

# C# and Visual Elements Naming Guidelines

Version 1.3 official adoption date: 9/17/2008

- A consistent naming pattern is one of the most important elements of predictability and discoverability in a managed class library. Widespread use and understanding of these naming guidelines should eliminate unclear code and make it easier for developers to understand shared code.
- C# and Visual Elements Naming guidelines is created to support the BITAC Charter – IT Architecture Operating Principles – Applications Architecture – Item 6: Consistent and Accurate Definitions.

## **SECTION I: C# Naming Guidelines:**

Category	Naming Guidelines	Example
Namespaces to reference as DLL or project name.	<p>It is recommended to prefix the namespace with WA.ECY follow by Program (sub-agency) abbreviation and application abbreviation such as WA.ECY.TCP.MYEIM.DataAccess. However it is also an option to use the following convention to reduce typing and to reflect the practical usage.</p> <ul style="list-style-type: none"> <li>- For a namespace shared outside the agency, the convention is as mentioned above.</li> <li>- For a namespace shared within the agency and other Programs (sub-agencies), it can be prefixed with ECY such as ECY.ADS.DataLoader.Import.</li> <li>- For a namespace shared within the Program (sub-agency), it can be prefixed with Program abbreviation such as WR.WRAT.Logging.</li> <li>- For a namespace used within the application, it can be prefix with application name such as: MyApp.Utilities.</li> </ul>	<pre> WA.ECY.ADS.EIM.Import.DataSets ECY.ADS.EIM.Editor.Utilities TCP.MYEIM.Utilities MyApp.Print MyApp.Business.Employer                     </pre>

<p><b>Note:</b> Do not use the same name for a namespace and a class. A namespace name does not have to parallel an assembly name.</p>		
Assemblies	If the assembly contains a single name space, or has an entire self-contained root namespace, name the assembly the same name as the namespace.	AppliedIS.TimeCard.BusinessRules.dll IrritatedVowel.Controllers.dll
Classes	Pascal Case, no underscores or leading "C" or "cls". Try to avoid abbreviations, and try to always use nouns.	InstanceHandler MethodListDataSet (typed dataset) ApplicationException(derived class) RequestHandlerDelegate(Delegate class)
<p><b>Note:</b> Classes should not have the same name as the namespace in which they reside. Derived classes - where appropriate, use a compound word to name a derived class. Classes may begin with an "I" only if the letter following the "I" is not capitalized; otherwise it looks like an Interface.</p>		
Interfaces	Follow class naming conventions, but start the name with "I" and capitalize the letter following the I	IServiceProvider IComponent
Struct	Pascal Case. Try to avoid abbreviations, and try to always use nouns.	Book Employee
Attributes	Follow class naming conventions, but add Attribute to the end of the name	ObsoleteAttribute
Enumerations	Follow class naming conventions. Do not use an "Enum" suffix on Enum type names. Use a singular name for most Enum types, but use a plural name for Enum types that are bit fields.	SearchOptions (bitwise flags) AcceptRejectRule (normal enum)
Static Fields	Follow class naming conventions. It is recommended that you use static properties instead of public static fields whenever possible.	static string CityName

Parameters	<p>Camel Case, no Hungarian Notation. Use descriptive names. It should be obvious what the parameter does by its name. Do not use reserved parameters. Reserved parameters are private parameters that might be exposed in a future version if they are needed. Instead, if more data is needed in a future version of your class library, add a new overload for a method</p>	<pre>Connection.Open(string connectionString)</pre>
Methods	<p>Pascal Case, use descriptive verb or phrase.</p>	<pre>public void RemoveAll() private string ParseEmailAddress(string emailAddress)</pre>
Properties	<p>Follow class naming conventions. Consider creating a property with the same name as its underlying type.</p>	<pre>Public Color BackColor {     get{...}     set{...} }</pre>
Events	<p>Pascal Case, use an EventHandler suffix on event handler names. The EventHandler has two parameters: sender and e. Name an event argument class with the EventArgs suffix, and use verbs to describe the event that is, or has occurred. Do not use a prefix or suffix on the event declaration on the type: Close, not OnClose. In general, you should provide a protected method called OnXxx on types with events that can be overridden in a derived class.</p>	<pre>Public Delegate void RowEventHandler(object sender, e RowEventArgs);  public class RowEventArgs: EventArgs {     DataRow dr;      public RowEventArgs(DataRow row)     {         this.dr = row     }      public DataRow CurrentRow     {         get {return dr;}} }</pre>
Private	<p>Camel Case and prefix private variable with a “_”</p>	<pre>_firstName, _employee</pre>

Variables		
Local Variables	Camel Case	firstName, employee
Constant	All upper case with words separated by underscores	WATER_MAX_DEPTH, WATER_MIN_DEPTH

## **SECTION II: Visual Elements Naming Guidelines:**

### **Preferred method:**

Currently in Visual Studio, when we drag a visual element from the tool box to our design or code page, it will name it with element type then followed by a number. We will take advantage of this by replacing the number with the Entity Name. Such as:

TextBoxEmployeeName  
LabelAddress  
GridViewEmployer  
ButtonAddCustomer

**Abbreviation method:** Please refer to the abbreviation list in the section III for commonly used elements. When we abbreviate, we use Camel Case convention. Such as:

tbxEmployeeName  
lblAddress  
gvEmployer  
btnAddCustomer

### **If the element is not in the list:**

- For single noun element types, we can abbreviate by removing all the vowels with the exception if the first letter of the element is a vowel. Such as:

ImagePerson = imgPerson  
ViewPerson = vwPerson

- For compound noun element types, combine the first letter of each noun. Such as:  
MultiViewFacilitySite= mvFacilitySite

## **II.1 Common abbreviation for visual elements:**

Button		btn
CheckBox		cbx
DropDownList		ddl
DataGrid		dgrid
GridView	gv	
GroupBox	gbx	
Image		img
Label		lbl
ListBox		lbx
Panel		pnl
RadioButton		rbtn
TextBox		tbx

## **III. Styles terminology:**

**Pascal case:** The first letter in the identifier and the first letter of each subsequent concatenated word are capitalized.

Example: BackColor, DataSet

**Camel case:** The first letter of an identifier is lowercase and the first letter of each subsequent concatenated word is capitalized.

Example:    numberOfDays, isValid

**Uppercase:** All letters in the identifier are capitalized.

Example:    ID, PI

Further reading: <http://msdn2.microsoft.com/en-gb/library/ms229045.aspx>

## Document History

Date	Version	Editor	Change
1/22/2007	1	Son Tran	Adopted by Technical Architecture team
3/12/2007	1.2	Son Tran	Adopted by Technical Architecture team
8/25/2008 9/02/2008 9/14/2008	1.3	Son Tran	Review by Strategic Architecture team Review by Architecture Work Group Approved to release by Enterprise Manager (Debbie Stewart)