C# scope of variables

Variables may be declared:

- within main
- within the { } sections of a condition or loop (if, do etc.)
- within a function body

A variable exists within a stated scope, this is usually the boundary of a set of { }s. C# does not allow a variable to be declared with the same name within a function. Some languages do allow this.

```
static int scope (int num )
{
            num++;
            return num;
}
static void Main(string[] args)
{
            int num;
            num = scope(1);
            Console.WriteLine(num);
            num = scope(1);
            Console.WriteLine(num);
            num = scope(1);
            Console.WriteLine(num);
            Console.ReadLine();
}
```

Here there are 2 variables both of type int and both named num. When the code runs it can be seen that the value of num within the function is reset to 0 each time that it runs. The code would be easier to folllow if the int within the main and the function had different names.

The above code is legal, C# will not alllow this:

```
int num =4;
int num = 5; //danger
```

C# is also not happy if the variable is declared again within a decision or loop. Some languages would allow this but not C#.

```
int num =4;
if (num == 4)
{ //this will not work
      int num = 5;
}
```

The following is OK because num is not declared twice:

```
int num =4;
if (num == 4)
{ //OK now
    num = 5;
}
```