Hayden Lepla

CS-383

3/29/18

Team Lead 5

Team Zero's code standards and documentation

Team Zero will use the official code style convention for C# created my Microsoft's .NET team and will use Allman style for enforcement of braces (scope of code). Code style conventions serve several purposes such as:

- They create a consistent look to the code, so that readers can focus on content, not layout.
- They enable readers to understand the code more quickly by making assumptions based on previous experience.
- They facilitate copying, changing, and maintaining the code.
- They demonstrate C# best practices.

This document illustrates the guidelines for our code samples and documentation of code style.

Naming Conventions

General Structure of a C# program: C# programs can consist of one or more files. Each file can contain zero or more namespaces. A namespace can contain types such as classes, structs, interfaces, enumerations, and delegates, in addition to other namespaces. The following is the skeleton of a C# program that contains all of these elements.

```
// A skeleton of a C# program
using System;
namespace YourNamespace
{
    class YourClass
    {
    }

    struct YourStruct
    {
    }
    interface IYourInterface
    {
    }

    delegate int YourDelegate();

    enum YourEnum
    {
    }

    namespace YourNestedNamespace
    {
        struct YourStruct
        {
        }
    }

    class YourMainClass
    {
        static void Main(string[] args)
        {
            //Your program starts here...
        }
    }
}
```

Variable Names: Variable names must be in the form of camelCase. The camelCasing convention, used only for variable names, capitalizes the second character of each word except the first word, as shown in the following examples.

```
propertyDescriptor
```

```
ioStream
```

```
htmlTag
```

As the example also shows, two-letter acronyms that begin a camel-cased identifier are both lowercase. Furthermore, variable names must be somewhat descriptive to what action is being performed by that variable.

Function names: Function names should use the PascalCase convention. PascalCase is where function names capitalize the first letter of every name. If the name is a compounded word, then every new word must be capitalized. Example:

```
public class StreamReader { ... }
public class Object {
public virtual string ToString();
}
```

Commenting styles: Code comments should do the following: Be placed on a separate line and not at the end of a line of code, begin with an uppercase letter, end comment text with a period, insert one space between the comment delimiter (//) and the comment text, and do not create formatted blocks of asterisks around the comments. Example:

```
// The following declaration creates a query. It does not run
// the query.
```

Indentation styles: Indentation should be co-paired with the Allman style bracket convention. Indentation depth must be kept constant at 4 spaces. Example:

```
while (x == y)
{
    something();
    somethingelse();
}
```

Error handling: A C# exception is a response to an exceptional circumstance that arises while a program is running, such as an attempt to divide by zero. Exceptions provide a way to transfer control from one part of a program to another. C# exception handling is built upon four keywords: **try**, **catch**, **finally**, and **throw**. Assuming a block raises an exception, a method catches an exception using a combination of the try and catch keywords. The code must follow this idea. Here is an example:

```
try {
   // statements causing exception
} catch( ExceptionName e1 ) {
   // error handling code
} catch( ExceptionName e2 ) {
   // error handling code
} catch( ExceptionName eN ) {
   // error handling code
} finally {
   // statements to be executed
}
```