C# static code analysis

Unique rules to find Bugs, Vulnerabilities, Security Hotspots, and Code Smells in your C# code

"ConstructorArgument" parameters should exist in constructors

Analyze your code

Bug Major xaml wpf

When creating a custom Markup Extension that accepts parameters in WPF, the `ConstructorArgument` markup must be used to identify the discrete properties that match these parameters. However since this is done via a string, the compiler will not notice if there are typos.

This rule raises an issue when the string argument to `ConstructorArgumentAttribute` doesn't match any parameter of any constructor.

Noncompliant Code Example

```
using System;

namespace myLibrary
{
  public class MyExtension : MarkupExtension
  {
    public MyExtension() { }

    public MyExtension(object value1)
    {
      Value1 = value1;
    }

    [ConstructorArgument("value2")]   // Noncompliant
    public object Value1 { get; set; }
  }
}
```

Compliant Solution

```
using System;

namespace myLibrary
{
  public class MyExtension : MarkupExtension
  {
    public MyExtension() { }

    public MyExtension(object value1)
    {
      Value1 = value1;
    }

    [ConstructorArgument("value1")]
    public object Value1 { get; set; }
  }
}
```

Sections of code should not be commented out

Code Smell

Unused method parameters should be removed

Code Smell

Empty arrays and collections should be returned instead of null

Code Smell

Unused private types or members should be removed

Code Smell

Available In:

sonarlint | sonarcloud | sonarqube