الباب الحادى عشر

قضبان الانزلاق وسطر الأدوات

Scrolling and Toolbars

مفتتح

في هذا الباب:

سوف نتعلم كيفية إضافة قضبان الانزلاق التى تمكننا من تحريك محتويات النافذة رأسياً وأفقياً عندما تزيد المحتويات عن حدود الشاشة.

كما سنعرف تفصيلات نظام الإحداثيات المستخدم فى النوافذ بنوعيه: الإحداثيات الطبيعية والإحداثيات المنطقية ، وكيفية التحويل بينهما.

سوف نتدرب أيضاً على إنشاء سطور الأدوات (Toolbar) والحالة (Status Bar) وإنشاء المقابض المسئولة عن استخدام أزرار الأدوات.

تذكر:

١. للاطلاع على الصيغة التفصيلية لإحدى الدوال أو الفصائل ، ضع مؤشر الفأر فوقها ثم اضغط زر اللوحة F1.

٢. لعرض الخريطة الكاملة لشجرة فصائل المؤسسة MFC ابحث فى شاشة النجدة (باستخدام البطاقة Search) عن "Hierarchy Chart".

٣. للاطلاع على الكود الكامل لأحد المشروعات ، افتح المشروع من القرص المصاحب للكتاب.

(١١-١) برنامج الرسم ــ الطراز ٤

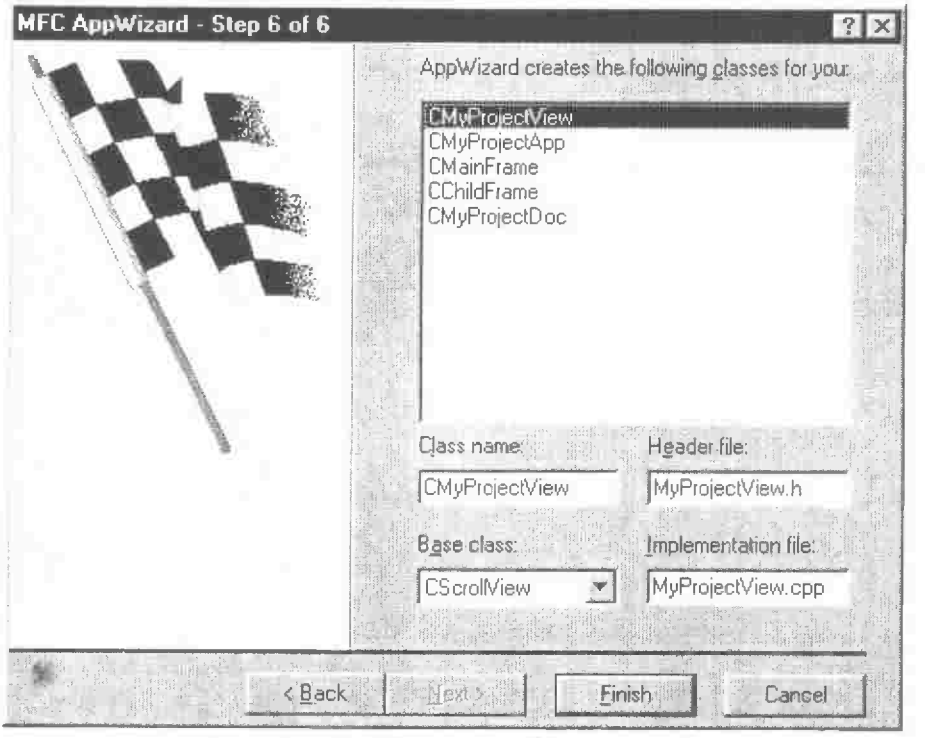
فى هذا الباب سوف نستخدم نفس البرنامج الذى أنشأناه من قبل فى الفصول السابقة MyPaintBrush لكى نمنحه إمكانية الانزلاق باستخدام قضبان الانزلاق (Scroll Bars).

ولكى تبدأ العمل فى هذا المشروع فإنك قد تستخدم أحد طريقتين:

- إما أن تستخدم نفس البرنامج الذى أنشأته فى الباب العاشر وتستكمل الرحلة من حيث انتهيت.
- أو تنسخ الدوسيه \ExamplesVCPP\Ch10\MyPaintBrush الموجود على القرص المصاحب للكتاب ، إلى القرص الصلب ، وتبدأ منه العمل (ويمكنك تغيير اسمه إلى اسم جديد).

إضافة قضبان الانزلاق إلى مشروع جديد

إذا كان المشروع جديداً ، ففى إمكانك أن تمنحه خاصية قصبات الانزلاق أثناء إنشاء البرنامج باستخدام الشاشة السادسة من شاشلت الساحر. وذلك باختيار الفصيلة CSrollView بدلا من الفصيلة CView كما بالشكل التالى ، حيث استخدمنا هذه الخطوة فى بناء مشروع بالاسم MyProject. أما إضافة قضبان الانزلاق إلى مشروع موجود فهذا هو موضوعنا الأساسى فى الفقرات التالية.



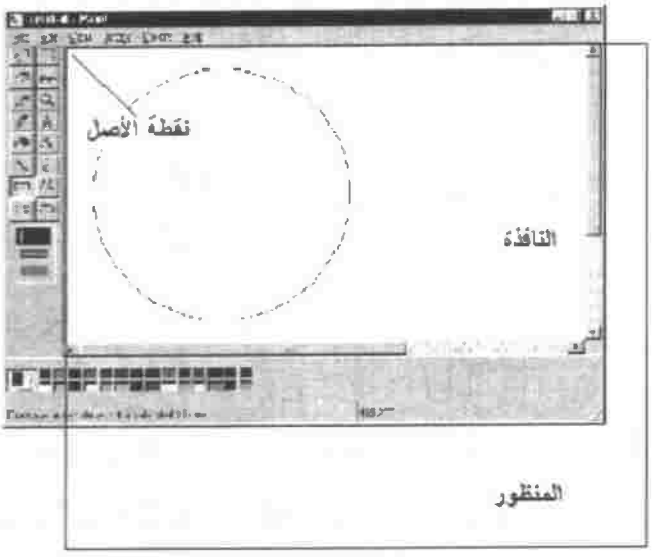
شكل (١٠-١) إضافة قضبان الانزلاق أثناء خلق المشروع

الانزلاق وعلاقته بنظام الإحداثيات

ذكرنا فى الباب السابع الفرق بين الإحداثيات الطبيعية (التى قد تسمى أيضاً بإحداثيات الجهاز Device Coordinates) وبين الإحداثيات المنطقية. وموضوع الانزلاق يرتبط مباشرة بنظم الإحداثيات المستخدمة فى النوافذ. ولنتصور أن نظام الإحداثيات المنطقية عبارة عن مستطيل له نقطة أصل ومحاور كالموضحة بالشكل التالى ــ سوف نصطلح عليه باسم المنظور (Viewport).

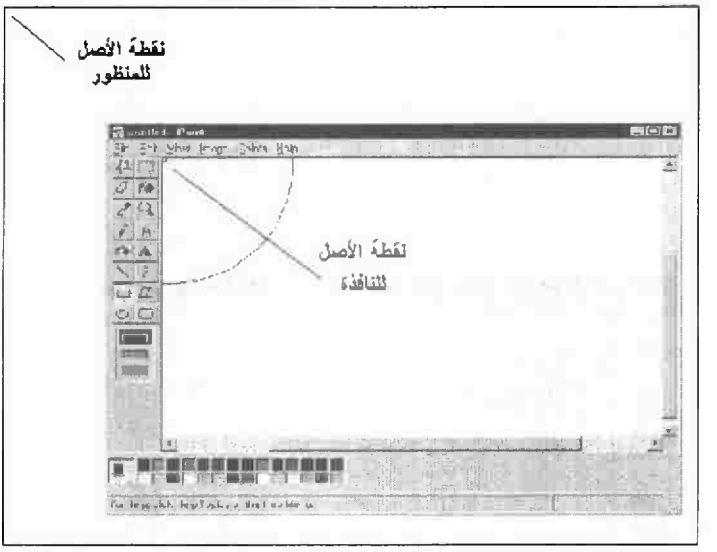
ولنفرض أنك فتحت برنامج الرسم Paint (المتضمن مع النوافذ) ثم رسمت به دائرة ما كما هو موضح بالشكل. فى بداية الأمر تتطابق

كل من نقطتى الأصل للنافذة (الإحداثيات الطبيعية) والمنظور (الإحداثيات المنطقية) ، بحيث تكون نقطة الأصل المشتركة فى الركن الأيسر العلوى لمنطقة العميل.



شكل (١٠-٢) الرسم على شاشة البرنامج فى البداية

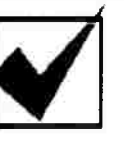
ولنفترض أننا حركنا المنزلق الرأسى إلى اسفل ، والمنزلق الأفقى إلى اليمين ، بحيث يصبح الرسم على شاشة البرنامج كما هو موضح بالشكل التالى. إن النتيجة التى نحصل عليها هى فى الحقيقة تحريك النافذة بمحاورها بالنسبة لنقطة الأصل المنطقية. وبالرغم من أنك لا ترى إلاّ ربع الدائرة فقط ، ولكن بعد الدائرة عن نقطة الأصل المنطقية لم يتغير. فالدائرة موجودة فى مكانها ولكن ما تراه على شاشة البرنامج هو جزء من المنظور.



شكل (١١-٣) تحريك المنظور باستخدام قضبان الانزلاق

وفى الحالة الأخيرة فإن إحداثيات مركز الدائرة بالنسبة للنافذة هى (0,0) ، أما بالنسبة للمنظور فقد تكون رقما مثل (95,100).

أما ما يحدث فى منطق البرنامج الذى يصنع هذه الأحداث أن الدالة OnDraw() المسئولة عن الرسم وإعادة الطلاء لا تتغير ، فهى ترسم الدائرة فى نفس المكان كل مرة ، منسوبا إلى المحاور المنطقية. أما عملية الانزلاق فتتولاها دوال المؤسسة MFC.



إن دوال الرسم مثل MoveTo() و LineTo() تستخدم الإحداثيات المنطقية ، أما ضغطات الفأر فى الأماكن المختلفة على الشاشة أو النافذة فهى تستخدم الإحداثيات الطبيعية ، وهذا هو مصدر الحاجة إلى تحويل الإحداثيات.

(١١-٢) إضافة قضبان الانزلاق إلى برنامج الرسم

فى هذه الفقرة سوف نضيف قضبان الانزلاق إلى البرنامج الموجود لدينا ، وسوف نقسم العمل إلى خطوات محددة:

الخطوة الأولى: الوراثة من الفصيلة CScrollView

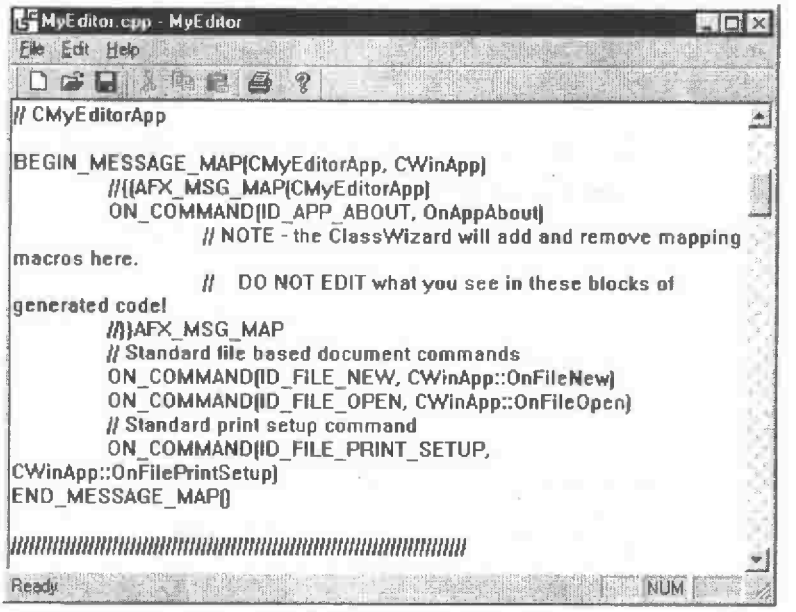
لتحقيق ذلك استبدل جميع تكرارات الكلمة CView بالكلمة CScrollView بكل من الملفات الآتية:

- ملف العناوين MyPaintBrushView.h
- ملف المصدر MyPaintBrushView.cpp

وتؤدى الوراثة من الفصيلة CScrollView إلى منح نافذة التطبيق خاصية قضبان الانزلاق مباشرة ، ومع ذلك فإن البرنامج يحتاج إلى إضافة بعض سطور الكود لتحويل الإحداثيات.

ملاحظة:

لاحظ أننا لا نحتاج إلى إضافة قضبان الانزلاق إلى ملف المحرر MyEditor وذلك لأنه يرث من الفصيلة CEditView التى تمنحه هذه الخصائص تلقائيا. وكما نرى بالشكل التالى أنه عند تحميل ملف محتو على عدد من السطور بما يزيد عن اتساع الشاشة فإن قضيب الانزلاق الرأسى يظهر تلقائيا ، كما أن خاصية الانتقال التلقائى إلى السطر التالى (Word Wrap) تغنى عن قضيب الانزلاق الأفقى.



شكل (١١-٤) خصائص الانزلاق لبرنامج المحرر

ملاحظة:

الخطوة الثانية: إعداد المنظور وتحويل إحداثى نقطة البداية

عند الضغط على زر الفأر الأيسر فإن معلومة الإحداثى يتم تسجيلها وتمريرها باستخدام البارامتر point إلى الدوال المختلفة. يلزم تحويل هذا الإحداثى من إحداثيات طبيعية منسوبة إلى النافذة إلى إحداثيات منطقية منسوبة إلى المنظور.

أضف الكود الموضح بالبنط الثقيل إلى الدالة OnLButtonDown() بالملف MyPaintBrushView.cpp:

```
// CMyPaintBrushView message handlers

void CMyPaintBrushView::OnLButtonDown(UINT nFlags, CPoint
point)
{
```

```
// TODO: Add your message handler code here and/or call default
// Init Variables:
// New code for coordinates conversion:          أضف السطور التالية:
        CClientDC ClientDC(this);
        OnPrepareDC(& ClientDC);
        ClientDC.DPtoLP(&point);
//                                                نهاية الكود الجديد
        m_NewPoint = point;
        m_PrevPoint = point;
        m_Moving =1;
        SetCapture();
```

- فى هذا الكود ، تستخدم العبارة CClientDC ClientDC(this) لخلق هدف منطقة عرض (Display Context) مرتبط بنافذة المشهد. ويحقق المؤشر this هذا الارتباط.
- أما العبارة التالية فهى تحتوى على الدالة:

CScrollView ::OnPrepareDC()

ووظيفتها ضبط نقطة الأصل للمنظور ، تبعا لموضع قضبان الانزلاق.

- وتحتوى العبارة الثالثة على الدالة:

CDC::DPtoLP()

وتستخدم فى التحويل من إحداثيات الجهاز (DP) إلى إحداثيات منطقية (LP) منسوبة إلى المنظور الذى تم إعداده فى الخطوة السابقة.

ترجم البرنامج للتأكد من عدم وجود أخطاء ، ولكن لا تتوقع أن يعمل بنجاح حتى الآن.

الخطوة الثالثة: تحويل إحداثيات حركة الفأر

عندما تسحب الفأر أثناء الضغط على الزر ، فإنك ترسل معلومات الإحداثيات لنقط المسار. ومن اللازم تحويل هذه النقط أيضا إلى إحداثيات منطقية بنفس الطريقة. وفى سطور الكود التالى قد أضفنا نفس الدوال إلى الدالة OnMouseMove() بنفس الملف. ولتلاحظ أن هدف منطقى العرض ClientDC كان موجودا بهذه الدالة من الأصل.

```
void CMyPaintBrushView::OnMouseMove(UINT nFlags, CPoint
point)
{
        // TODO: Add your message handler code here and/or call
default
//      Add these lines:
        if (m_Moving)
        {
                CClientDC ClientDC(this);
// Code for Coordinate Conversion:                    أضف هذه السطور
                OnPrepareDC(&ClientDC);
                ClientDC.DPtoLP(&point);
//                                                   نهاية الكود الجديد
```

الخطوة الرابعة: تحويل إحداثى نهاية الخط

أما الإحداثى الأخير الذى يحتاج إلى تحويل فهو إحداثى نهاية الخط عندما تطلق الزر. أضف عبارات مماثلة بنفس الملف إلى الدالة OnLButtonUp() كما هو موضح بالشكل التالى:

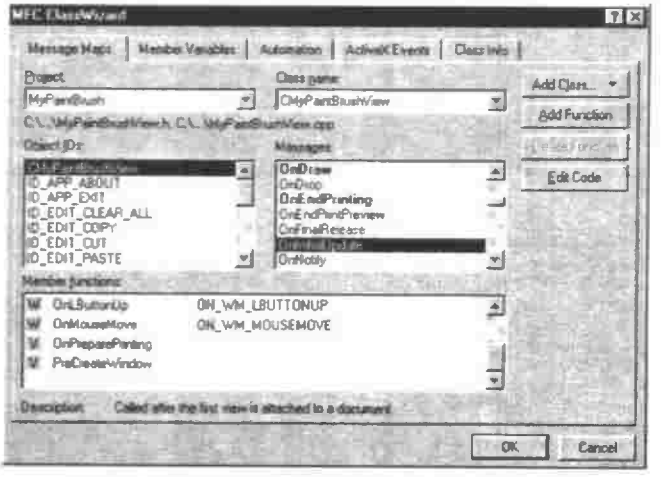
```
void CMyPaintBrushView::OnLButtonUp(UINT nFlags, CPoint point)
{
        // TODO: Add your message handler code here and/or call
default
// Add these lines:
```

```
        if (m_Moving)
        {
                m_Moving = 0;
                ::ReleaseCapture();
                ::ClipCursor(NULL);
                CClientDC ClientDC(this);
// Code for Coordinate Conversion:                    أضف هذه السطور:
                OnPrepareDC(&ClientDC);
                ClientDC.DPtoLP(&point);
//                                                    نهاية الكود الجديد
                ClientDC.SetROP2(R2_NOT);
```

الخطوة الخامسة: تحديد حيز الرسم

عندما تسحب المنزلق الرأسى أو الأفقى إلى أقصى مداه ، فمن اللازم أن تتزامن هذه الحركة مع أبعاد المنظور بحيث تشاهد أقصى حدود الرسم. ولتحقيق ذلك يلزم تسجيل معلومة مساحة المنظور المطلوبة. لتحقيق ذلك اتبع الآتى:

- استخدم ساحر الفصائل (Class Wizard) لركوب الدالة CView::OnInitialUpdate() كما بالشكل التالى.



شكل (١١-٥) إضافة الدالة OnInitialUpdate()

- اختر اسم المشروع ليكون MyPaintBrush ، و الفصيلة لتكون CMyPaintBrushView ، و رقم التعـــارف للـــهدف (Object ID) ليكون CMyPaintBrushView ، ثم اختر الدالـــة OnInitialUpdate() من صندوق الرسائل.
- اضغط الزر Add Function.
- اضغط الزر Edit Code لكى تنتقل إلى سطور الدالة الموضحة بالشكل التالى. أضف العبارات الموضحة بالبنط الثقيل.

```
void CMyPaintBrushView::OnInitialUpdate()
{
        CScrollView::OnInitialUpdate();

// TODO: Add your specialized code here and/or call the base class
                                                        أضف هذه السطور
        SIZE Size = {800, 600};
        SetScrollSizes(MM_TEXT, Size);
                                                        نهاية الكود الجديد
}
```

وتستدعى الدالة OnInitialUpdate() عادة لتحديث المشهد قبل عرضه على الشاشة مباشرة ، سواء كان المشهد جديـــدا أم موجـــودا مـــن الأصل. ويستخدم متغير المنشأ SIZE فى تخزين أبعاد المنظور. وقد استخدمنا هنا الأرقام ٦٠٠ و ٨٠٠ ، ولكنها ليست ملزمة ، ويمكنك أن تختار أرقاما أخرى مثل ٦٤٠ و ٤٨٠.

فيما يلى إعلان المنشأ SIZE المستخدم فى تحديد أبعاد المنظور:

```
typedef struct tagSIZE {
    int cx;
    int cy;
} SIZE;
```

أما الدالة CScrollView::SetScrollSizes() فهى تستخدم لتمرير منشأ الأبعاد علاوة على طريقة طور الخريطة (Mapping Mode) الذى يمكن تحديده بأحد الثوابت الموضحة بالجدول التالى ، وقد اخترنا منها الثابت MM_TEXT الذى يناظر نظام المحاور المعتاد (قطة الأصل فى الركن الأيسر العلوى) ووحدات القياس بالبكسلة (Pixel).

طور الخريطة	وحدة القياس	اتجاه المحور الرأسى
MM_TEXT	1 pixel	إلى أسفل
MM_HIMETRIC	0.01 mm	إلى أعلى
MM_TWIPS	1/1440 in	إلى أعلى
MM_HIENGLISH	0.001 in	إلى أعلى
MM_LOMETRIC	0.1 mm	إلى أعلى
MM_LOENGLISH	0.01 in	إلى أعلى

جدول (١١-١) ثوابت طور الخريطة (Mapping Mode Constants)

جرب تشغيل البرنامج الآن وشاهد قضبان الانزلاق. غير أيضا مساحة المنظور إلى ٤٨٠ و ٦٤٠ ، ثم اضغط على زر تكبير النافذة وشاهد الفرق بين المنظورين.

(١١-٣) إضافة سطر الأدوات وسطر الحالة
Toolbar-Status Bar

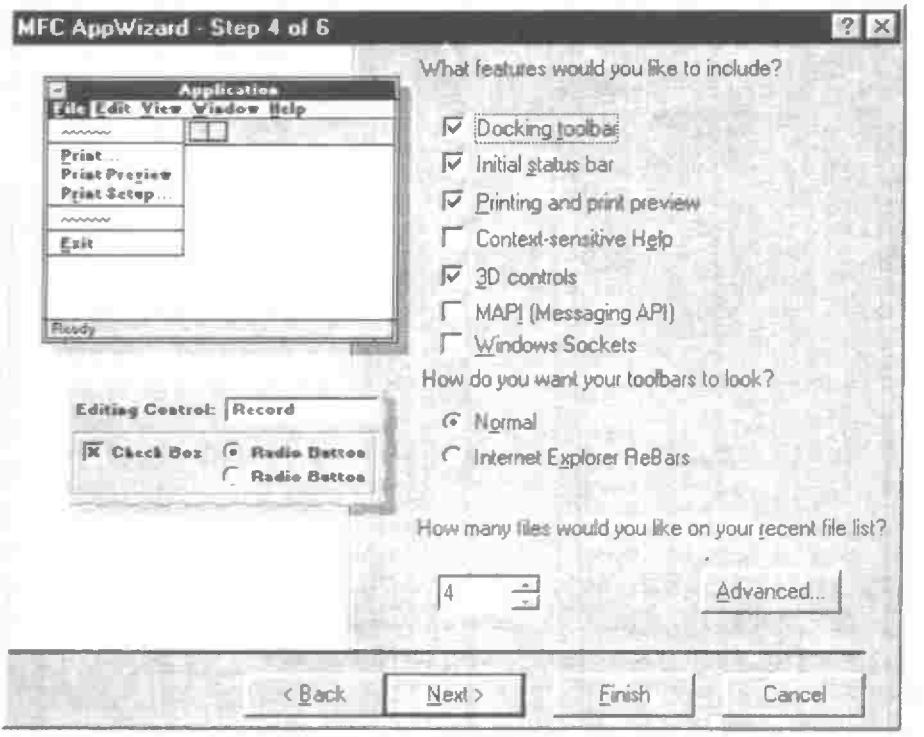
فى هذه الفقرة نتدرب على إنشاء سطر للأدوات سواء عند إنشـــاء برنامج جديد أو عند تطوير برنامج موجود أصلا.

أولا: سطر الأدوات فى برنامج جديد

عند تشغيل الساحر ، استمر فى الانتقال إلى الشاشات التالية حتـــى تصل إلى الشاشة الرابعة الموضحة بالشكل التالى.

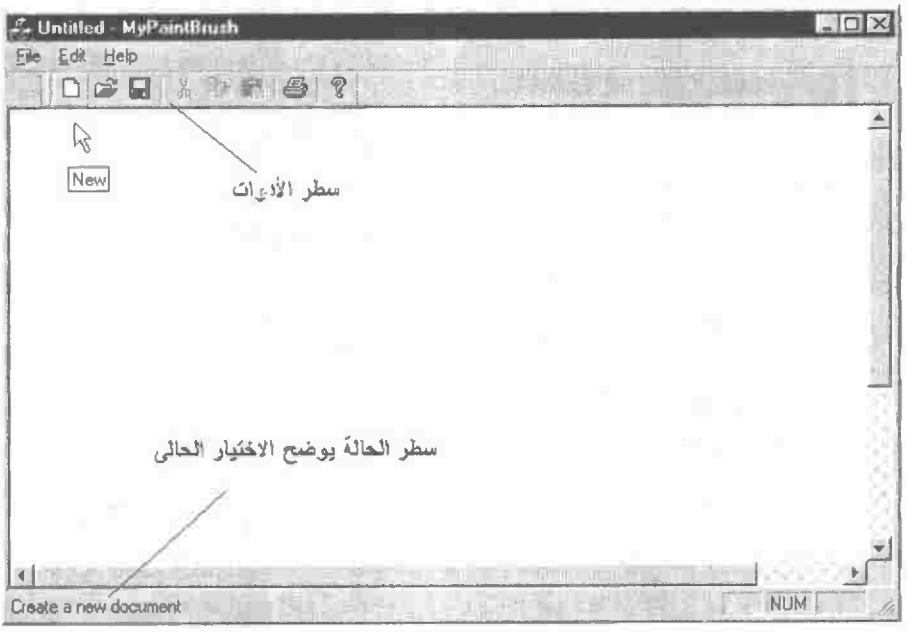
اختر من هذه الشاشة صناديق الاختيار التالية (إذا لـــم تكـــن هـــى الاختيارات سابقة التعريف):

- Docking Toolbar: سطر الأدوات العائم.
- Initial Status bar: سطر الحالة.



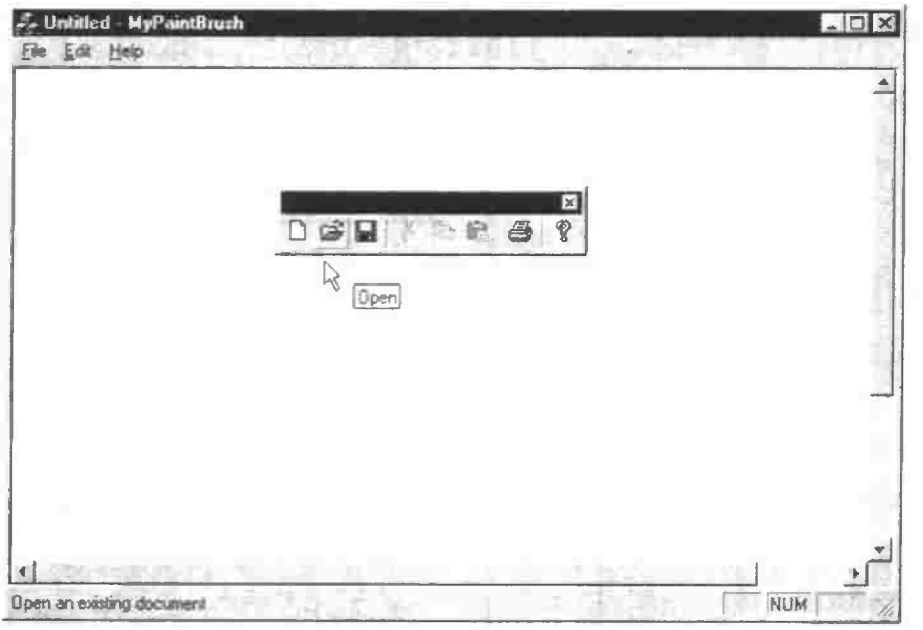
شكل (١١-٦) إنشاء سطر الأدوات وسطر الحالة في مشروع جديد

وبذلك فإن المشروع الجديد سوف يحتوى على كل من سطر الأدوات وسطر الحالة (سابقى التعريف) كما بالشكل التالى. وكما نرى بالشكل أنه عند اختيار أحد الأدوات (مثل أيقونة الملف الجديد "New" فى هذا المثال) فإن اسم الأيقونة يظهر فى صندوق خاص بجوارها ، كما أن سطر الحالة يصبح محتويا على شرح للاختيار الحالى.



شكل (١١-٧) سطر الحالة يبين الاختيار الحالي بسطر الأدوات

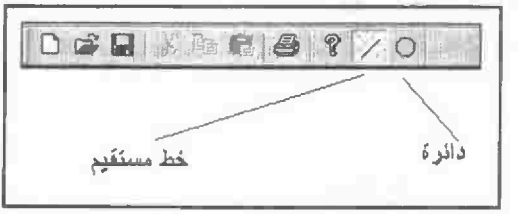
ولأن سطر الأدوات عائم (Docked) فإنك تستطيع سحبه وتحريكـــه إلى أى موضع من النافذة. أنظر الشكل التالى.



شكل (١١-٨) سطر الأدوات بعد تحريكه من مكانه

ثانيا: إضافة سطر أدوات إلى برنامج موجود

سوف نستخدم برنامج الرسم MyPaintBrush فى إضافة سطر الأدوات ، وبالطبع فإننا فى هذه الحالة لن نكتفى بالأدوات سابقة التعريف (الموجودة أصلا) بل سنضيف أيقونات جديدة ونربطها بالبرنامج. والنتيجة التى نرغب فى الوصول إليها موضحة بالشكل التالى ، وهى عبارة عن إضافة زرين من أزرار الأدوات واحد لرسم الخط المستقيم وآخر لرسم الدائرة. وبالطبع فإنك تستطيع أن تضيف أشكالا هندسية أخرى مثل البيضاوى والمستطيل والمربع إلى آخره. وكننا سوف نكتفى بهذه الأزرار فى المثال الحالى.



شكل (١١-٩) إضافة أزرار أدوات

وهذه هى الخطوات:

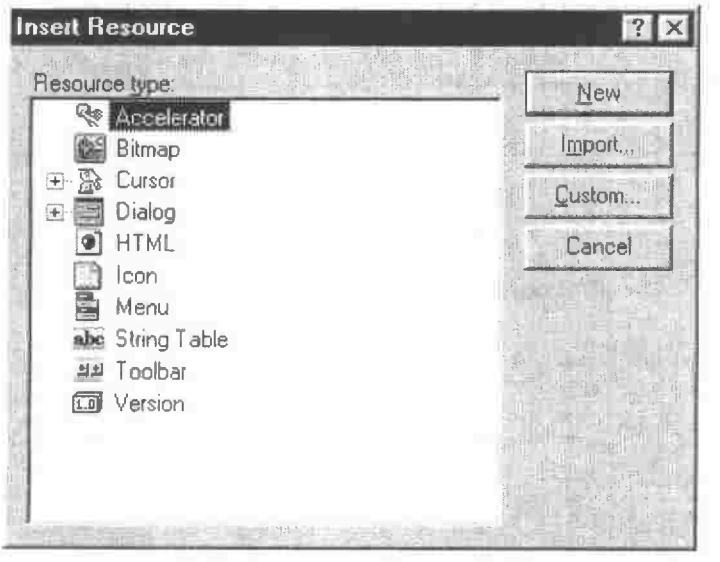
الخطوة الأولى: إضافة أزرار سطر الأدوات

أولا: إنشاء سطر جديد للأدوات

١.افتح المشروع MyPaintBrush.

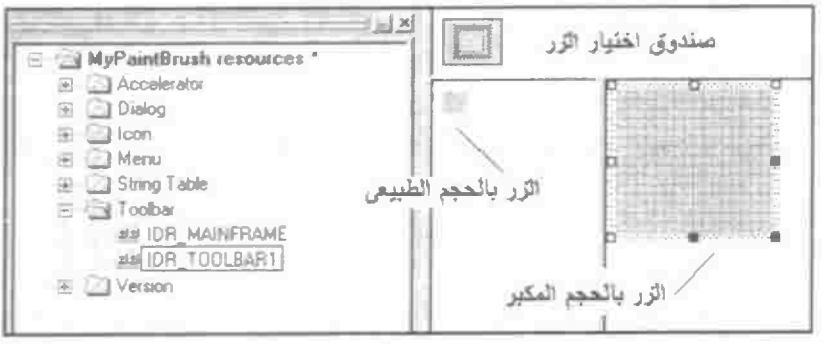
٢.اختر أمر القائمة Insert Resource.

٣.من صندوق الحوار الذى يظهر ، اختر سطر الأدوات (Toolbar). أنظر الشكل التالى.



شكل (١١-١٠) صندوق حوار الموارد

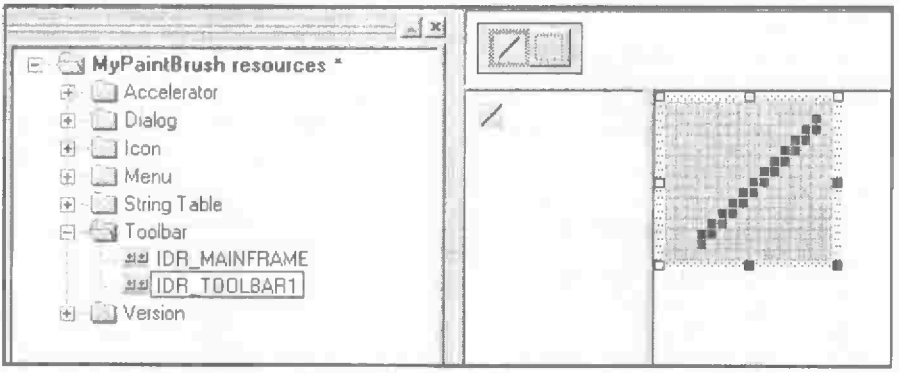
٤. يظهر فى القسم الأيسر من الشاشة عنصر جديد فـــى شـــجرة الموارد بالاسم IDR_TOOLBAR1. أنظر الشكل التالى.



شكل (١١-١٢) سطر الأدوات بمشهد الموارد

٥. يحتوى الشكل السابق على ثلاثة أقسام فى النافذة اليمنى. واحـــد مخصص للرسم وهو يظهر الزر بالحجم الكبير. وآخر لمشاهدة شكل الزر بالحجم الطبيعى. أما القسم العلوى فيحتـــوى علـــى جميع الأزرار التى نضيفها إلى ســـطر الأدوات ، ولكنـــه فـــى الوقت الحالى لا يحتوى إلا على الزر الجديد المزمع إضافتـــه. وبالطبع فإن هناك سطرا لأدوات الرسم جاهز على اســـتخدامه فى رسم أيقونة الزر ، وهو يظهر عائما فى بيئة الأستوديو.

٦. ارسم المنظر المرغوب إظهاره على الزر باســـتخدام المشـــهد المكبر. والشكل التالى يوضح رسم أيقونـــة الخـــط المســـتقيم. ولتلاحظ أنه عندما تبدأ الرسم الجديد ، تظـــهر أيقونـــة جديـــدة فارغة على اليمين.

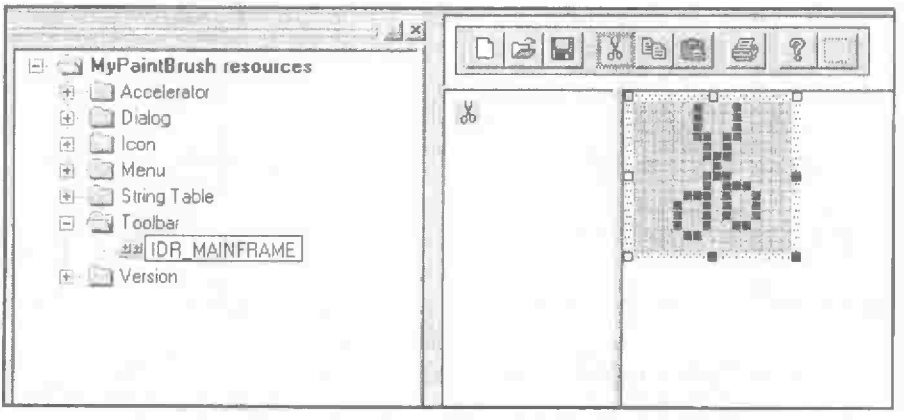


شكل (١١-١٣) رسم أيقونة الخط المستقيم

ثانيا: إضافة أزرار جديدة إلى سطر الأدوات سابق التعريف:

١. افتح المشروع MyPaintBrush.

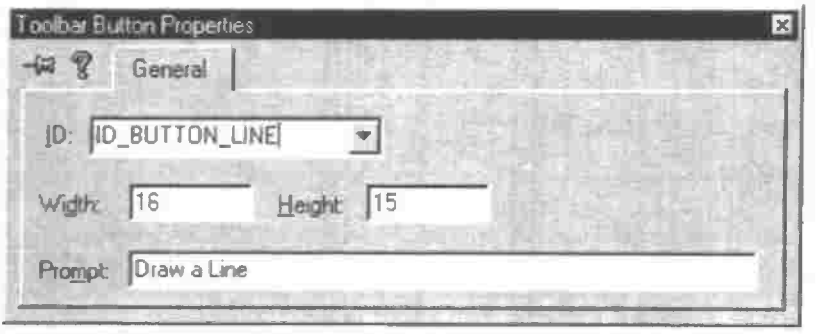
٢. افتح مشهد الموارد ، واختر منه العنصر Toolbar ، ثم العنصر IDR_MAINFRAME ، فيظهر على اليمين سطر الأدوات سابق التعريف ونرى به مجموعة الأزرار المعروفة وعلى يمينها أيقونة فارغة جاهزة على الرسم. أنظر الشكل التالى.



شكل (١١-١٤) إضافة أيقونة إلى سطر الأدوات سابق التعريف

٣.اختر الأيقونة الفارغة فيظهر المربع الفارغ فى المشهد المكبر ، وارسم أيقونة الخط المستقيم.

٤.اضغط ضغطة مزدوجة على أى نقطة خالية فى قسم الأيقونة المكبرة ، فتظهر نافذة لخصائص الأيقونة الجارى رسمها. غير رقم التعارف الموجود إلى ID_BUTTON_LINE. أنظر الشكل التالى.



شكل (١١-١٥) نافذة خصائص زر من أزرار سطر الأدوات

٥.امنح وصفا ما لوظيفة كل زر باستخدام نافذة الخصائص كما هو موضح بالشكل التالى. إن هذا الوصف (مثل Draw a Line) يظهر فى سطر الحالة عند اختيار الزر.

٦.كرر العملية السابقة لإضافة زر الدائرة ، وامنحه الاسم ID_BUTTON_CIRCLE ثم كرر العملية لإضافة أية أزرار جديدة ترغب فى إضافتها مع منحها الاسم المناسب.

١.قم ببناء وتنفيذ البرنامج عند هذا الحد للتأكد من سلامة عملـــه ، وسوف تشاهد سطر الأدوات وقد جدت عليه الأيقونات الجديـــدة ، لكنها تظهر معتمة (غير عاملة).

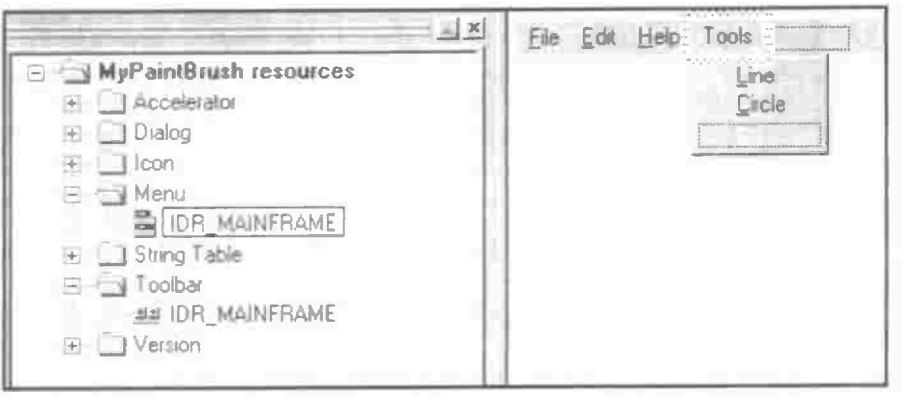
الخطوة الثانية: إضافة أوامر الرسم إلى القائمة

عادة تناظر أزرار سطر الأدوات أوامر مماثلة من أوامر القائمـــة ، لأن سطر الأدوات ما هو إلا طريقة سريعة لتنفيذ المهام. فـــالأوامر New و Open و Save كلها موجودة بالقائمة ، لكن وجودها بســـطر الأدوات يمنحك بديلا سريعا لتنفيذها. وفى هذه الفقرة سوف نضيف أمرين من أوامر القائمة ، واحد لرسم الخط Line والآخـــر لرســـم الدائرة Circle.

٢.افتح مشهد الموارد (Resource View).

٣.اختر القائمة (Menu) ، ثم اختر منها العنصـــر IDR_MAINFRAME ، فتظهر على اليمين نافذة تحرير القوائم.

٤.استخدم الطرق التى مررنا بها من قبل لإضافة أمر قائمة جديـــد بالاسم Tools ، يتفرع منه الاختيـــاران Line و Circle ، بحيـــث يبدو المشهد كما فى الشكل التالى.

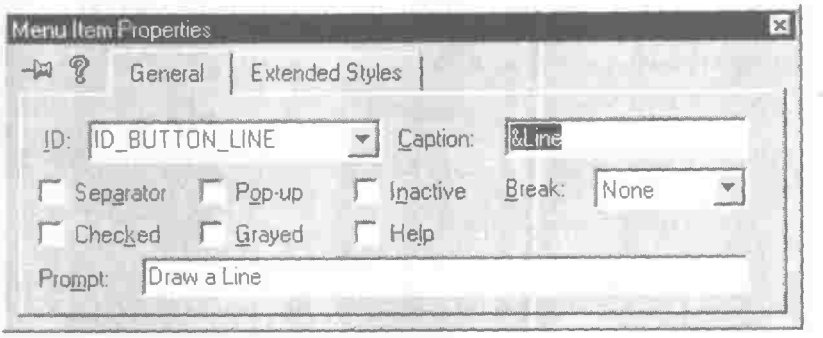


شكل (١١-١٦) المشهد النهائي لأوامر القائمة

٥. باستخدام نافذة الخصائص (التى تتوصل إليها بضغطة مزدوجة على اختيار القائمة) لكل أمر من أوامر القائمة امنح الأوامر Line و Circle نفس رقم التعارف الذى منحناه للأزرار المنظرة أى:

الاختيار	رقم التعارف
Line	ID_BUTTON_LINE
Circle	ID_BUTTON_CIRCLE

٦. يمكنك أيضا أن تمنح أيضا وصفا ما لوظيفة كل زر باستخدام هذه النافذة كما هو موضح بالشكل التالى. إن هذا الوصف هو نفسه الوصف المرتبط بخصائص الزر ويجوز تعديله من أى من النافذتين.



شكل (١١-١٧) نافذة خصائص أمر القائمة Line

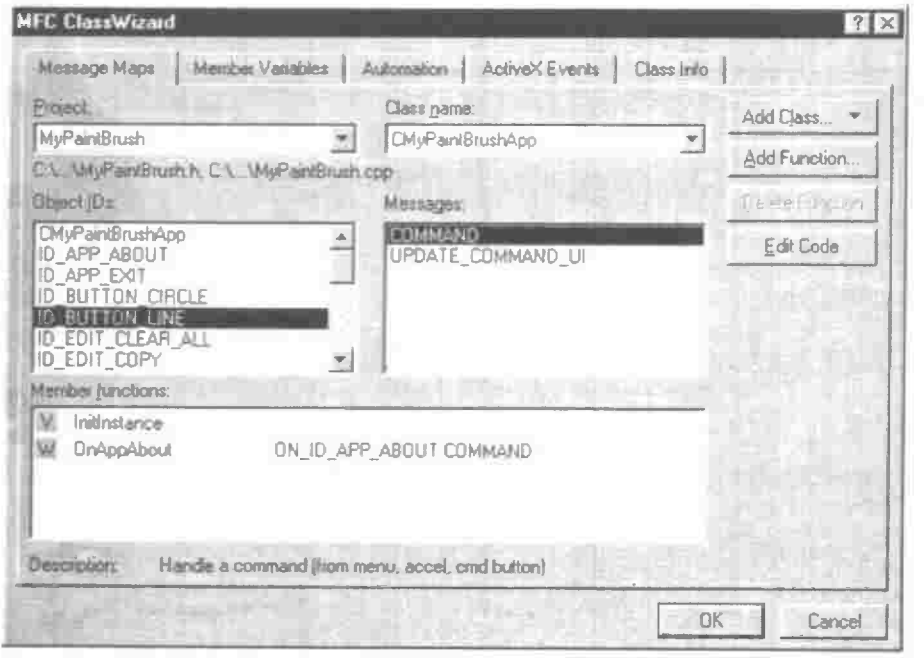
٧.قم ببناء البرنامج وتنفيذه فتشاهد أوامر القائمة الجديـــدة Line و Circle وقد ظهرت بقائمة الأدوات Tools ، ولكنها تبدو معتمـــة (غير عاملة).

الخطوة الثالثة: إضافة المقابض

١.قم بتشغيل ساحر الفصائل واختر من النافذة فصيلة التطبيق: CMyPaintBrushApp ، ثم اختر رقم التعارف ID_BUTTON_LINE من قسم أرقام التعارف ، كما بالشكل التالى. كما نرى بالشكل أن هناك مقبضين لهذا الرقم هما: **COMMAND** و **UPDATE_COMMAND_UI**. وهذه هى المقابض المعتادة لكل هدف من الأهداف فى بيئة النوافذ.

٢.أضف المقبضين باستخدام الأمر Add Function.

٣.كرر العمل بالنسبة لرقم تعارف الدائرة ID_BUTTON_CIRCLE.



شكل (١١-١٨) نافذة ساحر الفصائل ــ إضافة المقابض

تؤدى إضافة هذه المقابض إلى توليد أربعة من الدوال الجديدة بفصيلة التطبيق لكنها دوال فارغة من المحتويات. هذه الدوال سوف تكون هى المسئولة عن ربط الأزرار أو أوامر القائمة بالأحداث المناظرة ، وذلك بعد أن نضيف إليها الكود المناسب ، وهى:

- OnButtonLine()
- OnUpdateButtonLine()
- OnButtonCircle()
- OnUpdateButtonCircle()

وهذه الأوعية موضحة بالشكل التالى.

```
void CMyPaintBrushApp::OnButtonLine()          دالة زر الخط المستقيم
{
// TODO: Add your command handler code here
}
                                                دالة تحديث زر الخط المستقيم
void CMyPaintBrushApp::OnUpdateButtonLine(CCmdUI* pCmdUI)
{
// TODO: Add your command update UI handler code he re
}
                                                دالة زر الدائرة
void CMyPaintBrushApp::OnButtonCircle()
{
// TODO: Add your command handler code here
}
                                                دالة تحديث زر الدائرة
void CMyPaintBrushApp::OnUpdateButtonCircle(CCmdUI*
pCmdUI)
{
// TODO: Add your command update UI handler code here
}
```

الخطوة الرابعة: إضافة الكود إلى الدوال

قبل أن نضيف الكود إلى هذه المجموعة من أوعية الدوال ، فلنلـــق نظرة على ما هو موجود من الكود الذى كتبه لنا الساحر.

١.افتح ملف العناوين لفصيلة الإطار العام CMainFrame فتجد بـــها البيانات الأعضاء الخاصة بسطرى الحالـــة والأدوات ، وهـــى مشتقة من الفصيلتين **CToolBar** و **CStatusBar** ، وهى أعضـــاء جاهزة يضيفها الساحر من عنده:

```
m_wndStatusBar
m_wndToolBar
```

وفى إمكانك أن تستخدم هذه الأعضاء أو أن تستبدلها بأعضـــاء جديدة من عندك. وهذا هو الكود الذى أنشأه لك الساحر:

```
class CMainFrame : public CFrameWnd
{
protected: // create from serialization only
        CMainFrame();
        DECLARE_DYNCREATE(CMainFrame)
...

protected:  // control bar embedded members
        CStatusBar  m_wndStatusBar;    البيانات الأعضاء:
        CToolBar    m_wndToolBar;
```

٢.افتح الملف MainFrame.cpp ، فتجد به الدالة **OnCreate()** التـــى تتحكم فى خصائص سطر الأدوات ونصها كالآتى:

```
int CMainFrame::OnCreate(LPCREATESTRUCT lpCreateStruct)
{
```

```
        if (CFrameWnd::OnCreate(lpCreateStruct) == -1)
                return -1;
                                                خصائص سطر الأدوات:
        if (!m_wndToolBar.CreateEx(this, TBSTYLE_FLAT,
         WS_CHILD | WS_VISIBLE | CBRS_TOP
        | CBRS_GRIPPER | CBRS_TOOLTIPS | CBRS_FLYBY |
        CBRS_SIZE_DYNAMIC) ||
        !m_wndToolBar.LoadToolBar(IDR_MAINFRAME))
        {
                TRACE0("Failed to create toolbar\n");
                return -1;      // fail to create
        }

        if (!m_wndStatusBar.Create(this) ||
                !m_wndStatusBar.SetIndicators(indicators,
                  sizeof(indicators)/sizeof(UINT)))
        {
                TRACE0("Failed to create status bar\n");
                return -1;      // fail to create
        }
            احذف هذه السطور الثلاثة للحصول على سطر أدوات غير عائم (قياسى):
// TODO: Delete these three lines if you don't want the toolbar to
        //  be dockable
        m_wndToolBar.EnableDocking(CBRS_ALIGN_ANY);
        EnableDocking(CBRS_ALIGN_ANY);
        DockControlBar(&m_wndToolBar);

        return 0;
}
```

٣.كما نرى فى الكود السابق أن السطور الثلاثة المشار إليها (فى المؤخرة) هى التى تؤدى إلى إنشاء سطر الأدوات العائم ، وبحذفها تحصل على سطر قياسى. كما أن خصائص السطر تحددها الدالة **CreateEx()** من خلال عبارة الشرطية تحتوى على الصفات المطلوبة مفصولة بالمؤثر المنطقى "|". وفى إمكانك أن تحذف بعض الصفات أو تضيف إليها. وهذا هو ملخصها:

اسم الخاصية	الوظيفة
TBSTYLE_FLAT	لإنشاء زر مسطح ــ للاطلاع على الأنـــواع الأخرى اطلع على ثوابت أزرار سطر الحالـة بشاشة النجدة.
WS_CHILD	لجعل سطر الأدوات نافذة ابنة لنافذة الإطـــار العام.
WS_VISIBLE	لجعل السطر مرئياً.
CBRS_TOP	لوضع السطر مبدئياً على قمة نافذة الإطـــار العام.
CBRS_GRIPPER	لإنشاء مقبض (Gripper) على يسار الســـطر حتى يمكن سحبه منه.
CBRS_TOOLTIPS	لإظهار النص المحتوى على اسم الزر عنـــد الإشارة إليه بالفأر.
CBRS_FLYBY	لإظهار النص المحتوى على اسم الزر عنـــد الضغط عليه بالفأر
CBRS_SIZE_DYNAMIC)	لتمكين المستخدم من تغيير شـــكل الســطر وترتيب الأيقونات به.

جدول (١١-٢) خصائص أزرار سطر الأدوات

٤. **إضافة دالة بناء لأوامر الرسم:** فى هذه الفقرة نضيـــف الكـــود اللازم لشحن دالة التطبيق بالاختيار الحالى لزر سطر الأدوات. وهذا يتكون من جزئين:

- أولاً: إعلان عضواً جديداً يمثل متغير الزر وذلك بملف العناوين "MyPaintBrush.h". أنظر شريحة الكود التالية.

```
class CMyPaintBrushApp : public CWinApp
{
public:                                                    العضو الجديد
        UINT m_CurrentDrawing;
```

- ثانيا: تخصيص هذا المتغير لرقم تعارف الزر المستخدم حالياً ، وذلك بملف التطبيق "MyPaintBrushApp.cpp".

```
CMyPaintBrushApp::CMyPaintBrushApp()
{
// TODO: add construction code here,
// Place all significant initialization in InitInstance
// New code to initialize drawing:                         شحن العضو
        m_CurrentDrawing = ID_BUTTON_LINE;
}
```

٥. **إضافة الكود اللازم لرسم الخط والدائرة:** فى شريحة الكود التالية نضيف الكود اللازم إلى الدوال الأربعة التى تؤدى إلى تشغيل الأزرار بسطر الأدوات وذلك باستخدام العضو الجديد m_CurrentDrawing وربطه برقم التعارف المعين للخط أو للدائرة. كما أن الدالة OnUpdateButtonLine() تؤدى إلى تحديث حركة الزر وإظهاره كزر مضغوط أو غير مضغوط. ولو أنك حذفت الدالة الأخيرة سوف تلاحظ عدم تغير شكل الزر عند الضغط عليه.

```
void CMyPaintBrushApp::OnButtonLine()
{
// TODO: Add your command handler code here
// New code for the Line button:
  m_CurrentDrawing = ID_BUTTON_LINE;
}
void CMyPaintBrushApp::OnUpdateButtonLine(CCmdUI* pCmdUI)
{
  // TODO: Add your command update UI handler code here
  // New code for the Line button:
  pCmdUI->SetCheck(m_CurrentDrawing ==
ID_BUTTON_LINE ? 1: 0);
}

void CMyPaintBrushApp::OnButtonCircle()
{
// TODO: Add your command handler code here
// New code for the Circle button:
  m_CurrentDrawing = ID_BUTTON_CIRCLE;
}
void CMyPaintBrushApp::OnUpdateButtonCircle(CCmdUI*
pCmdUI)
{
// TODO: Add your command update UI handler code here
// New code for the Circle button:
pCmdUI->SetCheck(m_CurrentDrawing ==
ID_BUTTON_CIRCLE ? 1: 0);
}
```

لاحظ أن دالتى تحديث الزر لكل من الخط والدائـــرة ، تســتخدمان الدالة **SetCheck()** لاختبار وضبط حالة الزر بحسب مـــا إذا كـــان مضغوطاً أم لا. وهذه الدالة تابعة للفصيلة **CCmdUI** ، وهى فصيلـة مستقلة (ليست موروثة) وهـــى تســتخدم أساســاً مــع المقبــض **ON_UPDATE_COMMAND_UI**.

إن المهمة التى أنجزناها فى هذا البرنامج هى خلق أزرار سطر الأدوات التى تستجيب للضغط عليها وترتبط بمقابض معينة. ومع ذلك فإن هذا الطراز من برنامج الرسم لا يرسم إلاّ الخطوط المستقيمة. ولكى يؤدى زر رسم الدائرة عمله كاملاً فلابد من إجراء نفس الخطوات التى اتبعناها مع الخط المستقيم بدءاً من إنشاء فصيلة للدائرة وتحديد كيفية رسمها ومسحها. وسوف نترك لك إكمال هذه المهمة كتدريب. ومع ذلك فإن دوال رسم الأشكال المختلفة سوف نعرضها فى الأبواب القادمة.

الموجز

١. تعلمنا فى هذا الباب كيفية إضافة قضبان الانزلاق التى تمنحك خاصية الانزلاق رأسياً وأفقياً عندما تزيد المحتويات عن حدود الشاشة.

٢. وفى أثناء جولتنا بموضوع الانزلاق عرفنا خصائص نظام الإحداثيات المستخدم فى النوافذ بنوعيه: الإحداثيات الطبيعية والإحداثيات المنطقية. وعرفنا كيفية التحويل بينهما.

٣. عرفنا كيفية استخدام بيئة الموارد فى إنشاء سطر أدوات جديد أو فى إضافة سطر أدوات إلى السطر سابق التعريف الموجود بالعنصر IDR_MAINFRAME. كما عرفنا كيفية استخدام ساحر الفصائل فى إنشاء المقابض المسئولة عن استخدام أزرار الأدوات ، وعرفنا كيفية إضافة الكود اللازم إلى هذه المقابض.

٤. عرفنا أيضاً أن أزرار سطر الحالة يمدنا بها الساحر تلقائيـــاً وأنه لا ضرورة تدعو إلى إنشائها من جديد.

٥. استخدمنا فى هذه الجولة الفصائل والدوال والثوابت الآتية:

المنشآت:

SIZE (لتخزين أبعاد المنظور)

الفصائل:

CScrollView
CClientDC
CDC
CToolBar
CScrollBar
CCmdUI
CWnd

الدوال:

CClientDC::OnPrepareDC()
CDC::DPtoLP
CView::OnInitialUpdate()
CScrollView::SetScrollSizes()
CToolbar::CreateEx()
CCmdUI::SetCheck()
CWnd::OnCreate()
CWnd::OnMouseMove()
CWnd::OnLButtonUp()