        TranslateMessage(&msg);
        DispatchMessage(&msg);
    }

    return 0;
}

```

14 - ثم اضغط على الزر F7

15 - ثم اضغط على Ctrl+F5



* هذا أبسط كود يمكن كتابته و هو لإنشاء نافذة بسيطة.

* الآن نبدأ الشرح المبسط لهذا المثال:

السطر الأول `#include<windows.h>` هو ملف رئيسي موجودة فيه مجموعة من الدوال و الفئات و كائنات و غيرها ...

السطر الثاني `LRESULT CALLBACK WndProc(HWND hWnd, UINT Msg, WPARAM wParam, LPARAM lParam)`

يمثل إجراء النافذة الذي يعالج رسائل النافذة و به أربع

بارامترات:

- البارامتر الأول `HWND hWnd` هو مقبض النافذة و هو المختصر من `Handle window`.
- البارامتر الثاني `UINT Msg` قيمة لا تحتوي على عدد سالب و يمكن تغييرها إلى `Unsigned int Msg`.
- البارامتر الثالث `WPARAM wParam` هو بارامتر بحجم 32 بت و هو مثل البارامتر الربع `LPARAM lParam` ... لن نحتاجها في الدروس الأول و سندخل إليه في وقت آخر .

و السطر الذي يليه هو `case WM_DESTROY:` هنا في حالة تدمير النافذة نقوم باستدعاء الدالة `PostQuitMessage(0);` و التي تقوم بإغلاق النافذة.

- السطر `return DefWindowProc(hWnd, Msg, wParam, lParam);` هنا إن لم تكن أي رسالة يقوم بأخذ الإجراءات الافتراضية .

* بعد ذلك تأتي `WinMain` و هي المكافئة للـ `main` في الـ `Console` و لها أربع البارامترات أيضا حيث :

- البارامتر الأول `HINSTANCE hInstance` هو مقبض البرنامج
- البارامتر الثاني `HINSTANCE hPrevInstance` لا نحتاجه في حالتنا.
- البارامتر الثالث `LPSTR lpCmdLine` هو سلسلة حروف و التي هي مأخوذة من `Long Point STRing`
- البارامتر الرابع `INT nShowCmd` هو قيمة صحيحة سنحتاجها فيما بعد .

* و الآن تأتي البنية `WNDCLASSEX` و بها 10 بارامترات و هي :

- البارامتر الأول هو `sizeof(WNDCLASSEX)` و هنا نأخذ حجم البنية `WNDCLASSEX` للنافذة .

- البارامتر الثاني هو 0 و هو نمط الفئة و سندخل في التفاصيل فيه في وقت آخر.

- البارامتر الثالث `WndProc` و هو أخذ إجراءات النافذة.

- البارامتر الرابع و الخامس 0,0 هما للمعلومات الإضافية للفئة و النافذة.

- السادس `hInstance` هو معالج تطبيقات النافذة .

- البارامتر السابع و الثامن `LoadIcon(NULL, IDI_WINLOGO)`,

`LoadCursor(NULL, IDC_ARROW)` هما للإيقونة و الثاني للمؤشر الفأرة .

- البارامتر التاسع `(COLOR_WINDOW+2) (HBRUSH)` هو لون خلفية النافذة.

- البارامتر العاشر `NULL` هو خاص بالقائمة و هو `NULL` لأنه لا يوجد لدين قائمة.

- البارامتر الحادي عشر `"ClassName"` و هو اسم فئة النافذة.

- البارامتر الأخير `NULL` هو الإيقونة الصغيرة.

* الآن نقوم بتسجيل لبنية `wc` في `RegisterClassEx(&wc);`

* ثم نقوم بإنشاء مقبض النافذة `hWnd` و تساوي الدالة

`CreateWindow` و التي لها 11 بارامتر و هي :

- البارامتر الأول `"ClassName"` هو اسم فئة النافذة.

- البارامتر الثاني "Chapter 1" هو عنوان النافذة الذي يظهر في شريط الذي يحتوي على زر التصغير و التكبير و الإغلاق.
- البارامتر ثالث WS_OVERLAPPEDWINDOW هو شكل النافذة.
- البارامتر الرابع و الخامس CW_USEDEFAULT, CW_USEDEFAULT هما موقع النافذة حيث الأول هي x أي (س) يعني من اليسار إلى يمين الشاشة و الثانية هي y أي (ع) من أعلى إلى أسفل الشاشة.
- البارامتر السادس و السابع هما حجم النافذة حيث 640 هي العرض و 480 هي الارتفاع.
- البارامتر الثامن و التاسع NULL, NULL هما مقبض النافذة و الثاني مقبض القائمة.
- البارامتر العاشر hInstance هو مقبض مثيل للتطبيقات.
- البارامتر الأخير NULL هو بيانات إنشاء النافذة.

* الآن يأتي ShowWindow(hWnd, nShowCmd); و UpdateWindow(hWnd); هما دوال لإظهار و تحديث النافذة.

* بعدها تأتي while(GetMessage(&msg, NULL, 0, 0)) و هي حلقة الرسائل و بها :

TranslateMessage(&msg); تعالج بعد الأحداث الخاصة بلوحة المفاتيح

و DispatchMessage(&msg); و هي تقوم بإخراج الرسائل.

شرح بعض الأماكن و الدوال في المثال الفصل الأول:

* نرجع الآن إلى `LoadCursor(NULL, IDC_ARROW);` سأضع الآن جدول فيه تقريبا جميع مؤشرات الفأرة مع القليل من الشرح:

Cursor	شرح
IDC_WAIT	مؤشر الانتظار
IDC_APPSTARTING	مؤشر عمادي مع مؤشر الوقت الزجاجة أي hour glass
IDC_CROSS	مؤشر على شكل علامة زائد كبيرة
IDC_IBEAM	مؤشر النص
IDC_NO	مؤشر دائري
IDC_SIZEALL	مؤشر التحريك و هو عبارة عن أربع أسهم كل منه في اتجاه
IDC_SIZENESW	مؤشر ذو سهمين متجهين من الشمال الشرقي إلى الجنوب الغربي و هو لتحكم في حجم النافذة
IDC_SIZENWSE	و هو أيضا مؤشر ذو سهمين فقط هو عكس IDC_SIZENESW في الاتجاه
IDC_SIZENS	مؤشر ذو سهمين متجهين من الأعلى إلى الأسفل...
IDC_SIZEWE	وهو عكس IDC_SIZENS

IDC_UPARROW	مؤشر من سهم واحد متجه إلى الأعلى
-------------	----------------------------------

* نرجع إلى `LoadIcon(NULL, IDI_APPLICATION)` . و الآن

جدول لأغلب الأيقونات و مع شرح بسيط لها:

Icon	شرح
IDI_APPLICATION	وهي الأيقونة الافتراضية
IDI_ASTERISK	أيقونة النجمة
IDI_EXCLAMATION	أيقونة التعجب
IDI_HAND	أيقونة اليد
IDI_QUESTION	أيقونة السؤال
IDI_WINLOGO	هي كما IDI_APPLICATION
IDI_ERROR	أيقونة في حالة الخطأ
IDI_INFORMATION	أيقونة معلومات
IDI_WARNING	أيقونة التحذير
IDI_QUESTION	أيقونة سؤال

* نرجع إلى `(HBRUSH) (COLOR_WINDOW+0)` هنا في الريم 0 يمكنك

التغير إلى أقصى رقم و هو 25 و كل رقم بلونه و توجد طريقة

أخرى لأخذ لون للخلفية `HBRUSH)GetStockObject(WHITE_BRUSH)` ;

(و في مكان `WHITE_BRUSH` يمكنك استعمال:

Color	شرح
BLACK_BRUSH	لون أسود

WHITE_BRUSH	لون أبيض
GRAY_BRUSH	لون رمادي
LTGRAY_BRUSH	رمادي مضيء صلب
DKGRAY_BRUSH	رمادي مظلم صلب
NULL_BRUSH	لا لون

* الآن نرجع إلى 0 الذي قلنا أنه نمط الفئة و نستطيع في مكانه وضع:

شرح	Style
إعادة سحب النافذة الكاملة إذا حركة أو التعديل في حجم عرض النافذة	CS_HREDRAW
إعادة سحب النافذة الكاملة إذا حركة أو التعديل في حجم طول النافذة	CS_VREDRAW
يجعل النافذة أكثر نسبا	CS_OWNDC
النقر المزدوج	CS_DBLCLKS
لا يستطيع المستخدم الخروج باستخدام Alt+F4	CS_NOCLOSE

* الآن WS_OVERLAPPEDWINDOW و التي قلنا أنها شكل النافذة:

شرح	Style
-----	-------

WS_CHILDWINDOW	نافذة صغيرة
WS_HSCROLL	نافذة مع scroll bar أفقي
WS_OVERLAPPEDWINDOW	نافذة مع عناصر النافذة العادية
WS_POPUP	نافذة PopUp
WS_VISIBLE	نافذة مبدئياً مرئي
WS_VSCROLL	نافذة مع scroll bar عمودي

* و توجد أيضا الدالة `CW_USEDEFAULT` هذه تقوم مثلا إن وضعتها في مكان موقع النافذة فسيتم اختيار موقع افتراضي أي في كل مرة يتغير مكانها.

* `ShowWindow(hWnd, nCmdShow);` هنا قلنا أنا هذه الدالة تقوم

بإظهار النافذة و في مكان `nCmdShow` يمكن وضع:

شرح	nCmdShow
إخفاء النافذة	SW_HIDE
تصغير النافذة عند تشغيل البرنامج	SW_MINIMIZE
إرجاع النافذة	SW_RESTORE
إظهار النافذة	SW_SHOW
تكبير النافذة عند تشغيل البرنامج	SW_SHOWMAXIMIZED
تصغير النافذة عند تشغيل البرنامج	SW_SHOWMINIMIZED
ينشط النافذة	SW_SHOWNA
إظهار النافذة	SW_SHOWNORMAL

* نعود الآن إلى الرسائل و الرسالة التي وضعناها في الفصل الأول

هي WM_DESTROY و الرسائل الباقية هي :

شرح	Window Message
في حالة التشغيل	WM_ACTIVATE
في حالة الخروج	WM_CLOSE
في حالة الإنشاء	WM_CREATE
في حالة التدمير	WM_DESTROY
في حالة التحريك	WM_MOVE
في حالة تحريك مؤشر الفأرة	WM_MOUSEMOVE
في حالة إن لم يتم الضغط على أي زر على لوحة المفاتيح	WM_KEYUP
في حال الضغط على أي زر من لوحة المفاتيح	WM_KEYDOWN
في حالة الوقت	WM_TIMER
في حالة حدوث أشياء خاصة	WM_USER
في حالة الكتابة أو الرسم	WM_PAINT
في حالة الخروج	WM_QUIT
في حالة تغير حجم النافذة	WM_SIZE

هنا أخي الكريم ننتهي و أتمنى أن يكون الشرح كافي ولو قليلا.

فقط الآن ما عليك فعله هو التطبيقات الكثيرة التي تساعدك على

التعود على هذا الدوال أحسن من حفظها.

و حاول أيضا التغير دائما في أسلوب البرمجة كي تستطيع الفهم
أكثر.

الفصل الثاني:

* الحمد لله على الانتهاء من الفصل الأول و هو أطول فصل. و في هذا الفصل سأشرح كيفية وضع نص في خلفية النافذة و هو درس بسيط جدا.

* و الآن قم بإنشاء مشروع جديد باسم "Chapter2".

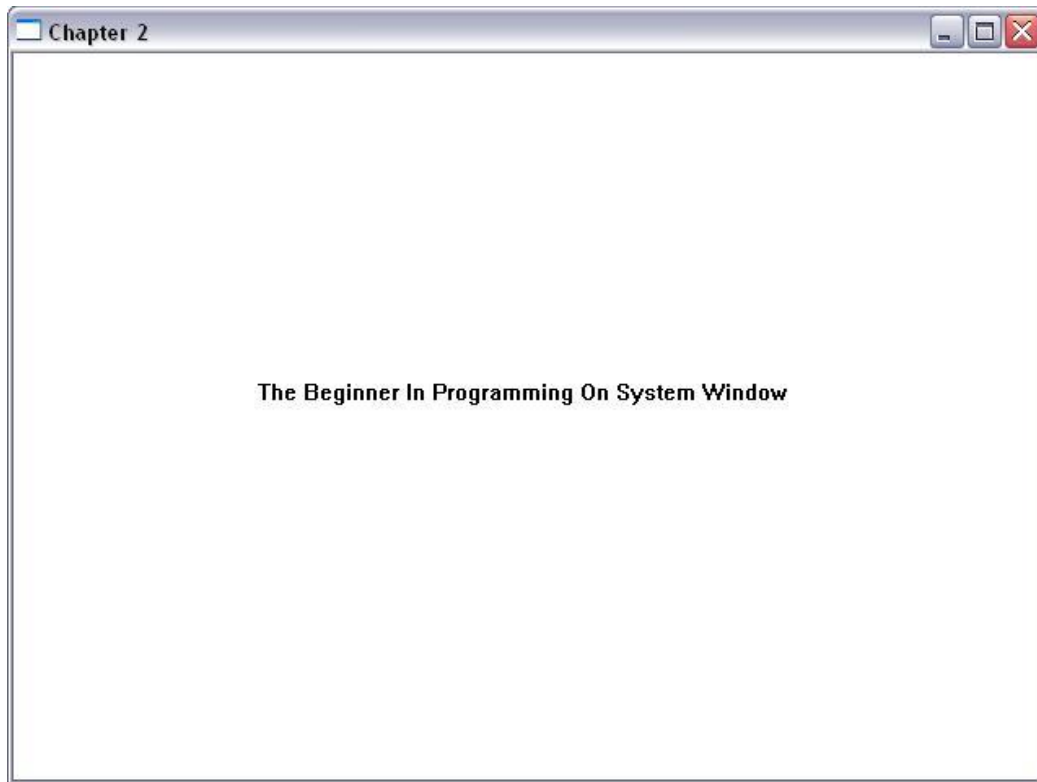
* ثم أكتب نفس الكود السابق فقط قم بالتغيير في

```
(HBRUSH)(COLOR_WINDOW+2),  
في الرقم 2 إلى الرقم 1 ثم أضف  
char text[] = "The Beginner In Programming On System  
Window";
```

ثم في إجراءات النافذة أضف الكود التالي:

```
case WM_PAINT:  
  
HDC          hDC;  
PAINTSTRUCT PS;  
hDC = BeginPaint(hWnd, &PS);  
TextOut( hDC, 150, 200, text, (sizeof(text)-1));  
EndPaint(hWnd, &PS);  
Break;
```

* اضغط على الزر F7 ثم اضغط على Ctrl+F5



* نبدأ الشرح بـ `HDC hdc` هو مقبض إلى أداة السياق.

- `PAINTSTRUCT pstruct` بنية الرسم لاستدعاء `BeginPaint`.

- `hDC = BeginPaint(hWnd, &PS);` هنا `hDC` يساوي `BeginPaint` و
التي سترسم في مقبض النافذة `hWnd`.

- `TextOut(hdc, 0, 0, message, (sizeof(message)-1));` هنا نقوم
بطبع النص على خلفية النافذة حيث `0` الأول هو موقع النص في `x` و
`0` الثاني هو `y`.

- `EndPaint(hWnd, &pstruct);` هنا ننهي الرسم.

* هنا ينتهي الفصل الثاني و الذي يقوم بطبع رسالة في خلفية
النافذة و التي ربما تحتاجها في المستقبل.