### Pascal case

Use pascal casing ("PascalCasing") when naming a class, record, or struct.

When naming an interface, use pascal casing in addition to prefixing the name with an I. This clearly indicates to consumers that it's an interface.

When naming public members of types, such as fields, properties, events, methods, and local functions, use pascal casing.

When writing positional records, use pascal casing for parameters as they're the public properties of the record.

**Camel case**

Use camel casing ("camelCasing") when naming private or internal fields, and prefix them with \_.

When working with static fields that are private or internal, use the s\_ prefix and for thread static use t\_.

When writing method parameters, use camel casing.

**Additional naming conventions**

Examples that don't include using directives, use namespace qualifications. If you know that a namespace is imported by default in a project, you don't have to fully qualify the names from that namespace. Qualified names can be broken after a dot (.) if they are too long for a single line.

You don't have to change the names of objects that were created by using the Visual Studio designer tools to make them fit other guidelines.

**Layout conventions**

Good layout uses formatting to emphasize the structure of your code and to make the code easier to read. Microsoft examples and samples conform to the following conventions:

Use the default Code Editor settings (smart indenting, four-character indents, tabs saved as spaces). For more information, see Options, Text Editor, C#, Formatting.

Write only one statement per line.

Write only one declaration per line.

If continuation lines are not indented automatically, indent them one tab stop (four spaces).

Add at least one blank line between method definitions and property definitions.

Use parentheses to make clauses in an expression apparent.

**Place the using directives outside the namespace declaration**

When a using directive is outside a namespace declaration, that imported namespace is its fully qualified name. That's more clear. When the using directive is inside the namespace, it could be either relative to that namespace or it's fully qualified name. That's ambiguous.

## **Commenting conventions**

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.

Don't create formatted blocks of asterisks around comments.

Ensure all public members have the necessary XML comments providing appropriate descriptions about their behavior.