**C# 6.0 Language Features – nameof Operator**

Welcome to C# 6.0 language features article series. As you might have heard about Visual Studio 2015 and .Net framework 4.6 announcements in various Microsoft events earlier, one of the most exciting news was the next version of C# language, which is 6.0 and new features added to it. C# has come a long way since 2002 when C# 1.0 was released along with Visual Studio. Since then with every major release of .Net framework, C# became better and better language, which has tremendously helped application developers to minimize the code complexity and focus on business functionality. With the new version of .Net framework and Visual Studio around the corner, you can start learning new language features today using Visual Studio 2015 preview release. There are many useful features added to the language and I don’t know at the moment, how many articles I will write in this series, but it will probably something like this.

* Static using statements
* Auto property initializer
* Null conditional operator
* String interpolation
* **nameof operator - This Article**
* Declaration expressions

**nameof Operator**

One of the common [bad] practices in any programming language is to use magic string within application program; however there are certain scenarios in which you cannot avoid using it. In some cases, such magic string refers to a specific element within the program. For e.g. while raising ArgumentNullException exception, we need to pass which argument cannot be null using a magic string.

1. static void DisplayName (string username)
2. {
3. if (username== null)
4. throw new ArgumentNullException("Username value cannot be null");
5. }

Similarly if you have done any XAML programming using MVVM pattern, you should be familiar with INotifyPropertyChangedinterface and you must have passed property name as a magic string to OnPropertyChanged event handler as shown in below code snippet.

1. public class CustomerViewModel : INotifyPropertyChanged
2. {
3. private int firstName;
4. public int FirstName
5. {
6. get { return firstName; }
7. set
8. {
9. firstName = value;
10. OnPropertyChanged("FirstName");
11. }
12. }
13. public void OnPropertyChanged(string propertyName)
14. {
15. if (PropertyChanged != null)
16. {
17. PropertyChanged(this, new PropertyChangedEventArgs(propertyName));
18. }
19. }
20. public event PropertyChangedEventHandler PropertyChanged;
21. }

Although there is nothing wrong in doing so, C# compiler cannot validate if these magic strings refers to appropriate program element. For e.g. if you rename the FirstName property in above code snippet, there is no way C# compiler can update the magic string in property setter based on the changes made…until C# 6.0.

The nameof operator allows developers to use a program element as text. You cannot specify a string literal / magic string directly with the nameof operator. It can be used to create the nameof expression to specify the name where expression may be a property-group or a method-group.

If you rename username parameter in the DisplayName function below,

1. static void DisplayName (string username)
2. {
3. if (name== null)
4. throw new ArgumentNullException(nameof(username));
5. }

If you rename the username parameter, you will also need to update the nameof expression otherwise program will not compile.

Similarly, we can also update the INotifyPropertyChanged property setter code to ensure that we can use nameof operator to refer to exact program element instead of a magic string.

1. public string FirstName
2. {
3. get { return firstName; }
4. set
5. {
6. if (firstName != value)
7. {
8. firstName = value;
9. OnPropertyChanged(nameof(FirstName));
10. };
11. }
12. }

I believe this feature will help developers to write cleaner and readable code. Let me know what you think about it in the comments section below.