

شرح البحث الأول في Direct3D :

```
#include<d3d9.h>
```

الملف الرئيسي الذي نحتاجه مع مكتبة d3d9.lib

```
LPDIRECT3D9 g_pD3D = NULL;
```

هنا نستعمل الكائن LPDIRECT3D9 لإنشاء أداة D3D (Device) و هي معرفة كمؤشرات إلى IDirect3D9 و التي نحتاجها في جميع برامج Direct3D المقبض : d3d9.h مستوردة من المكتبة : d3d9.lib تتطلب على الأقل نظام التشغيل : Windows 98

```
LPDIRECT3DDEVICE9 g_pd3dDevice = NULL;
```

تستعمل الكائن LPDIRECT3DDEVICE9 لإعادة استعمال الأداة و هي معرفة كمؤشرات أيضا إلى IDirect3DDevice9 و التي هي من أهم الكائنات و به أعمال كثير سنراها فيما بعد المقبض : d3d9.h مستوردة من المكتبة : d3d9.lib تتطلب على الأقل نظام التشغيل : Windows 98

```
HRESULT InitD3D( HWND hWnd)
```

نقوم الآن بتشغيل Direct3D و سنستدعيها عندما نقوم بإنشاء النافذة

```
if( NULL == ( g_pD3D = Direct3DCreate9( D3D_SDK_VERSION)))  
return E_FAIL;
```

إنشاء الكائن Direct3D و نحتاجه أيضا في إنشاء الأداة لـ Direct3D يجب دائما توجيه البارامتر Direct3DCreate9 على D3D_SDK_VERSION و هذا يخبر Direct3D بأن جميع المقابض صحيحة و مستعملة و إن لم توجد النسخة فإن Direct3DCreate9 سيفشل و هذا يعني فشل البرنامج بأكمله

```
D3DPRESENT_PARAMETERS d3dpp;
```

المقبض : **d3d9types.h**
أعطاء هذه الدالة المستعمل في هذا البحث هي :

```
d3dpp.Windowed = TRUE;
```

في هذه الحالة ستظهر النافذة عادية و في حالة FALSE فإنها ستظهر على شكل الشاشة الكاملة (مع بعض الإضافات)

```
d3dpp.SwapEffect = D3DSWAPEFFECT_DISCARD;
```

في هذه الحالة إستعملنا **D3DSWAPEFFECT_DISCARD** لتقديم الحاجر الخلفي إلى العرض

```
d3dpp.BackBufferFormat = D3DFMT_UNKNOWN;
```

D3DFMT_UNKNOWN طلب صيغة حاجر خلفية التي تشغل صيغة عرض المكتب الحالية.

```
if( FAILED( g_pD3D->CreateDevice(D3DADAPTER_DEFAULT,  
D3DDEVTYPE_HAL,  
hWnd,D3DCREATE_SOFTWARE_VERTEXPROCESSING,&d3dpp,  
&g_pd3dDevice)))  
return E_FAIL;
```

نقوم بإنشاء أداة **Direct3D**

D3DADAPTER_DEFAULT خاصة بوضعية العرض و هي إفتراضية أي أننا سنستعمل الوضعية الحالية للعرض للبرنامج

D3DDEVTYPE_HAL هنا نخبر المصرف بأننا نفضل **device** الأجهزة على البرنامج أي هي خاصة بكروت الشاشة (راجع الـ **Documentation**)

D3DCREATE_SOFTWARE_VERTEXPROCESSING لإخبار النظام بإستعمال معالجة **vertex**

الآن هذا مانحتاجه لكي نشغلي **Direct3D**

```
Void Cleanup()  
{  
if( g_pd3dDevice != NULL)  
g_pd3dDevice->Release();  
if( g_pD3D != NULL)  
g_pD3D->Release();  
}
```

في حالة توقف أو الخروج من البرنامج يقوم بتحرير جميع التشغيلات السابقة (D3D & d3ddevice)

```
VOID Render()
```

هنا نقوم بسحب مشاهدنا

```
g_pd3dDevice->Clear( 0, NULL, D3DCLEAR_TARGET,  
D3DCOLOR_XRGB(0,0,255), 1.0f, 0);
```

وضع اللون الأزرق للخلفية

```
if( SUCCEEDED( g_pd3dDevice->BeginScene()))
```

إبدأ المشهد

```
g_pd3dDevice->EndScene();
```

أنه المشهد

```
g_pd3dDevice->Present( NULL, NULL, NULL, NULL);
```

تقديم محتويات الخلفية للعرض

الآن نذهب إلى عامل الرسائل للنافذة

```
Case DestroyWindow:  
Cleanup();
```

هنا في حالة تدمير النافذة نقوم بإستدعاء الدالة Cleanup() و التي قلنا عليها أنها تقوم بتحرير كل من d3d & d3ddevice

```
case WM_PAINT:  
Render();
```

و هنا يقوم برسم ماهو موجود في الدالة Render() أي رسم المشاهد الموجودة بالدالة

```
WNDCLASSEX wc = {...,...,...};
```

```
HWND hWnd = CreateWindow (...,...,...);
```

إنشاء النافذة

```
if( SUCCEEDED( InitD3D( hWnd)))
```

تشغيل الـ D3D

```
ShowWindow( hWnd, SW_SHOWDEFAULT);  
UpdateWindow( hWnd );
```

إظهار و تحديث النافذة

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

حلقة الرسائل