The general rule we follow is "use Visual Studio defaults".

1. We use [Allman style](http://en.wikipedia.org/wiki/Indent\_style#Allman\_style) braces, where each brace begins on a new line. A single line statement block can go without braces but the block must be properly indented on its own line and it must not be nested in other statement blocks that use braces (See issue [381](https://github.com/dotnet/corefx/issues/381) for examples).

2. We use four spaces of indentation (no tabs).

3. We use `\_camelCase` for internal and private fields and use `readonly` where possible. Prefix instance fields with `\_`, static fields with `s\_` and thread static fields with `t\_`. When used on static fields, `readonly` should come after `static` (e.g. `static readonly` not `readonly static`).

4. We avoid `this.` unless absolutely necessary.

5. We always specify the visibility, even if it's the default (e.g. `private string \_foo` not `string \_foo`). Visibility should be the first modifier (e.g. `public abstract` not `abstract public`).

6. Namespace imports should be specified at the top of the file, \*outside\* of `namespace` declarations and should be sorted alphabetically.

7. Avoid more than one empty line at any time. For example, do not have two blank lines between members of a type.

8. Avoid spurious free spaces. For example avoid `if (someVar == 0)...`, where the dots mark the spurious free spaces. Consider enabling "View White Space (Ctrl+E, S)" if using Visual Studio, to aid detection.

9. If a file happens to differ in style from these guidelines (e.g. private members are named `m\_member` rather than `\_member`), the existing style in that file takes precedence.

10. We only use `var` when it's obvious what the variable type is (e.g. `var stream = new FileStream(...)` not `var stream = OpenStandardInput()`).

11. We use language keywords instead of BCL types (e.g. `int, string, float` instead of `Int32, String, Single`, etc) for both type references as well as method calls (e.g. `int.Parse` instead of `Int32.Parse`). See issue [391](https://github.com/dotnet/corefx/issues/391) for examples.

12. We use PascalCasing to name all our constant local variables and fields. The only exception is for interop code where the constant value should exactly match the name and value of the code you are calling via interop.

13. We use ```nameof(...)``` instead of ```"..."``` whenever possible and relevant.

14. Fields should be specified at the top within type declarations.

15. When including non-ASCII characters in the source code use Unicode escape sequences (\uXXXX) instead of literal characters. Literal non-ASCII characters occasionally get garbled by a tool or editor.

16. Do not use labels (e.g. for goto).