

HƯỚNG DẪN TẠO MODULE ĐƠN GIẢN TRONG DNN 4.9

Người viết: **Lê Văn Quân**

Email: levanquanna@yahoo.com, nguyendansoft@gmail.com

Mobile phone: 01689 227 100

1. Mở đầu

Tài liệu dành cho các bạn bắt đầu học viết module.

Đọc tài liệu này sau khi đã tìm hiểu về kiến trúc DNN, kiến trúc của Module DNN.

Để hiểu rõ cách viết Module, ở đây ta đưa ra một bài toán cụ thể trong thực tế, ví dụ: Viết Module cho phép nhập và hiển thị các danh mục(chẳng hạn như danh mục sản phẩm 1 cấp: Máy tính, Điện thoại, Mỹ phẩm, ...)

2. Chuẩn bị

Phải cài đặt trước:

- **DNN 4.9**
- **Visual Studio 2005** hoặc **Microsoft Visual Web Developer 2008 Express Edition**
- Gói **DotNetNuke_Community_04.09.04_StarterKit.vsi**

3. Bắt đầu viết

a. Phân tích yêu cầu bài toán

Theo như yêu cầu bài toán thì có 2 chức năng chính là:

- Nhập danh mục
- Hiển thị danh mục.

Như vậy ta cần thiết kế 2 chức năng phục vụ cho các yêu cầu trên.

b. Tạo bảng và các thủ tục liên quan

- Tạo bảng với các trường như sau:

	Column Name	Data Type	Allow Nulls
►	Id	int	<input type="checkbox"/>
	Name	nvarchar(200)	<input checked="" type="checkbox"/>
			<input type="checkbox"/>

Đặt tên là: **BeginModule_List**

- Các thủ tục:

+ Thủ tục thêm danh mục:

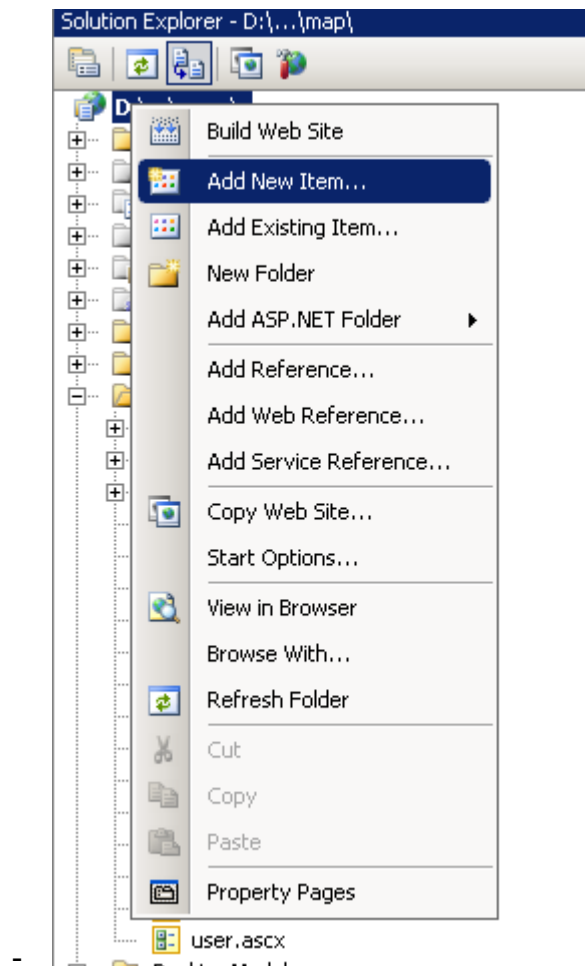
```
ALTER PROCEDURE BeginModule_List_AddItem
(
    @Name nvarchar(200)
)
AS
insert into BeginModule_List([Name]) values(@Name)
RETURN
```

+ Thủ tục lấy tất cả các danh mục

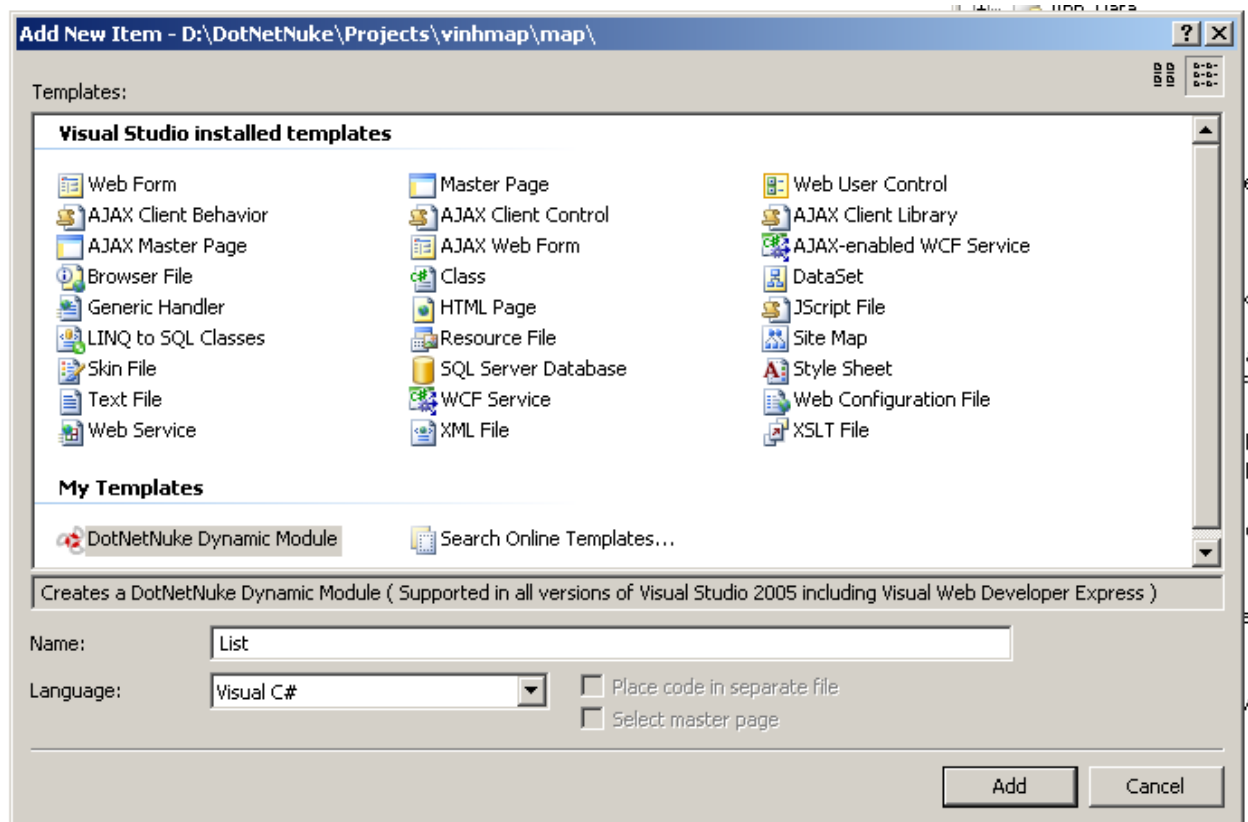
```
ALTER PROCEDURE BeginModule_List_GetItems
AS
select * from BeginModule_List
RETURN
```

c. Thêm module mới

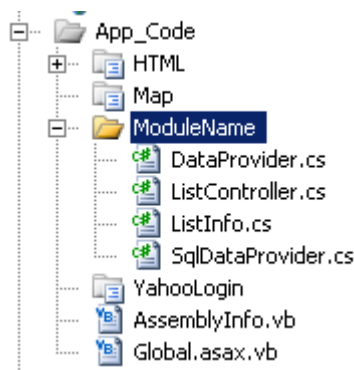
- Mở Porject bằng Visual Studio 2005 hoặc Microsoft Visual Web Developer 2008 Express Edition
- Trong cửa sổ Solution Explorer nhấp chuột phải vào project như sau:

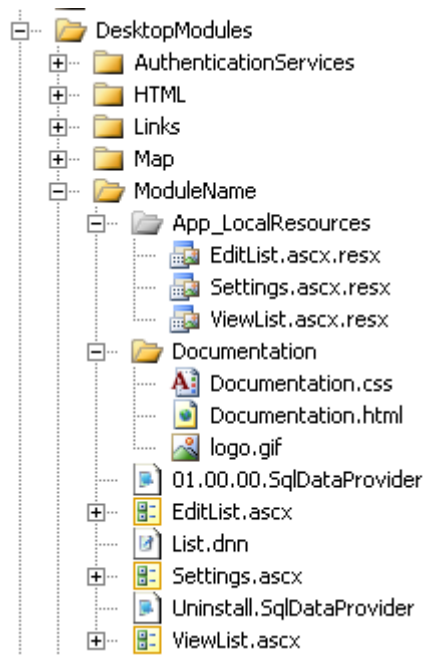


- Chọn **Add New Item** xuất hiện hộp thoại, nhập các thông tin như sau:



- Nhấn Add để thêm mới Module
- Sau khi thêm module, xuất hiện hai thư mục mới trong cây thư mục project như sau:



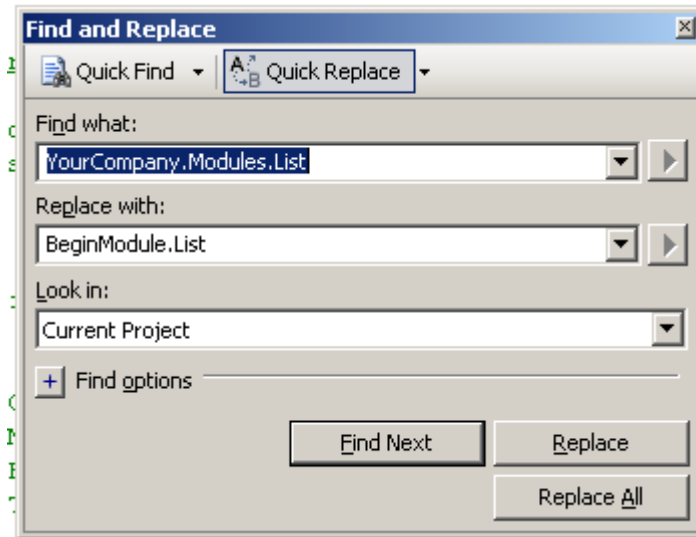


- Tiến hành đổi tên 2 thư mục này, ví dụ: **BeginModule.List**
- Khai báo thư mục chứa các class của module bằng cách edit trong file **web.config** như sau:

```
<codeSubDirectories>  
  <add directoryName="HTML" />  
  <add directoryName="Map" />  
  <add directoryName="YahooLogin" />  
  <add directoryName="BeginModule.List" />  
</codeSubDirectories>
```

- Đổi **namespace** trong các class của module theo tùy ý, ví dụ: **BeginModule.List**

Mẹo:



Nhấn **Replace All** để thay thế tất cả một cách nhanh chóng.

d. **Định nghĩa các thuộc tính và phương thức giao tiếp với Data Base**

- Định nghĩa các thuộc tính trong class **ListInfo** như sau:

```

using System;
using System.Configuration;
using System.Data;

namespace BeginModule.List
{
    public class ListInfo
    {
        #region Private Members

        private int _Id;
        private string _Name;

        #endregion

        Constructors

        #region Public Properties

        public int Id { get { return _Id; } set { _Id = value; } }
        public string Name { get { return _Name; } set { _Name = value; } }

        #endregion

    }
}

```

- Tiếp đến ta cần định nghĩa 3 loại phương thức gồm:

+ Các phương thức trừu tượng: Được viết trong class **DataProvider**

```

#region Abstract methods

    public abstract void AddItem(string Name);
    public abstract IDataReader GetItems();

#endregion

```

+ Định nghĩa các phương thức thực thi: Được viết trong class **SqlDataProvider**

```
#region Public Methods

    public override void AddItem(string Name)
    {
        SqlHelper.ExecuteNonQuery(ConnectionString, GetFullyQualifiedName("BeginModule_List_AddItem"), Name);
    }
    public override IEnumerable<IDataReader> GetItems()
    {
        return (IEnumerable<IDataReader>)SqlHelper.ExecuteReader(ConnectionString, GetFullyQualifiedName("BeginModule_List_GetItems"));
    }

#endregion
```

Lưu ý tại class này có hằng số *ModuleQualifier*="YourCompany":

```
public class SqlDataProvider : DataProvider
{

    #region Private Members

        private const string ProviderType = "data";
        private const string ModuleQualifier = "YourCompany_";
```

Gán nó thành một chuỗi rỗng:

```
public class SqlDataProvider : DataProvider
{

    #region Private Members

        private const string ProviderType = "data";
        private const string ModuleQualifier = "";
```

+ Định nghĩa các phương thức giao tiếp: Được viết trong class **ListController**

```
public void AddItem(ListInfo l)
{
    SqlDataProvider.Instance().AddItem(l.Name);
}
public List<ListInfo> GetItems()
{
    return CBO.FillCollection<ListInfo>(SqlDataProvider.Instance().GetItems());
}
```

e. Thiết kế giao diện người dùng và sử dụng các phương thức đã định nghĩa

Ta sẽ tiến hành thiết kế một màn hình hiển thị tất cả các danh mục, trên đó có nút **[Thêm danh mục]** để cho phép người dùng nhấn vào đây thì xuất hiện màn hình nhập danh mục. Màn hình hiển thị danh mục như sau:

DANH SÁCH CÁC DANH MỤC
Databound
Databound
Databound
Databound
Databound

Thêm danh mục

```
<asp:GridView ID="grvList" runat="server" AutoGenerateColumns="False"
Width="300px">
  <Columns>
    <asp:TemplateField HeaderText="DANH SÁCH CÁC DANH MỤC">
      <ItemTemplate>
        <p>
          <%#Eval("Name") %>
        </p>
      </ItemTemplate>
    </asp:TemplateField>
  </Columns>
</asp:GridView>
<br />
<asp:Button ID="btnAddNewList" runat="server" Text="Thêm danh mục" />
```

+ Khai báo mới một đối tượng điều khiển giao tiếp:

```
#region Private Members
private ListController lstCtr = new ListController();
#endregion
```

+ Lấy danh mục lên ngay khi trang được tải lên:

```
protected void Page_Load(System.Object sender, System.EventArgs e)
{
    try
    {
        if (!Page.IsPostBack)
        {
            grvList.DataSource = lstCtr.GetItems();
            grvList.DataBind();
        }
    }
    catch (Exception exc) //Module failed to load
    {
        Exceptions.ProcessModuleLoadException(this, exc);
    }
}
```

+ Xử lý sự kiện click vào nút **[Thêm danh mục]** thì chuyển đến màn hình nhập danh mục:

```
protected void btnAddNewList_Click(object sender, EventArgs e)
{
    Response.Redirect(DotNetNuke.Common.Globals.NavigateURL("EditList", "mid", this.ModuleId.ToString()));
}
```

Đề ý tại chỗ khoanh tròn màu đỏ: Đây là Key nhận biết điều khiển trình bày màn hình nhập danh mục, Key này được đăng ký khi định nghĩa module(sẽ nói đến trong phần sau còn bây giờ thì cứ coi như ta đã hiểu về Key này), các thông số khác là bắt buộc.

- Xử lý chức năng nhập danh mục

+ Màn hình nhập liệu:

```

<table style="width:100%;">
    <tr>
        <td class="style1">
            Tên danh mục:</td>
        <td>
            <asp:TextBox ID="txtName" runat="server" Width="190px"></asp:TextBox>
            <asp:RequiredFieldValidator ID="RequiredFieldValidator1" runat="server"
                ControlToValidate="txtName" ErrorMessage="*"></asp:RequiredFieldValidator>
        </td>
    </tr>
    <tr>
        <td class="style1">
            &nbsp;  </td>
        <td>
            <asp:Button ID="btnSave" runat="server" Text="Ghi lại" />
            &nbsp; <asp:Button ID="btnReturn" runat="server" Text="Quay lại"
                CausesValidation="False" />
        </td>
    </tr>
</table>

```

+ Xử lý nút **[Ghi lại]**:

```

protected void btnSave_Click(object sender, EventArgs e)
{
    //Khai báo một đối tượng điều khiển giao tiếp mới
    ListController lstCtr = new ListController();

    //Khai báo một đối tượng danh mục mới
    ListInfo l = new ListInfo();

    //Gán giá trị cho thuộc tính đối tượng danh mục
    l.Name = txtName.Text.Trim();

    //Thêm danh mục
    lstCtr.AddItem(l);
}

```

+ Xử lý nút **[Quay lại]**:

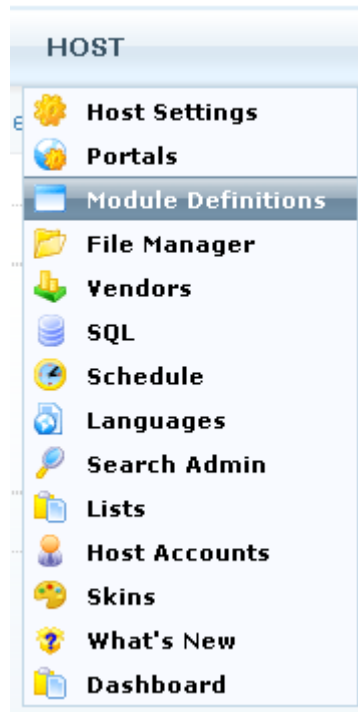
```

protected void btnReturn_Click(object sender, EventArgs e)
{
    Response.Redirect(DotNetNuke.Common.Globals.NavigateURL(this.TabId));
}

```

f. Định nghĩa module(đăng ký module với hệ thống)

- Vào **Host/Module Definitions**



Xuất hiện danh sách các module đã có trước đó:

Module Definitions

Install New Module

Create Module Definition

Import Module Definition

Help

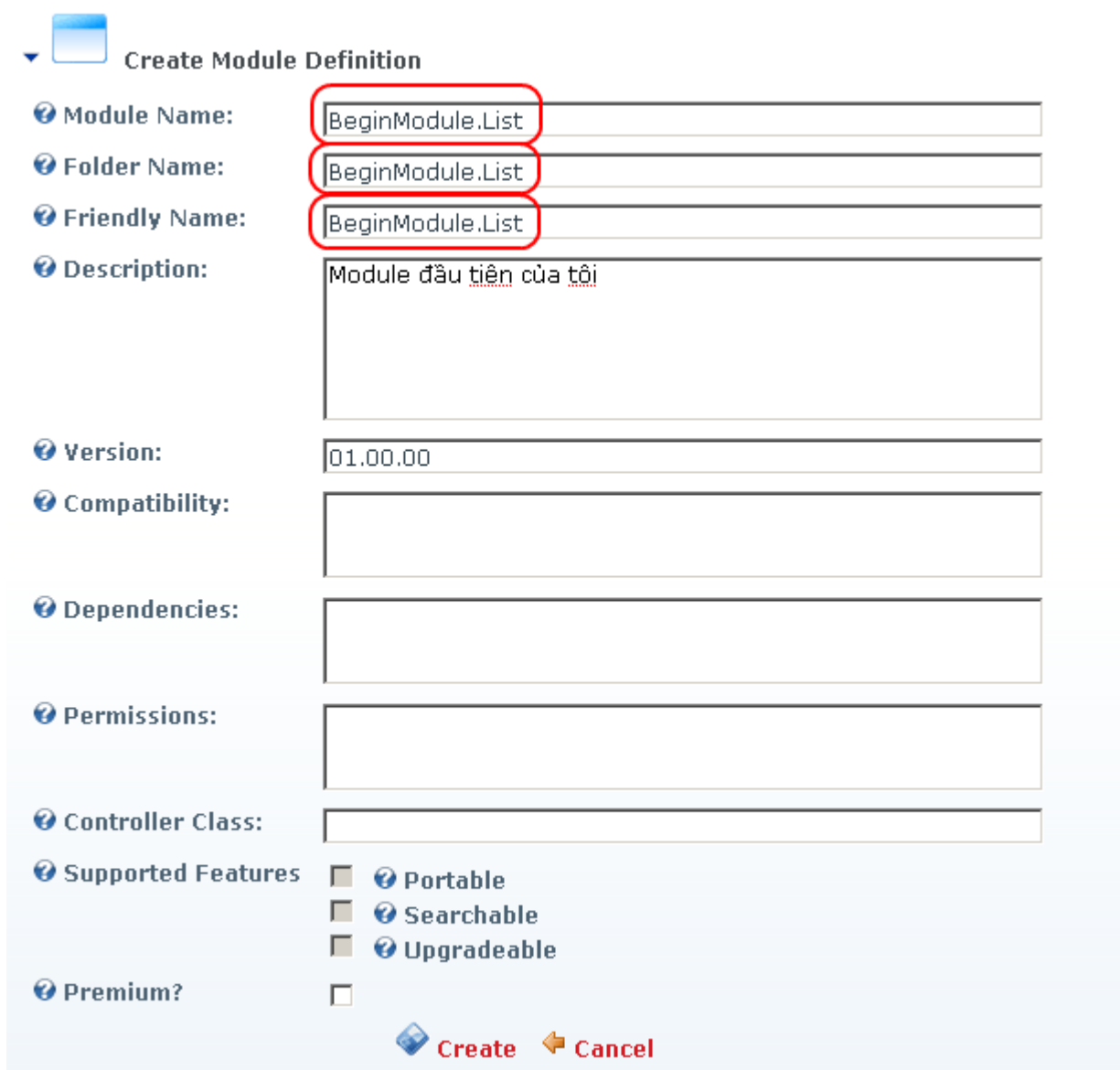
Online Help

Print

Locales: <Not Specified>

	Description	Version
	DotNetNuke Community Edition	04.09.04
[Skin Objects]	Skin Objects are User Controls which can be used to provide custom functionality to your Skin files.	
Account Login	Allows users to login to the portal.	01.00.00
Banners	Banner advertising is managed through the Vendors module in the Admin tab. You can select the number of banners to display as well as the banner type.	01.00.00
Feed Explorer	Allows users to browse RSS feeds using a tabbed user interface	01.00.00
Links	This module renders a list of hyperlinks. Links includes an edit page, which allows authorized users to edit the Links data stored in the SQL database.	04.00.01
Map	Map	01.00.00
Search Input	The Search Input module provides the ability to submit a search to a given search results module.	01.00.00
Search Results	The Search Results module provides the ability to display search results.	01.00.00
Text/HTML	This module renders a block of HTML or Text content. The Html/Text module allows authorized users to edit the content either inline or in a separate administration page. The content is stored in the database. According to module settings, tokens can be used, that get replaced during display.	04.08.01
User Account	Allows users to register and manage their account.	01.00.00
YahooLogin	YahooLogin	07.00.00

Chọn **Create Module Definition**, xuất hiện màn hình định nghĩa module, nhập các thông tin như sau:



Create Module Definition

Module Name: BeginModule.List

Folder Name: BeginModule.List

Friendly Name: BeginModule.List

Description: Module đầu tiên của tôi

Version: 01.00.00

Compatibility:

Dependencies:



Permissions:

Controller Class:


Supported Features

- ☐ Portable
- ☐ Searchable
- ☐ Upgradeable

Premium? ☐

 **Create**  **Cancel**

Đề ý các trường khoanh màu đỏ là bắt buộc, nhấn **Create** để tạo module, ta lại có:



Edit Module Definition

Module Name:

BeginModule.List

Folder Name:

BeginModule.List

Friendly Name:

BeginModule.List

Description:

Module đầu tiên của tôi

Version:

01.00.00

Compatibility:

Dependencies:

Permissions:

Controller Class:

Supported Features


☐ Portable


☐ Searchable


☐ Upgradeable

Premium?

☐


 Update

 Cancel

 Uninstall

☐ Delete Files?

New Definition:

 Add Definition

Để ý thấy lúc này xuất hiện thêm phần khoanh màu đỏ, tại khung này có thể thêm nhiều định nghĩa cho module bằng cách nhập tên định nghĩa vào **New Definition** và nhấn **Add Definition**, ví dụ nhập tên định nghĩa là: **BeginModule.List** ta có:

Definitions: ✖ Delete Definition

New Definition: ✚ Add Definition

Default Cache Time: 🔧 Update Cache Properties

Control	Title	Source
✚ Add Control		

Để ý thấy xuất hiện thêm phần khoanh màu đỏ, tại đây ta có thể thêm các điều khiển cho định nghĩa module. Nhấn vào **Add Control** để chuyển sang màn hình thêm điều khiển:

Edit Module Control

Module:

Definition:

Key:

Title:

Source:

Type:

View Order:

Icon:

Help URL:

Supports Partial Rendering? ☐

Update Cancel Delete

Để ý các trường khoanh màu đỏ là bắt buộc

Lưu ý: Mỗi định nghĩa module có 1 hoặc nhiều điều khiển(Control), cụ thể như sau:

. Key: Là thông tin đại diện cho 1 điều khiển(có thể hiểu như là bí danh)

.Source: Nguồn chỉ định file vật lý của điều khiển

.Type: Là định nghĩa nhằm mục đích phân quyền cho người dùng, bao gồm:

+ *Skin Object*: Nếu chọn kiểu này thì điều khiển đó được coi như là 1 đối tượng phục vụ cho việc thiết kế Skin

+ *Anonymouse*: Cho phép tất cả mọi người dùng đều truy cập được vào điều khiển này (Tức là có thể xem được nội dung trên điều khiển này mà không bị hạn chế)

+ *View*: Mỗi 1 định nghĩa module bắt buộc phải có và suy nhất 1 điều khiển có *Type=View*, loại điều khiển này xuất hiện đầu tiên khi truy cập vào module (Có thể hiểu nó là bộ mặt của module), một điều cần lưu ý: **nếu điều khiển *Type=View* thì không cần nhập thuộc tính Key**

+ *Edit*: Điều khiển loại này thì chỉ có những member được phân quyền hoặc các member cao hơn như Admin và Host mới truy cập được

+ *Admin*: Chỉ có những member thuộc nhóm Admin hoặc Host mới có thể truy cập được điều khiển loại này

+ *Host*: Chỉ có những member là Super User (Quản trị host) mới truy cập được điều khiển loại này

Trong phạm vi module này ta có 2 điều khiển, ta định nghĩa như sau:

. Điều khiển phục vụ chức năng hiển thị danh mục:

▼ Edit Module Control

Module:	BeginModule.List
Definition:	BeginModule.List
Key:	
Title:	
Source:	DesktopModules/BeginModule.List/ViewList.ascx
Type:	View
View Order:	
Icon:	<Not Specified>
Help URL:	
Supports Partial Rendering?	<input type="checkbox"/>

Update Cancel Delete

Tại điều khiển này ta không nhập **Key** vì ta chọn **Type=View**

Source là đường dẫn chứa file vật lý xử lý chức năng hiển thị danh mục

Nhấn **Update** để cập nhật

.Điều khiển phục vụ chức năng nhập danh mục:

▼ Edit Module Control

Module: BeginModule.List

Definition: BeginModule.List

Key: EditList

Title:

Source: DesktopModules/BeginModule.List/EditList.ascx

Type: Edit

View Order:

Icon: <Not Specified>

Help URL:

Supports Partial Rendering? ☐

Update Cancel Delete

Nhập **Key**=**"EditList"** => Giá trị của Key là đặt theo tùy ý.

Source là đường dẫn chứa file vật lý xử lý chức năng nhập danh mục

Ta chọn **Type**=**Edit** nhằm mục đích chỉ có những member được phân quyền hoặc các member cao hơn như Admin và Host mới được truy cập điều khiển này(cụ thể là nhập danh mục)

Nhấn **Update** để cập nhật

Đến đây coi như quá trình tạo module kết thúc, ta có thể sử dụng module này.

Đây mới chỉ là một module ở mức cơ bản dành cho các bạn mới bắt đầu học viết module