Secrets
ABAP
Apex
C
C++
CloudFormation
COBOL
C#
CSS
Flex
Go
HTML
Java
JavaScript
Kotlin
Objective C
PHP
PL/I
PL/SQL
Python
RPG
Ruby
Scala
Swift
Terraform
Text
TypeScript
T-SQL
VB.NET
VB6
XML

C# static code analysis

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

All rules 409 | Vulnerability 34 | Bug 76 | Security Hotspot 28 | Code Smell 271 | Quick Fix 52

Tags

Search by name...

Code Smell

URI Parameters should not be strings
Code Smell

Custom attributes should be marked with "System.AttributeUsageAttribute"
Code Smell

Assemblies should explicitly specify COM visibility
Code Smell

Assemblies should be marked as CLS compliant
Code Smell

"Generic.List" instances should not be part of public APIs
Code Smell

Collections should implement the generic interface
Code Smell

Generic event handlers should be used
Code Smell

Event Handlers should have the correct signature
Code Smell

"Assembly.GetExecutingAssembly" should not be called
Code Smell

Arguments of public methods should be validated against null
Code Smell

Value types should implement "IEquatable<T>"
Code Smell

Finalizers should not be empty

Static fields should not be used in generic types

Analyze your code

Code Smell   Major

A static field in a generic type is not shared among instances of different closed constructed types, thus `LengthLimitedSingletonCollection<int>.instances` and `LengthLimitedSingletonCollection<string>.instances` will point to different objects, even though `instances` is seemingly shared among all `LengthLimitedSingletonCollection<>` generic classes.

If you need to have a static field shared among instances with different generic arguments, define a non-generic base class to store your static members, then set your generic type to inherit from the base class.

Noncompliant Code Example

```
public class LengthLimitedSingletonCollection<T> where T : n
{
  protected const int MaxAllowedLength = 5;
  protected static Dictionary<Type, object> instances = new

  public static T GetInstance()
  {
    object instance;

    if (!instances.TryGetValue(typeof(T), out instance))
    {
      if (instances.Count >= MaxAllowedLength)
      {
        throw new Exception();
      }
      instance = new T();
      instances.Add(typeof(T), instance);
    }
    return (T)instance;
  }
}
```

Compliant Solution

```
public class SingletonCollectionBase
{
  protected static Dictionary<Type, object> instances = new
}

public class LengthLimitedSingletonCollection<T> : Singleton
{
  protected const int MaxAllowedLength = 5;

  public static T GetInstance()
  {
    object instance;
```

Code Smell

"[ExpectedException]" should not be used

Code Smell

"this" should not be exposed from constructors

Code Smell

Types should not have members with visibility set higher than the type's visibility

Code Smell

Fields should be private

```
    if (!instances.TryGetValue(typeof(T), out instance))
    {
      if (instances.Count >= MaxAllowedLength)
      {
        throw new Exception();
      }
      instance = new T();
      instances.Add(typeof(T), instance);
    }
    return (T)instance;
  }
}
```

Exceptions

If the static field or property uses a type parameter, then the developer is assumed to understand that the static member is not shared among the closed constructed types.

```
public class Cache<T>
{
   private static Dictionary<string, T> CacheDictionary { ge
}
```

Available In:

sonarlint | sonarcloud | sonarqube