. NET Coding Standards and Guidelines

Version No: 1.2

**Revision History**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Version (x.y) | Date of Revision | Description of Change | Reason for Change | Affected Sections | Approved By |
| 1.1 | 1/10/18 | Baseline Version |  |  | Nagoor Inaganti |
| 1.2 | 10/10/18 | Tailoring Guidelines | Specifying the naming conventions | Guidelines | Nagoor Inaganti |

**Approval History**

|  |  |  |  |
| --- | --- | --- | --- |
| Version (x.y) | Prepared By | Reviewed By/Date | Approved By/Date |
| 1.1 | Vijaya Somarepetta | Monalisha Mishra  3/10/18 | Nagoor Inaganti  3/10/18 |
| 1.2 | Vijaya Somarepetta | Monalisha Mishra  15/10/18 | Nagoor Inaganti  15/10/18 |

Table of Contents

[1 Purpose 4](#_Toc382403076)

[2 Scope 4](#_Toc382403077)

[3 Guidelines Description 4](#_Toc382403078)

[3.1 Naming Conventions 4](#_Toc382403079)

[3.2 Namespace/Project Naming Conventions 5](#_Toc382403080)

[3.3 Class Naming Conventions 5](#_Toc382403081)

[3.4 Interface Naming Conventions 5](#_Toc382403082)

[3.5 Attribute Naming Conventions 6](#_Toc382403083)

[3.6 Method Naming Conventions 6](#_Toc382403084)

[3.7 Constant Naming Conventions 7](#_Toc382403085)

[3.8 Object Naming Conventions 7](#_Toc382403086)

[3.9 Parameter Naming Conventions 9](#_Toc382403087)

[3.10 Enumeration Naming Conventions 9](#_Toc382403088)

[3.11 Variable Naming Conventions 10](#_Toc382403089)

[3.12 Data Types 11](#_Toc382403090)

[3.13 Property Naming Conventions 12](#_Toc382403091)

[3.14 Static Field Naming Conventions 12](#_Toc382403092)

[3.15 Event Naming Conventions 12](#_Toc382403093)

[3.16 Exception Naming Conventions 13](#_Toc382403094)

[4 References 13](#_Toc382403095)

# Purpose

This .NET coding standards document will aid the developers, designers, reviewers of the Forté migration project to follow a standard set of guidelines and conventions that will be implemented during the course of migration of the application to .NET Environment. The user interface screens are expected to have same look and feel of the existing Forté environment. This document will be used as reference-

* By developers for writing manual code.
* By reviewers for creating a code review checklist for manual review.

For setting up the code-review rules in the automated testing and code-review tools.

# Scope

This document covers

* Naming conventions to be followed in migration of manual code.
* Standards to be followed for regular .NET code syntax.
* Code documentation standards and comments.
* Other optimization standards.

A number of the above standards will be reviewed automatically through code review tools.

# Guidelines Description

Coding conventions are programming guidelines that focus not on the logic of the program but on its physical structure and appearance. They make the code easier to read, understand, and maintain.

## Naming Conventions

The following conventions are used for naming identifiers.

* Pascal Case (First character of all words are Upper Case and other characters are lower case).

For example: BackColor.

* Camel Case (First character of all words should be Upper Case, except the first word other characters are lower case).

For example: backColor.

* Upper Case (All the characters are Upper Case).  
  Use Upper Case convention only for identifiers that consist of two or fewer letters.   
  For example: System.IO, System.Web.UI.
* Hungarian Type

Using Hungarian notation, variable names begin with one or more lowercase letters that denote the variable type.  
For example: lblUserName

## Namespace/Project Naming Conventions

The general rule for naming namespaces is to use the company name followed by the Application name and then Component Name and optionally the Module Name and Feature as follows.

CompanyName.ApplicationName.ComponentName.[ModuleName].[Feature]

Example: Arcelor.SWEPPS.BaseConstants

## Class Naming Conventions

* Use Pascal Case.
* Do not use a type prefix, such as C for class, on a class name.
* Do not use underscore character.

Example:

public class FileStream

{

}

## Interface Naming Conventions

* Use Pascal Case.
* Always prefix interface names with I.
* Use similar names when you define a class/interface pair where the class is a standard implementation of the interface. The names should differ only by the letter I prefix on the interface name.
* Do not use underscore character.

Example:

public interface IComponent

{

}

public class Component: IComponent

{

}

## Attribute Naming Conventions

* Use Pascal Case.
* Always suffix the attribute names with “Attribute”.

Example:

public class ObsoleteAttribute

{

}

## Method Naming Conventions

* Use Pascal case.
* Use abbreviations sparingly.
* Use underscore when the method name is big enough. This would help the developer to easily understand the method name.

Example:

public class ResourceReader

{

public string GetQueryString();

}

## Constant Naming Conventions

* Use ALL CAPS for declaration of constants.
* Use separate words within the constants with an underscore.

Example:

AP\_WIN\_MIN\_WIDTH, AP\_WIN\_MAX\_WIDTH

## Object Naming Conventions

Use Hungarian type notation for objects. Following are the prefixes for controls.

| **Control Name** | **Prefix (All Lower Case)** |
| --- | --- |
| ArrayList | al |
| Button | btn |
| Binding Object | bnd |
| Check Box | chk |
| Checked List Box | chkl |
| Combo Box | cmb |
| Data Table | dt |
| Data Row | dr |
| Data Row View | drv |
| Data Grid View | dg |
| Data Grid Table Style | dgts |
| Data Grid Column Style | dgcs |
| Data Grid Textbox Column | dgtc |
| Data Grid Checkbox Column | dgcc |
| Data Grid Button Column | dgbtn |
| Data Grid Combo Box Column | dgcmb |
| Data Grid Image Column | dgimg |
| Data Grid Link Column | dglnk |
| Masked TextBox | msktxt |
| MenuStrip, MenuItem | mnu |
| Date Time Picker | dtp |
| Form | frm |
| Group Box | grp |
| Image List Box | ilb |
| Label | lbl |
| LinkLabel | llbl |
| List Box | lst |
| List View | lvw |
| List View Item | lvi |
| List View Column Header | lvch |
| Month Calendar | mc |
| Panel | pnl |
| Picture Box | pic |
| Radio Button | rdo |
| Rich Text Box | rtf |
| Tab Control | tab |
| Tab Page | tpg |
| Text Box | txt |
| Timer | tmr |
| Tree View | tvw |
| User Control | uc |

Database object naming conventions

|  |  |
| --- | --- |
| **Database Object** | **Prefix (All Lower Case)** |
| Command | cmd |
| Connection | conn |
| DataReader | dr |
| DataSet | ds |
| Field | fld |
| Parameter | prm |
| Property | prop |
| Stream | strm |

## Parameter Naming Conventions

* Use Camel Case for parameter names.
* Use descriptive parameter names. Parameter names should be descriptive enough that the name of the parameter and its type can be used to determine its meaning in most scenarios.
* Use names that describe a parameter's meaning rather than names that describe a parameter's type.
* Do not use reserved parameters. If more data is needed in the next version a new overload can be added.
* Do not use casing to differentiate parameters.

Example:

public string GetQueryString (string qryCriteria);

## Enumeration Naming Conventions

* User Pascal Case for Enum types and value names.
* Use abbreviations sparingly
* Do not use an Enum suffix on Enum type names.

Example:  
 enum MailType

{

Html,

Plaintext,

Attachment

}

## Variable Naming Conventions

* Use Pascal Case for public variables.
* Use Hungarian Type notation for private variables with a prefix of underscore (\_).
* Use ALL CAPS only for a two-character abbreviation.
* Imply the variable is Boolean if it is. For example: isTrue.
* Prefix private variables with underscore(\_).
* Do not use casing to differentiate variables.

Example:

public string FirstName;

private string \_strFirstName;

Use the following prefixes for private variables and method parameters.

|  |  |
| --- | --- |
| **Data Type** | **Prefix (All Lower Case)** |
| Array | arr |
| Boolean | bln |
| Byte | byt |
| Collection Object | col |
| Control | ctl |
| Currency | cur |
| Date (TimeStamp) | dtm |
| Double | dbl |
| Enumerated Type | enm |
| FileStream | fs |
| Integer | int |
| Long | lng |
| Object | obj |
| Single | sng |
| StreamWriter | sw |
| StreamReader | sr |
| String | str |
| StringBuilder | sb |
| Structure | obj |

## Data Types

* Use the simplest data type, list, or object required. For example, use **int** over **long** unless you know you need to store 64bit values.
* Always use the built-in C# data type aliases, not the .NET common type system (CTS).

Example:

short NOT System.Int16

int NOT System.Int32

long NOT System.Int64

string NOT System.String

* Only declare member variables as private.
* Use properties to provide access to them with public, protected or internal access modifiers.
* Declare read-only or static read-only variables instead of constants for complex types.
* Avoid boxing and unboxing value types.

Example:  
 int count=1;

object refCount=count;

int newCount=(int) refCount;

* Floating point values should include at least one digit before the decimal place and one after.

Example:

totalPercent = 0. 5;

or totalPercent = 5.0;

## Property Naming Conventions

* Use Pascal case.
* Consider creating a property with the same name as its underlying type. For example, if you declare a property named Color, the type of the property should be “Color”.

Example**:**

public class SampleClass

{

public Color BackColor

{

}

}

## Static Field Naming Conventions

* Use Pascal case.
* It is recommended that you use static properties instead of public static fields whenever possible.

Example:

private static string CompanyName = “Company NAme”;

## Event Naming Conventions

* User Pascal case.
* Use an Event Handler suffix on event handler names.
* Name an event argument class with the EventArgs suffix.
* Do not use a prefix or suffix on the event declaration on the type. For example, use Close instead of OnClose.

Example:

public delegate void MouseEventHandler (object sender, MouseEventArgs e);

## Exception Naming Conventions

* Use Pascal Case.
* Always suffix the exception names with “Exception”.

Example:

public class CustomException

{

}

# References

None.