**EAMI Coding Convention**

Coding conventions serve to provide consistency, easy comprehension and ease of maintenance of the code base.

**C# Coding Convention:**

|  |  |
| --- | --- |
| **Category** | **Convention** |
| Naming Convention | When namespace is imported, do not include fully qualified names from that namespace. |
|  | Use “PascalCasing” for class names and method names. Example, public class ClientActivity {…} |
|  | Use “camelCasing” for method arguments and local variables. Example, int itemCount = 0; |
|  | Do not use “Hungarian” notation or any other type identification in identifiers. Example, int counter; string name; |
|  | Do not use “Screaming Caps” for constants or readonly variables. Example, avoid using,  Public static const string SHIPPING TYPE = “DropShip”; |
|  | Avoid using abbreviations. Exceptions are for names such as Id, Xml, Ftp, Uri etc. |
|  | Do not use underscores in identifiers. Exception is for private static variables. |
|  | Use predefined type names instead of system type names like int16, Single, UInt64 etc. |
|  | Do prefix interfaces with the letter “I”.  Interface names are noun (phrases) or adjective. |
|  | Do use noun or noun phrases to name a class. |
|  | Do organize namespaces with a clearly defined structure. |
|  | Vertically align curly brackets. |
|  | Declare all member variables at the top of a class, with static variables at the very top. |
|  | Use singular names for enums. Exception: bit field enums. |
|  | Do not suffix enum names with Enum. |
| Layout Convention | Use default code editor settings of VS. |
|  | Write only one declaration per line. |
|  | Write only one statement per line. |
|  | Add at least one blank line between method definitions and property definitions. |
|  | Use parenthesis to make a clause in an expression apparent. |
| Commenting Convention | Place the comment on a separate line, not at the end of a line of Code. |
|  | Begin comment text with an uppercase letter. |
|  | End comment text with a period. |
|  | Insert one space between the comment delimiter (//) and the comment text. |
|  | Do not create formatted blocks of asterisks around comments |
| Language Guidelines | Use “+” operator to concatenate strings and use StringBuilder object to concatenate strings in loops. |
|  | When the type of variable is clear from the context, use “var” in the declaration. |
|  | Do not use “var” when type is not apparent from the right side of the assignment. |
|  | Do not rely on the variable name to specify the type of the variable. |
|  | Avoid using “var” in place of dynamic. |
|  | Use implicit-typing or “var” for loop variables. |
|  | In general, use “int” rather than unsigned types. |
|  | Use concise syntax when you initialize arrays on the declaration line. |
|  | Use “try-catch “statement for most exception handling. |
|  | Simplify you code by using “using” statement to utilize automatic object disposal. This can be useful to avoid try-finally statement wherein finally block used only to call dispose method. |
|  | To avoid exceptions and increase performance by skipping unnecessary comparisons, use “[&&](https://docs.microsoft.com/en-us/dotnet/csharp/language-reference/operators/conditional-and-operator)” instead of “[&](https://docs.microsoft.com/en-us/dotnet/csharp/language-reference/operators/and-operator)” and “[||](https://docs.microsoft.com/en-us/dotnet/csharp/language-reference/operators/conditional-or-operator)” instead of “[|](https://docs.microsoft.com/en-us/dotnet/csharp/language-reference/operators/or-operator)” when you perform comparisons. |
|  | Never use go-to statements. |
|  | Document explicitly in code why you are swallowing an exception. |
|  | Do not comment bad code but rewrite it. |
|  | Do not suppress warnings in your code with pragma warning disable. |
|  | A method should protect itself against bad input. |
|  | Write short methods. A method should only contain one level of abstraction. |
|  | Use variables for a single, well-defined goal. |
|  | Use regions sporadically to organize well defined code. |
|  | Do not use regions inside methods, refactor instead. |

|  |  |
| --- | --- |
| **To Do** | **Trick** |
| Format current code area automatically | Press CTRL + E, D for VS 2015 and above. Press CTRL + K, D for VS 2013 and below. |
| Stop Outlining | Press CTRL + M, P |
| Toggle all outlining | Press CTRL + M, L |
| Collapse to definitions | Press CTRL + M, O |
| Comment selection | Press CTRL + K, C |
| Uncomment selection | Press CTRL + K, U |
| Search a file | Press CTRL + ; |
| Go to definition | F12 |

**Source:**

[Dot Factory](http://www.dofactory.com/reference/csharp-coding-standards), [**Microsoft**](https://docs.microsoft.com/en-us/dotnet/csharp/programming-guide/inside-a-program/coding-conventions)**,** [**StackOverflow**](https://stackoverflow.com/questions/2787035/coding-guidelines-best-practices)**.**

**SQL Server Conventions:** Upcoming!