

MDI

(Multiple Document Interfaces)

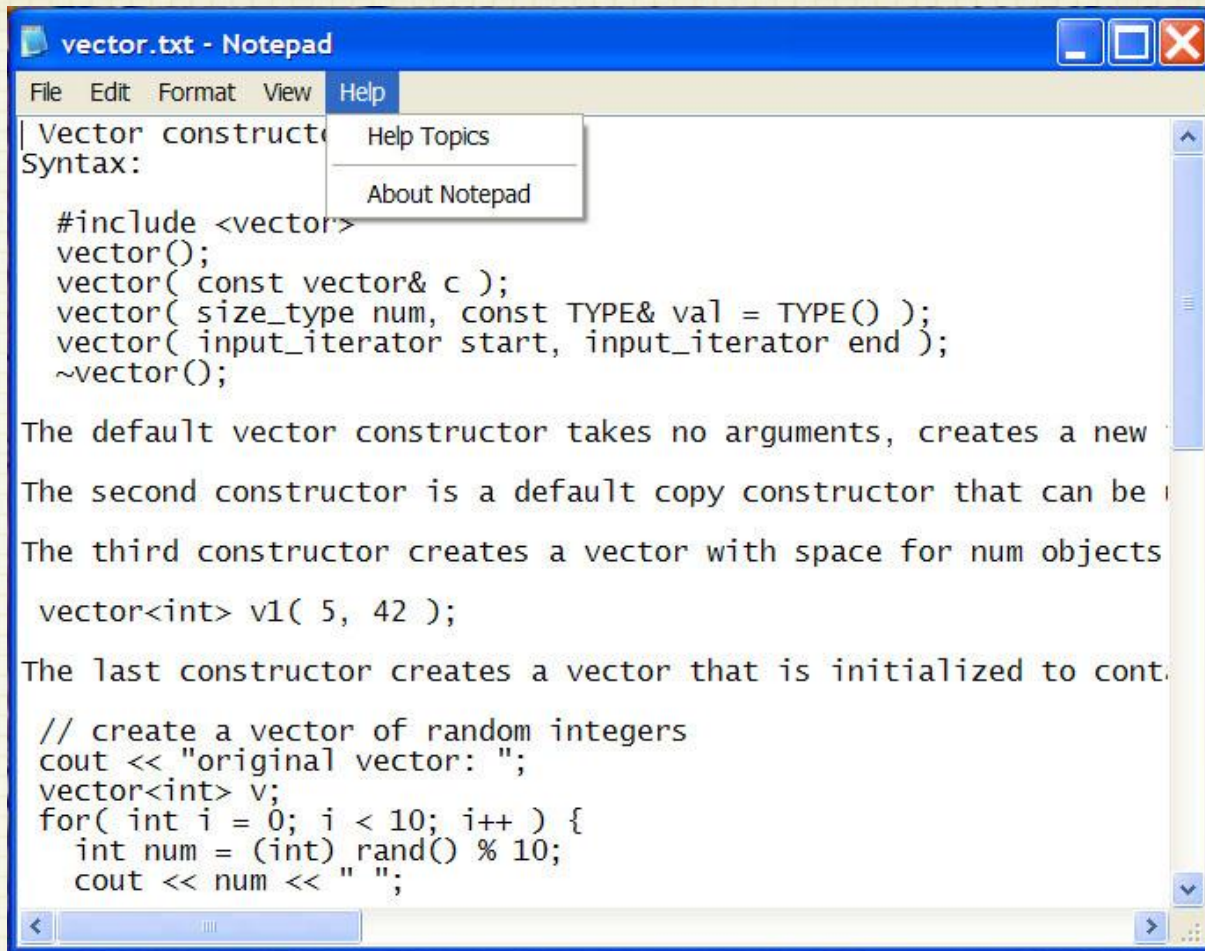
Tổng quát

- Phần lớn các ứng dụng của Windows đều rơi vào một trong 3 loại sau:
 - Single Document Interfaces (SDI)
 - Explorer Interfaces
 - Multiple Document Interfaces (MDI)

Single Document Interface

- Mỗi tài liệu sẽ được thể hiện ở một cửa sổ đơn.
- Trong Windows tiêu biểu cho loại giao diện này là Notepad hay Wordpad.

Cửa sổ SDI



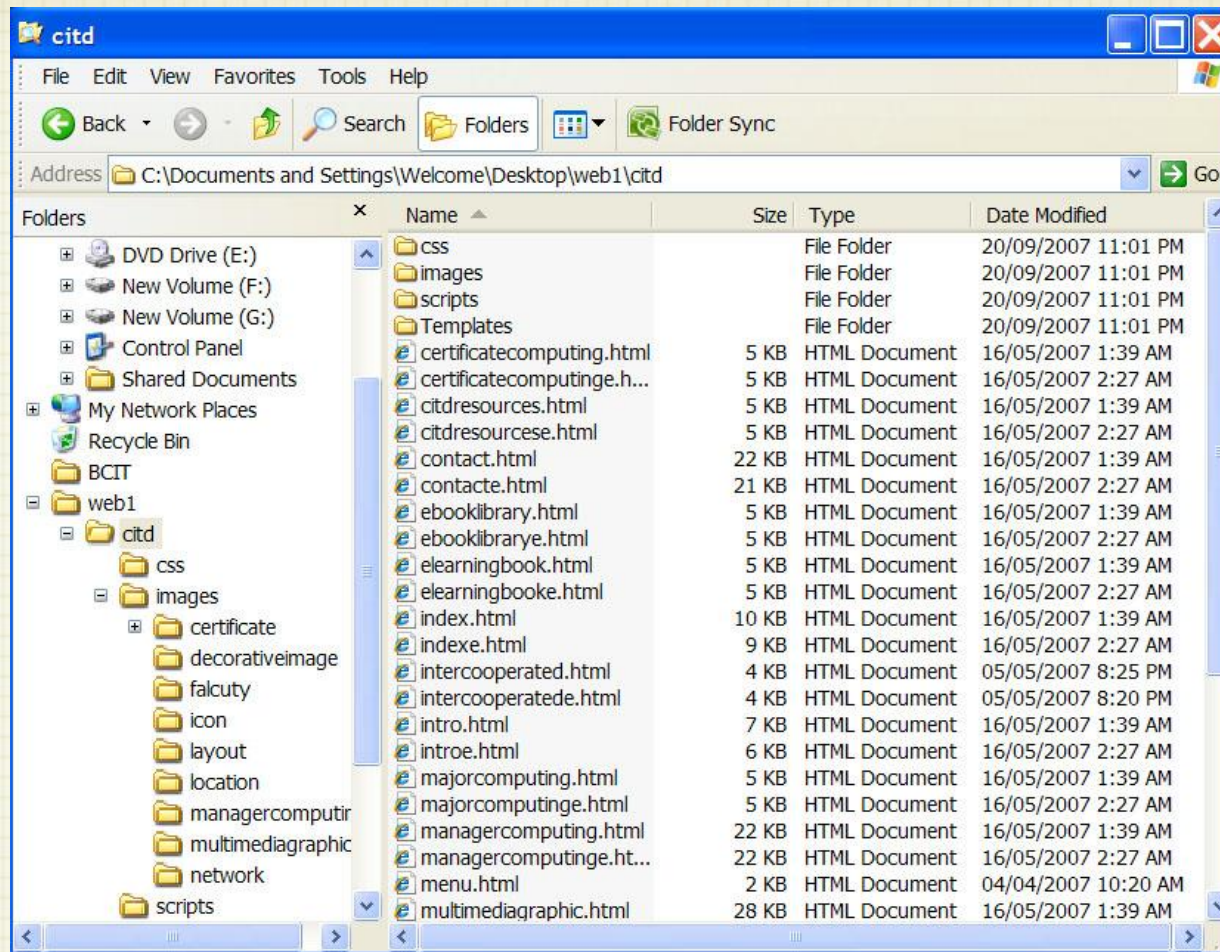
The screenshot shows a Notepad window with a blue title bar and a menu bar (File, Edit, Format, View, Help). The text content is as follows:

```
Vector constructor  
Syntax:  
  
#include <vector>  
vector();  
vector( const vector& c );  
vector( size_type num, const TYPE& val = TYPE() );  
vector( input_iterator start, input_iterator end );  
~vector();  
  
The default vector constructor takes no arguments, creates a new  
The second constructor is a default copy constructor that can be  
The third constructor creates a vector with space for num objects  
vector<int> v1( 5, 42 );  
  
The last constructor creates a vector that is initialized to cont.  
  
// create a vector of random integers  
cout << "original vector: ";  
vector<int> v;  
for( int i = 0; i < 10; i++ ) {  
    int num = (int) rand() % 10;  
    cout << num << " ";  
}
```

Explorer Interface

- Đây là cửa sổ mà thông tin sẽ được hiển thị theo một hệ thống phân cấp.
- Thông thường một TreeView control sẽ dùng để hiển thị hệ thống phân cấp này.
- Thông tin chi tiết của mỗi nút trên TreeView được chọn sẽ hiển thị trong một ListView control.

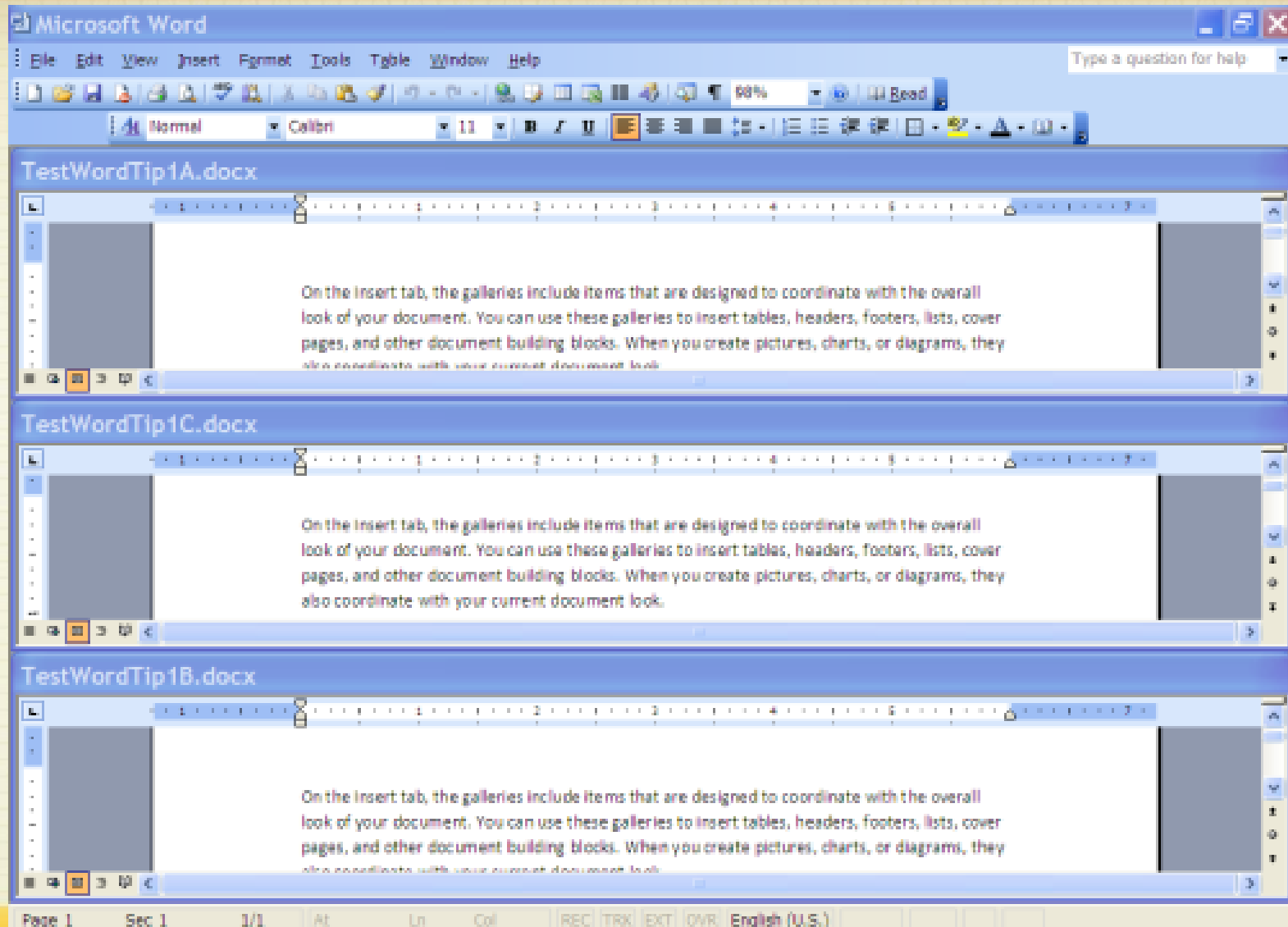
Cửa sổ Explorer Interface



Khái niệm cơ bản về MDI

- Multiple Document Interface (MDI) là một đặc tả quản lý tài liệu trong Microsoft Windows.
- Mô tả một cấu trúc cửa sổ và giao diện người dùng, cho phép người sử dụng làm việc với nhiều tài liệu trong một ứng dụng đơn.

Cửa sổ MDI



Ví dụ

- Windows duy trì nhiều ứng dụng Windows trong một màn hình đơn.
- Một ứng dụng MDI duy trì nhiều cửa sổ tài liệu trong một vùng client đơn.

Khái niệm cơ bản về MDI

- Đặc tả MDI đã xuất hiện từ Windows 2.0, nhưng các ứng dụng MDI lúc đó rất khó viết và cần nhiều công sức lập trình.
- Từ Windows 3.0 trở đi, nhiều chức năng đã được mở rộng và hỗ trợ được đưa vào.

Các thành phần của MDI

- Cửa sổ ứng dụng chính của một chương trình MDI theo kiểu cổ điển gồm có
 - Thanh tiêu đề.
 - Một trình đơn.
 - Một đường viền thay đổi kích thước.
 - Một icon trình đơn hệ thống.
 - Các nút minimize/maximize/close.

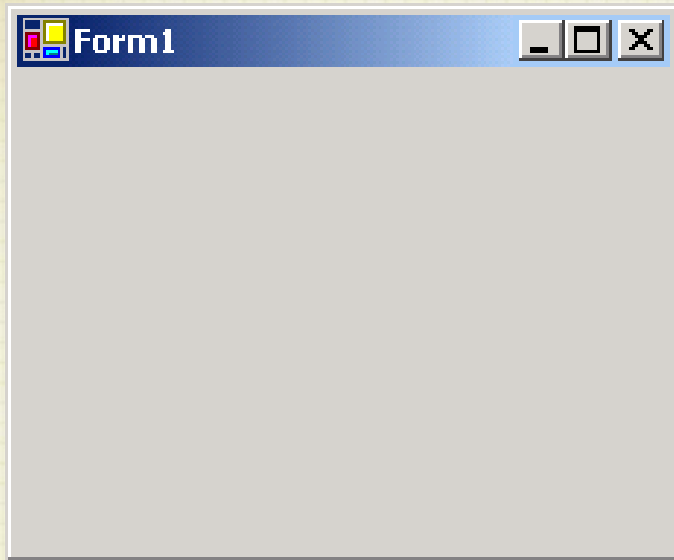
MDI

- Vùng client thường được gọi là “workspace” và không sử dụng trực tiếp để hiển thị output của chương trình.
- Workspace có thể không chứa hay chứa nhiều cửa sổ con, mỗi cửa sổ con hiển thị một tài liệu.
- Các cửa sổ con rất giống các cửa sổ ứng dụng bình thường và các cửa sổ ứng dụng chính của một chương trình MDI
- Ở tại một thời điểm, chỉ một cửa sổ tài liệu được kích hoạt và nó xuất hiện trước tất cả các cửa sổ tài liệu khác.
- Tất cả các cửa sổ tài liệu con được giới hạn bởi vùng workspace và không bao giờ xuất hiện bên ngoài cửa sổ ứng dụng.

Các loại MDI

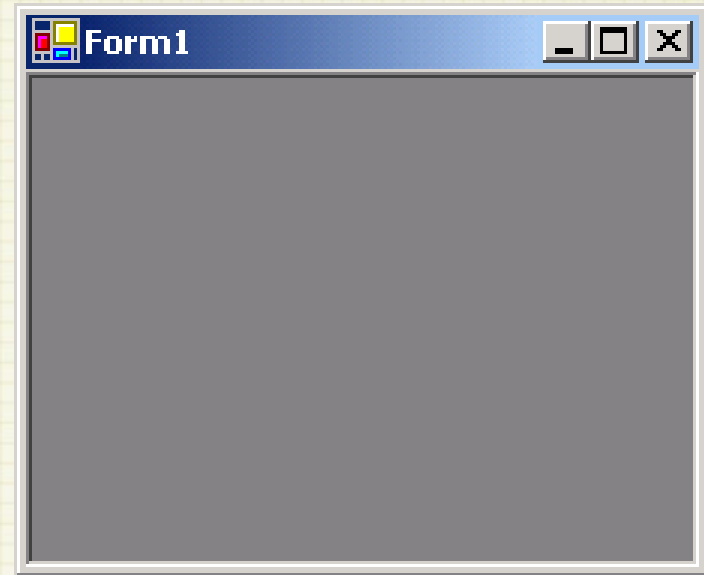
- Document application
- Workspace application

MDI



SDI

Single Document Interface



MDI

Multiple Document Interface

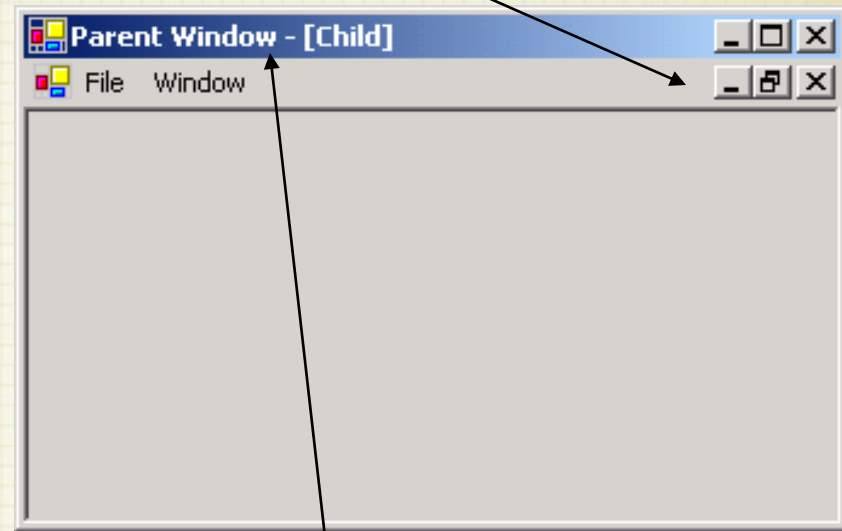
MDI

Parent's icons: minimize, maximize and close



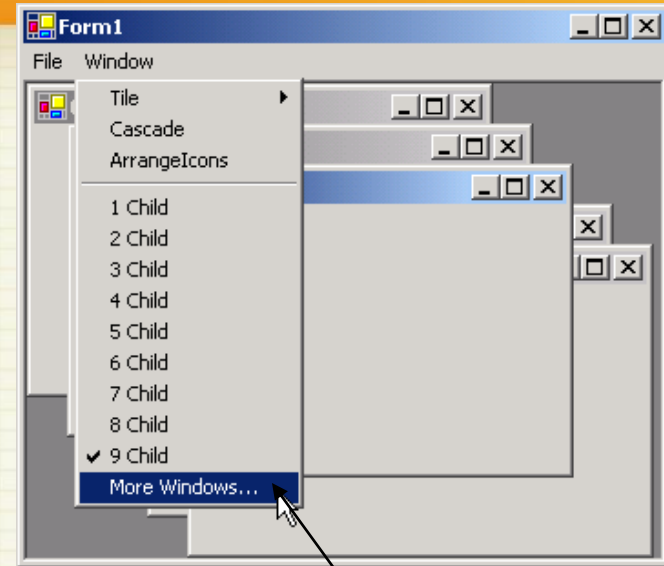
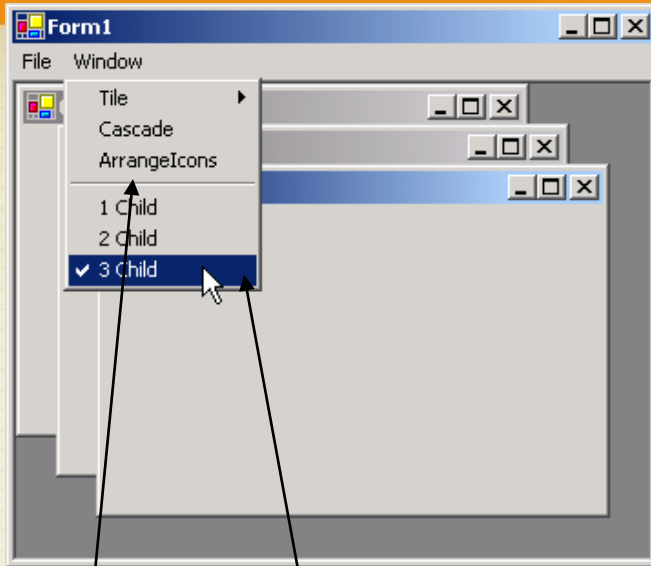
Minimized child's icons: restore, maximize and close

Maximized child's icons: minimize, restore and close



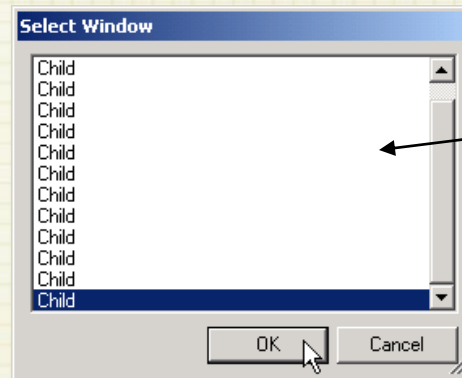
Parent's title bar displays maximized child

MDI

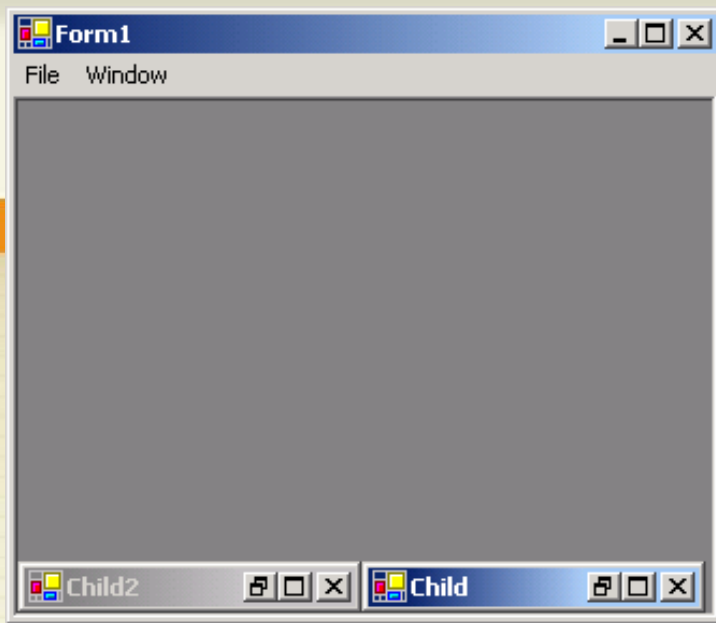


Separator bar and
child windows

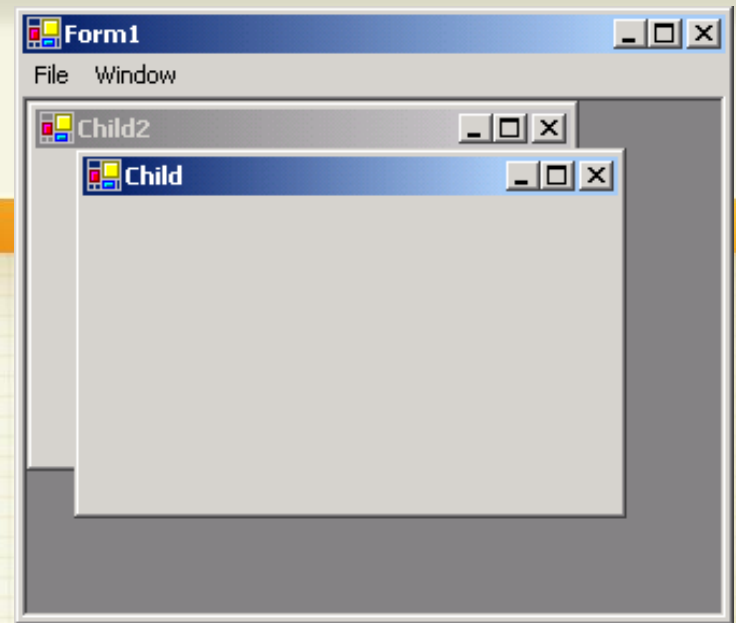
Child windows list



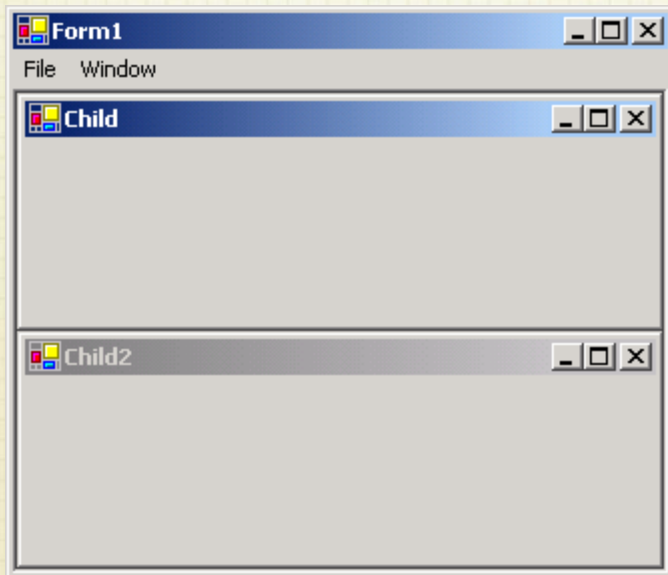
9 or more child windows
enables the More
Windows... option



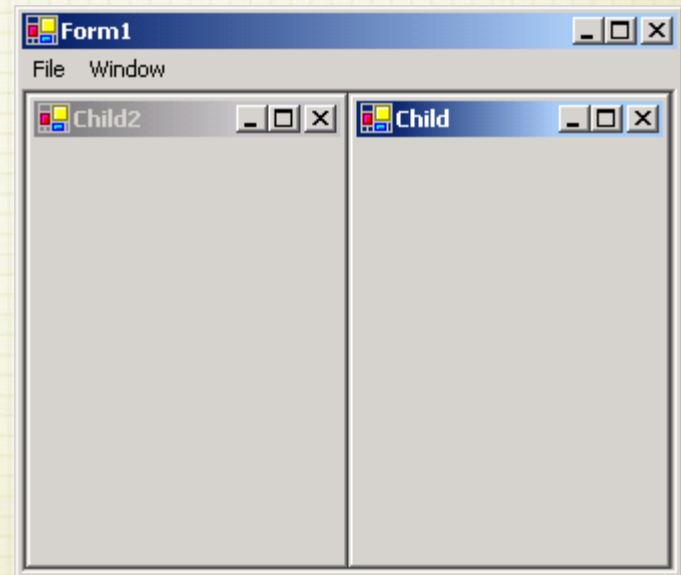
ArrangeIcons



Cascade



TileHorizontal



TileVertical

MDI trên .NET

- Trên .NET, việc phân biệt các cửa sổ bình thường và cửa sổ MDI không thật rõ ràng.
- Có thể biến đổi bất cứ cửa sổ nào thành cửa sổ MDI Parent bằng cách đặt
`this.IsMdiContainer = true;`

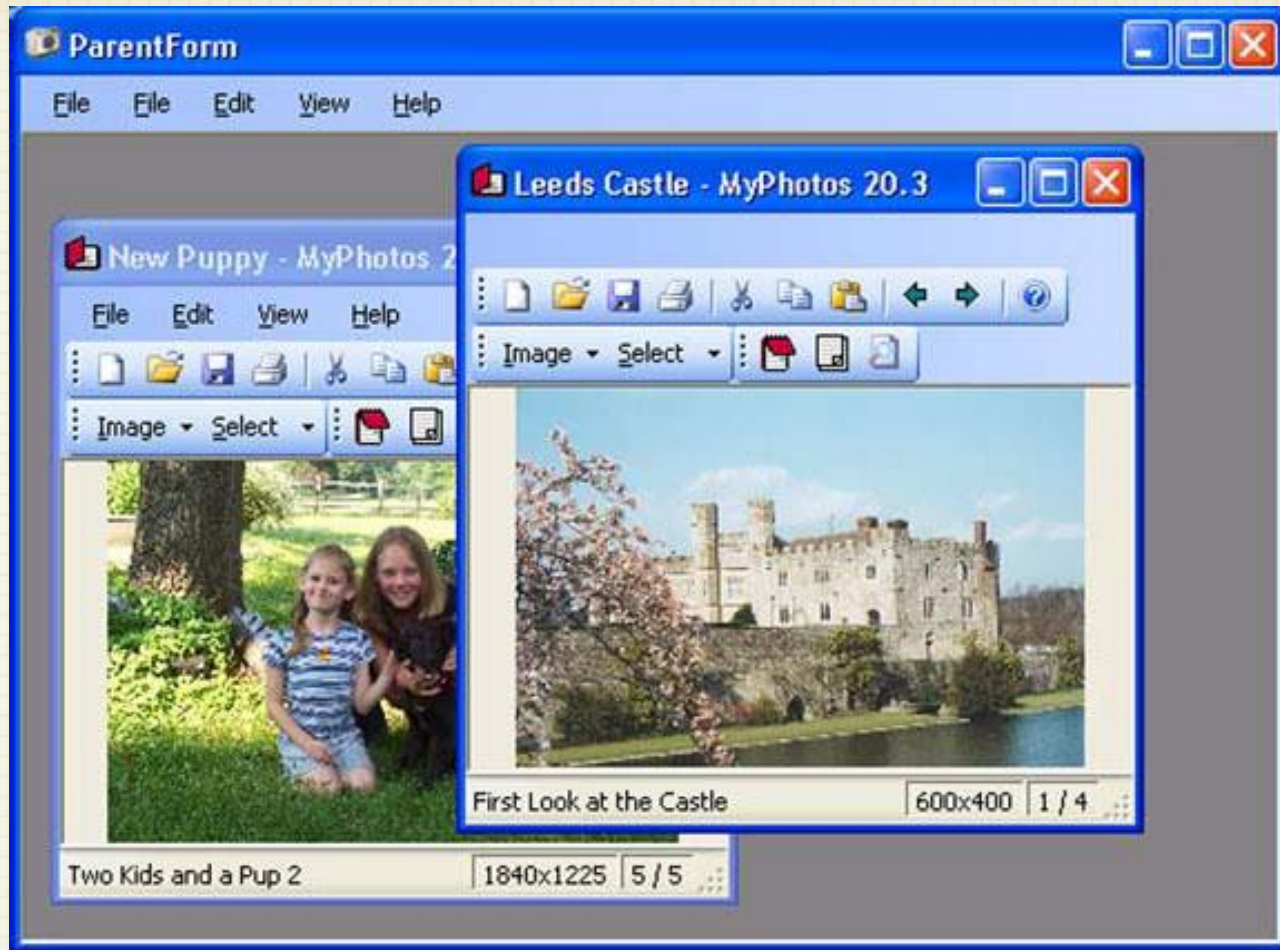
MDI trên .NET

- Khi được hiển thị như là một MDI container, biểu mẫu sẽ trở thành màu xám.
- Muốn thêm một cửa sổ mới như là một MDI Child, chỉ cần đặt thuộc tính MDI Parent của biểu mẫu Child trong hàm Parent_Load()

MDI trên .NET

```
private void Parent_Load (object sender,  
    System.EventArgs e)  
{  
    Child frmChild = new Child();  
    frmChild.MdiParent = this;  
    frmChild.Show();  
}
```


Minh hoạ MDI Form



Các lớp thường dùng

Lớp	Loại	Tên	Mô tả
Form	Properties	ActiveMdiChild	Gets the MDI child window that is currently active.
		IsMdiChild	Gets whether the form is an MDI child.
		IsMdiContainer	Gets whether the form is an MDI container form
		MdiChildren	Gets the set of MDI children contained by this form as an array of <i>Form</i> object.
		MdiParent	Gets or sets the MDI container for this form. If set, then this form is an MDI child form.
		MergedMenu	Gets the <i>MainMenu</i> object representing the current merged menu for an MDI container form
	Methods	LayoutMdi	Arranges the MDI children within this form using a given layout style.

Lớp	Loại	Tên	Mô tả
	Events	MdiChildActive	Occurs when an MDI child form is activated or deactivated with an MDI application . Note that MDI children do not receive the Activated and Deactivated events.
Menu	Properties	MdiListItem	Gets the <i>MenuItem</i> object contained by this menu that displays a list of MDI child forms for the associated form object.
	Method	MergeMenu	Merges the <i>MenuItem</i> objects in a given menu with those contained by this menu.
MenuItem	Properties	MdiList	Gets or sets whether this menu should be populated with a list of MDI child forms contained by the associated form.
		MergeOrder	Gets or sets the relative position of this menu item when it is merged with another menu.
		MergeType	Gets or sets how this menu should be merged with other menus. The default is <i>MergeType.Add</i>

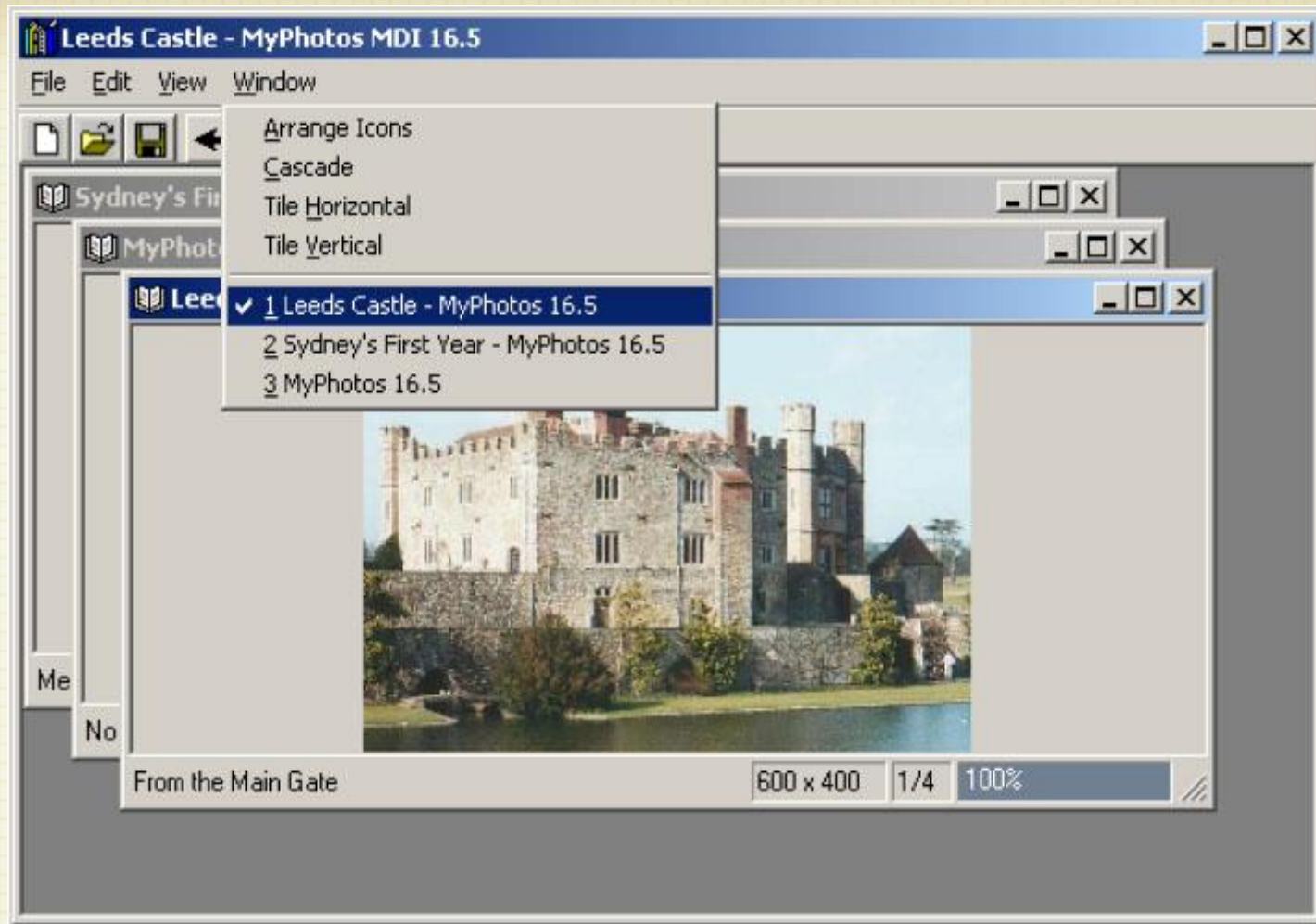
MDI trên .NET

- Trong một project có thể có nhiều MDI Parent.
- Có thể chuyển đổi một MDI Child từ MDI Parent này sang MDI Parent khác bằng cách thay đổi thuộc tính MdiParent.

Sắp xếp các MDI Child

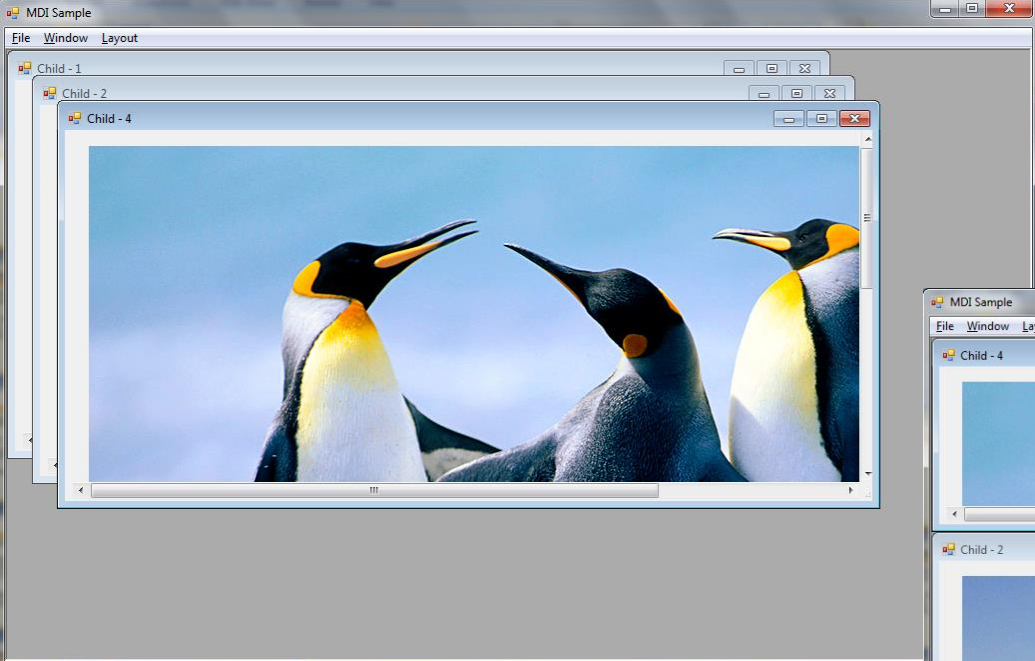
- Để tạo một danh sách các MDI Child, chỉ cần thêm một top-level menu item (thường mang tên Window), rồi cho thuộc tính MdiList = true.
- Bộ máy Windows.Forms sẽ tự động thêm một item vào cuối submenu cho mỗi cửa sổ MDI Child.

Người dùng có thể chuyển từ cửa sổ này sang cửa sổ khác bằng cách sử dụng trình đơn.

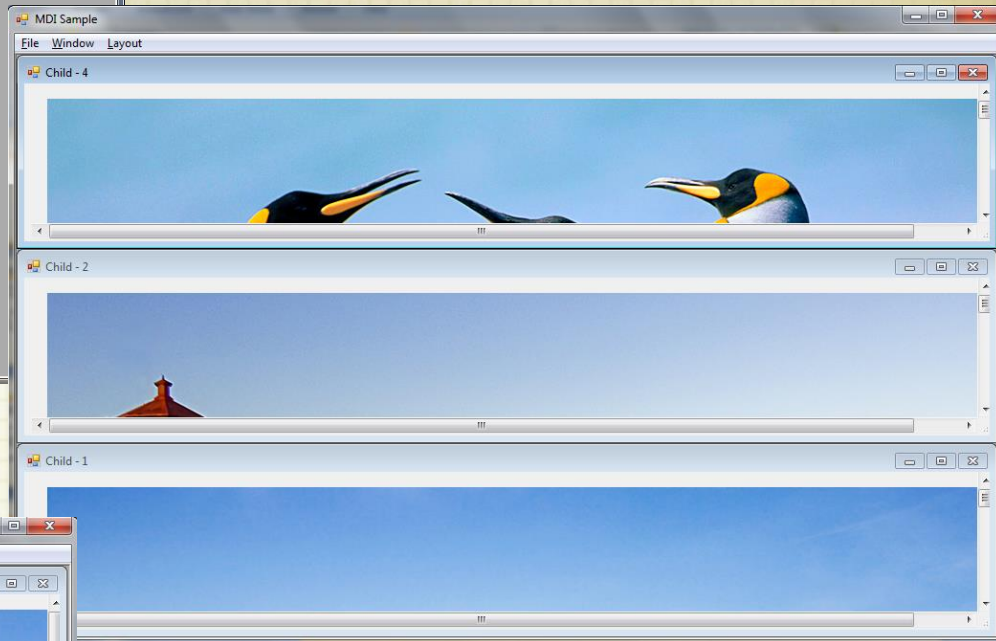


Sắp xếp các Child Form

- Có nhiều các sắp xếp các child form trên parent form
 - Cascade
 - TileHorizontal
 - TileVertical
 - ArrangeIcons



Cascade



TileHorizontal



TileVertical

Sắp xếp các Child Form

- Nếu muốn sắp xếp theo kiểu Cascade hoặc Tile, cần thêm các chức năng này vào trình đơn.
- Mỗi MDI container đều hỗ trợ hàm `LayoutMdi()`, hàm này nhận giá trị từ Enumeration **MdiLayout** và sắp xếp tự động các cửa sổ.

Sắp xếp các Child Form

- **Đoạn chương trình sắp xếp theo kiểu Cascade**

```
private void mnuCascade_Click ( Object sender, System.EventArgs e)
{
    this.LayoutMdi (MdiLayout.Cascade)
}
```

- **Đoạn chương trình sắp xếp theo kiểu Tile Horizontal**

```
private void mnuTileH_Click ( Object sender , System.EventArgs e)
{
    this.LayoutMdi (MdiLayout.TileHorizontal)
}
```

Sắp xếp các Child Form

- Ngoài ra, có thể tạo ra các cách sắp xếp riêng tùy theo mỗi ứng dụng.
- Ví dụ, đoạn chương trình sau cho phép thu nhỏ lại tất cả các cửa sổ đang mở.

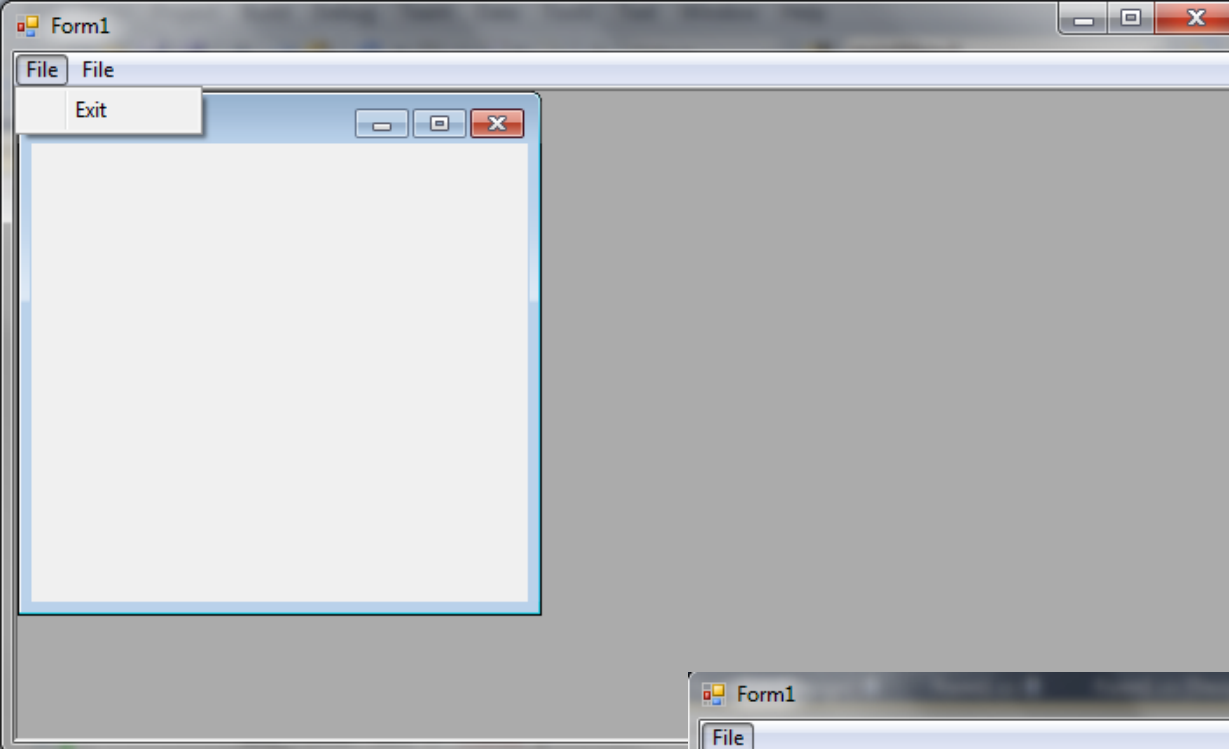
```
private void mnuMinimizeAll_Click (Object sender,  
    System.EventArgs e)  
{  
    foreach (Form frm in this.MdiChildren) {  
        frm.WindowState = FormWindowState.Minimized;  
    }  
}
```

Merge Menu

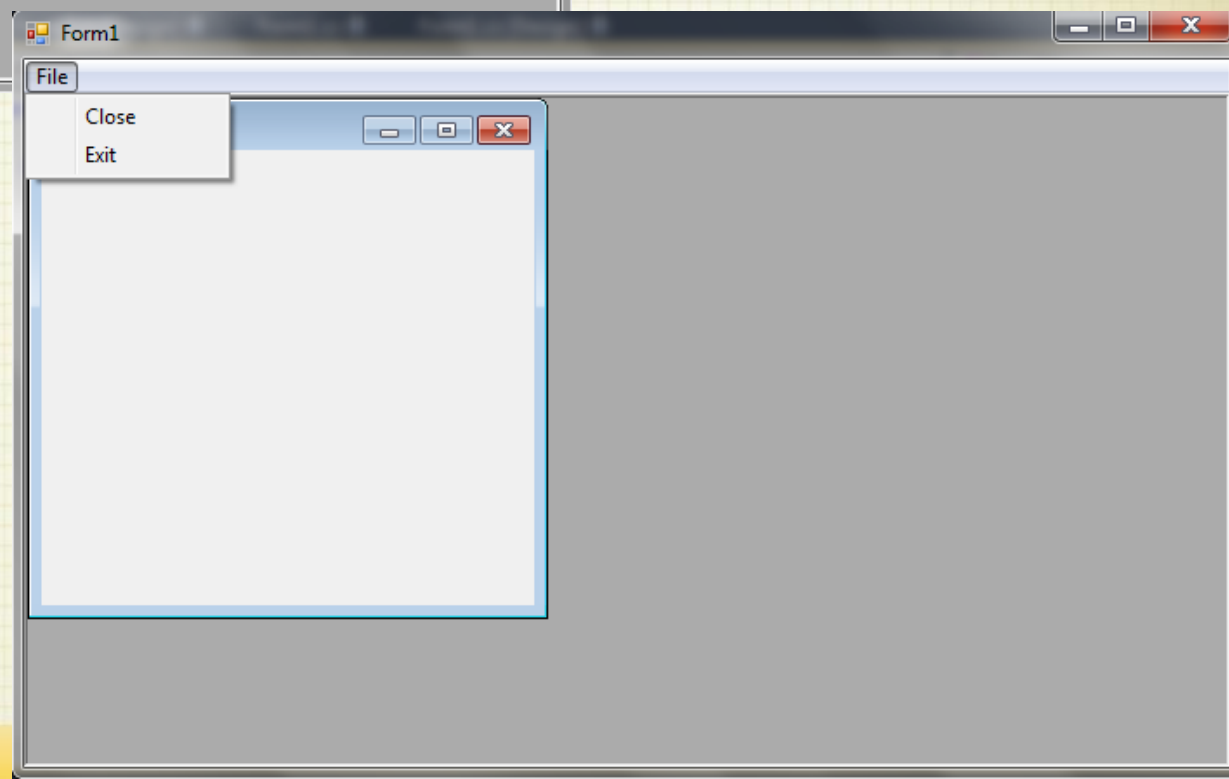
- Nếu trên cả Parent Form và Child Form đều có menu thì khi Child Form được show, menu của Child Form sẽ được thêm vào phía sau menu của Parent Form
- Nếu trên Child Form và Parent Form có các menu item có caption giống nhau thì sẽ xuất hiện cả 2 menu item đó trên cùng 1 thanh menu (vd có cả 2 menu File trên cùng một menu)

Merge Menu

- Để tránh tình trạng này ta có thể trộn chung 2 menu của Parent Form và Child Form lại với nhau:
`mnu_PFile.MergeType = MenuMerge.MergeItems; //Parent`
`mnu_CFile.MergeType = MenuMerge.MergeItems; //Child`
- Có thể định vị trí cho các item con sau khi trộn bằng thuộc tính MergeOrder
`mnu_Exit.MergeOrder = 2;`



Menu chưa được merge



Menu được merge