

# Namespace NF.UnityLibs.Utils.RoslynCode Analysis.CodeFixProviders

## Classes

[DateTimeNowCodeFixProvider](#)

[MathRoundCodeFixProvider](#)

[TodoStyleCodeFixProvider](#)

# Class DateTimeNowCodeFixProvider

Namespace: [NF.UnityLibs.Utils.RoslynCodeAnalysis.CodeFixProviders](#)








Assembly: NF.UnityLibs.Utils.RoslynCodeAnalysis.dll

```
[ExportCodeFixProvider("C#", new string[] { }, Name = "DateTimeNowCodeFixProvider")]  
[Shared]  
public class DateTimeNowCodeFixProvider : CodeFixProvider
```

## Inheritance

[object](#)  ← [CodeFixProvider](#)  ← DateTimeNowCodeFixProvider

## Inherited Members

[object.Equals\(object\)](#)  , [object.Equals\(object, object\)](#)  , [object.GetHashCode\(\)](#)  ,  
[object.GetType\(\)](#)  , [object.MemberwiseClone\(\)](#)  , [object.ReferenceEquals\(object, object\)](#)  ,  
[object.ToString\(\)](#) 

## Properties

### FixableDiagnosticIds

A list of diagnostic IDs that this provider can provide fixes for.



```
public override sealed ImmutableArray<string> FixableDiagnosticIds { get; }
```

### Property Value

[ImmutableArray](#)  <[string](#)  >

## Methods

### GetFixAllProvider()

Gets an optional [FixAllProvider](#)  that can fix all/multiple occurrences of diagnostics fixed by this code fix provider. Return null if the provider doesn't support fix all/multiple occurrences. Otherwise, you can return any of the well known fix all providers from [Well Known Fix All Providers](#)  or implement your own fix all provider.

```
public override sealed FixAllProvider GetFixAllProvider()
```

Returns

[FixAllProvider](#)

## RegisterCodeFixesAsync(CodeFixContext)

Computes one or more fixes for the specified [CodeFixContext](#).

```
public override sealed Task RegisterCodeFixesAsync(CodeFixContext context)
```

Parameters

context [CodeFixContext](#)

A [CodeFixContext](#) containing context information about the diagnostics to fix. The context must only contain diagnostics with a [Id](#) included in the [FixableDiagnosticIds](#) for the current provider.

Returns

[Task](#)

# Class MathRoundCodeFixProvider

Namespace: [NF.UnityLibs.Utils.RoslynCodeAnalysis.CodeFixProviders](#)








Assembly: NF.UnityLibs.Utils.RoslynCodeAnalysis.dll

```
[ExportCodeFixProvider("C#", new string[] { }, Name = "MathRoundCodeFixProvider")]  
[Shared]  
public class MathRoundCodeFixProvider : CodeFixProvider
```

## Inheritance

[object](#)  ← [CodeFixProvider](#)  ← MathRoundCodeFixProvider

## Inherited Members

[object.Equals\(object\)](#)  , [object.Equals\(object, object\)](#)  , [object.GetHashCode\(\)](#)  ,  
[object.GetType\(\)](#)  , [object.MemberwiseClone\(\)](#)  , [object.ReferenceEquals\(object, object\)](#)  ,  
[object.ToString\(\)](#) 

## Properties

### FixableDiagnosticIds

A list of diagnostic IDs that this provider can provide fixes for.



```
public override sealed ImmutableArray<string> FixableDiagnosticIds { get; }
```

### Property Value

[ImmutableArray](#)  <[string](#)  >

## Methods

### GetFixAllProvider()

Gets an optional [FixAllProvider](#)  that can fix all/multiple occurrences of diagnostics fixed by this code fix provider. Return null if the provider doesn't support fix all/multiple occurrences. Otherwise, you can return any of the well known fix all providers from [Well Known Fix All Providers](#)  or implement your own fix all provider.

```
public override sealed FixAllProvider GetFixAllProvider()
```

Returns

[FixAllProvider](#)

## RegisterCodeFixesAsync(CodeFixContext)

Computes one or more fixes for the specified [CodeFixContext](#).

```
public override sealed Task RegisterCodeFixesAsync(CodeFixContext context)
```

Parameters

**context** [CodeFixContext](#)

A [CodeFixContext](#) containing context information about the diagnostics to fix. The context must only contain diagnostics with a [Id](#) included in the [FixableDiagnosticIds](#) for the current provider.

Returns

[Task](#)

# Class TodoStyleCodeFixProvider

Namespace: [NF.UnityLibs.Utils.RoslynCodeAnalysis.CodeFixProviders](#)








Assembly: NF.UnityLibs.Utils.RoslynCodeAnalysis.dll

```
[ExportCodeFixProvider("C#", new string[] { }, Name = "TodoStyleCodeFixProvider")]  
[Shared]  
public class TodoStyleCodeFixProvider : CodeFixProvider
```

## Inheritance

[object](#)  ← [CodeFixProvider](#)  ← TodoStyleCodeFixProvider

## Inherited Members

[object.Equals\(object\)](#)  , [object.Equals\(object, object\)](#)  , [object.GetHashCode\(\)](#)  ,  
[object.GetType\(\)](#)  , [object.MemberwiseClone\(\)](#)  , [object.ReferenceEquals\(object, object\)](#)  ,  
[object.ToString\(\)](#) 

## Properties

### FixableDiagnosticIds

A list of diagnostic IDs that this provider can provide fixes for.



```
public override sealed ImmutableArray<string> FixableDiagnosticIds { get; }
```

### Property Value

[ImmutableArray](#)  <[string](#)  >

## Methods

### GetFixAllProvider()

Gets an optional [FixAllProvider](#)  that can fix all/multiple occurrences of diagnostics fixed by this code fix provider. Return null if the provider doesn't support fix all/multiple occurrences. Otherwise, you can return any of the well known fix all providers from [Well Known Fix All Providers](#)  or implement your own fix all provider.

```
public override sealed FixAllProvider GetFixAllProvider()
```

Returns

[FixAllProvider](#)

## RegisterCodeFixesAsync(CodeFixContext)

Computes one or more fixes for the specified [CodeFixContext](#).

```
public override sealed Task RegisterCodeFixesAsync(CodeFixContext context)
```

Parameters

context [CodeFixContext](#)

A [CodeFixContext](#) containing context information about the diagnostics to fix. The context must only contain diagnostics with a [Id](#) included in the [FixableDiagnosticIds](#) for the current provider.

Returns

[Task](#)

# Namespace NF.UnityLibs.Utils.RoslynCode Analysis.DiagnosticAnalyzers

## Classes

[DateTimeNowAnalyzer](#)

[MathRoundAnalyzer](#)

[TodoStyleAnalyzer](#)



# Class DateTimeNowAnalyzer

Namespace: [NF.UnityLibs.Utils.RoslynCodeAnalysis.DiagnosticAnalyzers](#)

Assembly: NF.UnityLibs.Utils.RoslynCodeAnalysis.dll

```
[DiagnosticAnalyzer("C#", new string[] { })]  
public class DateTimeNowAnalyzer : DiagnosticAnalyzer
```

## Inheritance

[object](#) ← [DiagnosticAnalyzer](#) ← DateTimeNowAnalyzer

## Inherited Members

[DiagnosticAnalyzer.Equals\(object\)](#), [DiagnosticAnalyzer.GetHashCode\(\)](#),  
[DiagnosticAnalyzer.ToString\(\)](#), [object.Equals\(object, object\)](#), [object.GetType\(\)](#),  
[object.MemberwiseClone\(\)](#), [object.ReferenceEquals\(object, object\)](#)

## Properties

### SupportedDiagnostics

Returns a set of descriptors for the diagnostics that this analyzer is capable of producing.

```
public override ImmutableArray<DiagnosticDescriptor> SupportedDiagnostics { get; }
```

### Property Value

[ImmutableArray](#) <[DiagnosticDescriptor](#)>

## Methods

### Initialize(AnalysisContext)

Called once at session start to register actions in the analysis context.

```
public override void Initialize(AnalysisContext context)
```

## Parameters

context [AnalysisContext](#)

# Class MathRoundAnalyzer

Namespace: [NF.UnityLibs.Utils.RoslynCodeAnalysis.DiagnosticAnalyzers](#)

Assembly: NF.UnityLibs.Utils.RoslynCodeAnalysis.dll

```
[DiagnosticAnalyzer("C#", new string[] { })]  
public class MathRoundAnalyzer : DiagnosticAnalyzer
```

## Inheritance

[object](#) ← [DiagnosticAnalyzer](#) ← MathRoundAnalyzer

## Inherited Members

[DiagnosticAnalyzer.Equals\(object\)](#), [DiagnosticAnalyzer.GetHashCode\(\)](#),  
[DiagnosticAnalyzer.ToString\(\)](#), [object.Equals\(object, object\)](#), [object.GetType\(\)](#),  
[object.MemberwiseClone\(\)](#), [object.ReferenceEquals\(object, object\)](#)

## Properties

### SupportedDiagnostics

Returns a set of descriptors for the diagnostics that this analyzer is capable of producing.

```
public override ImmutableArray<DiagnosticDescriptor> SupportedDiagnostics { get; }
```

### Property Value

[ImmutableArray](#) <[DiagnosticDescriptor](#)>

## Methods

### Initialize(AnalysisContext)

Called once at session start to register actions in the analysis context.

```
public override void Initialize(AnalysisContext context)
```

## Parameters

context [AnalysisContext](#)

# Class TodoStyleAnalyzer

Namespace: [NF.UnityLibs.Utils.RoslynCodeAnalysis.DiagnosticAnalyzers](#)

Assembly: NF.UnityLibs.Utils.RoslynCodeAnalysis.dll

```
[DiagnosticAnalyzer("C#", new string[] { })]  
public class TodoStyleAnalyzer : DiagnosticAnalyzer
```

## Inheritance

[object](#) ← [DiagnosticAnalyzer](#) ← TodoStyleAnalyzer

## Inherited Members

[DiagnosticAnalyzer.Equals\(object\)](#), [DiagnosticAnalyzer.GetHashCode\(\)](#),  
[DiagnosticAnalyzer.ToString\(\)](#), [object.Equals\(object, object\)](#), [object.GetType\(\)](#),  
[object.MemberwiseClone\(\)](#), [object.ReferenceEquals\(object, object\)](#)

## Properties

### SupportedDiagnostics

Returns a set of descriptors for the diagnostics that this analyzer is capable of producing.

```
public override ImmutableArray<DiagnosticDescriptor> SupportedDiagnostics { get; }
```

### Property Value

[ImmutableArray](#) <[DiagnosticDescriptor](#)>

## Methods

### Initialize(AnalysisContext)

Called once at session start to register actions in the analysis context.

```
public override void Initialize(AnalysisContext context)
```

## Parameters

context [AnalysisContext](#)