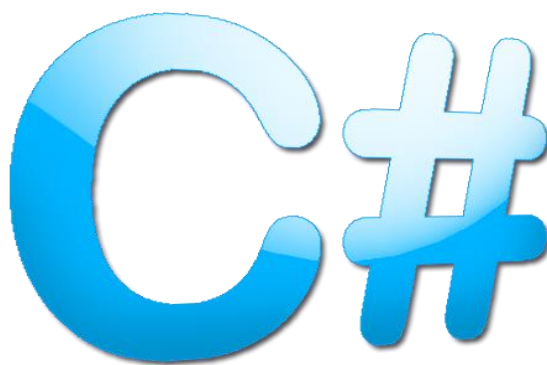


TRƯỜNG CAO ĐẲNG KỸ THUẬT CAO THẮNG
KHOA ĐIỆN TỬ TIN HỌC
BỘ MÔN TIN HỌC
∞📖∞

TRẦN THANH TUẤN - DƯƠNG TRỌNG ĐÌNH

LẬP TRÌNH WINDOWS NÂNG CAO

(Lưu hành nội bộ)



TP HCM – NĂM 2014

LỜI GIỚI THIỆU

Lập trình windows nâng cao là môn học cần thiết cho lập trình viên, nhằm cung cấp kiến thức trong lập trình windows forms, thao tác với các điều khiển nâng cao, kết nối cơ sở dữ liệu, mô hình 3 lớp... Ngoài ra, môn học còn là nền tảng cho các môn học lập trình web với Asp.net.

*Tài liệu Lập Trình Windows Nâng Cao này gồm 5 chương:
Chương 1: Một số điều khiển nâng cao*

Chương 2: Xây dựng ứng dụng MDI

Chương 3: ADO.NET

Chương 4: Local Report

Chương 5: Mô hình 3 lớp

Các chương được trình bày chi tiết với nhiều ví dụ minh họa, giúp người đọc nắm vững được các nền tảng lý thuyết căn bản cũng như có khả năng vận dụng để xây dựng các ứng dụng thực tiễn. Chúng tôi hy vọng tài liệu này sẽ giúp cho sinh viên ngành Công nghệ thông tin trường Cao Đẳng Kỹ Thuật Cao Thắng học môn Lập trình windows nâng cao và các môn học liên quan một cách hiệu quả hơn.

Trong quá trình biên soạn, chúng tôi đã nhận được các ý kiến đóng góp quý báu từ đồng nghiệp cũng như sự hỗ trợ của nhà trường. Chúng tôi xin chân thành cảm ơn và hy vọng sẽ tiếp tục nhận được các ý kiến đóng góp nhằm bổ sung, hiệu chỉnh nội dung để tài liệu này có thể trở thành tài liệu học và tham khảo bổ ích cho các em sinh viên cũng như giảng viên ngành Công nghệ thông tin trường Cao Đẳng Kỹ Thuật Cao Thắng.

Nhóm tác giả

Bộ môn Tin Học - Khoa Điện Tử - Tin Học

Trường Cao Đẳng Kỹ Thuật Cao Thắng

Trần Thanh Tuấn – Dương Trọng Đỉnh

MỤC LỤC

CHƯƠNG 1: MỘT SỐ ĐIỀU KHIỂN NÂNG CAO	1
1.1. ListView	1
1.1.1. Khái niệm	1
1.1.2. Một số thuộc tính	2
1.1.3. Một số phương thức	5
1.1.4. Một số sự kiện	5
1.1.5. Lớp ListViewItem	6
1.1.6. Ví dụ minh họa	7
1.2. TreeView	16
1.2.1. Khái niệm	16
1.2.2. Một số thuộc tính	16
1.2.3. Một số phương thức	18
1.2.4. Một số sự kiện	19
1.2.5. Lớp TreeNode	19
1.2.6. Ví dụ minh họa	22
CHƯƠNG 2: XÂY DỰNG ỨNG DỤNG MDI	27
2.1. Tổng quan ứng dụng MDI	27
2.2. Một số thuộc tính	27
2.3. Một số phương thức	28
2.4. Một số sự kiện	28
2.5. Ví dụ minh họa	28
CHƯƠNG 3: ADO.NET	32
3.1. Tổng quan ADO.Net	32
3.1.1. Giới thiệu	32
3.1.2. Các thành phần của ADO.Net	32
3.2. Làm việc với mô hình kết nối (connected)	34
3.2.1. Đối tượng Connection	34
3.2.2. Đối tượng Command	35
3.2.3. Đối tượng DataReader	37
3.2.4. Các bước lập trình ứng dụng thao tác với CSDL	39
3.2.5. Ví dụ minh họa	39
3.3. Làm việc với mô hình ngắt kết nối (dis-connected)	48
3.3.1. Đối tượng DataAdapter	48
3.3.2. Đối tượng DataSet	50
3.3.3. Đối tượng DataTable	51
3.3.4. Đối tượng DataView	53
3.3.5. Điều khiển DataGridView	54
3.3.6. Điều khiển Combobox nâng cao	57
3.3.7. Ví dụ minh họa	59
CHƯƠNG 4: LOCAL REPORT	63
4.1. Khái niệm report	63

4.2. Local Report	63
4.2.1. Một số thuộc tính	64
4.2.2. Một số phương thức	64
4.2.3. Một số sự kiện	64
4.3. ReportParameter	64
4.3.1. Một số thuộc tính	64
4.3.2. Phương thức khởi tạo	65
4.4. Điều khiển ReportViewer	65
4.4.1. Một số thuộc tính	65
4.4.2. Một số phương thức	66
4.4.3. Một số sự kiện	67
4.5. Ví dụ minh họa	68
4.5.1. Tạo report đơn giản	68
4.5.2. Tạo report có tham số (Parameter)	72
4.5.3. Tạo report có Group	74
4.5.4. Một số thao tác trong Local Report	77
CHƯƠNG 5: MÔ HÌNH 3 LỚP	82
5.1. Tổng quan mô hình 3 lớp (Three Layers)	82
5.1.1. Giới thiệu	82
5.1.2. Các thành phần của mô hình 3 lớp	82
5.2. Ví dụ minh họa	83
PHỤ LỤC : DANH SÁCH CÁCH ĐẶT TÊN GỢI Ý CHO CÁC ĐIỀU KHIỂN	96
TÀI LIỆU THAM KHẢO	98

CHƯƠNG 1: MỘT SỐ ĐIỀU KHIỂN NÂNG CAO

1.1. ListView

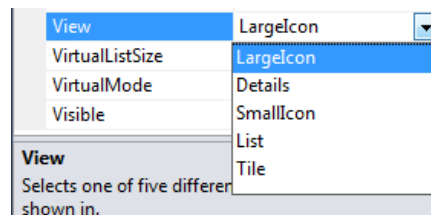
1.1.1. Khái niệm

ListView là điều khiển được dùng để hiển thị danh sách các phần tử với nhiều hình ảnh khác nhau.

Mỗi phần tử trong ListView là đối tượng thuộc lớp ListViewItem.

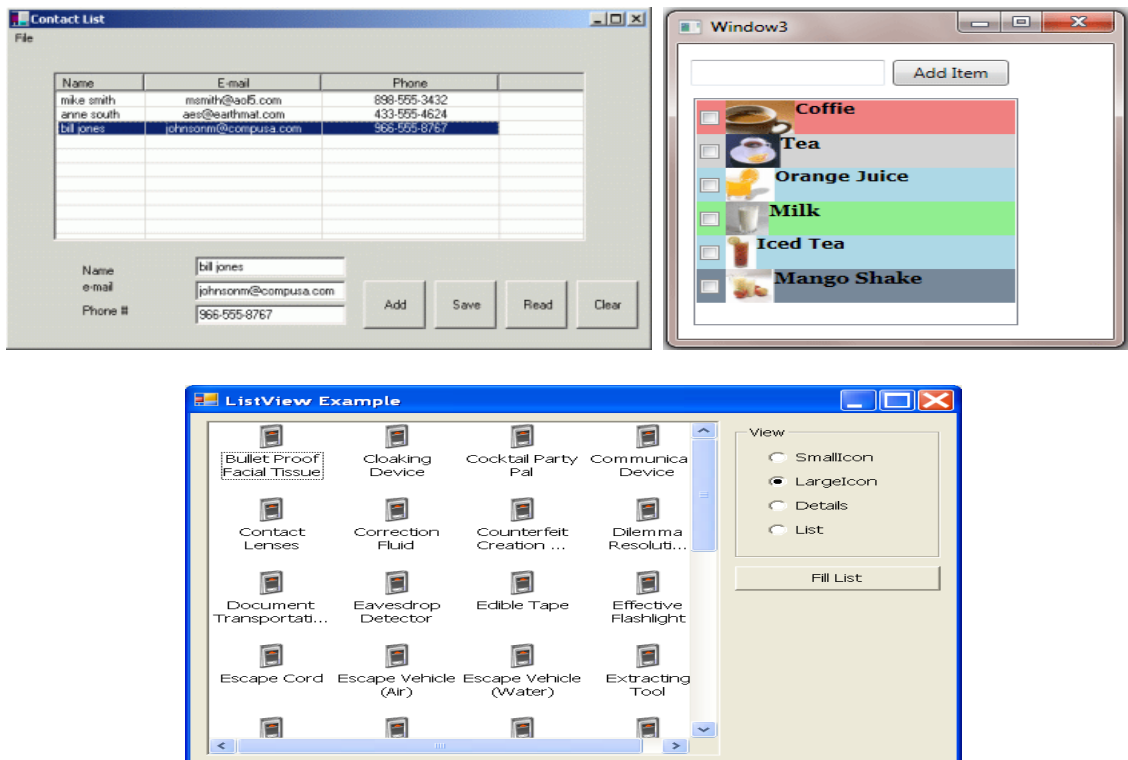
ListView hỗ trợ nhiều khung nhìn khác nhau:

- LargeIcon
- SmallIcon
- Details
- List
- Tile



Lớp ListView thuộc namespace: System.Windows.Forms.

Một số ví dụ về ListView:



1.1.2. Một số thuộc tính

1.1.2.1. CheckBoxes

- ❖ Kiểu dữ liệu: bool
- ❖ Mô tả:
 - Cho phép hiển thị checkbox bên cạnh mỗi phần tử trong ListView
 - Giá trị mặc định là false
 - Lưu ý: Checkbox sẽ không hiển thị trong khung nhìn Tile

1.1.2.2. CheckItems

- ❖ Kiểu dữ liệu: ListView.CheckedListViewItemCollection
- ❖ Mô tả:
 - Danh sách các phần tử được chọn khi check vào Checkbox
 - Được sử dụng khi thuộc tính CheckBoxes = true

1.1.2.3. Columns

- ❖ Kiểu dữ liệu: ListView.ColumnHeaderCollection
- ❖ Mô tả:
 - Danh sách các cột trong ListView
 - Các cột sẽ được hiển thị trong khung nhìn Details

1.1.2.4. FocusItem

- ❖ Kiểu dữ liệu: ListViewItem
- ❖ Mô tả:
 - Phần tử được trỏ đến trong ListView
 - Nếu không có phần tử nào được trỏ đến thì FocusItem = null

1.1.2.5. FullRowSelect

- ❖ Kiểu dữ liệu: bool
- ❖ Mô tả:
 - Cho phép tô hết dòng của phần tử được chọn trong ListView
 - Giá trị mặc định là false
 - Được sử dụng trong khung nhìn Details

1.1.2.6. GridLines

- ❖ Kiểu dữ liệu: bool
- ❖ Mô tả:
 - Cho phép hiển thị các đường kẻ quanh các phần tử trong ListView
 - Giá trị mặc định là false
 - Được sử dụng trong khung nhìn Details

1.1.2.7. Items

- ❖ Kiểu dữ liệu: ListView.ListViewItemCollection
- ❖ Mô tả:
 - Danh sách các phần tử trong ListView

1.1.2.8. LabelEdit

- ❖ Kiểu dữ liệu: bool
- ❖ Mô tả:
 - Cho phép chỉnh sửa nội dung của nhãn của phần tử trong ListView khi thực thi (run time)
 - Giá trị mặc định là false

1.1.2.9. LargeImageList

- ❖ Kiểu dữ liệu: ImageList
- ❖ Mô tả:
 - Danh sách các hình ảnh được dùng trong khung nhìn LargeIcon
 - Giá trị mặc định là null

1.1.2.10. MultiSelect

- ❖ Kiểu dữ liệu: bool
- ❖ Mô tả:
 - Cho phép chọn nhiều phần tử trong ListView
 - Giá trị mặc định là false
 - Lưu ý: Dùng phím SHIFT để chọn nhiều phần tử liên tiếp nhau, phím CTRL để chọn nhiều phần tử rời rạc nhau

1.1.2.11. Scrollable

- ❖ Kiểu dữ liệu: bool
- ❖ Mô tả:
 - Cho phép hiển thị thanh cuộn trên điều khiển ListView
 - Giá trị mặc định là false

1.1.2.12. SelectedIndices

- ❖ Kiểu dữ liệu: ListView.SelectedIndexCollection
- ❖ Mô tả:
 - Danh sách chỉ số của các phần tử được chọn trong ListView hay còn gọi là mảng chỉ số chọn
 - Lưu ý: chỉ số có kiểu dữ liệu là kiểu số nguyên (int)

1.1.2.13. SelectedItems

- ❖ Kiểu dữ liệu: ListView.SelectedListViewItemCollection hay còn gọi là mảng chọn
- ❖ Mô tả:
 - Danh sách các phần tử được chọn trong ListView

1.1.2.14. SmallImageList

- ❖ Kiểu dữ liệu: ImageList
- ❖ Mô tả:
 - Danh sách các hình ảnh được sử dụng trong khung nhìn SmallIcon
 - Giá trị mặc định là null

1.1.2.15. View

- ❖ Kiểu dữ liệu: View
- ❖ Mô tả:
 - Cách hiển thị các phần tử trong ListView
 - Các giá trị của View là: LargeIcon, SmallIcon, Details, List, Tile
 - Giá trị mặc định là LargeIcon

1.1.3. Một số phương thức

1.1.3.1. Phương thức khởi tạo

- ❖ Cú pháp: `public ListView()`
- ❖ Mô tả:
 - Dùng để tạo một điều khiển ListView

1.1.3.2. *Clear()*

- ❖ Cú pháp: `public void Clear()`
- ❖ Mô tả:
 - Xóa tất cả phần tử và cột khỏi điều khiển ListView

1.1.4. Một số sự kiện

1.1.4.1. *SelectedIndexChanged*

Xảy ra khi danh sách các chỉ số của các phần tử được chọn bị thay đổi hay nói cách khác khi mảng chỉ số chọn `SelectedIndices` thay đổi.

Lưu ý: Khi người dùng click chuột vào 1 phần tử mà không giữ phím ctrl để chọn nhiều, mảng chọn `SelectedItems` sẽ bị xóa rồi sau đó mới thêm phần tử được chọn vào mảng. Như vậy, sự kiện này sẽ được “fire” hai lần khi người dùng click chọn. Đây là lí do mà chúng ta cần thêm câu lệnh kiểm tra trong sự kiện:

```
if (lvwListView.SelectedItems.Counts > 0)
{
    // your code here...
}
```

Với câu lệnh if trên, chúng ta hiểu rằng code chỉ được thực hiện khi số lượng phần tử của mảng chọn lớn hơn không – nghĩa là chỉ thực hiện khi phần tử mới được thêm vào mảng.

1.1.4.2. *Click*

Xảy ra khi click vào điều khiển ListView.

1.1.4.3. *DoubleClick*

Xảy ra khi click đúp vào điều khiển ListView.

1.1.5. Lớp ListViewItem

Lớp ListViewItem là đối tượng thể hiện cho một phần tử trong ListView. Hay nói cách khác, mỗi phần tử của ListView có kiểu là ListViewItem.

Lớp ListViewItem thuộc namespace System.Windows.Forms.

1.1.5.1. Một số thuộc tính

❖ Checked

- Kiểu dữ liệu: bool
- Có giá trị true nếu phần tử được check vào checkbox, ngược lại có giá trị false (mặc định)

❖ Focused

- Kiểu dữ liệu: bool
- Có giá trị true nếu phần tử được trở đến, ngược lại có giá trị false (mặc định)

❖ ImageIndex

- Kiểu dữ liệu: int
- Chỉ số của hình ảnh trong ImageList dùng để hiển thị với phần tử. Giá trị mặc định là -1

❖ ImageList

- Kiểu dữ liệu: ImageList
- Danh sách các hình ảnh dùng để hiển thị với các phần tử

❖ Index

- Kiểu dữ liệu: int
- Chỉ số của phần tử trong danh sách các phần tử của điều khiển ListView

❖ SubItems

- Kiểu dữ liệu: ListViewItem.ListViewSubItemCollection
- Danh sách các phần tử con của phần tử trong ListView. Có thể thêm (Add), xóa (Remove) hay đếm số lượng phần tử con (Count)
- Tham khảo thêm tại:

<http://msdn.microsoft.com/en-us/library/system.windows.forms.listviewitem.listviewsubitemcollection.aspx>

❖ Text

- Kiểu dữ liệu: string
- Nội dung văn bản của phần tử

1.1.5.2. Một số phương thức

❖ Phương thức khởi tạo

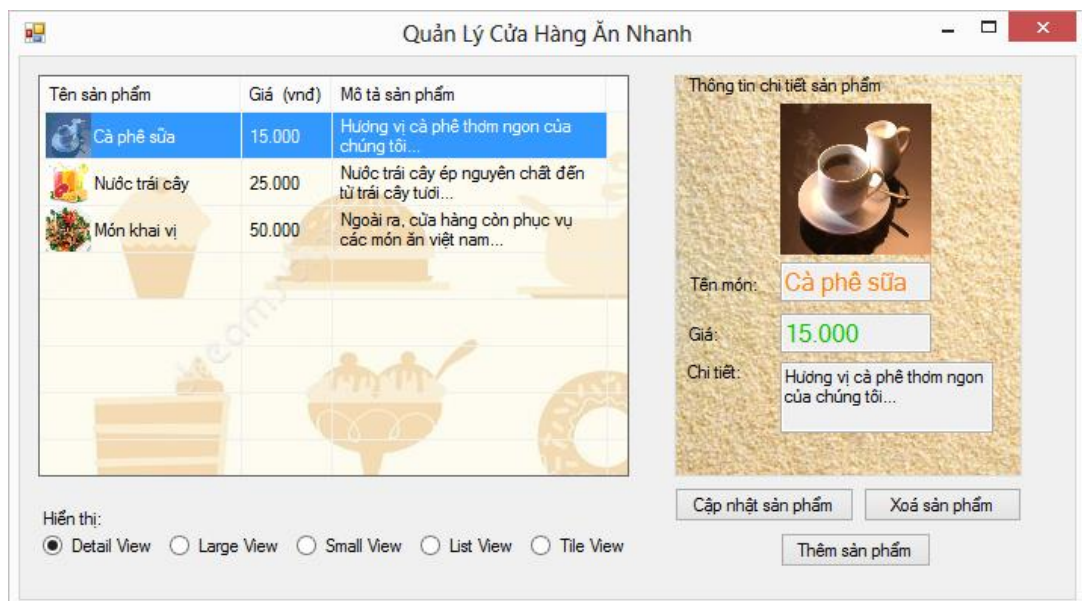
- `public ListViewItem()`
- `public ListViewItem(string text)`
- `public ListViewItem(string text, int imageIndex)`

❖ Tham khảo thêm tại:

<http://msdn.microsoft.com/en-us/library/system.windows.forms.listviewitem.aspx>

1.1.6. Ví dụ minh họa

Viết ứng dụng “*Quản Lý Cửa Hàng Ăn Nhanh*”, với giao diện như hình sau:



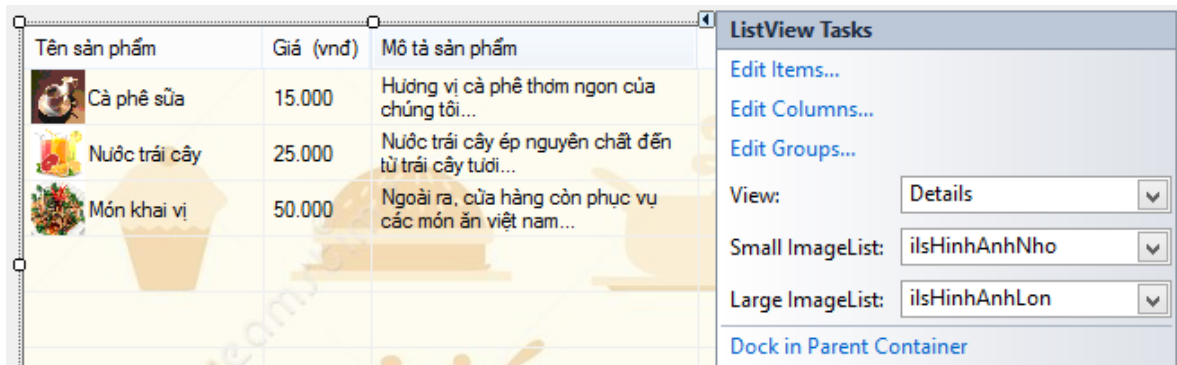
1.1.6.1. Tạo ListView (dùng Visual Studio thiết kế)

Đưa điều khiển ListView từ hộp công cụ (Toolbox) vào Form

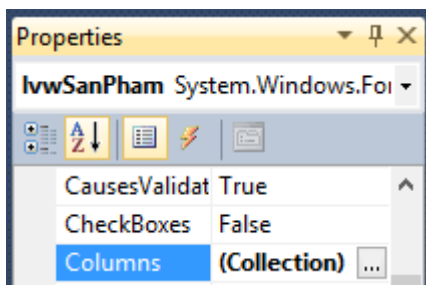
Đặt tên ListView: lvwDSSanPham

Thêm các cột vào ListView:

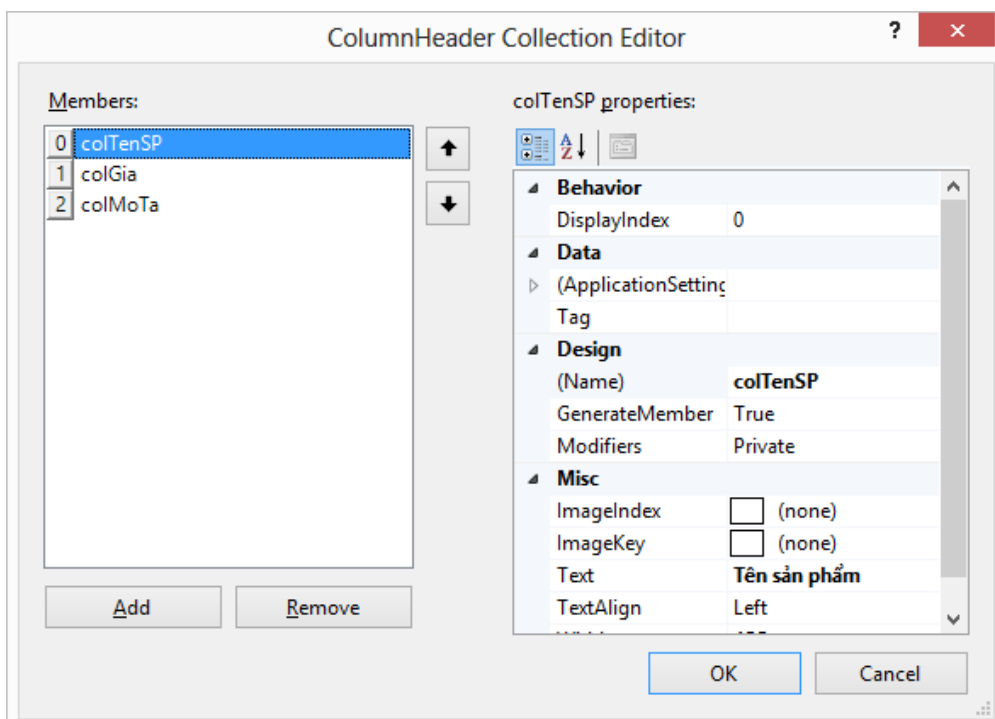
Nhấn chuột vào hình tam giác ở góc trên bên phải của điều khiển ListView, chọn Edit Columns...



Hoặc, nhấn chuột vào nút [...] bên cạnh thuộc tính Columns của ListView trong bảng Properties



Trong hộp thoại columnHeader Collection Editor, nhấn vào nút Add để thêm cột mới vào ListView

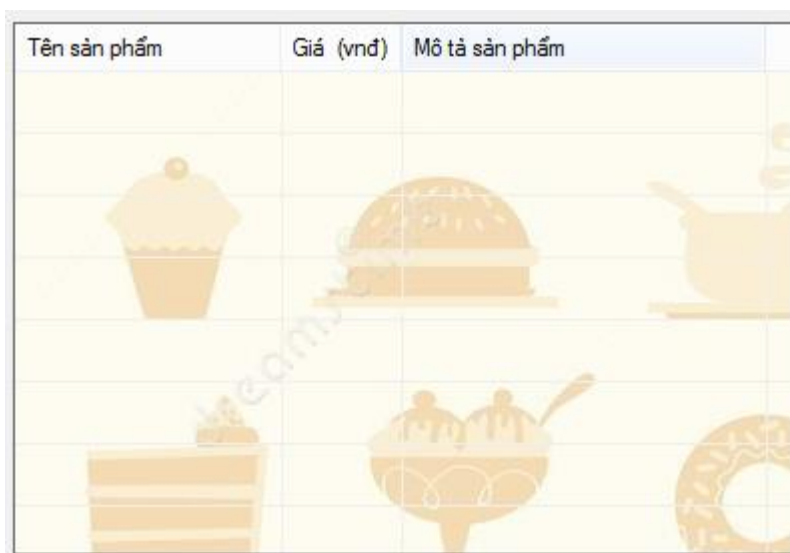


Một số thuộc tính trong ColumnHeader:

- DisplayIndex: thứ tự hiển thị của cột trên ListView
- Name: tên cột
- Text: tiêu đề cột
- TextAlign: canh lề nội dung trong cột
- Width: độ rộng cột

Khi thêm đủ các cột, nhấn nút OK để hoàn tất.

Kết quả:



1.1.6.2. Thêm cột vào ListView (Lập trình)

Kéo điều khiển ListView từ hộp công cụ (Toolbox) vào Form

Đặt tên ListView, ví dụ: lvwDSSanPham

Để tạo các cột cho ListView khi Form được load lên, chúng ta viết mã nguồn thêm các cột cho ListView trong hàm xử lý sự kiện FormLoad của form.

Mã nguồn tạo cột *Tên sản phẩm*:

```
// Tạo cột
ColumnHeader colTenSP = new ColumnHeader();
colTenSP.Text = "Tên sản phẩm";

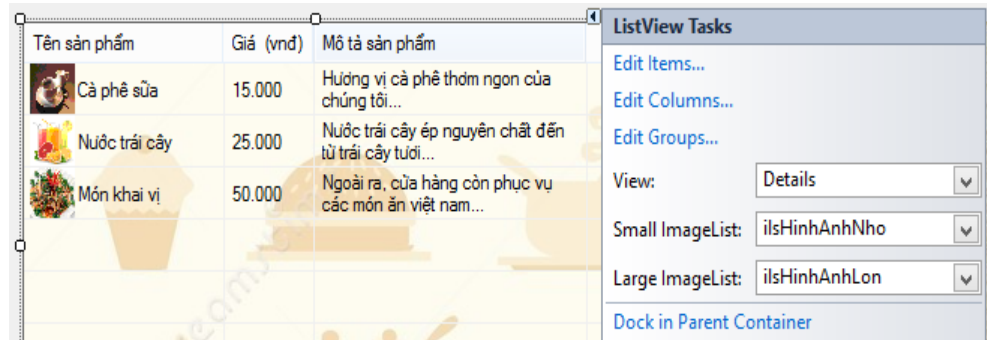
// Thêm cột vào listview
lvwSanPham.Columns.Add(colTenSP);
```

Tương tự như cột *Tên sản phẩm*, chúng ta sẽ tạo và thêm các cột còn lại vào danh sách các cột của ListView lvwDSSanPham.

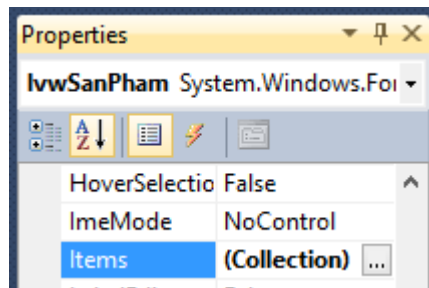
1.1.6.3. Thêm phần tử vào ListView

❖ Thêm phần tử khi design:

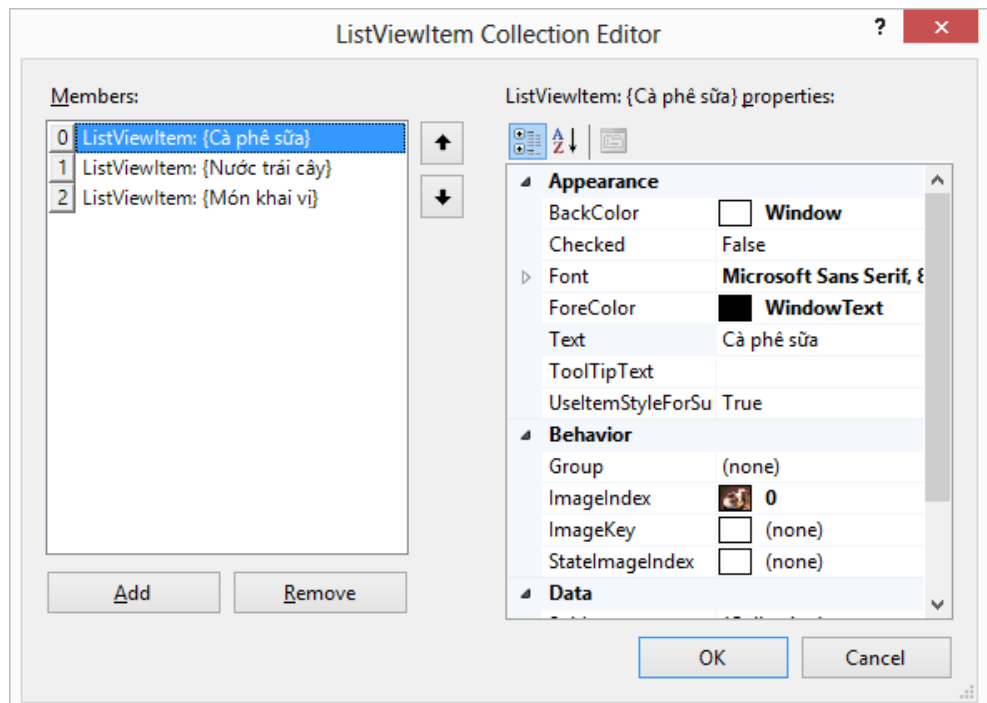
- Nhấn chuột vào hình tam giác ở góc trên bên phải của điều khiển ListView, chọn Edit Items...



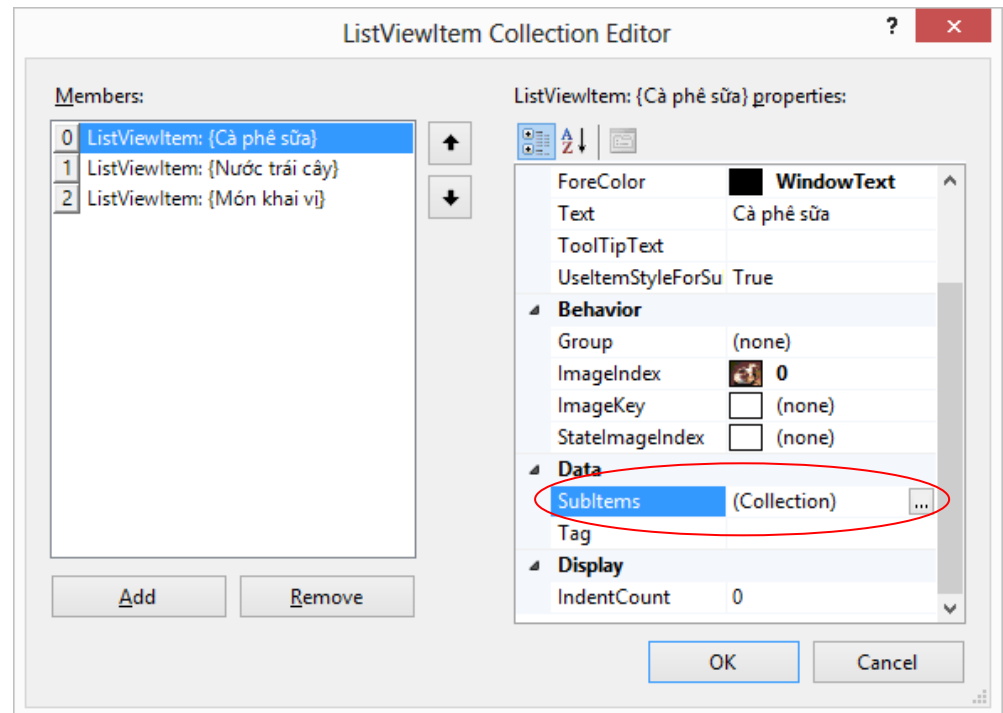
- Hoặc, nhấn chuột vào nút [...] bên cạnh thuộc tính Items của ListView trong bảng Properties



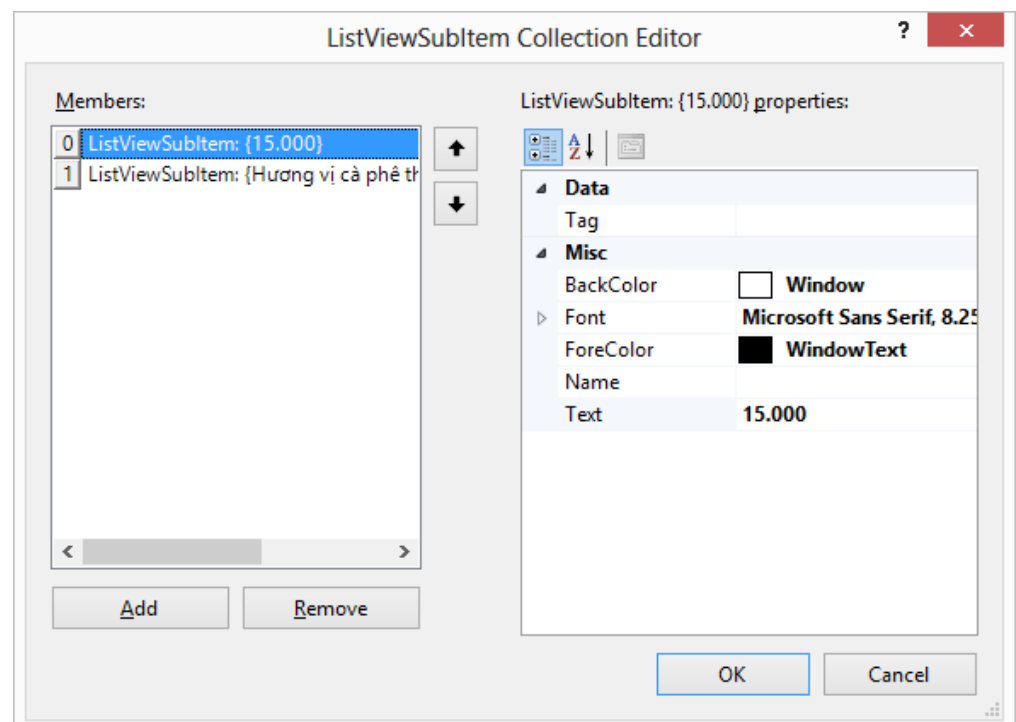
- Trong hộp thoại ListViewItem Collection Editor, nhấn vào nút Add để thêm phần tử vào ListView



Để thêm subitems cho phần tử, nhấn chuột vào nút [...] bên cạnh thuộc tính SubItems của ListViewItem trong bảng Properties



Trong hộp thoại ListViewSubItem Collection Editor, nhấn vào nút Add để thêm subitem cho phần tử



❖ Thêm phần tử khi lập trình:

○ Các bước cơ bản:

- 1. Tạo đối tượng ListViewItem
- 2. Tạo các subitem cho đối tượng ListViewItem (nếu có)

- 3. Thêm đối tượng ListViewItem vào ListView

- Mã nguồn ví dụ:

```
// 1. Tạo listviewitem với giá trị của cột đầu tiên
ListViewItem lvi = new ListViewItem(txtTenMon.Text);
// 2. Tạo các subitem cho listviewitem
// 2.1. Lấy giá sản phẩm từ txtGia
lvi.SubItems.Add(txtGia.Text);
// 2.2. Lấy chi tiết sản phẩm từ txtChiTiet
lvi.SubItems.Add(txtChiTiet.Text);
// 3. Thêm listviewitem vào ListView
lvwSanPham.Items.Add(lvi);
```

1.1.6.4. Cập nhật phần tử trong listview

Ví dụ, cập nhật thông tin từ các điều khiển trong groupbox Thông tin chi tiết cho phần tử được chọn trong listview

```
// Lấy phần tử được chọn trong ListView
ListViewItem lvi = lvwDSSanPham.SelectedItems[0];
// Cập nhật các thông tin
lvi.SubItems[0].Text = txtTenMon.Text;
lvi.SubItems[1].Text = txtGia.Text;
lvi.SubItems[2].Text = txtChiTiet.Text;
```

1.1.6.5. Xóa phần tử khỏi ListView

- ❖ Xóa một phần tử thứ 5 trong danh sách

```
// Lấy ra listviewitem thứ 5 trong danh sách
ListViewItem lvi = lvwSanPham.Items[4];
// Xóa listviewitem thứ 5 khỏi danh sách
lvwSanPham.Items.Remove(lvi);
```

hoặc:

```
// Xóa listviewitem thứ 5 khỏi danh sách
lvwSanPham.Items.RemoveAt(4);
```

- ❖ Xóa một phần tử được chọn trên ListView

```
// Lấy ra listviewitem được chọn trong danh sách
ListViewItem lvi = lvwSanPham.SelectedItems[0];
// Xóa listviewitem thứ 5 khỏi danh sách
lvwSanPham.Items.Remove(lvi);
```

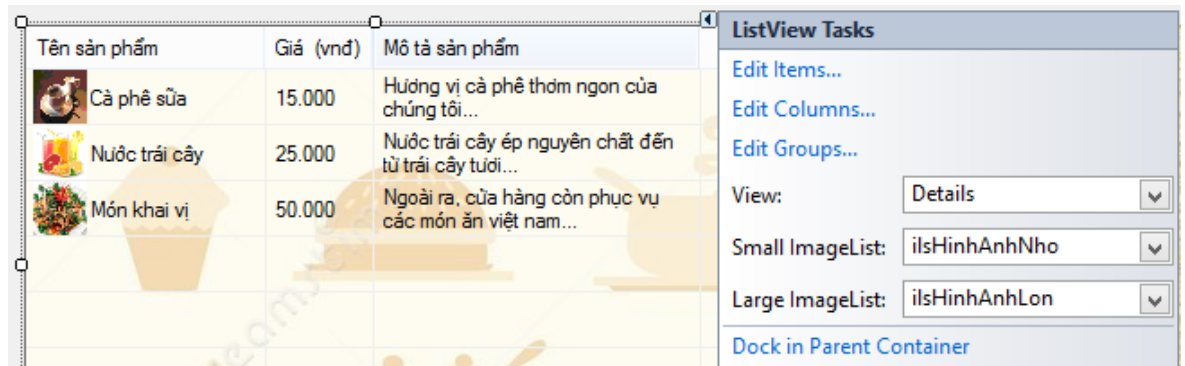
- ❖ Xóa nhiều phần tử được chọn trên ListView

```
// Lấy ra số lượng phần tử được chọn
int iSoLuong = lvwSanPham.SelectedItems.Count;
// Duyệt và xóa từng phần tử của danh sách được chọn
for(int i = 0; i < iSoLuong; i++)
{
    lvwSanPham.Items.Remove(lvwSanPham.SelectedItems[0]);
}
```


1.1.6.6. Thêm hình ảnh vào trước mỗi phần tử trong ListView

- ❖ Sử dụng điều khiển ImageList chứa danh sách các hình ảnh. Ví dụ trong ứng dụng có ImageList ilsHinhAnhLon (hiển thị hình ảnh cho các khung nhìn: LargeIcon và Tile) và ilsHinhAnhNho (hiển thị hình ảnh cho các khung nhìn: Details, SmallIcon và List).

Thiết lập ImageList khi design:



Hoặc code:

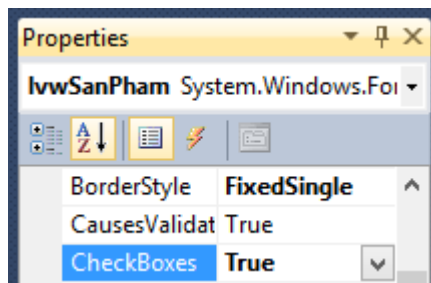
```
lvwDSSanPham.LargeImageList = ilsHinhAnhLon;  
lvwDSSanPham.SmallImageList = ilsHinhAnhNho;
```

- ❖ Ví dụ: thiết lập hình ảnh đầu tiên trong danh sách cho phần tử thứ 5 trong listview

```
lvwDSSanPham.Items[4].ImageIndex = 0;
```

1.1.6.7. Thêm checkbox vào trước mỗi phần tử trong ListView




Thiết lập ImageList khi design:



Hoặc code:




```
lvwDSSanPham.CheckBoxes = true;
```

Kết quả:

Tên sản phẩm	Giá (vnđ)	Mô tả sản phẩm
<input type="checkbox"/>  Cà phê sữa	15.000	Hương vị cà phê thơm ngon của chúng tôi...
<input type="checkbox"/>  Nước trái cây	25.000	Nước trái cây ép nguyên chất đến từ trái cây tươi...
<input type="checkbox"/>  Món khai vị	50.000	Ngoài ra, cửa hàng còn phục vụ các món ăn việt nam...

1.1.6.8. Thay đổi các khung nhìn trên ListView

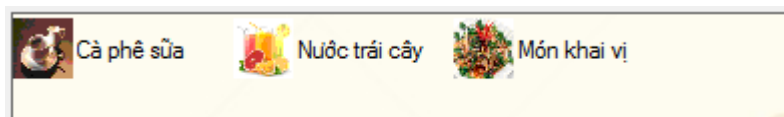
```
// Khung nhìn Details  
lvwSanPham.View = View.Details;
```

Tên sản phẩm	Giá (vnđ)	Mô tả sản phẩm
 Cà phê sữa	15.000	Hương vị cà phê thơm ngon của chúng tôi...
 Nước trái cây	25.000	Nước trái cây ép nguyên chất đến từ trái cây tươi...
 Món khai vị	50.000	Ngoài ra, cửa hàng còn phục vụ các món ăn việt nam...

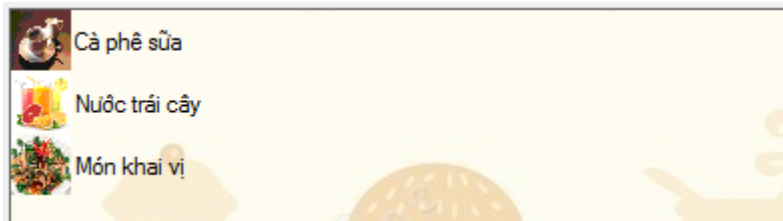
```
// Khung nhìn LargeIcon  
lvwSanPham.View = View.LargeIcon;
```



```
// Khung nhìn SmallIcon  
lvwSanPham.View = View.SmallIcon;
```



```
// Khung nhìn List
lvwSanPham.View = View.List;
```



```
// Khung nhìn Tile
lvwSanPham.View = View.Tile;
```



1.1.6.9. Truy xuất nội dung các ô của một phần tử trên ListView

Ví dụ: lấy thông tin chi tiết của một phần tử được chọn và hiển thị lên các điều khiển trong groupbox Thông tin chi tiết sản phẩm.

```
if (lvwSanPham.SelectedItems.Count > 0)
{
    // Lấy phần tử được chọn
    ListViewItem lvw = lvwSanPham.SelectedItems[0];

    // Lấy các thông tin chi tiết
    // -- Ảnh
    picChiTiet.Image =
        ilsHinhAnhLon.Images[lvw.ImageIndex];
    // -- Tên
    txtTenMon.Text = lvw.SubItems[0].Text;
    // -- Giá
    txtGia.Text = lvw.SubItems[1].Text;
    // -- Chi tiết
    txtChiTiet.Text = lvw.SubItems[2].Text;
}
```

1.1.6.10. Duyệt tất cả các phần tử của ListView

```
for (int i = 0; i < lvwSanPham.Items.Count; i++)  
{  
    // you code here...  
}
```

Hoặc:

```
foreach (ListViewItem lvi in lvwSanPham.Items)  
{  
    // you code here...  
}
```

Lưu ý: không dùng foreach để xóa phần tử trong ListView

1.2. TreeView

1.2.1. Khái niệm

TreeView là điều khiển cho phép trình bày danh sách phần tử ở dạng cây

Mỗi phần tử của cây được gọi là một node (là đối tượng thuộc lớp TreeNode)

Lớp TreeView thuộc namespace: System.Windows.Forms

1.2.2. Một số thuộc tính

1.2.2.1. CheckBoxes

- ❖ Kiểu dữ liệu: bool
- ❖ Mô tả:
 - Cho phép hiển thị checkbox bên trái mỗi node trong TreeView
 - Giá trị mặc định là false
 - Checkbox được dùng để chọn nhiều node trong TreeView cùng lúc

1.2.2.2. FullRowSelect

- ❖ Kiểu dữ liệu: bool
- ❖ Mô tả:
 - Cho phép tô vùng chọn của node được chọn theo chiều rộng của điều khiển TreeView thay vì theo độ dài nhãn của node
 - Giá trị mặc định là false
 - Thuộc tính FullRowSelect sẽ bị bỏ qua nếu thuộc tính ShowLines = true

1.2.2.3. ImageIndex

- ❖ Kiểu dữ liệu: int
- ❖ Mô tả:
 - Chỉ số của hình ảnh (trong ImageList) được dùng làm hình ảnh mặc định cho các node trong TreeView

1.2.2.4. ImageList

- ❖ Kiểu dữ liệu: ImageList
- ❖ Mô tả:
 - Chứa danh sách các hình ảnh được sử dụng cho các node trong TreeView
 - Giá trị mặc định là null

1.2.2.5. LabelEdit

- ❖ Kiểu dữ liệu: bool
- ❖ Mô tả:
 - Cho phép chỉnh sửa nội dung của nhãn của node trong TreeView khi thực thi (run time)
 - Giá trị mặc định là null

1.2.2.6. Nodes

- ❖ Kiểu dữ liệu: TreeNodeCollection
- ❖ Mô tả:
 - Danh sách các node trong TreeView
 - Mỗi node trong TreeView lại có một danh sách các node con (TreeNodeCollection)

1.2.2.7. Scrollable

- ❖ Kiểu dữ liệu: bool
- ❖ Mô tả:
 - Cho phép hiển thị thanh cuộn trên điều khiển TreeView
 - Giá trị mặc định là false

1.2.2.8. *SelectedNode*

- ❖ Kiểu dữ liệu: `TreeNode`
- ❖ Mô tả:
 - Node được chọn trong `TreeView`
 - Nếu không có node nào được chọn thì `SelectedNode = null`

1.2.2.9. *ShowLines*

- ❖ Kiểu dữ liệu: `bool`
- ❖ Mô tả:
 - Cho phép hiển thị đường kẻ giữa các node trong `TreeView`
 - Giá trị mặc định là `false`

1.2.2.10. *ShowPlusMinus*

- ❖ Kiểu dữ liệu: `bool`
- ❖ Mô tả:
 - Cho phép hiển thị nút có dấu cộng `[+]` và dấu trừ `[-]` dùng để hiện/ẩn các node con
 - Giá trị mặc định là `false`

1.2.2.11. *ShowRootLines*

- ❖ Kiểu dữ liệu: `bool`
- ❖ Mô tả:
 - Cho phép hiển thị đường nối giữa các node cha trong `TreeView`
 - Giá trị mặc định là `false`
 - Nếu `ShowRootLines = false` thì thuộc tính `ShowPlusMinus` sẽ không có tác dụng

1.2.3. Một số phương thức

1.2.3.1. *Phương thức khởi tạo*

- ❖ Cú pháp: `public TreeView()`
- ❖ Mô tả:
 - Dùng để tạo một điều khiển `TreeView`

1.2.3.2. CollapseAll()

- ❖ Cú pháp: `public void CollapseAll()`
- ❖ Mô tả:
 - Thu gọn tất cả node con trong TreeView

1.2.3.3. ExpandAll()

- ❖ Cú pháp: `public void ExpandAll()`
- ❖ Mô tả:
 - Hiển thị tất cả node con trong TreeView

1.2.3.4. GetNodeCount(bool includeSubTrees)

- ❖ Cú pháp: `public int GetNodeCount(bool includeSubTrees)`
- ❖ Mô tả:
 - Đếm số lượng node trong TreeView
 - Tham số: includeSubTrees (true: đếm cả các node của cây con, false: không đếm các node của cây con)

1.2.4. Một số sự kiện

1.2.4.1. AfterSelect

Xảy ra sau khi node trong TreeView được chọn

1.2.4.2. AfterCollapse

Xảy ra sau khi node trong TreeView được thu gọn

1.2.4.3. AfterExpand

Xảy ra sau khi node trong TreeView được mở rộng

1.2.4.4. Click

Xảy ra khi click vào điều khiển TreeView

1.2.5. Lớp TreeNode

TreeNode là lớp đối tượng thể hiện cho một node trong TreeView. Hay nói cách khác, mỗi node trong TreeView có kiểu dữ liệu là TreeNode.

Lớp TreeNode thuộc namespace System.Windows.Forms.

1.2.5.1. Một số thuộc tính

❖ Checked

- Kiểu dữ liệu: bool
- Có giá trị true nếu node được check vào checkbox, ngược lại có giá trị false (mặc định)

❖ FirstNode

- Kiểu dữ liệu: TreeNode
- Là node đầu tiên trong danh sách các node con của node hiện hành. Nếu node hiện hành không có node con nào thì FirstNode = null

❖ FullPath

- Kiểu dữ liệu: string
- Là chuỗi bao gồm các nhãn của các node dẫn đến node hiện hành bắt đầu từ node gốc của TreeView. Ngăn cách giữa các nhãn của mỗi node là dấu “\”. Ví dụ: “Country\VietNam\TPHoChiMinh”, trong đó node gốc của TreeView là Country, node hiện hành là TPHoChiMinh.

❖ ImageIndex

- Kiểu dữ liệu: int
- Chỉ số của hình ảnh (trong ImageList) của node khi chưa được chọn

❖ Index

- Kiểu dữ liệu: int
- Chỉ số của node trong cây con

❖ LastNode

- Kiểu dữ liệu: TreeNode
- Là node cuối cùng trong danh sách các node con của node hiện hành. Nếu node hiện hành không có node con nào thì LastNode = null

❖ Level

- Kiểu dữ liệu: int

- Độ sâu của node trong cây. Node gốc có Level = 0
- ❖ NextNode
 - Kiểu dữ liệu: TreeNode
 - Là node kế tiếp (cùng node cha) của node hiện hành. Nếu node hiện hành không có node kế tiếp nào thì NextNode = null
- ❖ Nodes
 - Kiểu dữ liệu: TreeNodeCollection
 - Danh sách các node con của node hiện hành
- ❖ Parent
 - Kiểu dữ liệu: TreeNode
 - Là node cha của node hiện hành. Nếu node hiện hành là node gốc thì Parent = null
- ❖ PrevNode
 - Kiểu dữ liệu: TreeNode
 - Là node kế trước (cùng node cha) của node hiện hành. Nếu node hiện hành không có node kế trước nào thì PrevNode = null
- ❖ SelectedImageIndex
 - Kiểu dữ liệu: int
 - Chỉ số của hình ảnh (trong ImageList) của node khi được chọn
- ❖ Text
 - Kiểu dữ liệu: string
 - Nội dung (văn bản) được hiển thị của node

1.2.5.2. Một số phương thức

- ❖ Phương thức khởi tạo
 - `public TreeNode()`
 - `public TreeNode(string text)`
 - `public TreeNode(string text, int imageIndex, int selectedImageIndex)`
- ❖ `public void Collapse()`

Thu gọn node hiện hành và các node con của node hiện hành

❖ `public void Collapse(bool ignoreChildren)`

Nếu `ignoreChildren = true` thì thu gọn node hiện hành, không thu gọn các node con của node hiện hành. Ngược lại giống với phương thức `Collapse()`

❖ `public void Expand()`

Mở rộng node hiện hành, không mở rộng các node con của node hiện hành

❖ `public void ExpandAll()`

Mở rộng node hiện hành và tất cả node con của node hiện hành

1.2.6. Ví dụ minh họa

Phát triển từ ứng dụng “*Quản Lý Cửa Hàng Ăn Nhanh*” với giao diện như sau:

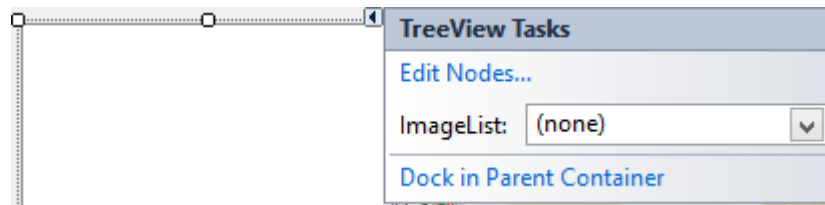


1.2.6.1. Tạo TreeView (design)

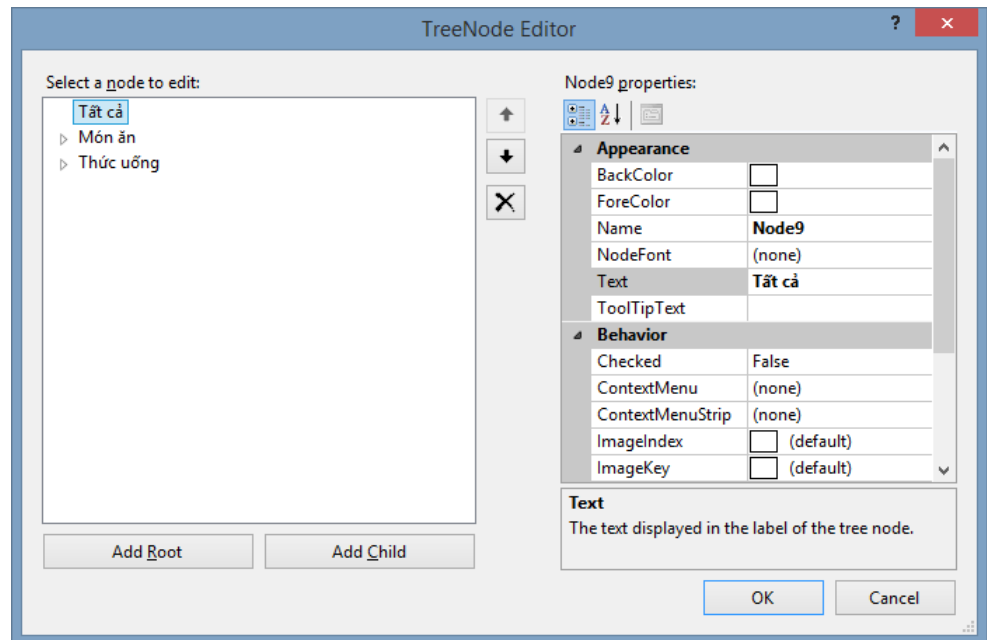
❖ Kéo điều khiển TreeView từ hộp công cụ (Toolbox) vào Form và đặt tên cho điều khiển TreeView. Ví dụ trong ứng dụng TreeView có tên là `treLoaiSanPham`.

❖ Thêm node vào điều khiển TreeView:

- Nhấn chuột vào hình tam giác góc trên bên phải của điều khiển TreeView, chọn `Edit Nodes...`

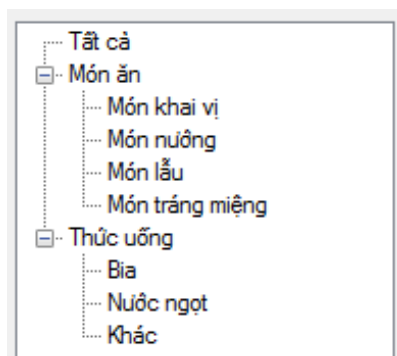


Trong hộp thoại TreeNode Editor, nhấn vào nút:



- Add Root: để thêm node gốc vào treeview
- Add Child: để thêm node con vào node được chọn trong treeview

❖ Kết quả:



1.2.6.2. Thêm Node vào TreeView (lập trình)

❖ Thêm node cha

```
// 1. Tạo đối tượng TreeNode với nhãn của node
TreeNode tn = new TreeNode("Tất cả");
// 2. Thêm đối tượng treenode vào treeview
treLoaiSanPham.Nodes.Add(tn);
```

Tương tự thêm các node “Thức ăn”, “Món uống” vào TreeView

❖ Thêm node con

- Thêm node “Món khai vị” vào node “Món ăn”

```
// 1. Lấy node Món ăn (index = 1) trong treeview
TreeNode tnRoot = treLoaiSanPham.Nodes[1];
// 2. Tạo node Món khai vị
TreeNode tnChild = new TreeNode("Món khai vị");
// 3. Thêm node Món khai vị vào node Món ăn
tnRoot.Nodes.Add(tnChild);
```

- Thêm node có nhãn được nhập từ TextBox txtLoaiSanPham vào node được chọn trong TreeView

```
// 1. Lấy node được chọn trong treeview
TreeNode tnRoot = treLoaiSanPham.SelectedNode;
// 2. Tạo node mới
TreeNode tnChild = new TreeNode(txtLoaiSanPham.Text);
// 3. Thêm node mới vào node được chọn
tnRoot.Nodes.Add(tnChild);
```

1.2.6.3. Truy xuất đến Node trong TreeView

❖ Lấy node được chọn

```
// Lấy node được chọn trong treeview
TreeNode tnSel = treLoaiSanPham.SelectedNode;
```

❖ Lấy node “Món ăn”

```
// Lấy node Món ăn (index = 1) trong treeview
TreeNode tnRoot = treLoaiSanPham.Nodes[1];
```

❖ Lấy node “Món lẩu”

```
// Lấy node Món ăn (index = 1) trong treeview
TreeNode tnRoot = treLoaiSanPham.Nodes[1];
// Lấy node Món lẩu (index = 2) trong node Món ăn
TreeNode tnChild = tnRoot.Nodes[2];
```

Hoặc

```
TreeNode tn = treLoaiSanPham.Nodes[1].Nodes[2];
```

1.2.6.4. Cập nhật Node

Ví dụ: cập nhật nhãn của node được chọn trong TreeView bằng nội dung được nhập từ TextBox txtLoaiSanPham

```
// 1. Lấy node được chọn trong treeview
TreeNode tnSel = treLoaiSanPham.SelectedNode;
// 2. Cập nhật nhãn
tnSel.Text = txtLoaiSanPham.Text;
```

1.2.6.5. Xóa Node khỏi Treeview

Ví dụ: xóa node được chọn trong TreeView

```
// 1. Lấy node được chọn trong treeview
TreeNode tnSel = treLoaiSanPham.SelectedNode;
// 2. Xóa node
treLoaiSanPham.Nodes.Remove(tnSel);
```

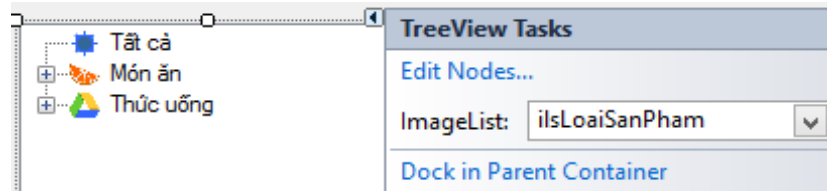
1.2.6.6. Thêm hình ảnh vào trước mỗi Node trong Treeview

Để có thể hiển thị hình ảnh trước mỗi node trong TreeView thì ta sẽ sử dụng điều khiển ImageList chứa danh sách các hình ảnh cho các node trong TreeView.

Ví dụ, trong ứng dụng sử dụng ImageList có tên ilsLoaiSanPham cho TreeView treLoaiSanPham.

❖ Gán ImageList cho TreeView

- Design:

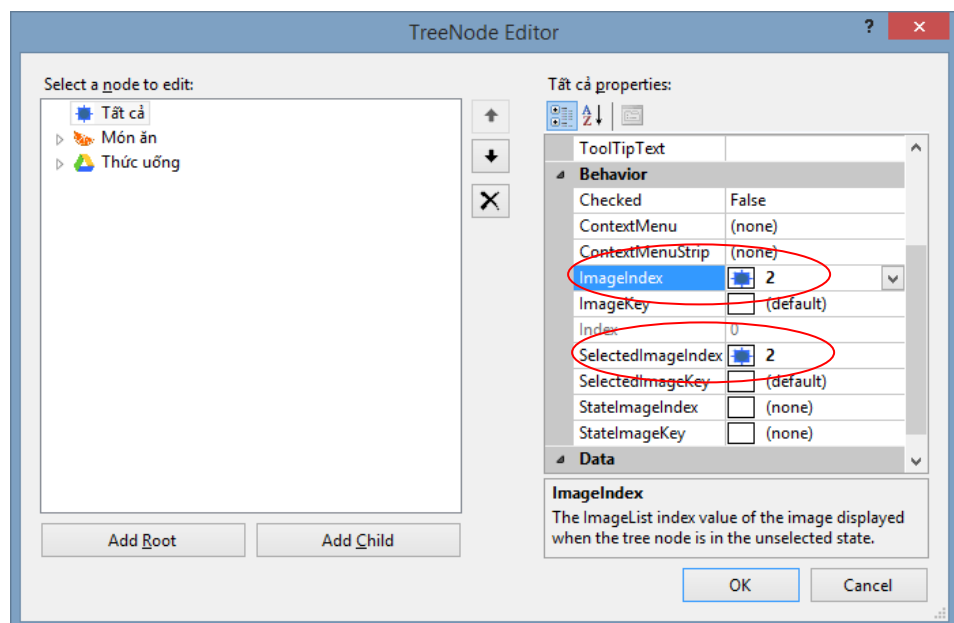


- Hoặc code:

```
treLoaiSanPham.ImageList = ilsLoaiSanPham;
```

❖ Gán hình ảnh cho các Node

- Design:



- Code:

Ví dụ: gán hình ảnh cho node “Tất cả”

```
// 1. Lấy node Tất cả (index = 0)
TreeNode tn = treLoaiSanPham.Nodes[0];
// 2. Gán hình ảnh có index = 2 trong imagelist cho node
// 2.1. Gán hình ảnh khi node không được chọn
tn.ImageIndex = 2;
// 2.2. Gán hình ảnh khi node được chọn
tn.SelectedImageIndex = 2;
```

1.2.6.7. Xử lý khi nhấn chuột vào Node trong TreeView

Lưu ý: để làm việc này, ta sẽ viết code xử lý sự kiện AfterSelect của TreeView

Ví dụ: Hiển thị danh sách các sản phẩm theo loại sản phẩm được chọn trong TreeView.

Ghi chú: Để biết sản phẩm loại nào, ta sử dụng thuộc tính Tag của ListViewItem và TreeNode để gán loại sản phẩm.

```
private void treLoaiSanPham_AfterSelect(object sender, TreeViewEventArgs e)
{
    if (treLoaiSanPham.SelectedNode.Tag != null)
    {
        // xoá hết sản phẩm ở lvwSanPham
        lvwSanPham.Items.Clear();

        if (treLoaiSanPham.SelectedNode.Tag.ToString() == "-1")
        {
            // thêm vào lvwSanPham tất cả các phần tử
            // của mảng lviAll
            foreach (ListViewItem lviItem in lviAll)
            {
                lvwSanPham.Items.Add((ListViewItem)lviItem.Clone());
            }
        }
        else
        {
            // tìm các sản phẩm có cùng loại (tag) với treeview
            // node (loại sản phẩm) và thêm vào lvwSanPham
            foreach (ListViewItem lviItem in lviAll)
            {
                if (lviItem.Tag.ToString() ==
                    treLoaiSanPham.SelectedNode.Tag.ToString())
                {
                    lvwSanPham.Items.Add((ListViewItem)
                                            lviItem.Clone());
                }
            }
        }
    }
}
```

CHƯƠNG 2: XÂY DỰNG ỨNG DỤNG MDI

2.1. Tổng quan ứng dụng MDI

MDI (Multiple Document Interface) là một giao diện lập trình cho phép người dùng làm việc với nhiều cửa sổ cùng lúc. Mỗi cửa sổ là một không gian riêng biệt, người dùng có thể làm việc với các cửa sổ khác nhau bằng việc di chuyển con trỏ từ cửa sổ này đến cửa sổ khác.

Ứng dụng MDI giống như màn hình Desktop của Windows. Tuy nhiên, các cửa sổ con của ứng dụng MDI không bao giờ xuất hiện bên ngoài cửa sổ ứng dụng.

Ứng dụng MDI có duy nhất một cửa sổ cha (MDI Parent) và có thể có nhiều cửa sổ con (MDI Child).

Một số ưu điểm của ứng dụng MDI:

- Các cửa sổ con được quản lý bởi một cửa sổ cha duy nhất
- Hệ thống Menu, Thanh công cụ (Toolbar) được sử dụng chung cho nhiều cửa sổ con
- Đóng cửa sổ cha thì các cửa sổ con cũng được đóng lại

2.2. Một số thuộc tính

Thuộc tính của MDI Parent

Thuộc tính	Kiểu dữ liệu	Mô tả
ActiveMdiChild	Form	trả về MDI Child Form đang được kích hoạt
IsMdiContainer	bool	xác định form là MDI Parent hoặc không
MdiChildren	Form[]	trả về mảng chứa các MDI Child Form

Thuộc tính của MDI Child

Thuộc tính	Kiểu dữ liệu	Mô tả
IsMdiChild	bool	trả về MDI Parent Form
MdiParent	Form	xác định form là MDI Child hoặc không

2.3. Một số phương thức

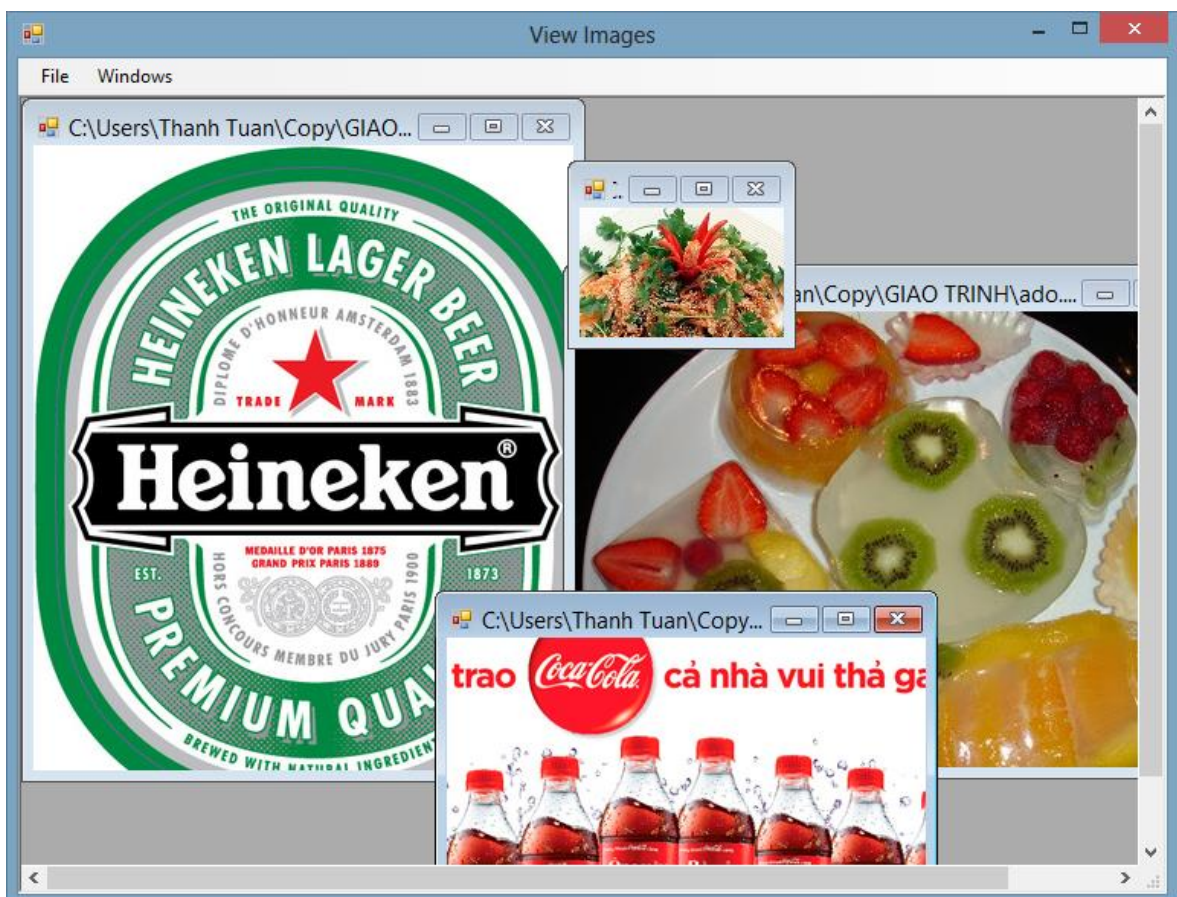
Phương thức	Mô tả
<code>void LayoutMdi(MdiLayout value)</code>	<p>xác định cách hiển thị các MDI Child Form.</p> <p>Một số giá trị của MdiLayout:</p> <ul style="list-style-type: none">• ArrangeIcons: sắp xếp các biểu tượng của các Child Form• Cascade: sắp xếp các cửa sổ Child Form theo dạng hình mái ngói• TileHorizontal: sắp xếp các cửa sổ Child Form theo chiều ngang• TileVertical: sắp xếp các cửa sổ Child Form theo chiều dọc

2.4. Một số sự kiện

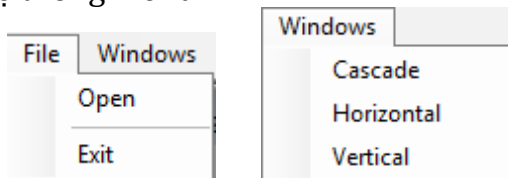
Sự kiện	Mô tả
MdiChildActive	xảy ra khi MDI Child Form được đóng lại hoặc được kích hoạt

2.5. Ví dụ minh họa

Xây dựng ứng dụng View Images với giao diện như hình bên dưới



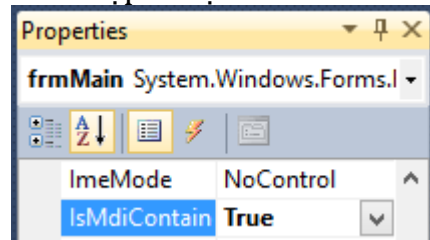
Hệ thống Menu:



Các bước thực hiện:

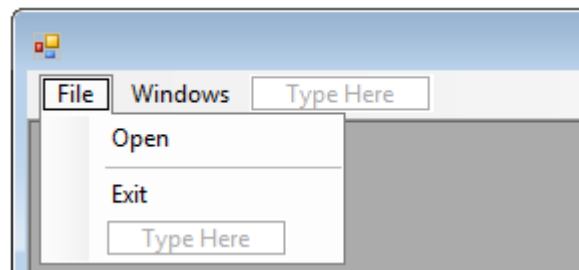
❖ Tạo MDI Parent Form frmMain

Thiết lập thuộc tính IsMdiContain của Form thành True



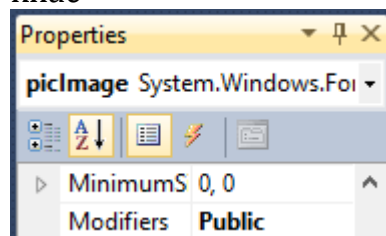
❖ Tạo hệ thống Menu

- Đưa điều khiển MenuStrip vào Form frmMain
- Thiết kế hệ thống menu như yêu cầu (Nhập menu vào các ô Type Here)

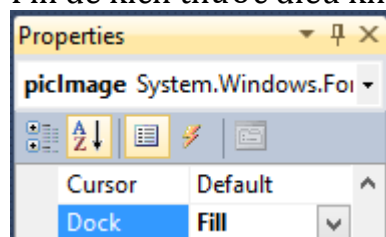


❖ Tạo Form frmViewImage

- Thêm Form mới vào project và đặt tên là frmViewImage
- Đưa điều khiển PictureBox vào Form frmViewImage và đặt tên là ptbImage
 - Thiết lập thuộc tính Modifiers của PictureBox picImage thành Public để có thể truy cập được đối tượng ở những lớp khác



- Thiết lập thuộc tính Dock của PictureBox ptbImage thành Fill để kích thước điều khiển vừa với Form



❖ Xử lý sự kiện click vào menu File/Open

```
// Khởi tạo Dialog để chọn hình
OpenFileDialog ofd = new OpenFileDialog();
// Lọc những loại tập tin hình ảnh
ofd.Filter = "JPG Files|.jpg|JPEG Files|.jpeg|PNG
              Files|.png|GIF Files|.gif|BMP Files|.bmp";
// Cho phép chọn nhiều tập tin trong Dialog
ofd.Multiselect = true;
if (ofd.ShowDialog() == System.Windows.Forms.DialogResult.OK)
{
    // Duyệt các tất cả các file được mở
    foreach (string fileName in ofd.FileNames)
    {
        // Thêm Child Form vào MDI Parent
        //- Khởi tạo Child Form
        frmViewImage frm = new frmViewImage();
        frm.Text = ofd.FileName;
        frm.picImage.ImageLocation = fileName;
        frm.Size = Image.FromFile(fileName).Size;
        //- Gán MDI Parent cho Child Form
        frm.MdiParent = this;
        //- Hiển thị Child Form
        frm.Show();
    }
}
```

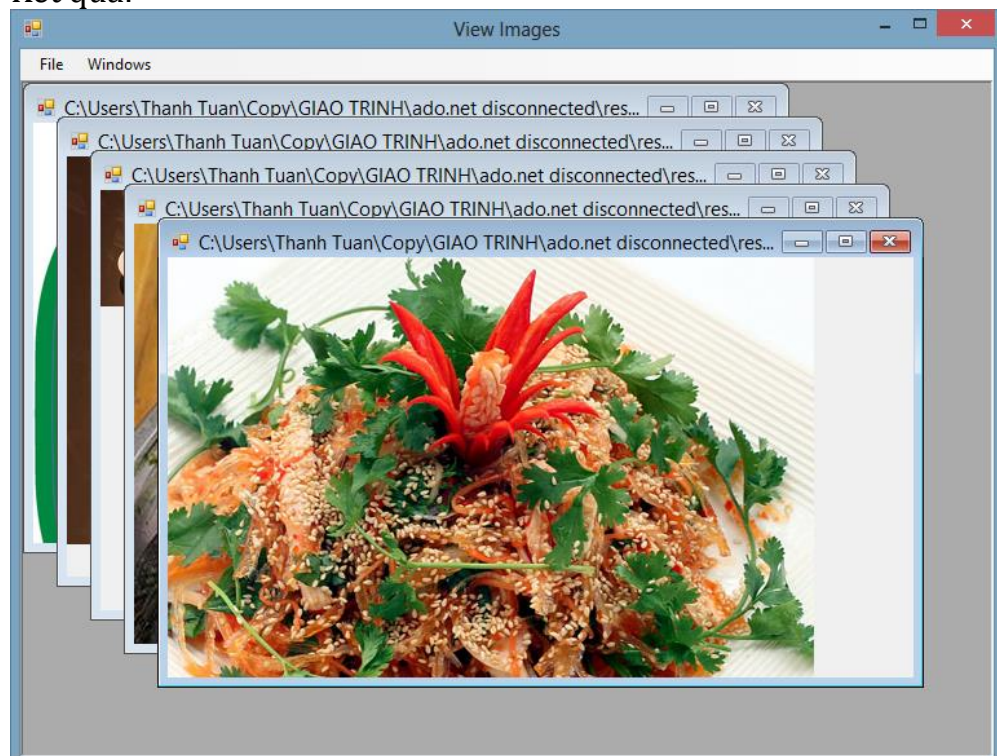
❖ Xử lý sự kiện click vào menu File/Exit

```
this.Close();
```

❖ Xử lý sự kiện click vào menu Windows/Cascade

```
this.LayoutMdi(MdiLayout.Cascade);
```

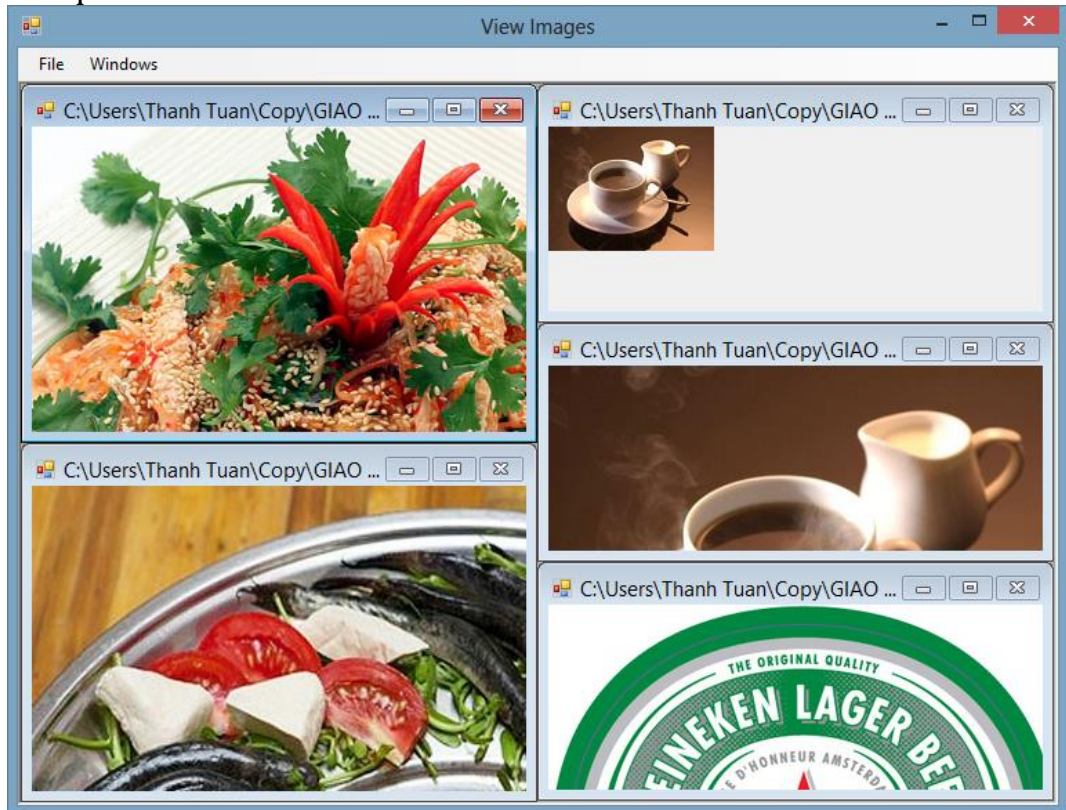
Kết quả:



- ❖ Xử lý sự kiện click vào menu Windows/Tile Horizontal

```
this.LayoutMdi(MdiLayout.TileHorizontal);
```

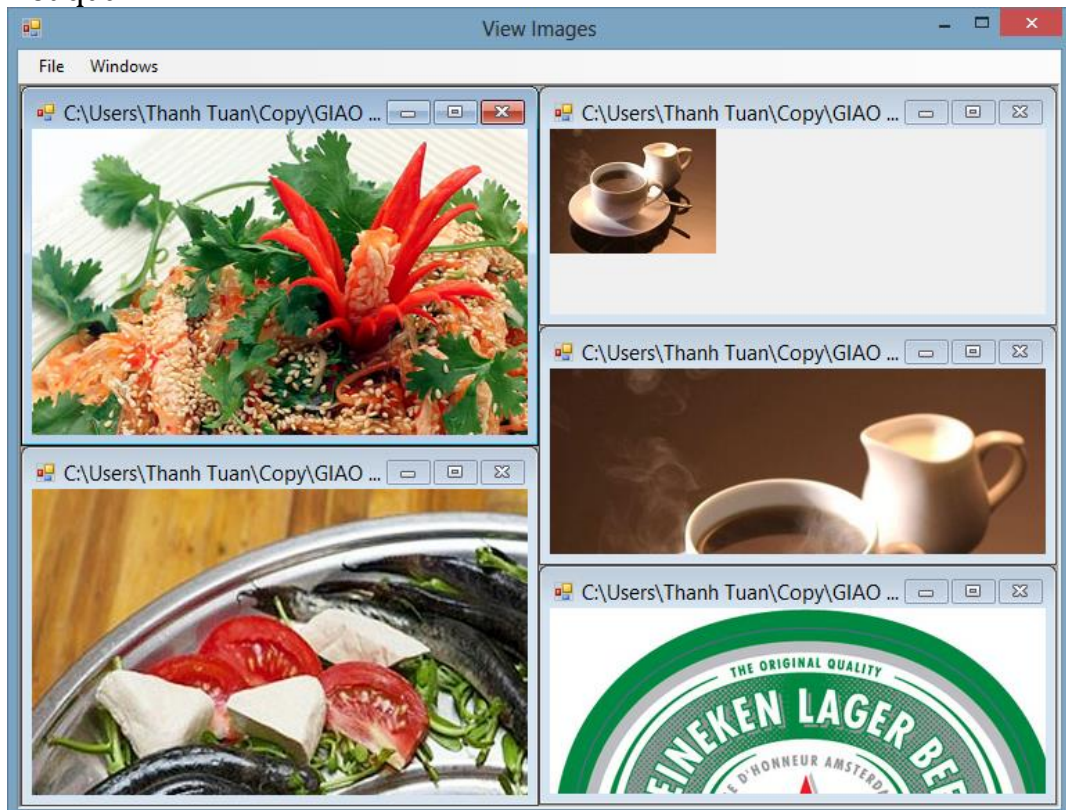
Kết quả:



- ❖ Xử lý sự kiện click vào menu Windows/Tile Vertical

```
this.LayoutMdi(MdiLayout.TileVertical);
```

Kết quả:



CHƯƠNG 3: ADO.NET

3.1. Tổng quan ADO.Net

3.1.1. Giới thiệu

ADO.Net là một thành phần trong dotNet Framework

ADO.Net tập hợp các lớp đối tượng cho phép thao tác với dữ liệu nguồn

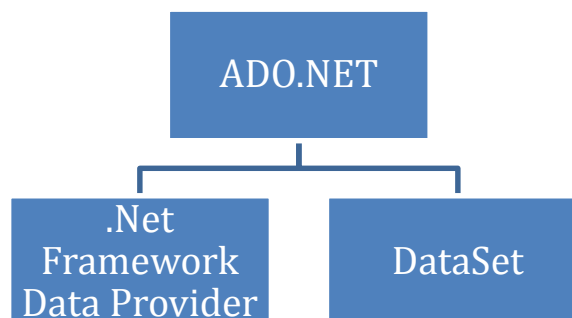
Dữ liệu nguồn có thể là một cơ sở dữ liệu lưu trữ trong các hệ quản trị cơ sở dữ liệu như MS Access, MS SQL Server, Oracle... hoặc tập tin XML, Excel...

ADO.Net hỗ trợ mô hình truy cập ngắt kết nối (disconnected model)

ADO.Net sử dụng XML để tương tác với cơ sở dữ liệu, nghĩa là dữ liệu trong cơ sở dữ liệu được chuyển sang định dạng XML để thực hiện các thao tác liên quan đến cơ sở dữ liệu như truy vấn, cập nhật...

3.1.2. Các thành phần của ADO.Net

ADO.Net có 2 thành phần chính .Net Framework Data Provider và DataSet



3.1.2.1. .Net Framework Data Provider

.Net Framework Data Provider được dùng để thực hiện việc kết nối và duy trì kết nối đến nguồn dữ liệu

Các dotNet Framework Data Provider

Loại CSDL	Tên Provider	Namespace
ODBC	.Net Framework Data Provider for ODBC	System.Data.Odbc
Access, Excel, MS SQL Server...	.Net Framework Data Provider for OLEDB	System.Data.OleDb
MS SQL Server	.Net Framework Data Provider	System.Data.SqlClient

	for SQL Server	
Oracle	.Net Framework Data Provider for Oracle	System.Data.OracleClient

Các đối tượng trong .Net Framework Data Provider gồm:

- ❖ Connection
- ❖ Command
- ❖ DataReader
- ❖ DataAdapter

3.1.2.2. DataSet

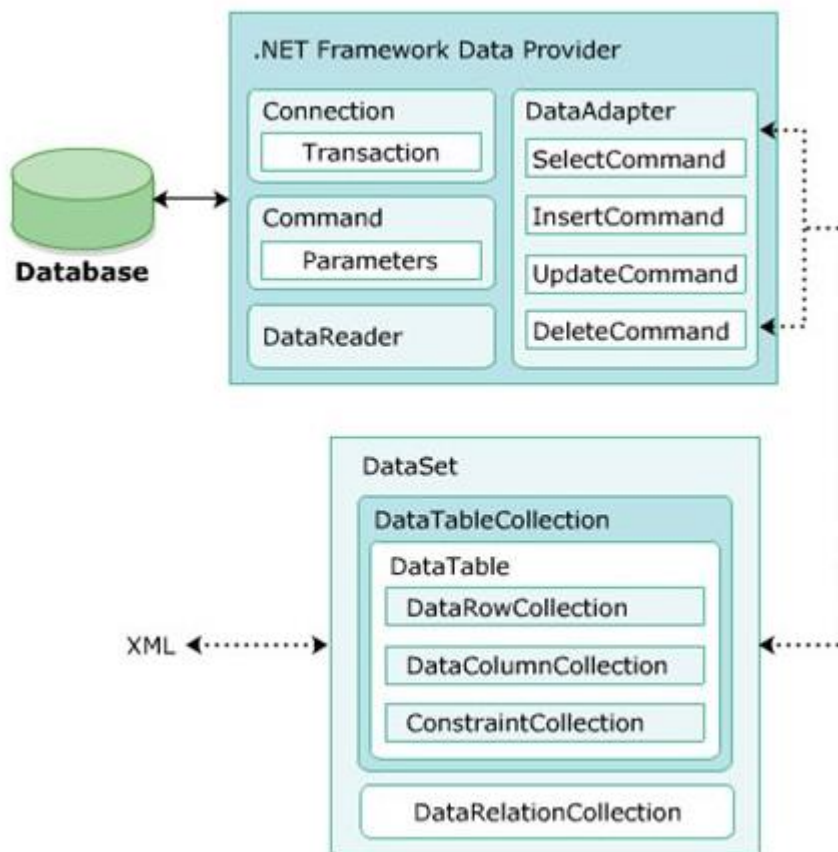
Dataset dùng để hỗ trợ mô hình truy cập ngắt kết nối.

DataSet cho phép chúng ta copy dữ liệu trong cơ sở dữ liệu ở máy database server (kết nối đến database server được mở), lưu vào DataSet ở máy client (máy cài đặt ứng dụng), sau khi copy xong, kết nối đến database server được đóng lại, khi đó ứng dụng có thể thao tác dữ liệu trên máy client, khi dữ liệu thao tác xong, thực hiện kết nối đến database server, cập nhật dữ liệu từ DataSet ngược lại databaser server.

Các thành phần trong DataSet:

- ❖ DataTableCollection: tập hợp các DataTable
- ❖ DataRelationCollection: tập hợp các mối quan hệ giữa các DataTable

3.1.2.3. Sơ đồ tổng quát các thành phần của ADO.Net



3.2. Làm việc với mô hình kết nối (connected)

3.2.1. Đối tượng Connection

Đối tượng Connection dùng để tạo một kết nối giữa ứng dụng với cơ sở dữ liệu.

Tùy thuộc vào .Net Framework Data Provider thì sẽ có những lớp đối tượng Connection tương ứng. Ví dụ như OleDbConnection, SqlConnection... Các lớp đối tượng Connection này có các thuộc tính và phương thức giống nhau, chỉ khác nhau về tên gọi.

3.2.1.1. Khai báo đối tượng Connection

Cách 1: `XYZConnection <tên_biến> = new XYZConnection();`

Cách 2: `XYZConnection <tên_biến> = new
XYZConnection(<chuỗi_kết_nối>);`

Trong đó:

- ❖ XYZ có thể là Sql, OleDb, Odbc, Oracle
- ❖ Với các loại cơ sở dữ liệu khác nhau sẽ có các <chuỗi_kết_nối> khác nhau

- MS Access 2003: **Provider=Microsoft.Jet.OLEDB.4.0;**
Data Source=<đường_dẫn_đến_tập_tin_CSDL>;
- MS SQL Server: **Server=myServerAddress;**
Database=myDataBase; User Id=myUsername;
Password=myPassword;

3.2.1.2. Một số thuộc tính

❖ **ConnectionString**

- Kiểu dữ liệu: string
- Chuỗi kết nối đến cơ sở dữ liệu

❖ **State**

- Kiểu dữ liệu: ConnectionState
- Trạng thái của đối tượng Connection (Broken, Closed, Connecting, Executing, Fetching, Open)

3.2.1.3. Một số phương thức

❖ **Open()**

- Cú pháp: **public void Open();**
- Mở kết nối đến cơ sở dữ liệu

❖ **Close()**

- Cú pháp: **public void Close();**
- Đóng kết nối cơ sở dữ liệu

3.2.2. Đối tượng Command

Đối tượng Command dùng để thực thi một câu lệnh SQL như SELECT, INSERT, UPDATE, DELETE, hoặc một Store Procedure.

Tùy thuộc vào .Net Framework Data Provider thì sẽ có những lớp đối tượng Command tương ứng.

3.2.2.1. Khai báo đối tượng Command

Cách 1: **XYZCommand <tên_biến> = new XYZCommand();**

Cách 2: **XYZCommand <tên_biến> = new XYZCommand (<câu_lệnh_SQL>,
<đối_tượng_Connection>);**

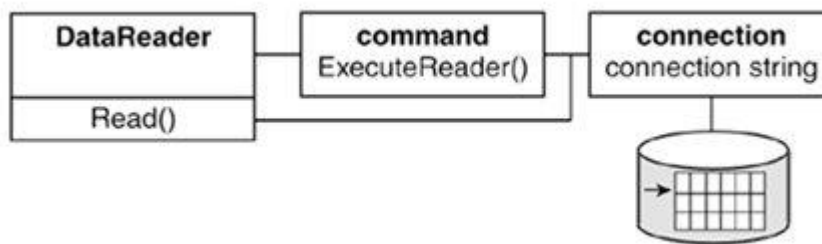
3.2.2.2. Một số thuộc tính

- ❖ CommandText
 - Kiểu dữ liệu: string
 - Câu lệnh SQL hoặc Store Procedure để thực thi trên CSDL
- ❖ Connection
 - Kiểu dữ liệu: XYZConnection
 - Đối tượng Connection được sử dụng bởi đối tượng Command
- ❖ CommandType
 - Kiểu dữ liệu: CommandType
 - Kiểu câu lệnh mà đối tượng Command sẽ thực thi (Text, StoreProcedure)
- ❖ Parameters
 - Kiểu dữ liệu: XYZParameterCollection
 - Danh sách các tham số trong câu lệnh SQL. Các tham số trong câu lệnh SQL có thể được thêm (Add), xóa (Remove), ...

3.2.2.3. Một số phương thức

- ❖ ExecuteNonQuery()
 - Cú pháp: `public int ExecuteNonQuery()`
 - Thực thi các câu lệnh SQL INSERT, UPDATE, DELETE. Kết quả trả về là số dòng được thực thi
- ❖ ExecuteScalar()
 - Cú pháp: `public object ExecuteScalar()`
 - Thực thi câu lệnh SQL SELECT. Kết quả trả về một giá trị duy nhất (giá trị của cột đầu tiên trong dòng đầu tiên của kết quả truy vấn) có kiểu object
- ❖ ExecuteReader()
 - Cú pháp: `public XYZDataReader ExecuteReader()`
 - Thực thi câu lệnh SQL SELECT. Kết quả trả về là đối tượng XYZDataReader chứa kết quả truy vấn SELECT

3.2.3. Đối tượng DataReader



DataReader dùng để đọc dữ liệu từ cơ sở dữ liệu.

DataReader đọc các dòng dữ liệu tuần tự từ đầu đến cuối (không theo chiều ngược lại).

DataReader chỉ đọc dữ liệu ra, không cập nhật ngược lại dữ liệu vào cơ sở dữ liệu.

Tùy thuộc vào .Net Framework Data Provider thì sẽ có những lớp đối tượng DataReader tương ứng.

3.2.3.1. Tạo đối tượng DataReader

```
XYZDataReader <tên_biến> = <đối_tượng_Command>.ExecuteReader();
```

3.2.3.2. Một số thuộc tính

- ❖ <đối_tượng_DataReader>[<chỉ số cột> | "tên_cột"]
 - Kiểu dữ liệu: object
 - Giá trị của ô có chỉ số cột hoặc tên cột tương ứng
- ❖ FieldCount
 - Kiểu dữ liệu: int
 - Số cột trong dòng mà đối tượng DataReader đang đọc
- ❖ HasRow
 - Kiểu dữ liệu: bool
 - Có giá trị true nếu đối tượng DataReader có chứa ít nhất một dòng dữ liệu, ngược lại có giá trị false

❖ IsClosed

- Kiểu dữ liệu: bool
- Có giá trị true nếu đối tượng DataReader đã bị đóng lại, ngược lại có giá trị false

3.2.3.3. Một số phương thức

❖ Close()

- Cú pháp: `public void Close()`
- Đóng đối tượng DataReader

❖ Một số phương thức lấy các giá trị với các kiểu dữ liệu tương ứng với tham số được truyền vào là chỉ số của cột

- `public bool GetBoolean(int index)`
- `public byte GetByte(int index)`
- `public char GetChar(int index)`
- `public DateTime GetDateTime(int index)`
- `public decimal GetDecimal(int index)`
- `public double GetDouble(int index)`
- `public float GetFloat(int index)`
- `public short GetInt16(int index)`
- `public int GetInt32(int index)`
- `public long GetInt64(int index)`
- `public string GetString(int index)`
- `public object GetValue(int index)`

❖ IsDBNull(int index)

- Cú pháp: `public bool IsDBNull(int index)`
- Tham số truyền vào là chỉ số cột
- Có giá trị true nếu giá trị của cột trong dòng hiện hành là DBNull, ngược lại có giá trị false

❖ Read()

- Cú pháp: `public bool Read()`
- Có giá trị true nếu đối tượng DataReader đọc được một dòng dữ liệu, ngược lại có giá trị false

3.2.4. Các bước lập trình ứng dụng thao tác với CSDL

- ❖ Bước 1: Tạo và mở kết nối với CSDL
- ❖ Bước 2: Tạo đối tượng truy vấn, thao tác với CSDL
- ❖ Bước 3: Try vấn, thao tác CSDL và xử lý kết quả trả về nếu có
- ❖ Bước 4: Đóng kết nối CSDL

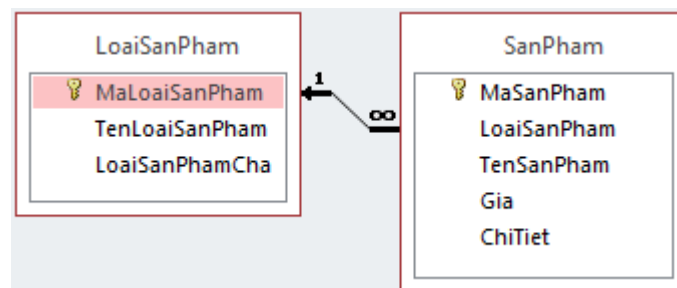
3.2.5. Ví dụ minh họa

Phát triển ứng dụng “*Quản Lý Cửa Hàng Ăn Nhanh*” với danh sách các sản phẩm và loại sản phẩm được lưu ở trong tập tin MS Access CuaHang.mdb

- ❖ Giao diện ứng dụng:

Tên sản phẩm	Giá (vnd)	Mô tả sản phẩm
Cà phê sữa	15000	Hương vị cà phê thơm ngon của ...
Nước trái cây	25000	Nước trái cây ép nguyên chất để...
Heineken	14000	Bia Heineken xuất xứ từ Hà Lan...
Lẩu cá kèo	100000	Lẩu cá kèo của cửa hàng với ngu...
Bách tuộc nướng	45000	Món bạch tuộc nướng ướp gia vị ...
Coca cola	8000	Nước uống giải khát coca cola...
Thạch trái cây	25000	Món thạch trái cây thơm ngon bổ ...

- ❖ Mô hình quan hệ CSDL:



❖ Mô tả chi tiết các bảng:

LOAISANPHAM		
Tên cột	Kiểu dữ liệu	Mô tả
<u>MaLoaiSanPham</u>	Chuỗi	Mã loại sản phẩm
TenLoaiSanPham	Chuỗi	Tên loại sản phẩm
LoaiSanPhamCha	Chuỗi	Loại sản phẩm cha Tất cả = tc, Món ăn = ma, Thức uống = tu

SANPHAM		
Tên cột	Kiểu dữ liệu	Mô tả
<u>MaSanPham</u>	Số tự động tăng	Mã sản phẩm
LoaiSanPham	Chuỗi	Mã loại sản phẩm
TenSanPham	Chuỗi	Tên sản phẩm
Giá	Số nguyên	Giá sản phẩm
ChiTiet	Chuỗi	Mô tả sản phẩm

❖ Các bước thực hiện:

- Thiết kế giao diện:
 - TreeView treLoaiSanPham có 3 node: Tất cả (Tag = tc), Món ăn (Tag = ma), Thức uống (Tag = tu).
 - ListView lvwDDSanPham có cột thứ tư là Mã sản phẩm có width = 0.
- Tạo lớp đối tượng kết nối đến CSDL CuaHang.mdb

```
public class DataProvider
{
    public static OleDbConnection TaoKetNoi(string
                                         strChuoiKetNoi)
    {
        OleDbConnection conn = new
                                OleDbConnection(strChuoiKetNoi);
        conn.Open();
        return conn;
    }
}
```

- Tạo các lớp tương ứng với mỗi bảng trong CSDL CuaHang.mdb
 - Mã nguồn lớp LoaiSanPham

```

public class LoaiSanPham
{
    // Các thuộc tính
    public string MaLoaiSanPham { set; get; }
    public string TenLoaiSanPham { set; get; }
    public string LoaiSanPhamCha { set; get; }

    // Phương thức khởi tạo
    public LoaiSanPham()
    {
        MaLoaiSanPham = "";
        TenLoaiSanPham = "";
        LoaiSanPhamCha = "";
    }

    // Các phương thức thao tác với CSDL
    // --Lấy danh sách loại sản phẩm
    public List<LoaiSanPham> LayDanhSach()
    {
        List<LoaiSanPham> lstLoaiSanPham = new
                                                List<LoaiSanPham>();

        // B.1: Tạo kết nối CSDL
        OleDbConnection conn =
                                DataProvider.TaoKetNoi();

        // B.2: Tạo đối tượng try vấn CSDL
        OleDbCommand com = new OleDbCommand();
        com.CommandText = "SELECT * FROM
                                LOAISANPHAM";

        com.Connection = conn;
        // Hoặc:
        // OleDbCommand com = new
        //                                OleDbCommand("SELECT * FROM
        //                                LOAISANPHAM", conn);

        // B.3: Truy vấn CSDL và xử lý kết quả trả về
        // -----Truy vấn
        OleDbDataReader dr = com.ExecuteReader();
        // -----Xử lý kết quả
        while (dr.Read())
        {
            LoaiSanPham loai = new LoaiSanPham();
            // Mã loại sản phẩm
            if (!dr.IsDBNull(0))
                loai.MaLoaiSanPham = (string)dr[0];
            // Tên loại sản phẩm
            if (!dr.IsDBNull(1))
                loai.TenLoaiSanPham = (string)dr[1];
            // Loại sản phẩm cha
            if (!dr.IsDBNull(2))
                loai.LoaiSanPhamCha = (string)dr[2];
            // Thêm loại sản phẩm vào danh sách loại

```

```

        // sản phẩm
        lstLoaiSanPham.Add(loai);
    }
    dr.Close();
    // B.4: Đóng kết nối CSDL
    conn.Close();
    return lstLoaiSanPham;
}
}

```

- Mã nguồn lớp SanPham

```

public class SanPham
{
    // Các thuộc tính
    public int MaSanPham { set; get; }
    public string LoaiSanPham { set; get; }
    public string TenSanPham { set; get; }
    public int Gia { set; get; }
    public string ChiTiet { set; get; }

    // Phương thức khởi tạo
    public SanPham()
    {
        MaSanPham = 0;
        LoaiSanPham = "";
        TenSanPham = "";
        Gia = 0;
        ChiTiet = "";
    }

    // Các phương thức thao tác với CSDL
    // --Lấy danh sách sản phẩm
    // ----Nếu mã loại strMaLoai = "" ==> lấy tất cả
    // danh sách sản phẩm trong CSDL
    // ----Ngược lại lấy danh sách theo loại
    public List<SanPham> LayDanhSach(string strMaLoai
                                     = "")
    {
        List<SanPham> lstSanPham = new
                                     List<SanPham>();

        // B.1: Tạo kết nối CSDL
        OleDbConnection conn =
            DataProvider.TaoKetNoi();
        // B.2: Tạo đối tượng try vấn CSDL
        OleDbCommand com = new OleDbCommand();
        com.CommandText = "SELECT * FROM SANPHAM";
        // --Nếu có mã loại thì thêm vào điều kiện
        if (strMaLoai != "")
            com.CommandText += string.Format(" WHERE
                                             LOAISANPHAM = '{0}'", strMaLoai);
        com.Connection = conn;

        // B.3: Truy vấn CSDL và xử lý kết quả trả về
        // -----Truy vấn
    }
}

```

```

OleDbDataReader dr = com.ExecuteReader();
// -----Xử lý kết quả
while (dr.Read())
{
    SanPham sp = new SanPham();
    // Mã sản phẩm
    if (!dr.IsDBNull(0))
        sp.MaSanPham = (int)dr[0];
    // Loại sản phẩm
    if (!dr.IsDBNull(1))
        sp.LoaiSanPham = (string)dr[1];
    // Tên sản phẩm
    if (!dr.IsDBNull(2))
        sp.TenSanPham = (string)dr[2];
    // Giá
    if (!dr.IsDBNull(3))
        sp.Gia = (int)dr[3];
    // Chi tiết
    if (!dr.IsDBNull(4))
        sp.ChiTiet = (string)dr[4];
    // Thêm sản phẩm vào danh sách sản phẩm
    lstSanPham.Add(sp);
}
dr.Close();
// B.4: Đóng kết nối CSDL
conn.Close();
return lstSanPham;
}

// --Thêm sách sản phẩm
public void ThemSanPham()
{
    // B.1: Tạo kết nối CSDL
    OleDbConnection conn =
        DataProvider.TaoKetNoi();
    // B.2: Tạo đối tượng try vắn CSDL với
    // parameter (?)
    OleDbCommand com = new OleDbCommand();
    com.CommandText = "INSERT INTO
        SANPHAM(LoaiSanPham, TenSanPham,
        Gia, ChiTiet) VALUES(?,?,?,?)";
    com.Connection = conn;
    //----Khai báo các parameter theo đúng thứ tự
    com.Parameters.AddWithValue("@loaisp",
        this.LoaiSanPham);
    com.Parameters.AddWithValue("@tensp",
        this.TenSanPham);
    com.Parameters.AddWithValue("@gia",
        this.Gia);
    com.Parameters.AddWithValue("@chitiet",
        this.ChiTiet);
    // B.3: Thực thi thao tác với CSDL
    com.ExecuteNonQuery();
}

```

```

        // B.4: Đóng kết nối CSDL
        conn.Close();
    }

    // --Cập nhật sách sản phẩm
    public void CapNhatSanPham()
    {
        // B.1: Tạo kết nối CSDL
        OleDbConnection conn =
            DataProvider.TaoKetNoi();
        // B.2: Tạo đối tượng try vắn CSDL với
        // parameter (?)
        OleDbCommand com = new OleDbCommand();
        com.CommandText = "UPDATE SANPHAM SET
            LoaiSanPham = ?, TenSanPham = ?,
            Gia = ?, ChiTiet = ? WHERE
            MASANPHAM = ?";

        com.Connection = conn;
        //----Khai báo các parameter theo đúng thứ tự
        com.Parameters.AddWithValue("@loaisp",
            this.LoaiSanPham);
        com.Parameters.AddWithValue("@tensp",
            this.TenSanPham);
        com.Parameters.AddWithValue("@gia",
            this.Gia);
        com.Parameters.AddWithValue("@chitiet",
            this.ChiTiet);
        com.Parameters.AddWithValue("@masp",
            this.MaSanPham);
        // B.3: Thực thi thao tác với CSDL
        com.ExecuteNonQuery();
        // B.4: Đóng kết nối CSDL
        conn.Close();
    }

    // --Xóa sách sản phẩm
    public void XoaSanPham()
    {
        // B.1: Tạo kết nối CSDL
        OleDbConnection conn =
            DataProvider.TaoKetNoi();
        // B.2: Tạo đối tượng try vắn CSDL
        OleDbCommand com = new OleDbCommand();
        com.CommandText = string.Format("DELETE FROM
            SANPHAM WHERE MASANPHAM = {0}",
            this.MaSanPham);

        com.Connection = conn;
        // B.3: Thực thi thao tác với CSDL
        com.ExecuteNonQuery();
        // B.4: Đóng kết nối CSDL
        conn.Close();
    }
}

```


- Xử lý sự kiện Load của Form:
 - Đưa danh sách đối tượng Loại sản phẩm vào TreeView treLoaiSanPham và Combobox cboLoaiSanPham

```
// Load danh sách loại sản phẩm
LoaiSanPham lsp = new LoaiSanPham();
List<LoaiSanPham> lstLoaiSanPham =
    lsp.LayDanhSach();
// Đưa danh sách lên TreeView treLoaiSanPham
foreach (LoaiSanPham loai in lstLoaiSanPham)
{
    // Tạo Node
    TreeNode node = new
        TreeNode(loai.TenLoaiSanPham);
    node.Tag = loai.MaLoaiSanPham;

    // Tìm và thêm vào node có mã loại sản phẩm
    // (Tag) giống mã loại sản phẩm cha của đối
    // tượng
    foreach (TreeNode tn in treLoaiSanPham.Nodes)
    {
        if (tn.Tag.ToString() ==
            loai.LoaiSanPhamCha)
        {
            tn.Nodes.Add(node);
            break;
        }
    }
}
// Đưa danh sách lên Combobox cboLoaiSanPham
cboLoaiSanPham.DataSource = lstLoaiSanPham;
cboLoaiSanPham.DisplayMember = "TenLoaiSanPham";
cboLoaiSanPham.ValueMember = "MaLoaiSanPham";
```

- Đưa danh sách tất cả sản phẩm trong CSDL vào ListView lvwSanPham

```
// Lấy danh sách tất cả sản phẩm trong CSDL
SanPham sp = new SanPham();
List<SanPham> lstSanPham = sp.LayDanhSach();
// Đưa danh sách sản phẩm lên ListView lvwSanPham
foreach (SanPham s in lstSanPham)
{
    // Tạo đối tượng ListViewItem
    ListViewItem lvi = new
        ListViewItem(s.TenSanPham);
    // Thêm các subitem
    lvi.SubItems.Add(s.Gia.ToString());
    lvi.SubItems.Add(s.ChiTiet);
    lvi.SubItems.Add(s.MaSanPham.ToString());
    // Gán Loại Sản phẩm vào Tag của ListViewItem
    lvi.Tag = s.LoaiSanPham;
```

```

// Thêm đối tượng ListViewItem vào ListView
lvwSanPham.Items.Add(lvi);
}

```

- Xử lý sự kiện AfterSelect của TreeView treLoaiSanPham khi chọn nhấn chọn Loại sản phẩm để hiển thị danh sách Sản phẩm trên ListView lvwDSSanPham tương ứng với loại được chọn

```

if (treLoaiSanPham.SelectedNode.Tag != null)
{
    List<SanPham> lstSanPham = new List<SanPham>();
    SanPham sp = new SanPham();
    // Xoá hết sản phẩm ở lvwSanPham
    lvwSanPham.Items.Clear();

    if (treLoaiSanPham.SelectedNode.Tag.ToString() == "tc")
    {
        // Lấy danh sách tất cả sản phẩm
        lstSanPham = sp.LayDanhSach();
    }
    else
    {
        // Lấy danh sách sản phẩm theo mã loại sản phẩm (Tag)
        lstSanPham = sp.LayDanhSach(treLoaiSanPham.SelectedNode.Tag.ToString());
    }

    // Đưa danh sách sản phẩm lên ListView lvwSanPham
    foreach (SanPham s in lstSanPham)
    {
        // Tạo đối tượng ListViewItem
        ListViewItem lvi = new ListViewItem(s.TenSanPham);
        // Thêm các subitem
        lvi.SubItems.Add(s.Gia.ToString());
        lvi.SubItems.Add(s.ChiTiet);
        lvi.SubItems.Add(s.MaSanPham.ToString());
        // Gán Loại Sản phẩm vào Tag của ListViewItem
        lvi.Tag = s.LoaiSanPham;
        // Thêm đối tượng ListViewItem vào ListView
        lvwSanPham.Items.Add(lvi);
    }
}

```

- Xử lý thêm sản phẩm mới

```
// Tạo đối tượng sản phẩm
SanPham sp = new SanPham();
sp.LoaiSanPham = cboLoaiSanPham.SelectedValue.ToString();
sp.TenSanPham = txtTenMon.Text;
sp.Gia = int.Parse(txtGia.Text);
sp.ChiTiet = txtChiTiet.Text;

// Thêm vào bảng SANPHAM trong CSDL
sp.ThemSanPham();

// Thêm sản phẩm mới vào listview
ListViewItem lvi = new ListViewItem(txtTenMon.Text);
lvi.SubItems.Add(txtGia.Text);
lvi.SubItems.Add(txtChiTiet.Text);
lvi.SubItems.Add(sp.MaSanPham.ToString());
lvi.Tag = cboLoaiSanPham.SelectedValue.ToString();
lvwSanPham.Items.Add(lvi);
```

- Xử lý cập nhật sản phẩm

```
// Cập nhật thông tin sản phẩm được chọn trên listview
ListViewItem lvi = lvwSanPham.SelectedItems[0];
lvi.SubItems[0] = txtTenMon.Text;
lvi.SubItems[1].Text = txtGia.Text;
lvi.SubItems[2].Text = txtChiTiet.Text;
lvi.Tag = cboLoaiSanPham.SelectedValue.ToString();

// Tạo đối tượng sản phẩm
SanPham sp = new SanPham();
sp.MaSanPham = int.Parse(lvi.SubItems[3].Text);
sp.LoaiSanPham = cboLoaiSanPham.SelectedValue.ToString();
sp.TenSanPham = txtTenMon.Text;
sp.Gia = int.Parse(txtGia.Text);
sp.ChiTiet = txtChiTiet.Text;

// Cập nhật vào bảng SANPHAM trong CSDL
sp.CapNhatSanPham();
```

- Xử lý xóa sản phẩm

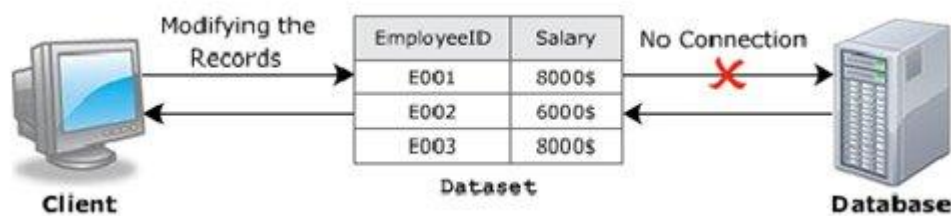
```
// Lấy ra sản phẩm được chọn
ListViewItem lvi = lvwSanPham.SelectedItems[0];

// Xóa sản phẩm khỏi bảng SANPHAM trong CSDL
SanPham sp = new SanPham();
sp.MaSanPham = int.Parse(lvi.SubItems[3].Text);
sp.XoaSanPham();

// Xóa sản phẩm khỏi ListView
lvwSanPham.Items.Remove(lvi);
```

3.3. Làm việc với mô hình ngắt kết nối (dis-connected)

ADO.Net hỗ trợ mô hình truy cập dữ liệu ngắt kết nối (disconnected data access). DataSet sẽ đọc và lưu trữ dữ liệu trong vùng nhớ đệm (cached memory), sau đó kết nối sẽ bị ngắt, dữ liệu trong vùng nhớ đệm sau khi thao tác xong, kết nối sẽ được mở và dữ liệu trong DataSet được cập nhật trở lại cơ sở dữ liệu.

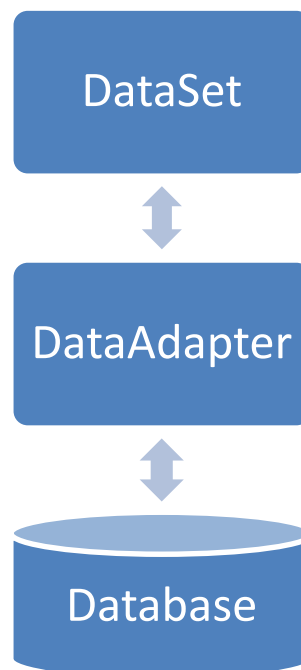


3.3.1. Đối tượng DataAdapter

Đối tượng DataAdapter dùng để thực hiện 2 chức năng:

- ❖ Đọc dữ liệu từ cơ sở dữ liệu vào DataSet
- ❖ Cập nhật dữ từ DataSet ra cơ sở dữ liệu

Có thể nói DataAdapter là cầu nối trung gian giữa cơ sở dữ liệu và DataSet



Tùy vào loại .Net Framework Data Provider sẽ có lớp đối tượng DataAdapter tương ứng

- Đọc dữ liệu từ cơ sở dữ liệu và đổ vào một DataTable trong DataSet dataSet. Kết quả trả về số dòng dữ liệu được đổ vào DataSet

❖ Update()

- Cú pháp:
 - `public int Update(DataSet dataSet)`
 - `public int Update(DataTable dataTable)`
 - `public int Update(DataSet dataSet, string srcTable)`
- Cập nhật các thay đổi của dữ liệu trong DataSet (DataTable) ra cơ sở dữ liệu

3.3.2. Đối tượng DataSet

Dữ liệu lưu trong DataSet được tổ chức dưới dạng một tập các bảng.

Mỗi bảng trong DataSet được biểu diễn bởi một đối tượng DataTable, mỗi dòng trong DataTable được biểu diễn bởi DataRow và mỗi cột được biểu diễn bởi DataColumn.

Đối tượng DataRelation để biểu diễn mối quan hệ giữa các DataTable.

Đối tượng Constraint để thiết lập ràng buộc trên DataTable để đảm bảo tính toàn vẹn dữ liệu.

Lớp DataSet nằm trong namespace System.Data.

3.3.2.1. Khởi tạo đối tượng DataSet

Cách 1: `DataSet <tên_biến> = new DataSet();`

Cách 2: `DataSet <tên_biến> = new DataSet(<tên_DataSet>);`

Lưu ý: với cách 1 thì DataSet được tạo sẽ có tên là NewDataSet

3.3.2.2. Một số thuộc tính

❖ DataSetName

- Kiểu dữ liệu: string
- Thiết lập/lấy tên của DataSet

❖ Tables

- Kiểu dữ liệu: DataTableCollection
- Danh sách các DataTable của DataSet

❖ Relations

- Kiểu dữ liệu: DataRelationCollection
- Tập hợp các mối liên kết giữa các DataTable trong DataSet

3.3.2.3. Một số phương thức

❖ AcceptChanges()

- Cú pháp: `public void AcceptChanges()`
- Thực hiện các thay đổi dữ liệu trên DataSet

❖ RejectChanges()

- Cú pháp: `public void RejectChanges()`
- Hủy bỏ các thay đổi dữ liệu trên DataSet

❖ Clear()

- Cú pháp: `public void Clear()`
- Xóa tất cả các dòng dữ liệu trong các DataTable trong DataSet

❖ ReadXML()

- Cú pháp: `public XMLReadMode ReadXML(string filename)`
- Đọc dữ liệu từ tập tin XML vào DataSet

❖ WriteXML()

- Cú pháp: `public void WriteXML(string filename)`
- Ghi dữ liệu từ DataSet ra tập tin XML

3.3.3. Đối tượng DataTable

Lớp đối tượng DataTable biểu diễn một bảng trong DataSet.

Đối tượng DataTable dùng để lưu dữ liệu từ nguồn dữ liệu.

DataTable tổ chức dữ liệu thành các dòng và cột. Mỗi đối tượng trong DataTable được biểu diễn bởi lớp đối tượng DataRow, mỗi cột trong DataTable biểu diễn bởi lớp đối tượng DataColumn.

Lớp DataTable nằm trong namespace System.Data.

3.3.3.1. Khởi tạo đối tượng DataTable

Cách 1: `DataTable <tên_biến> = new DataTable();`

Cách 2: `DataTable <tên_biến> = new DataTable(<tên_DataTable>);`

3.3.3.2. Một số thuộc tính

❖ Columns

- Kiểu dữ liệu: DataColumnCollection
- Danh sách các cột trong DataTable

❖ DataSet

- Kiểu dữ liệu: DataSet
- Đối tượng DataSet chứa DataTable hiện hành

❖ DefaultView

- Kiểu dữ liệu: DataView
- Đối tượng DataView của DataTable hiện hành

❖ PrimaryKey

- Kiểu dữ liệu: DataColumn[]
- Danh sách các cột làm khóa chính của DataTable

❖ Rows

- Kiểu dữ liệu: DataRowCollection
- Danh sách các dòng dữ liệu trong DataTable

❖ TableName

- Kiểu dữ liệu: string
- Tên của DataTable

3.3.3.3. Một số phương thức

❖ AcceptChanges()

- Cú pháp: `public void AcceptChanges()`
- Thực hiện các thay đổi dữ liệu trên DataTable

❖ RejectChanges()

- Cú pháp: `public void RejectChanges()`
- Hủy bỏ các thay đổi dữ liệu trên DataTable

- ❖ **Clear()**
 - Cú pháp: `public void Clear()`
 - Xóa tất cả các dòng dữ liệu trong DataTable
- ❖ **Copy()**
 - Cú pháp: `public DataTable Copy()`
 - Sao chép cấu trúc và dữ liệu của DataTable hiện hành
- ❖ **GetChanges()**
 - Cú pháp: `public DataTable GetChanges()`
 - Một DataTable mới gồm những thay đổi từ DataTable hiện hành sẽ được tạo ra. Nếu trong DataTable hiện hành không có sự thay đổi nào thì kết quả trả về là null
- ❖ **ImportRow()**
 - Cú pháp: `public void ImportRow(DataRow row)`
 - Thêm một DataRow vào DataTable hiện hành
- ❖ **Merge()**
 - Cú pháp: `public void Merge(DataTable dt)`
 - Gộp DataTable dt vào DataTable hiện hành
- ❖ **NewRow()**
 - Cú pháp: `public DataRow NewRow()`
 - Tạo ra một dòng mới có cấu trúc của DataTable hiện hành
- ❖ **ReadXML()**
 - Cú pháp: `public XMLReadMode ReadXML(string filename)`
 - Đọc dữ liệu từ tập tin XML vào DataTable
- ❖ **WriteXML()**
 - Cú pháp: `public void WriteXML(string filename)`
 - Ghi dữ liệu từ DataTable ra tập tin XML

3.3.4. Đối tượng DataView

DataView là lớp đối tượng được dùng cho việc sắp xếp, lọc, tìm kiếm và thay đổi các dòng dữ liệu của một bảng.

Lớp DataView nằm trong namespace System.Data.

3.3.4.1. Khởi tạo đối tượng DataView

Cách 1: `DataView <tên_biến> = new DataView(<đối_tượng_DataTable>);`

Cách 2: `DataView <tên_biến> = <đối_tượng_DataTable>.DefaultView;`

3.3.4.2. Một số thuộc tính

❖ Count

- Kiểu dữ liệu: int
- Số lượng dòng dữ liệu có trong DataView

❖ RowFilter

- Kiểu dữ liệu: string
- Biểu thức dùng để lọc dữ liệu trên DataView

❖ Sort

- Kiểu dữ liệu: string
- Tên các cột được sắp xếp và thứ tự sắp xếp

❖ Table

- Kiểu dữ liệu: DataTable
- Đối tượng DataTable tạo DataView

3.3.4.3. Một số phương thức

❖ Delete()

- Cú pháp: `public void Delete(int position)`
- Xóa một dòng tại vị trí position

3.3.5. Điều khiển DataGridView

3.3.5.1. Khái niệm

DataGridView là một điều khiển chuyên dùng cho việc hiển thị dữ liệu dưới dạng bảng với các chức năng:

- ❖ Có thể biểu diễn dữ liệu trên bảng trong Textbox, CheckBox hoặc ComboBox
- ❖ Có khả năng sử dụng DataSource để kết nối với bất kì cơ sở dữ liệu nào, với bất kì bảng nào

- ❖ Dữ liệu trên điều khiển được lưu sẵn trong bộ nhớ cache nên tăng tốc độ xử lý mà vẫn đảm bảo khả năng xử lý hàng trăm bản ghi một lúc

Lớp DataGridView nằm trong namespace System.Windows.Forms.

3.3.5.2. Một số thuộc tính

- ❖ AllowUserToAddRows
 - Kiểu dữ liệu: bool
 - Cho phép (true) hoặc không cho phép (false) người dùng thêm dòng mới trực tiếp trên DataGridView
- ❖ AllowUserToDeleteRows
 - Kiểu dữ liệu: bool
 - Cho phép (true) hoặc không cho phép (false) người dùng xóa các dòng trực tiếp trên DataGridView
- ❖ AllowUserToOrderColumns
 - Kiểu dữ liệu: bool
 - Cho phép (true) hoặc không cho phép (false) người dùng thay đổi thứ tự các cột trên DataGridView
- ❖ AllowUserToResizeColumns
 - Kiểu dữ liệu: bool
 - Cho phép (true) hoặc không cho phép (false) người dùng thay đổi kích thước các cột trên DataGridView
- ❖ AllowUserToResizeRows
 - Kiểu dữ liệu: bool
 - Cho phép (true) hoặc không cho phép (false) người dùng thay đổi kích thước các dòng trên DataGridView
- ❖ ColumnCount
 - Kiểu dữ liệu: int
 - Số lượng cột hiển thị trên DataGridView
- ❖ Columns
 - Kiểu dữ liệu: DataGridViewColumnCollection
 - Danh sách cột trên DataGridView

- ❖ CurrentCell
 - Kiểu dữ liệu: DataGridViewCell
 - Ô đang được trỏ đến trên DataGridView
- ❖ CurrentRow
 - Kiểu dữ liệu: DataGridViewRow
 - Dòng chứa ô đang được trỏ đến trên DataGridView
- ❖ DataSource
 - Kiểu dữ liệu: Object
 - Nguồn dữ liệu của DataGridView
- ❖ MultiSelect
 - Kiểu dữ liệu: bool
 - True: cho phép người dùng chọn nhiều dòng trên DataGridView;
False: ngược lại
- ❖ ReadOnly
 - Kiểu dữ liệu: bool
 - True: không thay đổi được dữ liệu trên DataGridView ; False:
ngược lại
- ❖ RowCount
 - Kiểu dữ liệu: int
 - Số lượng dòng hiển thị trên DataGridView
- ❖ Rows
 - Kiểu dữ liệu: DataGridViewRowCollection
 - Danh sách dòng trên DataGridView
- ❖ SelectedCells
 - Kiểu dữ liệu: DataGridViewSelectedCellCollection
 - Danh sách các ô được chọn trên DataGridView
- ❖ SelectedRows
 - Kiểu dữ liệu: DataGridViewSelectedRowCollection
 - Danh sách các dòng được chọn trên DataGridView

3.3.5.3. Một số phương thức

- ❖ ClearSelection()
 - Cú pháp: `public void ClearSelection()`
 - Bỏ chọn các ô trên DataGridView
- ❖ SelectAll()
 - Cú pháp: `public void SelectAll ()`
 - Chọn tất cả các ô trên DataGridView

3.3.5.4. Một số sự kiện

- ❖ Click
 - Xảy ra khi nhấn chuột vào DataGridView
- ❖ CellClick
 - Xảy ra khi nhấn chuột vào ô bất kỳ trên DataGridView
- ❖ CellContentClick
 - Xảy ra khi nhấn chuột vào ô có chứa nội dung trên DataGridView
- ❖ CellValueChanged
 - Xảy ra khi nội dung của ô trên DataGridView bị thay đổi
- ❖ UserAddedRow
 - Xảy ra sau khi người dùng thêm dòng mới vào DataGridView
- ❖ UserDeletedRow
 - Xảy ra sau khi người dùng xóa dòng trên DataGridView

3.3.6. Điều khiển Combobox nâng cao

3.3.6.1. Một số thuộc tính nâng cao

- ❖ DataSource
 - Kiểu dữ liệu: Object
 - Nguồn dữ liệu của Combobox
- ❖ DisplayMember
 - Kiểu dữ liệu: string
 - Tên trường/thuộc tính của đối tượng được dùng để hiển thị giá trị lên Combobox

❖ ValueMember

- Kiểu dữ liệu: string
- Tên trường/thuộc tính của đối tượng được dùng để làm giá trị thực cho phần tử trong Combobox

❖ SelectedItem

- Kiểu dữ liệu: Object
- Phần tử được chọn trong Combobox

❖ SelectedValue

- Kiểu dữ liệu: Object
- Giá trị thực của phần tử được chọn trong Combobox

3.3.6.2. Một số thao tác trên Combobox

❖ Gán nguồn dữ liệu là một danh sách các đối tượng vào combobox

```
<điều_khiển_combobox>.DataSource = <danh_sách_đối_tượng>;  
<điều_khiển_combobox>.DisplayMember = "<tên_trường_hiển_thị>";  
<điều_khiển_combobox>.ValueMember = "<tên_trường_giá_trị>";
```

❖ Lấy phần tử được chọn và ép về kiểu dữ liệu ban đầu của phần tử

```
(<kdl_phần_tử><tên_biến> =  
    (<kdl_phần_tử><điều_khiển_combobox>.SelectedItem;
```

3.3.7. Ví dụ minh họa

Phát triển ứng dụng “*Quản Lý Cửa Hàng Ăn Nhanh*” với CSDL CuaHang.mdb và có giao diện như hình bên dưới

The screenshot shows a Windows application titled "Quản Lý Cửa Hàng Ăn Nhanh". It has three radio buttons for search: "Tất cả sản phẩm" (selected), "Tìm theo loại sản phẩm", and "Tìm theo tên sản phẩm". Below these are input fields and a "Tìm" button. A section titled "Thông tin chi tiết sản phẩm" contains fields for "Tên món:" (Lẩu cá kèo), "Giá:" (100000), "Loại sản phẩm:" (Món lẩu), and "Chi tiết:" (Lẩu cá kèo của cửa hàng với nguyên liệu cá kèo tươi ngon...). Below this is a table titled "Danh sách sản phẩm" with columns: MaSanPham, LoaiSanPham, TenSanPham, Gia, and ChiTiet. The table contains five rows of data, with the second row (MaSanPham: 4, LoaiSanPham: la, TenSanPham: Lẩu cá kèo) highlighted in blue. At the bottom are "Lưu" and "Hủy" buttons.

	MaSanPham	LoaiSanPham	TenSanPham	Gia	ChiTiet
	11	bi	Tiger	15000	Bia Tiger...
▶	4	la	Lẩu cá kèo	100000	Lẩu cá kèo của ...
	5	ng	Bạch tuộc nướng	45000	Món bạch tuộc n...
	6	nn	Coca cola	8000	Nước uống giải k...
	7	tm	Sương sa	20000	Món thạch trái câ...
*					

- ❖ Các thao tác: Thêm sản phẩm mới, cập nhật thông tin sản phẩm, xóa sản phẩm được thực hiện trực tiếp trên DataGridView dgdSanPham
- ❖ Khai báo các biến toàn cục cho Form

```
// Các biến toàn cục
private string strChuoKetNoi;
private DataSet dsCuaHang;
// cầu nối của bảng SANPHAM và dataset dsCuaHang
private OleDbDataAdapter daSanPham;
// cầu nối của bảng LOAISANPHAM và dataset dsCuaHang
private OleDbDataAdapter daLoaiSP;
// khung nhìn của DataTable tblSANPHAM trong dataset dsCuaHang
private DataView dvSanPham;
```

- ❖ Khởi tạo các biến toàn cục trong hàm khởi tạo của Form

```
strChuoKetNoi = @"Provider=Microsoft.Jet.OLEDB.4.0; Data
Source=Data/CuaHang.mdb;";
dsCuaHang = new DataSet();
dvSanPham = new DataView();
daLoaiSP = new OleDbDataAdapter("SELECT * FROM LOAISANPHAM",
strChuoKetNoi);
```

```

// Xây dựng các đối tượng thực thi SQL cho DataAdapter
// daSanPham
daSanPham = new OleDbDataAdapter("SELECT * FROM SANPHAM",
                                   strChuoiKetNoi);
// Insert command
OleDbCommand comInsSanPham = new OleDbCommand("INSERT INTO
                                                SANPHAM(LoaiSanPham, TenSanPham, Gia, ChiTiet)
                                                VALUES(?,?,?,?)");
// Khai báo parameter theo cho Command theo cú pháp
// <command>.Parameters.Add(<tên_parameter>, <KDL_parameter>,
<độ_dài_paramter>, <tên_cột_tương_ứng_trong_CSDL>);

comInsSanPham.Parameters.Add("@loaisp", OleDbType.VarWChar, 2,
                              "LoaiSanPham");
comInsSanPham.Parameters.Add("@tensp", OleDbType.VarWChar, 30,
                              "TenSanPham");
comInsSanPham.Parameters.Add("@gia", OleDbType.Integer, 4,
                              "Gia");
comInsSanPham.Parameters.Add("@chitiet", OleDbType.VarWChar,
                              100, "ChiTiet");
comInsSanPham.Connection = new
                              OleDbConnection(strChuoiKetNoi);
daSanPham.InsertCommand = comInsSanPham;

// Update command
OleDbCommand comUpdSanPham = new OleDbCommand("UPDATE SANPHAM
SET LoaiSanPham = ?, TenSanPham = ?, Gia = ?, ChiTiet = ?
WHERE MASANPHAM = ?");
comUpdSanPham.Parameters.Add("@loaisp", OleDbType.VarWChar, 2,
                              "LoaiSanPham");
comUpdSanPham.Parameters.Add("@tensp", OleDbType.VarWChar, 30,
                              "TenSanPham");
comUpdSanPham.Parameters.Add("@gia", OleDbType.Integer, 4,
                              "Gia");
comUpdSanPham.Parameters.Add("@chitiet", OleDbType.VarWChar,
                              100, "ChiTiet");
comUpdSanPham.Parameters.Add("@masp", OleDbType.Integer, 4,
                              "MaSanPham");
comUpdSanPham.Connection = new
                              OleDbConnection(strChuoiKetNoi);
daSanPham.UpdateCommand = comUpdSanPham;

// Delete command
OleDbCommand comDelSanPham = new OleDbCommand("DELETE FROM
                                                SANPHAM WHERE MASANPHAM = ?");
comDelSanPham.Parameters.Add("@masp", OleDbType.Integer, 4,
                              "MaSanPham");
comDelSanPham.Connection = new
                              OleDbConnection(strChuoiKetNoi);
daSanPham.DeleteCommand = comDelSanPham;

```


❖ Xử lý sự kiện Load của Form:

- Đưa danh sách Loại sản phẩm vào Combobox cboLoaiSanPham và cboLoaiSanPhamCT

```
// Lấy dữ liệu trong bảng LOAISANPHAM đưa vào Dataset
// với tên của DataTable là tblLOAISANPHAM
daLoaiSP.Fill(dsCuaHang, "tblLOAISANPHAM");
// Đưa danh sách loại sản phẩm lên Combobox
// cboLoaiSanPham
cboLoaiSanPham.DataSource =
    dsCuaHang.Tables["tblLOAISANPHAM"].Copy();
cboLoaiSanPham.DisplayMember = "TenLoaiSanPham";
cboLoaiSanPham.ValueMember = "MaLoaiSanPham";

// Đưa danh sách loại sản phẩm lên Combobox
// cboLoaiSanPhamCT
cboLoaiSanPhamCT.DataSource =
    dsCuaHang.Tables["tblLOAISANPHAM"].Copy();
cboLoaiSanPhamCT.DisplayMember = "TenLoaiSanPham";
cboLoaiSanPhamCT.ValueMember = "MaLoaiSanPham";
```

- Đưa danh sách sản phẩm lên DataGridView dgdSanPham

```
// Lấy dữ liệu trong bảng SANPHAM đưa vào Dataset
// với tên của DataTable là tblSANPHAM
daSanPham.Fill(dsCuaHang, "tblSANPHAM");
// Đưa danh sách sản phẩm lên DataGridView dgdSanPham
dvSanPham = new DataView(dsCuaHang.Tables["tblSANPHAM"]);
dgdSanPham.DataSource = dvSanPham;
```

❖ Lọc sản phẩm

- Theo Loại sản phẩm được chọn từ Combobox cboLoaiSanPham

```
// Lấy mã loại sản phẩm được chọn trên combobox
// cboLoaiMonAn
DataRowView drvLoaiSP =
    (DataRowView)cboLoaiSanPham.SelectedItem;
string strMaLoaiSP =
    drvLoaiSP["MaLoaiSanPham"].ToString();

// Lọc danh sách sản phẩm theo loại sản phẩm
dvSanPham.RowFilter = string.Format("LOAISANPHAM =
    '{0}'", strMaLoaiSP);
```

- Theo tên sản phẩm có chứa từ khóa được nhập vào Textbox txtTuKhoa

```
dvSanPham.RowFilter = string.Format("TENSANPHAM like
    '%{0}%', txtTuKhoa.Text);
```

- Bỏ lọc sản phẩm

```
dvSanPham.RowFilter = "";
```

- ❖ Hủy những thay đổi trên DataGridView dgdSanPham nhưng chưa được lưu vào CSDL

```
dsCuaHang.Tables["tblSANPHAM"].RejectChanges();
```

- ❖ Lưu các thay đổi trên DataGridView dgdSanPham vào CSDL

```
daSanPham.Update(dsCuaHang.Tables["tblSANPHAM"]);
```

CHƯƠNG 4: LOCAL REPORT

4.1. Khái niệm report

Report được dùng để trình bày các bảng báo cáo, thống kê...

Một số phần mềm được sử dụng để thiết kế report:

- Microsoft Access
- Crystal Report
- C1Report (ComponetOne)
- XtraReport (Devexpress)
- ...

Để có thể thiết kế các loại report này trong Visual Studio 2010, chúng ta phải cài đặt thêm các gói phần mềm hỗ trợ để sử dụng.

Kể từ Visual Studio 2008, Microsoft đã thêm vào Local Report.

Trong chương này chúng ta sẽ tìm hiểu cách sử dụng Local Report để thiết kế các bảng báo cáo.

4.2. Local Report

Local Report dùng để trình bày các bảng báo cáo tại máy trạm mà không cần phải kết nối đến máy chủ Report.

Sử dụng điều khiển ReportViewer để hiển thị Local Report.

Local Report không có khả năng truy vấn dữ liệu. Dữ liệu cung cấp cho Local Report có thể là DataTable hoặc một tập hợp các đối tượng nghiệp vụ (business object).










Local Report có hỗ trợ tham số (parameter).

LocalReport là lớp thuộc namespace Microsoft.Reporting.WinForms

Các thành phần cơ bản trên Local Report (theo thứ tự):

- **Page Header:** phần đầu mỗi trang
- **Group Header:** phần đầu mỗi nhóm
- **Body:** phần chi tiết của báo cáo, chứa dữ liệu trong các trường/cột.
- **Group Footer:** phần cuối mỗi nhóm
- **Page Footer:** phần cuối mỗi trang

Một số điều khiển dùng để thiết kế report:

- Textbox 
- Line 
- Table 
- Matrix 
- Rectangle 
- List 
- Image 
- Subreport 
- Chart 

4.2.1. Một số thuộc tính

Thuộc tính	Kiểu dữ liệu	Mô tả
DataSources	ReportDataSourceCollection	Danh sách DataSource được dùng trong Report.
ReportEmbeddedResource	String	Tên của Report được nhúng vào chương trình.
ReportPath	String	Đường dẫn của Report.

4.2.2. Một số phương thức

Phương thức	Mô tả
<code>public int GetTotalPages()</code>	Trả về số trang của Report
<code>public void Refresh()</code>	Làm mới Report

4.2.3. Một số sự kiện

Phương thức	Mô tả
SubreportProcessing	xảy ra khi subreport được thực thi

4.3. ReportParameter

ReportParameter là lớp thể hiện cho đối tượng tham số trong Local Report

ReportParameter là lớp thuộc namespace Microsoft.Reporting.WinForms

4.3.1. Một số thuộc tính

4.3.1.1. Name

- ❖ Kiểu dữ liệu: string
- ❖ Mô tả: Tên của tham số

4.3.1.2. Value

- ❖ Kiểu dữ liệu: StringCollection
- ❖ Mô tả:
 - Danh sách giá trị của tham số
 - Mỗi phần tử trong danh sách có kiểu dữ liệu string, được thêm (Add), xóa (Remove)...

4.3.2. Phương thức khởi tạo

- ❖ `public ReportParameter()`
- ❖ `public ReportParameter(string name, string[] values)`

4.4. Điều khiển ReportViewer

ReportViewer là điều khiển dùng để hiển thị, quản lý việc in ấn, xuất bản report.

ReportViewer là lớp thuộc namespace Microsoft.Reporting.WinForms.

4.4.1. Một số thuộc tính

4.4.1.1. LocalReport

- ❖ Kiểu dữ liệu: LocalReport
- ❖ Mô tả: Là đối tượng Local Report được thiết lập trong điều khiển Report Viewer

4.4.1.2. PageCountMode

- ❖ Kiểu dữ liệu: PageCountMode
- ❖ Mô tả:
 - Cách tính tổng số trang của report được trình bày
 - Các giá trị của PageCountMode:
 - PageCountMode.Actual: hiển thị số trang thực của report ở trên thanh công cụ
 - PageCountMode.Estimated: là giá trị mặc định, có thể hiển thị số trang thực hoặc ước đoán số trang của report. Số trang của report được hiển thị trên thanh công cụ có thể bị thay đổi.

4.4.1.3. PrinterSettings

- ❖ Kiểu dữ liệu: PrinterSettings
- ❖ Mô tả:
 - Là tập hợp những thiết lập về máy in, cách thức in, số trang in...
 - Tham khảo thêm tại:
<http://msdn.microsoft.com/en-us/library/system.drawing.printing.printersettings.aspx>

4.4.1.4. Một số thuộc tính dùng để thiết lập việc hiển thị các nút điều khiển trên thanh công cụ

- ❖ ShowBackButton: Hiển thị nút Back
- ❖ ShowDocumentMapButton: Hiển thị sơ đồ tài liệu
- ❖ ShowExportButton: Hiển thị nút Export
- ❖ ShowFindControls: Hiển thị các điều khiển tìm kiếm
- ❖ ShowPageNavigationControls: Hiển thị các điều khiển điều hướng
- ❖ ShowPrintButton: Hiển thị nút in ấn
- ❖ ShowRefreshButton: Hiển thị nút Refresh
- ❖ ShowStopButton: Hiển thị nút dừng
- ❖ ShowToolBar: Hiển thị thanh công cụ
- ❖ ShowZoomControl: Hiển thị các điều khiển phóng to/thu nhỏ

Kiểu dữ liệu của các thuộc tính này đều là bool và có giá trị mặc định là true

4.4.2. Một số phương thức

4.4.2.1. Phương thức khởi tạo

4.4.2.2. Clear()

- ❖ Cú pháp: `public void Clear()`
- ❖ Mô tả: đưa ReportViewer về trạng thái ban đầu (mặc định)

4.4.2.3. GetTotalPages()

- ❖ Cú pháp: `public int GetTotalPages()`
- ❖ Mô tả: trả về tổng số trang của report

4.4.2.4. PageSetupDialog()

❖ Cú pháp: `public DialogResult PageSetupDialog()`

❖ Mô tả:

- Mở hộp thoại Page Setup
- Kết quả trả về là DialogResult.OK hoặc DialogResult.Cancel phụ thuộc vào người sử dụng

4.4.2.5. PrintDialog()

❖ Cú pháp: `public DialogResult PrintDialog()`

❖ Mô tả:

- Mở hộp thoại in ấn
- Nếu kết quả trả về là DialogResult.OK thì report sẽ được in, ngược lại thì report sẽ không được in

4.4.2.6. RefreshReport()

❖ Cú pháp: `public void RefreshReport()`

- Mô tả: làm mới report hiện hành trong Report Viewer

4.4.3. Một số sự kiện

4.4.3.1. ReportError

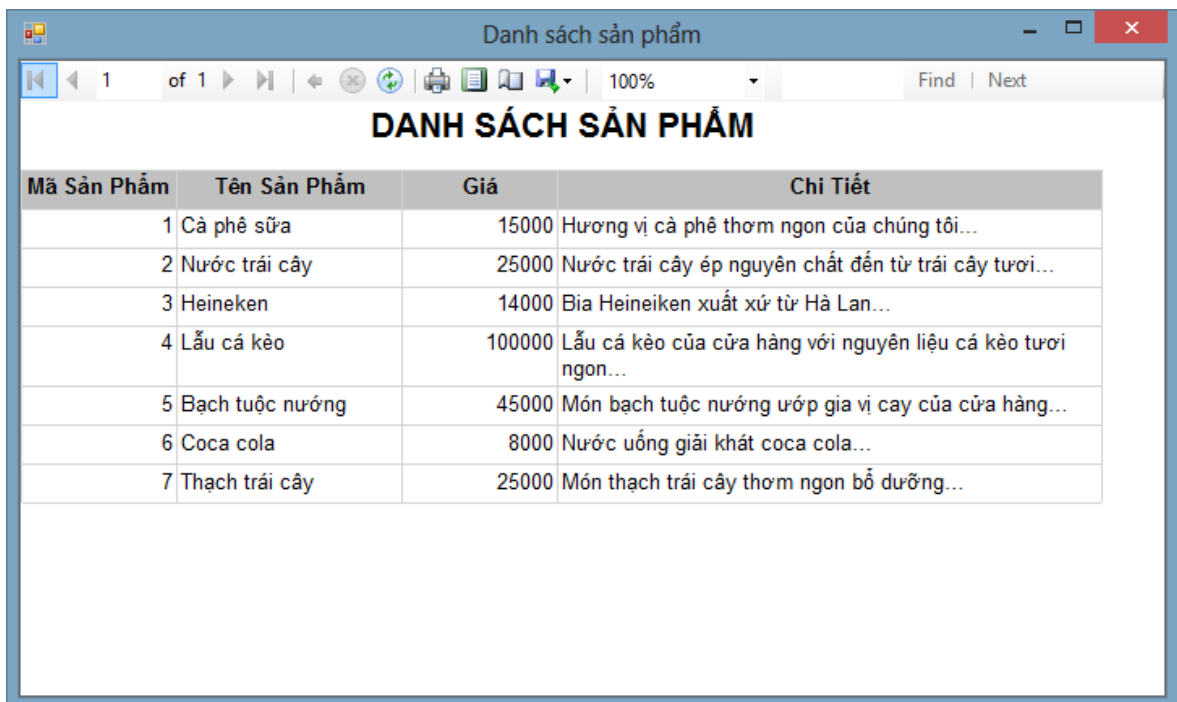
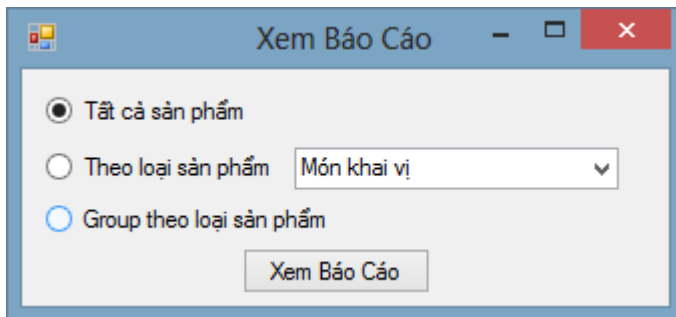
Xảy ra khi report bị lỗi

4.4.3.2. ReportRefresh

Xảy ra khi report được làm mới

4.5. Ví dụ minh họa

Sử dụng CSDL Quản lý cửa hàng ăn nhanh (CuanHang.mdb) và các lớp SanPham, LoaiSanPham trong phần ADO.NET - Làm việc với mô hình kết nối (connected) , viết ứng dụng “Xem Báo Cáo” với giao diện như hình bên dưới:

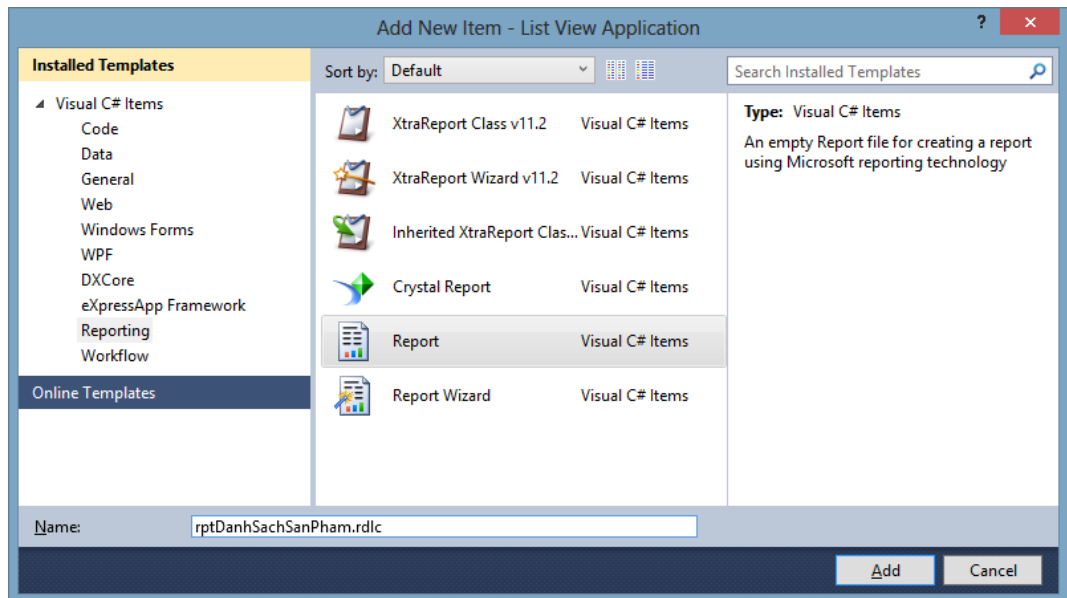


Mã Sản Phẩm	Tên Sản Phẩm	Giá	Chi Tiết
1	Cà phê sữa	15000	Hương vị cà phê thơm ngon của chúng tôi...
2	Nước trái cây	25000	Nước trái cây ép nguyên chất đến từ trái cây tươi...
3	Heineken	14000	Bia Heineken xuất xứ từ Hà Lan...
4	Lẩu cá kèo	100000	Lẩu cá kèo của cửa hàng với nguyên liệu cá kèo tươi ngon...
5	Bạch tuộc nướng	45000	Món bạch tuộc nướng ướp gia vị cay của cửa hàng...
6	Coca cola	8000	Nước uống giải khát coca cola...
7	Thạch trái cây	25000	Món thạch trái cây thơm ngon bổ dưỡng...

4.5.1. Tạo report đơn giản

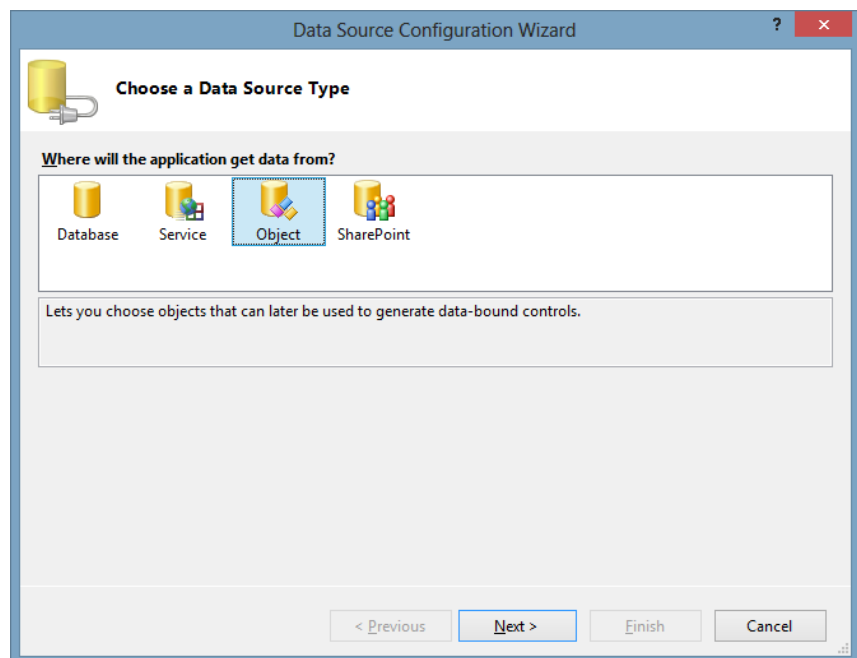
Tạo report hiển thị danh sách tất cả sản phẩm trong bảng SANPHAM

- ❖ Thêm tập tin Local Report rptDanhSachSanPham.rldc vào project

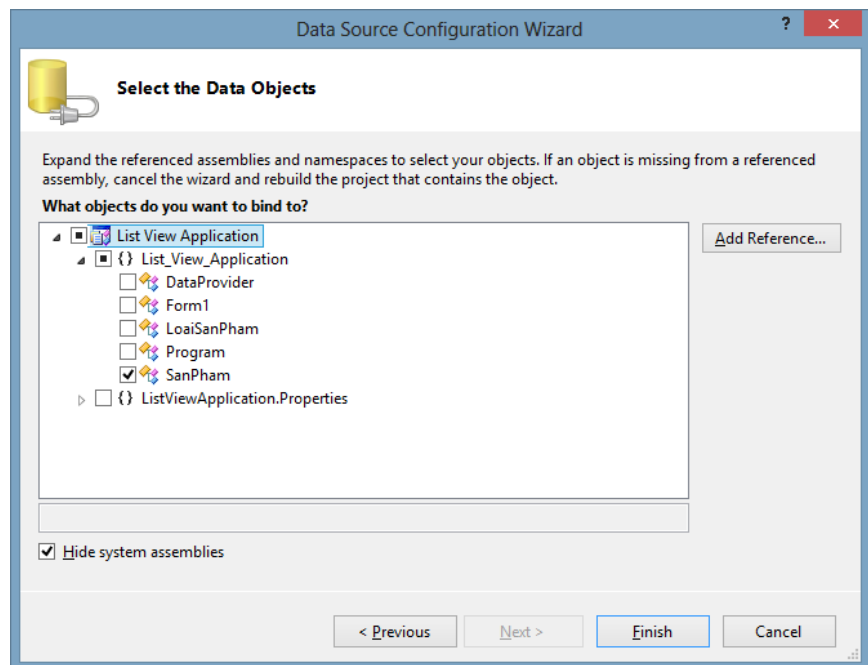


❖ Sử dụng các điều khiển trong hộp công cụ (Toolbox) để thiết kế report rptDanhSachSanPham

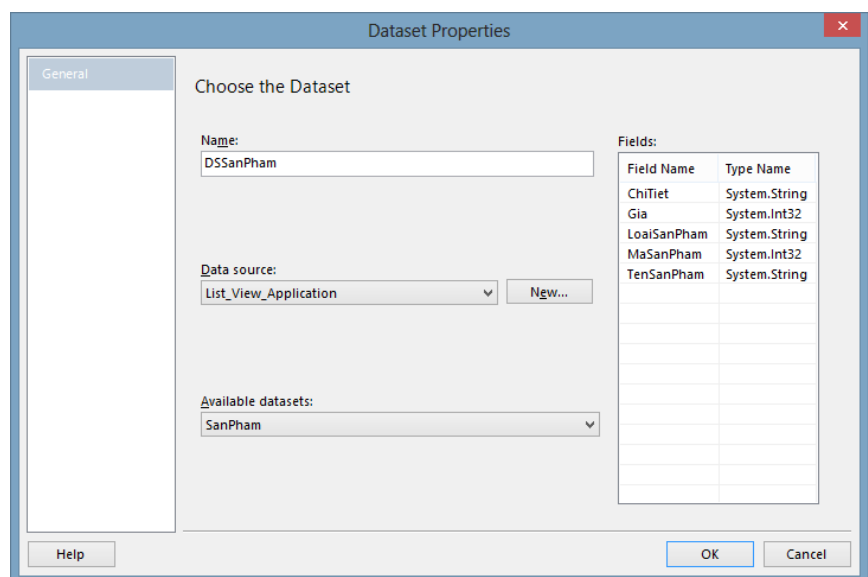
- Dùng điều khiển TextBox để tạo tiêu đề “DANH SÁCH SẢN PHẨM”
- Dùng điều khiển Table để tạo bảng danh sách sản phẩm
 - Đưa điều khiển Table vào Report
 - Thiết lập DataSource cho Table



- Tùy nguồn dữ liệu mà chọn loại DataSource (Database, Object, ...) thích hợp. Trong ví dụ này, ta sử dụng DataSource là Object.

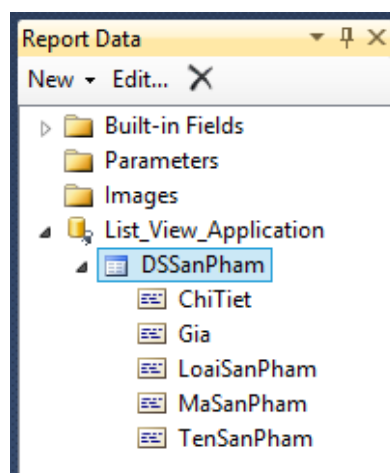


- Chọn lớp đối tượng thích hợp. Trong ví dụ là lớp SanPham



Đặt tên cho nguồn dữ liệu: DSSanPham

- Đưa các trường dữ liệu trong khung Report Data vào Table



- Kết quả thiết kế:

DANH SÁCH SẢN PHẨM			
Mã Sản Phẩm	Tên Sản Phẩm	Giá	Chi Tiết
[MaSanPham]	[TenSanPham]	[Gia]	[ChiTiet]

❖ Thiết kế Form để hiển thị Report

- Thêm Form có tên frmReport vào project
- Đưa điều khiển ReportView vào Form frmReport, đặt tên là: rpvReport

❖ Xử lý hiển thị Report lên Form frmReport

```
// Lấy danh sách tất cả sản phẩm trong bảng SANPHAM trong CSDL
SanPham sp = new SanPham();
List<SanPham> lstSanPham = sp.LayDanhSach();
// Chọn report cho report viewer
this.rpvReport.LocalReport.ReportEmbeddedResource =
    "ListViewApplication.rptDanhSachSanPham.rdlc";
// Đổ dữ liệu vào report
this.rpvReport.LocalReport.DataSources.Add(new
    ReportDataSource("DSSanPham", lstSanPham));
this.rpvReport.RefreshReport();
```

Lưu ý:

- Thuộc tính *ReportEmbeddedResource* của LocalReport là chuỗi có định dạng: <namespace_của_soluton>.<tên_tập_tin_report>.rdlc
- Phần tên của *ReportDataSource* phải là tên của nguồn dữ liệu của Table khi thiết kế Report.

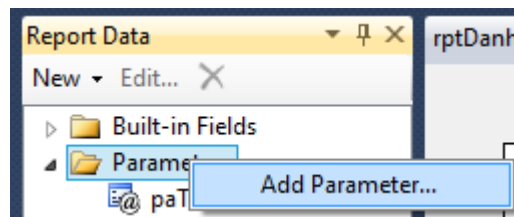
❖ Kết quả chạy chương trình:

DANH SÁCH SẢN PHẨM			
Mã Sản Phẩm	Tên Sản Phẩm	Giá	Chi Tiết
1	Cà phê sữa	15000	Hương vị cà phê thơm ngon của chúng tôi...
2	Nước trái cây	25000	Nước trái cây ép nguyên chất đến từ trái cây tươi...
3	Heineken	14000	Bia Heineken xuất xứ từ Hà Lan...
4	Lẩu cá kèo	100000	Lẩu cá kèo của cửa hàng với nguyên liệu cá kèo tươi ngon...
5	Bạch tuộc nướng	45000	Món bạch tuộc nướng ướp gia vị cay của cửa hàng...
6	Coca cola	8000	Nước uống giải khát coca cola...
7	Thạch trái cây	25000	Món thạch trái cây thơm ngon bổ dưỡng...

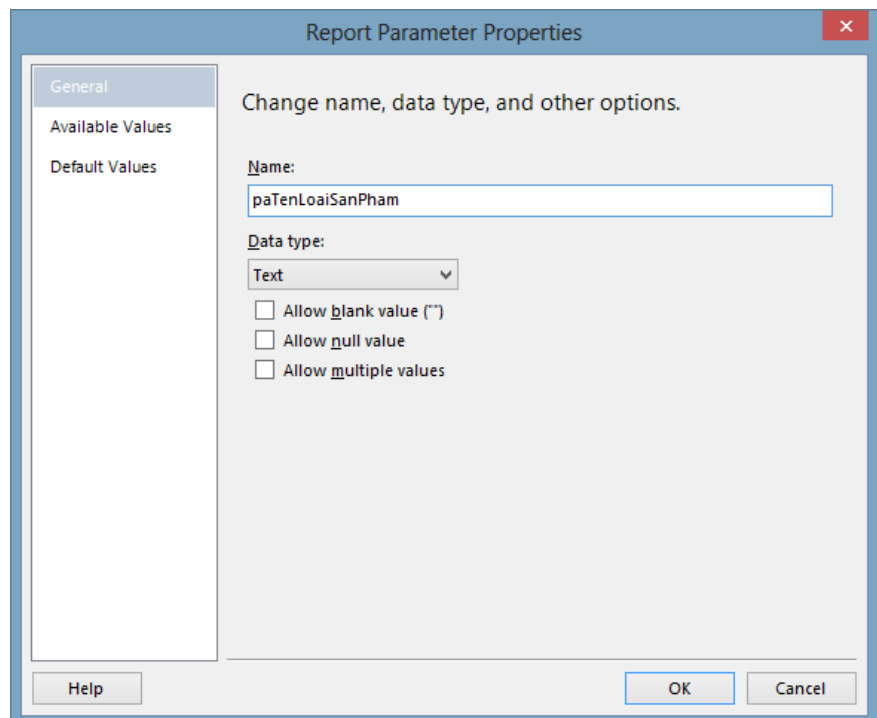
4.5.2. Tạo report có tham số (Parameter)

Tạo report hiển thị danh sách sản phẩm trong bảng SANPHAM của một loại sản phẩm được chọn từ chương trình và có tên loại sản phẩm

- ❖ Thêm tập tin Local Report rptDanhSachSanPhamTheoLoai.rldc vào project
- ❖ Thiết kế report rptDanhSachSanPhamTheoLoai.rldc tương tự như report rptDanhSachSanPham.rldc, nhưng thêm Tên Loại Sản Phẩm vào tiêu đề của Report
 - Thêm parameter paTenLoaiSP (Tên loại sản phẩm) vào Report:
 - Nhấn chuột phải vào Parameters trong khung Report Data, chọn Add Parameter...



- Gõ tên (Name) và chọn kiểu dữ liệu (Type) cho parameter



- Đưa các trường thích hợp trong Parameters trong khung Report Data vào Report

- Kết quả thiết kế:

DANH SÁCH SẢN PHẨM			
Loại sản phẩm:		@paTenLoaiSanPham	
Mã Sản Phẩm	Tên Sản Phẩm	Giá	Chi Tiết
[MaSanPham]	[TenSanPham]	[Gia]	[ChiTiet]

- ❖ Xử lý hiển thị Report lên Form frmReport

```
// Lấy các thông tin về loại sản phẩm được chọn trên Combobox
// cboLoaiSanPham
LoaiSanPham lsp = (LoaiSanPham) cboLoaiSanPham.SelectedItem;

// Lấy danh sách tất cả sản phẩm trong bảng SANPHAM thuộc loại
// đã chọn trong CSDL
SanPham sp = new SanPham();
List<SanPham> lstSanPham = sp.LayDanhSach(lsp.MaLoaiSanPham);
// Chọn report cho report viewer
this.rpvReport.LocalReport.ReportEmbeddedResource =
    "ListViewApplication.rptDanhSachSanPhamParameter.rdlc";
// Đổ dữ liệu vào report
this.rpvReport.LocalReport.DataSources.Add(new
    ReportDataSource("DSSanPham", lstSanPham));
// Gán giá trị cho parameter trong report
this.rpvReport.LocalReport.SetParameters(new
    ReportParameter("paTenLoaiSanPham", lsp.TenLoaiSanPham,
        false));
```

Lưu ý:

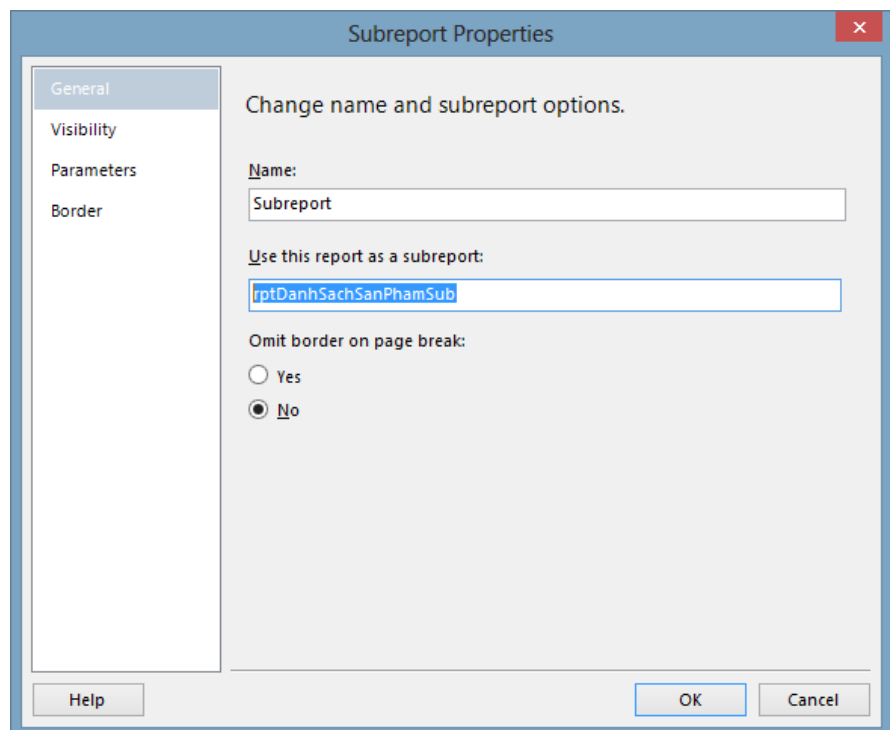
- Khi tạo đối tượng ReportParameter, tham số thứ 3 cho phép hiển thị (true) hoặc không hiển thị (false) hộp thoại yêu cầu nhập giá trị parameter
- ❖ Kết quả chạy chương trình:

Mã Sản Phẩm	Tên Sản Phẩm	Giá	Chi Tiết
1	Cà phê sữa	15000	Hương vị cà phê thơm ngon của chúng tôi...
2	Nước trái cây	25000	Nước trái cây ép nguyên chất đến từ trái cây tươi...

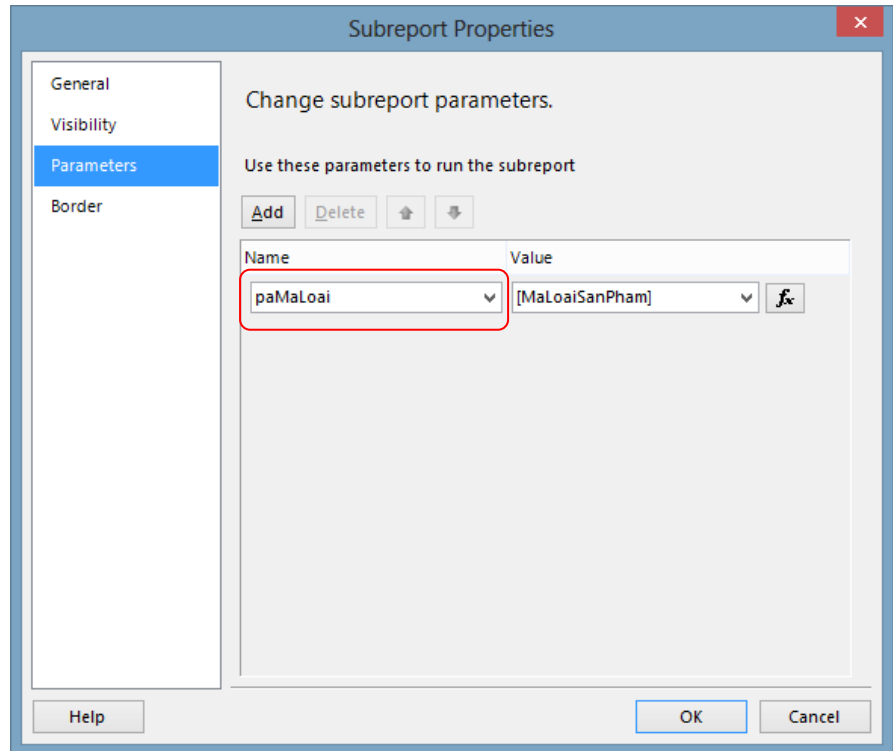
4.5.3. Tạo report có Group

Tạo report hiển thị danh sách tất cả sản phẩm trong bảng SANPHAM nhưng được gom theo từng loại sản phẩm

- ❖ Thêm tập tin Local Report rptDanhSachSanPhamGroup.rldc và rptDanhSachSanPhamSub.rldc vào project
- ❖ Tương tự, thiết kế report rptDanhSachSanPhamGroup.rldc như các report trên, nhưng:
 - Bỏ phần Header của Table, chỉ giữ lại ô chứa Tên Loại Sản Phẩm
 - Thêm điều khiển Subreport vào dòng cuối của Table
 - Thiết lập các thuộc tính cho Subreport
 - Tên Subreport: chỉ chứa tên tập tin report, không chứa phần mở rộng (.rldc)



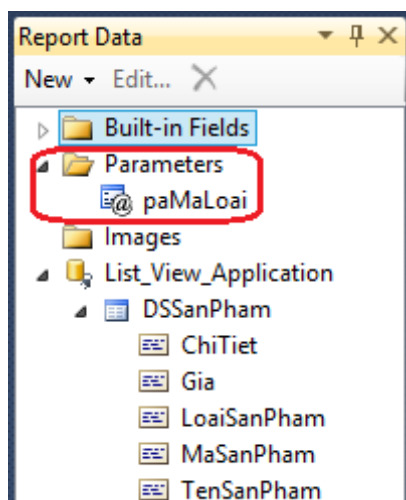
- Các parameter (nếu có)



- Kết quả thiết kế:



- ❖ Tương tự, thiết kế report rptDanhSachSanPhamSub.rdlc như các report trên, nhưng phải lưu ý: nếu điều khiển Subreport trong report rptDanhSachSanPhamGroup.rdlc có bao nhiêu parameter thì report rptDanhSachSanPhamSub.rdlc cũng phải khai báo bấy nhiêu parameter cùng tên tương ứng



Kết quả thiết kế:

Mã Sản Phẩm	Tên Sản Phẩm	Giá	Chi Tiết
[MaSanPham]	[TenSanPham]	[Gia]	[ChiTiet]

❖ Xử lý hiển thị Report lên Form frmReport

```
// Lấy danh sách tất cả loại sản phẩm trong bảng LOAISANPHAM
// trong CSDL
LoaiSanPham lsp = new LoaiSanPham();
List<LoaiSanPham> lstLoaiSanPham = lsp.LayDanhSach();
// Chọn report cho report viewer
this.rpvReport.LocalReport.ReportEmbeddedResource =
    "ListViewApplication.rptDanhSachSanPhamGroup.rdlc";
// Lắng nghe sự kiện phát sinh khi thực thi subreport
this.rpvReport.LocalReport.SubreportProcessing += new
    SubreportProcessingEventHandler(
        LocalReport_SubreportProcessing);
// Đổ dữ liệu vào report
this.rpvReport.LocalReport.DataSources.Add(new
    ReportDataSource("DSLoaiSanPham", lstLoaiSanPham));
```

Phương thức xử lý sự kiện SubreportProcessing

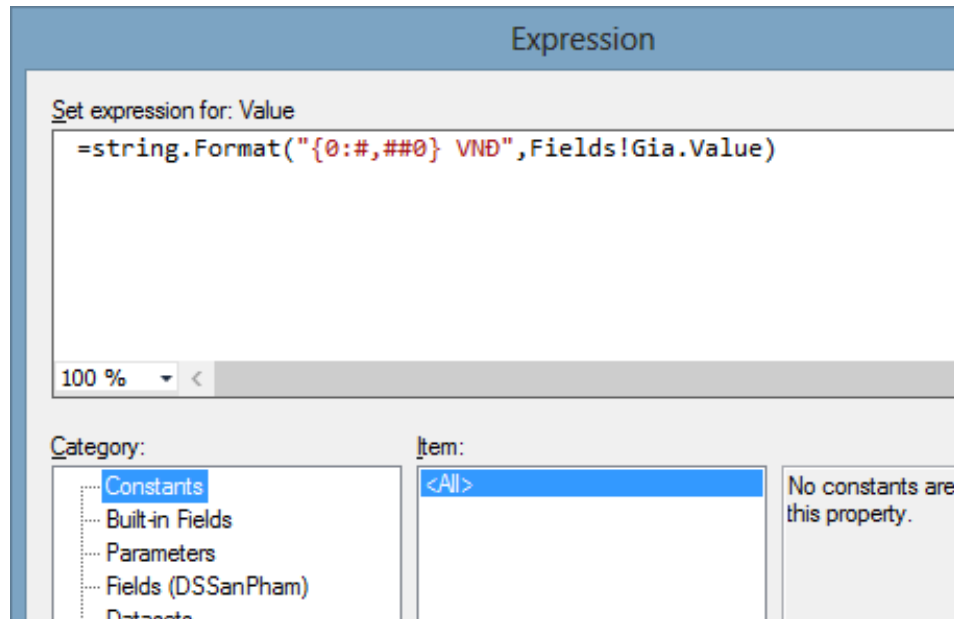
```
void LocalReport_SubreportProcessing(object sender,
    SubreportProcessingEventArgs e)
{
    // Lấy mã loại sản phẩm
    string strMaLoai = e.Parameters["paMaLoai"].Values[0];
    // Lấy danh sách sản phẩm theo loại sản phẩm
    SanPham sp = new SanPham();
    List<SanPham> lstSanPham = sp.LayDanhSach(strMaLoai);
    // Đổ dữ liệu vào Subreport
    e.DataSources.Add(new ReportDataSource("DSSanPham",
        lstSanPham));
}
```

❖ Kết quả chạy chương trình:

Mã Sản Phẩm	Tên Sản Phẩm	Giá	Chi Tiết
5	Bạch tuộc nướng	45000	Món bạch tuộc nướng ướp gia vị cay của cửa hàng...
4	Lẩu cá kèo	100000	Lẩu cá kèo của cửa hàng với nguyên liệu cá kèo tươi ngon...

4.5.4. Một số thao tác trong Local Report

- ❖ Định dạng hiển thị số dạng “#,##0 VNĐ” cho cột Giá trong Report
 - Bấm chuột phải vào ô cần định dạng, chọn Expression..., dùng phương thức Format của lớp string để định dạng giá trị

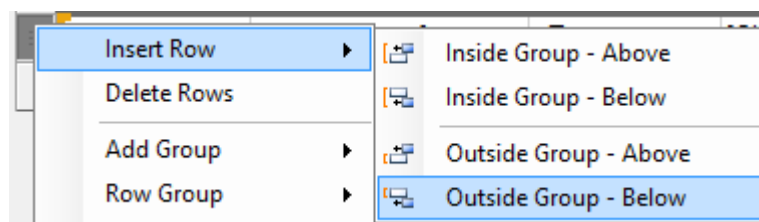


- Kết quả chạy chương trình:

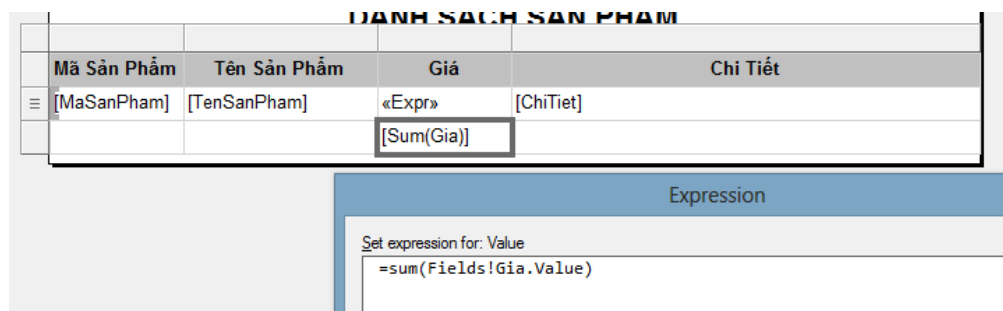
Mã Sản Phẩm	Tên Sản Phẩm	Giá	Chi Tiết
1	Cà phê sữa	15,000 VNĐ	Hương vị cà phê thơm ngon của chúng tôi...
2	Nước trái cây	25,000 VNĐ	Nước trái cây ép nguyên chất đến từ trái cây tươi...
3	Heineken	14,000 VNĐ	Bia Heineken xuất xứ từ Hà Lan...
4	Lẩu cá kèo	100,000 VNĐ	Lẩu cá kèo của cửa hàng với nguyên liệu cá kèo tươi ngon...
5	Bạch tuộc nướng	45,000 VNĐ	Món bạch tuộc nướng ướp gia vị cay của cửa hàng...
6	Coca cola	8,000 VNĐ	Nước uống giải khát coca cola...
7	Thạch trái cây	25,000 VNĐ	Món thạch trái cây thơm ngon bổ dưỡng...

Lưu ý: có thể dùng phương thức Format của lớp string để thay cho thao tác cộng chuỗi.

- ❖ Tính tổng Giá các sản phẩm
 - Thêm dòng vào cuối bảng: nhấn chuột phải vào dòng dữ liệu, chọn Insert Row ➔ Outsite Group - Below



- Tạo biểu thức tính tổng cho ô Giá của dòng vừa thêm

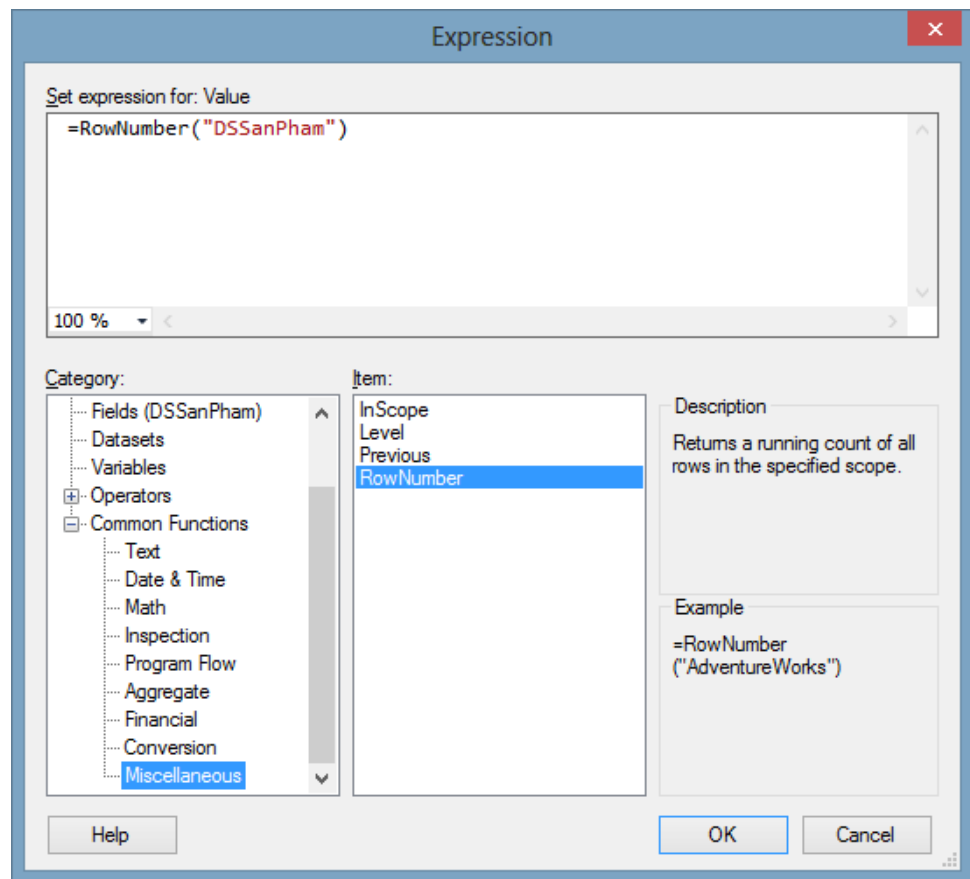


- Kết quả chạy chương trình:

Mã Sản Phẩm	Tên Sản Phẩm	Giá	Chi Tiết
1	Cà phê sữa	15,000 VNĐ	Hương vị cà phê thơm ngon của chúng tôi...
2	Nước trái cây	25,000 VNĐ	Nước trái cây ép nguyên chất đến từ trái cây tươi...
3	Heineken	14,000 VNĐ	Bia Heineken xuất xứ từ Hà Lan...
4	Lẩu cá kèo	100,000 VNĐ	Lẩu cá kèo của cửa hàng với nguyên liệu cá kèo tươi ngon...
5	Bạch tuộc nướng	45,000 VNĐ	Món bạch tuộc nướng ướp gia vị cay của cửa hàng...
6	Coca cola	8,000 VNĐ	Nước uống giải khát coca cola...
7	Thạch trái cây	25,000 VNĐ	Món thạch trái cây thơm ngon bổ dưỡng...
Tổng cộng:		232000	

❖ Thêm Cột số thứ tự vào trước cột Mã sản phẩm

- Thêm 1 cột vào trước cột Mã sản phẩm, nhập Header là: STT
- Thiết lập biểu thức cho ô dữ liệu của cột STT



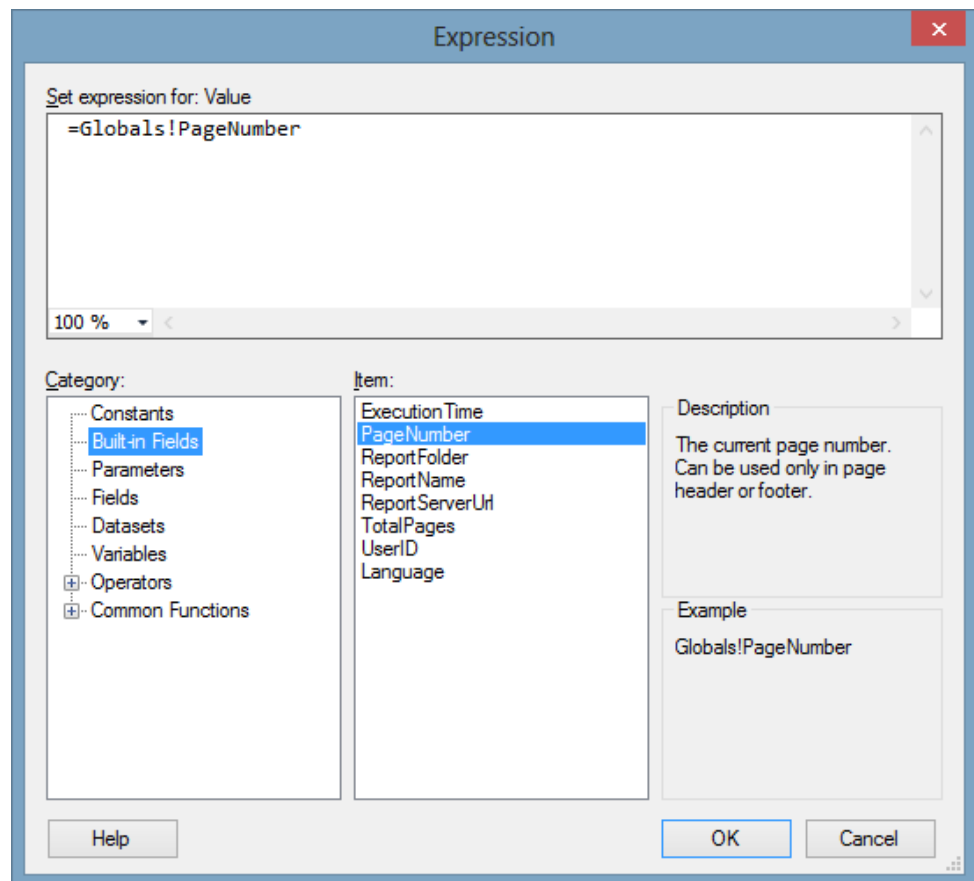
Lưu ý: Tham số của phương thức RowNumber là tên của nguồn dữ liệu của Table

- Kết quả chạy chương trình:

DANH SÁCH SẢN PHẨM				
STT	Mã Sản Phẩm	Tên Sản Phẩm	Giá	Chi Tiết
1	1	Cà phê sữa	15,000 VNĐ	Hương vị cà phê thơm ngon của chúng tôi...
2	2	Nước trái cây	25,000 VNĐ	Nước trái cây ép nguyên chất đến từ trái cây tươi...
3	3	Heineken	14,000 VNĐ	Bia Heineken xuất xứ từ Hà Lan...
4	4	Lẩu cá kèo	100,000 VNĐ	Lẩu cá kèo của cửa hàng với nguyên liệu cá kèo tươi ngon...
5	5	Bạch tuộc nướng	45,000 VNĐ	Món bạch tuộc nướng ướp gia vị cay của cửa hàng...
6	6	Coca cola	8,000 VNĐ	Nước uống giải khát coca cola...
7	7	Thạch trái cây	25,000 VNĐ	Món thạch trái cây thơm ngon bổ dưỡng...
Tổng cộng:			232000	

❖ Thêm số trang vào cuối mỗi trang

- Thêm Page Footer vào Report
- Thêm điều khiển TextBox vào Page Footer, và thiết lập biểu thức hiển thị số trang



- Kết quả thiết kế:

DANH SÁCH SẢN PHẨM				
STT	Mã Sản Phẩm	Tên Sản Phẩm	Giá	Chi Tiết
«Expr»	[MaSanPham]	[TenSanPham]	«Expr»	[ChiTiet]
	Tổng cộng:		[Sum(Gia)]	
[&PageNumber]				

- Kết quả chạy chương trình:

DANH SÁCH SẢN PHẨM				
STT	Mã Sản Phẩm	Tên Sản Phẩm	Giá	Chi Tiết
1	1	Cà phê sữa	15,000 VNĐ	Hương vị cà phê thơm ngon của chúng tôi...
2	2	Nước trái cây	25,000 VNĐ	Nước trái cây ép nguyên chất đến từ trái cây tươi...
3	3	Heineken	14,000 VNĐ	Bia Heineken xuất xứ từ Hà Lan...
4	4	Lẩu cá kèo	100,000 VNĐ	Lẩu cá kèo của cửa hàng với nguyên liệu cá kèo tươi ngon...
5	5	Bạch tuộc nướng	45,000 VNĐ	Món bạch tuộc nướng ướp gia vị cay của cửa hàng...
6	6	Coca cola	8,000 VNĐ	Nước uống giải khát coca cola...
7	7	Thạch trái cây	25,000 VNĐ	Món thạch trái cây thơm ngon bổ dưỡng...
	Tổng cộng:		232000	
1				

- Ghi chú: có thể áp dụng biểu thức sau để hiển thị số trang theo định dạng “Trang “ + pageNumber

```
=string.Format("Trang {0}", Globals!PageNumber)
```

❖ Thêm ngày tháng năm hiện tại

- Để lấy ngày giờ hiện tại, dùng biểu thức

```
=Now()
```

- Có thể dùng phương thức string.Format() để tùy chỉnh hiển thị ngày giờ

- Ví dụ:

```
=string.Format("Ngày {0:dd} Tháng {0:MM} Năm {0:yyyy}",  
Now())
```

CHƯƠNG 5: MÔ HÌNH 3 LỚP

5.1. Tổng quan mô hình 3 lớp (Three Layers)

5.1.1. Giới thiệu

Từ khi bắt đầu học các môn về lập trình nói chung, chúng ta đều tiếp cận theo thứ tự:

- ❖ Tất cả mọi thứ đều được chứa trong hàm main
- ❖ Chia nhỏ các đoạn mã nguồn thành các hàm với các chức năng riêng biệt
- ❖ Làm việc với các khái niệm trong hướng đối tượng

Nói tóm tắt, chúng ta thường đặt tất cả mọi thứ trong cùng một tập tin. Như thế với những chương trình lớn, chúng ta sẽ gặp khó khăn trong việc khoanh vùng để kiểm lỗi, quản lý mã nguồn chương trình.

Mô hình 3 lớp là một chuẩn mô hình trong việc phát triển phần mềm, giúp chúng ta có thể quản lý các thành phần trong hệ thống dễ dàng hơn, cũng như không bị ảnh hưởng bởi các thay đổi bằng cách nhóm các thành phần có cùng chức năng lại với nhau và phân chia trách nhiệm cho từng nhóm để công việc không bị chồng chéo, ảnh hưởng lẫn nhau.

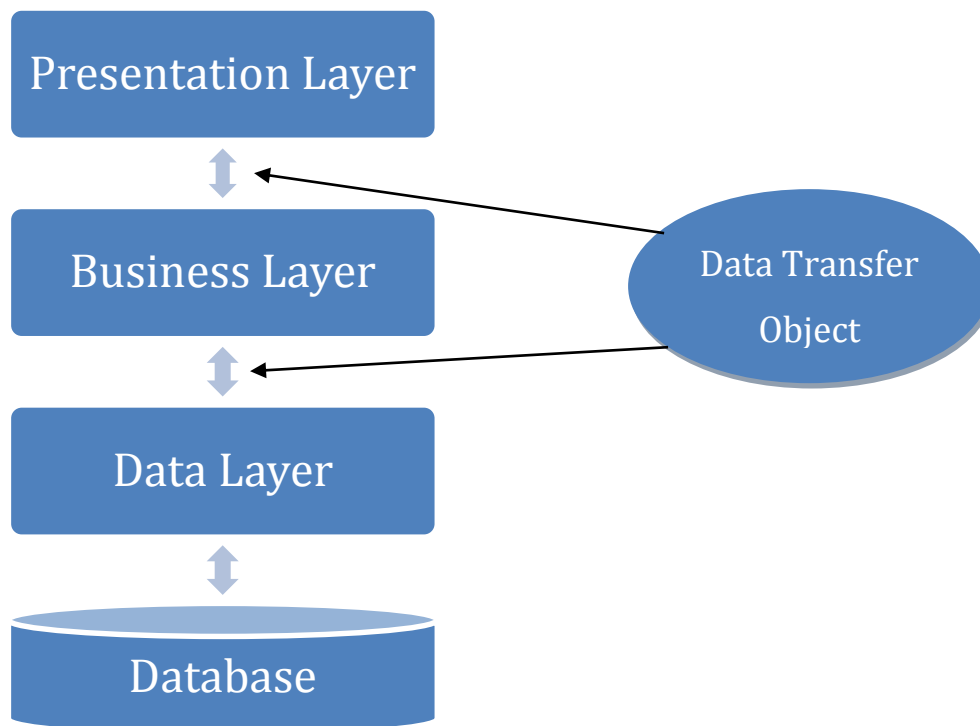
5.1.2. Các thành phần của mô hình 3 lớp

Mô hình 3 lớp gồm có các thành phần sau:

- ❖ Presentation Layer (Lớp giao diện)
 - Làm nhiệm vụ giao tiếp với người dùng cuối để thu thập dữ liệu, hiển thị kết quả thông qua các thành phần trong giao diện người sử dụng
- ❖ Business Layer (Lớp nghiệp vụ)
 - Thực hiện các nghiệp vụ chính của ứng dụng như tính toán, xử lý ngoại lệ...
 - Cung cấp các dịch vụ cho lớp Presentation
- ❖ Data Layer (Lớp dữ liệu)
 - Thực hiện các công việc truy xuất, lưu trữ dữ liệu của ứng dụng
 - Cung cấp các dịch vụ cho lớp Business

Để thông tin (dữ liệu) được chuyển qua lại giữa các lớp đồng nhất và tiện dụng, người ta xây dựng thêm một lớp (layer) gọi là Data Transfer Object (DTO) chứa các lớp đối tượng tương ứng với các bảng trong CSDL.

Sơ đồ của mô hình 3 lớp:

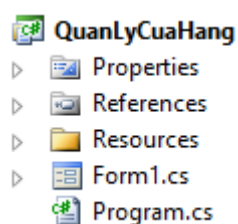


5.2. Ví dụ minh họa

Phát triển ứng dụng “*Quản Lý Cửa Hàng Ăn Nhanh*” tương tự như đã thực hiện ở phần ADO.NET – Làm việc với mô hình kết nối (connected), nhưng ứng dụng sẽ được phát triển theo mô hình 3 lớp.

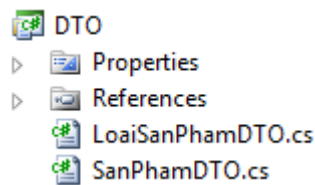
❖ Các bước thực hiện:

- Tạo một project Windows Forms Application, đặt làm project mặc định của Solution

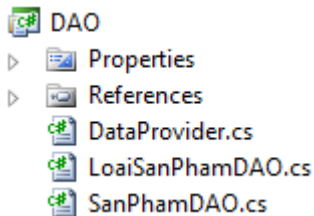


- Tạo các project loại Class Library trong cùng Solution với project loại Windows Form Application:

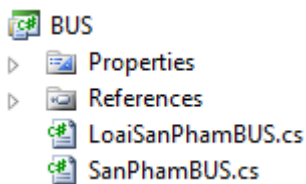
- DTO: Data Transfer Object



- DAO: Data Layer



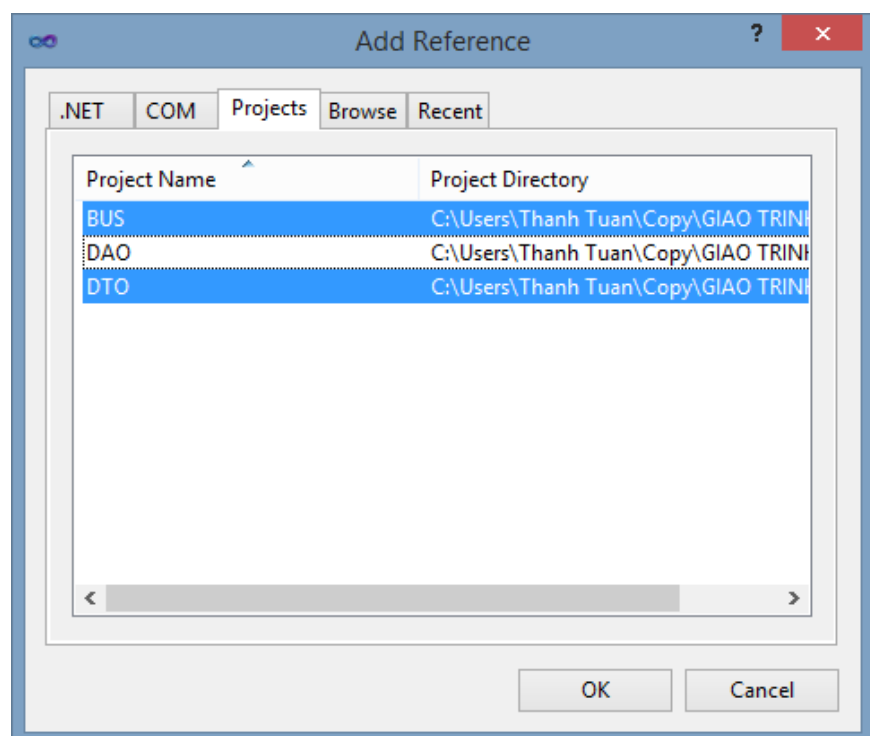
- BUS: Business Layer



- Add References cho các project với nhau:

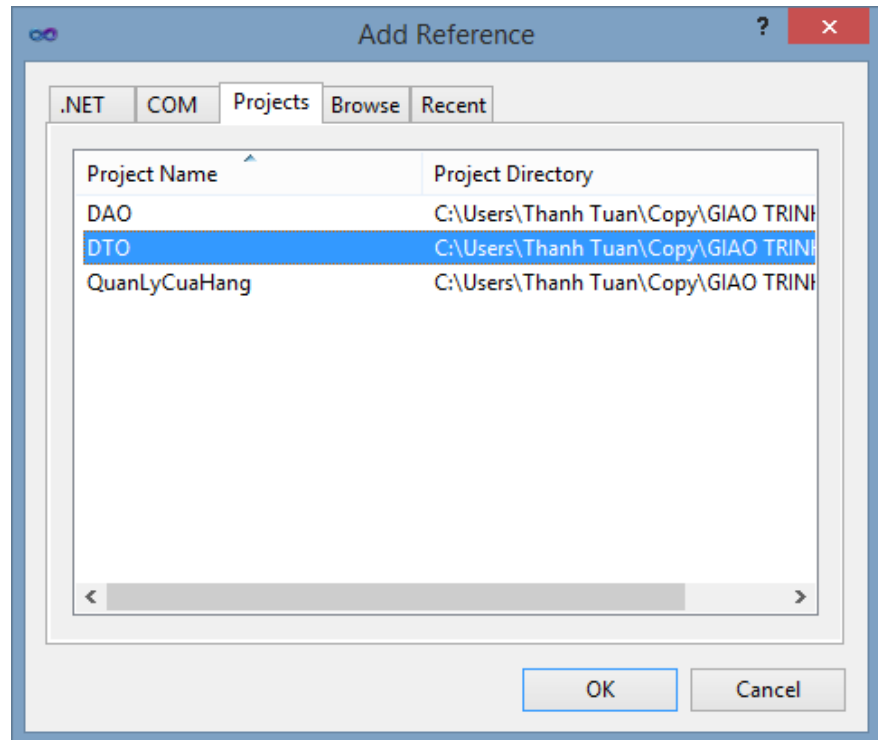
- QuanLyCuaHang (Presentation Layer)

Liên kết với BUS và DTO



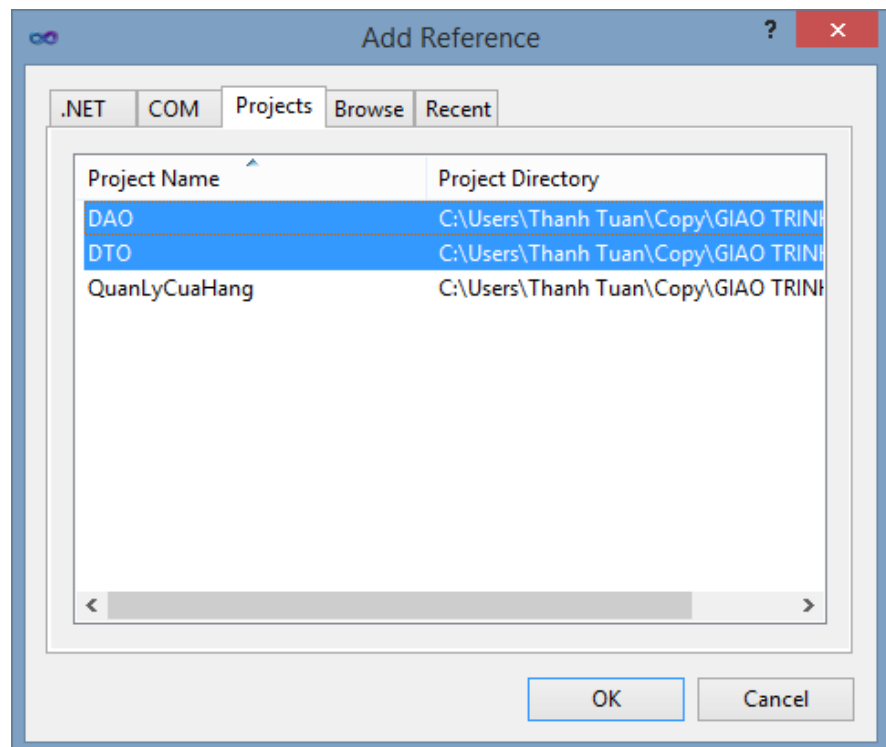
- DAO

Chỉ liên kết với DTO



- BUS

Liên kết với DAO và DTO



- Tạo các lớp trong các project:

- DTO

- Lớp LoaiSanPhamDTO

```
public class LoaiSanPhamDTO
{
    // Các thuộc tính
    public string MaLoaiSanPham { set; get; }
    public string TenLoaiSanPham { set; get; }
    public string LoaiSanPhamCha { set; get; }

    // Phương thức khởi tạo
    public LoaiSanPhamDTO()
    {
        MaLoaiSanPham = "";
        TenLoaiSanPham = "";
        LoaiSanPhamCha = "";
    }
}
```

- Lớp SanPhamDTO

```
public class SanPhamDTO
{
    // Các thuộc tính
    public int MaSanPham { set; get; }
    public string LoaiSanPham { set; get; }
    public string TenSanPham { set; get; }
    public int Gia { set; get; }
    public string ChiTiet { set; get; }

    // Phương thức khởi tạo
    public SanPhamDTO()
    {
        MaSanPham = 0;
        LoaiSanPham = "";
        TenSanPham = "";
        Gia = 0;
        ChiTiet = "";
    }
}
```

- DAO

- Lớp LoaiSanPhamDAO

```
public class LoaiSanPhamDAO
{
    // Các phương thức thao tác với CSDL
    // --Lấy danh sách loại sản phẩm
    public List<LoaiSanPhamDTO> LayDanhSach()
    {
        List<LoaiSanPhamDTO> lstLoaiSanPham
    }
}
```

```

        = new List<LoaiSanPhamDTO>();
// B.1: Tạo kết nối CSDL
OleDbConnection conn =
    DataProvider.TaoKetNoi();
// B.2: Tạo đối tượng try vấn CSDL
OleDbCommand com = new
    OleDbCommand();
com.CommandText = "SELECT * FROM
    LOAISANPHAM";
com.Connection = conn;
// B.3: Truy vấn CSDL và xử lý kết
// quả trả về
// -----Truy vấn
OleDbDataReader dr =
    com.ExecuteReader();
// -----Xử lý kết quả
while (dr.Read())
{
    LoaiSanPhamDTO loai = new
        LoaiSanPhamDTO();
    // Mã loại sản phẩm
    if (!dr.IsDBNull(0))
        loai.MaLoaiSanPham =
            (string)dr[0];
    // Tên loại sản phẩm
    if (!dr.IsDBNull(1))
        loai.TenLoaiSanPham =
            (string)dr[1];
    // Loại sản phẩm cha
    if (!dr.IsDBNull(2))
        loai.LoaiSanPhamCha =
            (string)dr[2];
    // Thêm loại sản phẩm vào
    // danh sách loại sản phẩm
    lstLoaiSanPham.Add(loai);
}
dr.Close();
// B.4: Đóng kết nối CSDL
conn.Close();
return lstLoaiSanPham;
}
}

```

- Lớp SanPhamDAO

```

public class SanPhamDAO
{
    // Các phương thức thao tác với CSDL
    // --Lấy danh sách sản phẩm
    // ----Nếu mã loại strMaLoai = "" ==> lấy
    // tất cả danh sách sản phẩm trong CSDL
    // ----Ngược lại lấy danh sách theo loại
    public List<SanPhamDTO>
        LayDanhSach(string strMaLoai = "")

```

```

{
    List<SanPhamDTO> lstSanPham = new
        List<SanPhamDTO>();
    // B.1: Tạo kết nối CSDL
    OleDbConnection conn =
        DataProvider.TaoKetNoi();
    // B.2: Tạo đối tượng try vấn CSDL
    OleDbCommand com = new
        OleDbCommand();
    com.CommandText = "SELECT * FROM
        SANPHAM";
    // --Nếu có mã loại thì thêm vào
    // điều kiện
    if (strMaLoai != "")
        com.CommandText +=
            string.Format(" WHERE
            LOAISANPHAM = '{0}'",
                strMaLoai);
    com.Connection = conn;

    // B.3: Truy vấn CSDL và xử lý kết
    // quả trả về
    // -----Truy vấn
    OleDbDataReader dr =
        com.ExecuteReader();
    // -----Xử lý kết quả
    while (dr.Read())
    {
        SanPhamDTO sp = new
            SanPhamDTO();
        // Mã sản phẩm
        if (!dr.IsDBNull(0))
            sp.MaSanPham = (int)dr[0];
        // Loại sản phẩm
        if (!dr.IsDBNull(1))
            sp.LoaiSanPham =
                (string)dr[1];
        // Tên sản phẩm
        if (!dr.IsDBNull(2))
            sp.TenSanPham =
                (string)dr[2];
        // Giá
        if (!dr.IsDBNull(3))
            sp.Gia = (int)dr[3];
        // Chi tiết
        if (!dr.IsDBNull(4))
            sp.ChiTiet =
                (string)dr[4];
        // Thêm sản phẩm vào danh sách
        // sản phẩm
        lstSanPham.Add(sp);
    }
    dr.Close();
}

```

```

        // B.4: Đóng kết nối CSDL
        conn.Close();
        return lstSanPham;
    }

    // --Thêm sách sản phẩm
    public void ThemSanPham(SanPhamDTO spObj)
    {
        // B.1: Tạo kết nối CSDL
        OleDbConnection conn =
            DataProvider.TaoKetNoi();
        // B.2: Tạo đối tượng try vắn CSDL
        // với parameter (?)
        OleDbCommand com = new
            OleDbCommand();
        com.CommandText = "INSERT INTO
            SANPHAM(LoaiSanPham,
            TenSanPham, Gia, ChiTiet)
            VALUES(?,?,?,?)";
        com.Connection = conn;
        //----Khai báo các parameter theo
        // đúng thứ tự
        com.Parameters.AddWithValue(
            @"loaisp", spObj.LoaiSanPham);
        com.Parameters.AddWithValue(
            @"tensp", spObj.TenSanPham);
        com.Parameters.AddWithValue(@"gia",
            spObj.Gia);
        com.Parameters.AddWithValue(
            @"chitiet", spObj.ChiTiet);
        // B.3: Thực thi thao tác với CSDL
        com.ExecuteNonQuery();
        // B.4: Đóng kết nối CSDL
        conn.Close();
    }

    // --Cập nhật sách sản phẩm
    public void CapNhatSanPham(SanPhamDTO
                                spObj)
    {
        // B.1: Tạo kết nối CSDL
        OleDbConnection conn =
            DataProvider.TaoKetNoi();
        // B.2: Tạo đối tượng try vắn CSDL
        // với parameter (?)
        OleDbCommand com = new
            OleDbCommand();
        com.CommandText = "UPDATE SANPHAM
        SET LoaiSanPham = ?, TenSanPham = ?,
            Gia = ?, ChiTiet = ?
            WHERE MASANPHAM = ?";
        com.Connection = conn;
        //----Khai báo các parameter theo
    }

```

```

        // đúng thứ tự
        com.Parameters.AddWithValue(
            @"loaisp", spObj.LoaiSanPham);
        com.Parameters.AddWithValue(
            @"tensp", spObj.TenSanPham);
        com.Parameters.AddWithValue(@"gia",
            spObj.Gia);
        com.Parameters.AddWithValue(
            @"chitiet", spObj.ChiTiet);
        com.Parameters.AddWithValue(
            @"masp", spObj.MaSanPham);
        // B.3: Thực thi thao tác với CSDL
        com.ExecuteNonQuery();
        // B.4: Đóng kết nối CSDL
        conn.Close();
    }
    // --Xóa sách sản phẩm
    public void XoaSanPham(int masp)
    {
        // B.1: Tạo kết nối CSDL
        OleDbConnection conn =
            DataProvider.TaoKetNoi();
        // B.2: Tạo đối tượng try vấn CSDL
        OleDbCommand com = new
            OleDbCommand();
        com.CommandText =
            string.Format("DELETE FROM
                SANPHAM WHERE MASANPHAM =
                    {0}", masp);
        com.Connection = conn;
        // B.3: Thực thi thao tác với CSDL
        com.ExecuteNonQuery();
        // B.4: Đóng kết nối CSDL
        conn.Close();
    }
}

```

- BUS

- Lớp LoaiSanPhamBUS

```

public class LoaiSanPhamBUS
{
    // Lấy danh sách loại sản phẩm
    public List<LoaiSanPhamDTO> LayDanhSach()
    {
        LoaiSanPhamDAO lspDao = new
            LoaiSanPhamDAO();
        return lspDao.LayDanhSach();
    }
}

```

- Lớp SanPhamBUS

```

public class SanPhamBUS
{
    // Lấy danh sách sản phẩm
    public List<SanPhamDTO>
        LayDanhSach(string strMaLoai = "")
    {
        SanPhamDAO spDao = new
                                SanPhamDAO();
        return
            spDao.LayDanhSach(strMaLoai);
    }

    // --Thêm sách sản phẩm
    public void ThemSanPham(SanPhamDTO
                                spObj)
    {
        // Kiểm tra dữ liệu đầu vào
        if(spObj.LoaiSanPham == "")
            throw new Exception("Chưa nhập
                                loại sản phẩm");
        if (spObj.TenSanPham == "")
            throw new Exception("Chưa nhập
                                tên sản phẩm");
        if (spObj.Gia <= 0)
            throw new Exception("Chưa nhập
                                giá sản phẩm");
        if (spObj.ChiTiet == "")
            throw new Exception("Chưa nhập
                                mô tả sản phẩm");

        //
        SanPhamDAO spDao = new
                                SanPhamDAO();
        spDao.ThemSanPham(spObj);
    }

    // --Cập nhật sách sản phẩm
    public void CapNhatSanPham(SanPhamDTO
                                spObj)
    {
        // Kiểm tra dữ liệu đầu vào
        if (spObj.LoaiSanPham == "")
            throw new Exception("Chưa nhập
                                loại sản phẩm");
        if (spObj.TenSanPham == "")
            throw new Exception("Chưa nhập
                                tên sản phẩm");
        if (spObj.Gia <= 0)
            throw new Exception("Chưa nhập
                                giá sản phẩm");
        if (spObj.ChiTiet == "")
            throw new Exception("Chưa nhập
                                mô tả sản phẩm");
    }
}

```

```

        //
        SanPhamDAO spDao = new SanPhamDAO();
        spDao.CapNhatSanPham(spObj);
    }

    // --Xóa sách sản phẩm
    public void XoaSanPham(int masp)
    {
        // Kiểm tra dữ liệu đầu vào
        if (masp <= 0)
            throw new Exception("Không tồn tại sản phẩm");

        //
        SanPhamDAO spDao = new SanPhamDAO();
        spDao.XoaSanPham(masp);
    }
}

```

- Xử lý sự kiện Load của Form:
 - Đưa danh sách đối tượng Loại sản phẩm vào TreeView treLoaiSanPham và Combobox cboLoaiSanPham

```

// Load danh sách loại sản phẩm
LoaiSanPhamBUS lspBus = new LoaiSanPhamBUS();
List<LoaiSanPhamDTO> lstLoaiSanPham =
    lspBus.LayDanhSach();
// Đưa danh sách lên TreeView treLoaiSanPham
foreach (LoaiSanPhamDTO lspDto in lstLoaiSanPham)
{
    // Tạo Node
    TreeNode node = new
        TreeNode(lspDto.TenLoaiSanPham);
    node.Tag = lspDto.MaLoaiSanPham;

    // Tìm và thêm vào node có mã loại sản phẩm
    // (Tag) giống mã loại sản phẩm cha của đối
    // tượng
    foreach (TreeNode tn in treLoaiSanPham.Nodes)
    {
        if (tn.Tag.ToString() ==
            lspDto.LoaiSanPhamCha)
        {
            tn.Nodes.Add(node);
            break;
        }
    }
}
// Đưa danh sách lên Combobox cboLoaiSanPham
cboLoaiSanPham.DataSource = lstLoaiSanPham;
cboLoaiSanPham.DisplayMember = "TenLoaiSanPham";

```



```
cboLoaiSanPham.ValueMember = "MaLoaiSanPham";
```

- Đưa danh sách tất cả sản phẩm trong CSDL vào ListView lvwSanPham

```
// Lấy danh sách tất cả sản phẩm trong CSDL
SanPhamBUS spBus = new SanPhamBUS();
List<SanPhamDTO> lstSanPham = spBus.LayDanhSach();
// Đưa danh sách sản phẩm lên ListView lvwSanPham
foreach (SanPham spDto in lstSanPham)
{
    // Tạo đối tượng ListViewItem
    ListViewItem lvi = new ListViewItem(spDto.TenSanPham);
    // Thêm các subitem
    lvi.SubItems.Add(spDto.Gia.ToString());
    lvi.SubItems.Add(spDto.ChiTiet);
    lvi.SubItems.Add(spDto.MaSanPham.ToString());
    // Gán Loại Sản phẩm vào Tag của ListViewItem
    lvi.Tag = spDto.LoaiSanPham;
    // Thêm đối tượng ListViewItem vào ListView
    lvwSanPham.Items.Add(lvi);
}
```

- Xử lý sự kiện AfterSelect của TreeView treLoaiSanPham khi chọn nhấn chọn Loại sản phẩm để hiển thị danh sách Sản phẩm trên ListView lvwDSSanPham tương ứng với loại được chọn

```
if (treLoaiSanPham.SelectedNode.Tag != null)
{
    List<SanPhamDTO> lstSanPham = new List<SanPhamDTO>();
    SanPhamBUS spBus = new SanPhamBUS();
    // Xoá hết sản phẩm ở lvwSanPham
    lvwSanPham.Items.Clear();

    if (treLoaiSanPham.SelectedNode.Tag.ToString() == "tc")
    {
        // Lấy danh sách tất cả sản phẩm
        lstSanPham = spBus.LayDanhSach();
    }
    else
    {
        // Lấy danh sách sản phẩm theo mã loại sản phẩm (Tag)
        lstSanPham = spBus.LayDanhSach(treLoaiSanPham.SelectedNode.Tag.ToString());
    }

    // Đưa danh sách sản phẩm lên ListView lvwSanPham
}
```

```

foreach (SanPham spDto in lstSanPham)
{
    // Tạo đối tượng ListViewItem
    ListViewItem lvi = new
        ListViewItem(spDto.TenSanPham);
    // Thêm các subitem
    lvi.SubItems.Add(spDto.Gia.ToString());
    lvi.SubItems.Add(spDto.ChiTiet);
    lvi.SubItems.Add(spDto.MaSanPham.ToString());
    // Gán Loại Sản phẩm vào Tag của ListViewItem
    lvi.Tag = spDto.LoaiSanPham;
    // Thêm đối tượng ListViewItem vào ListView
    lvwSanPham.Items.Add(lvi);
}
}

```

- Xử lý thêm sản phẩm mới

```

// Tạo đối tượng sản phẩm
SanPhamDTO spDto = new SanPhamDTO();
spDto.LoaiSanPham =
    cboLoaiSanPham.SelectedValue.ToString();
spDto.TenSanPham = txtTenMon.Text;
spDto.Gia = int.Parse(txtGia.Text);
spDto.ChiTiet = txtChiTiet.Text;

// Thêm vào bảng SANPHAM trong CSDL
SanPhamBUS spBus = new SanPhamBUS();
spBus.ThemSanPham(spDto);

// Thêm sản phẩm mới vào listview
ListViewItem lvi = new ListViewItem(txtTenMon.Text);
lvi.SubItems.Add(txtGia.Text);
lvi.SubItems.Add(txtChiTiet.Text);
lvi.SubItems.Add(spDto.MaSanPham.ToString());
lvi.Tag = cboLoaiSanPham.SelectedValue.ToString();
lvwSanPham.Items.Add(lvi);

```

- Xử lý cập nhật sản phẩm

```

// Cập nhật thông tin sản phẩm được chọn trên listview
ListViewItem lvi = lvwSanPham.SelectedItems[0];
lvi.SubItems[0] = txtTenMon.Text;
lvi.SubItems[1].Text = txtGia.Text;
lvi.SubItems[2].Text = txtChiTiet.Text;
lvi.Tag = cboLoaiSanPham.SelectedValue.ToString();

// Tạo đối tượng sản phẩm
SanPhamDTO spDto = new SanPhamDTO();
spDto.MaSanPham = int.Parse(lvi.SubItems[3].Text);
spDto.LoaiSanPham =
    cboLoaiSanPham.SelectedValue.ToString();
spDto.TenSanPham = txtTenMon.Text;
spDto.Gia = int.Parse(txtGia.Text);
spDto.ChiTiet = txtChiTiet.Text;

```

```
// Cập nhật vào bảng SANPHAM trong CSDL
SanPhamBUS spBus = new SanPhamBUS();
spBus. CapNhatSanPham (spDto);
```

- Xử lý xóa sản phẩm

```
// Lấy ra sản phẩm được chọn
ListViewItem lvi = lvwSanPham.SelectedItems[0];

// Xóa sản phẩm khỏi bảng SANPHAM trong CSDL
int iMaSP = int.Parse(lvi.SubItems[3].Text);
SanPhamBUS spBus = new SanPhamBUS();
spBus.XoaSanPham(iMaSP);

// Xóa sản phẩm khỏi ListView
lvwSanPham.Items.Remove(lvi);
```

PHỤ LỤC : DANH SÁCH CÁCH ĐẶT TÊN GỢI Ý CHO CÁC ĐIỀU KHIỂN

Kiểu điều khiển	Tiền tố	Ví dụ
ADO Data	ado	adoMaSinhVien
Check box	chk	chkKiemTra
Combo box, drop-down list box	cbo	cboLopHoc
Command button	cmd	cmdThoat
Common dialog	dlg	dlgFileOpen
Data	dat	datMaSinhVien
Data-bound combo box	dbcbo	dbcboNgonNgu
Data-bound grid	dbgrd	dbgrdKetQuaTruyVan
Data-bound list box	dblst	dblstLoaiCongViec
Data combo	dbc	dbcTacGia
Data grid	dgd	dgdTuaDe
Data list	dbl	dblNhaXuatBan
Data repeater	drp	drpDiaDiem
Date picker	dtp	dtpNgayPhatHanh
Directory list box	dir	dirNguon
File list box	fil	filNguon
Form	frm	frmLienLac
Frame	fra	fraNgonNgu
Graph	gra	graDoanhThu
Grid	grd	grdGiaSanPham
Horizontal scroll bar	hsb	hsbAmThanh
Image	img	imgAnhDaiDien
Image combo	imgcbo	imgcboSanPham
ImageList	ils	ilsTatCaHinhAnh
Label	lbl	lblThongDiep
List box	lst	lstSanPham
ListView	lvw	
Menu	mnu	mnuFileOpen
Month view	mvw	mvwThang

OLE container	ole	oleBangTinh
Picture box	pic	picAnhDaiDien
ProgressBar	prg	prgTaiTapTin
RichTextBox	rtf	rtfBaoCao
StatusBar	sta	staNgayThang
Text box	txt	txtHoTen
Timer	tmr	tmrThongBao
Toolbar	tlb	tlbDanhMuc
TreeView	tre	treThuMuc
Vertical scroll bar	vsb	vsbTiLe

TÀI LIỆU THAM KHẢO

❖ MSDN Online: <http://msdn.microsoft.com>

