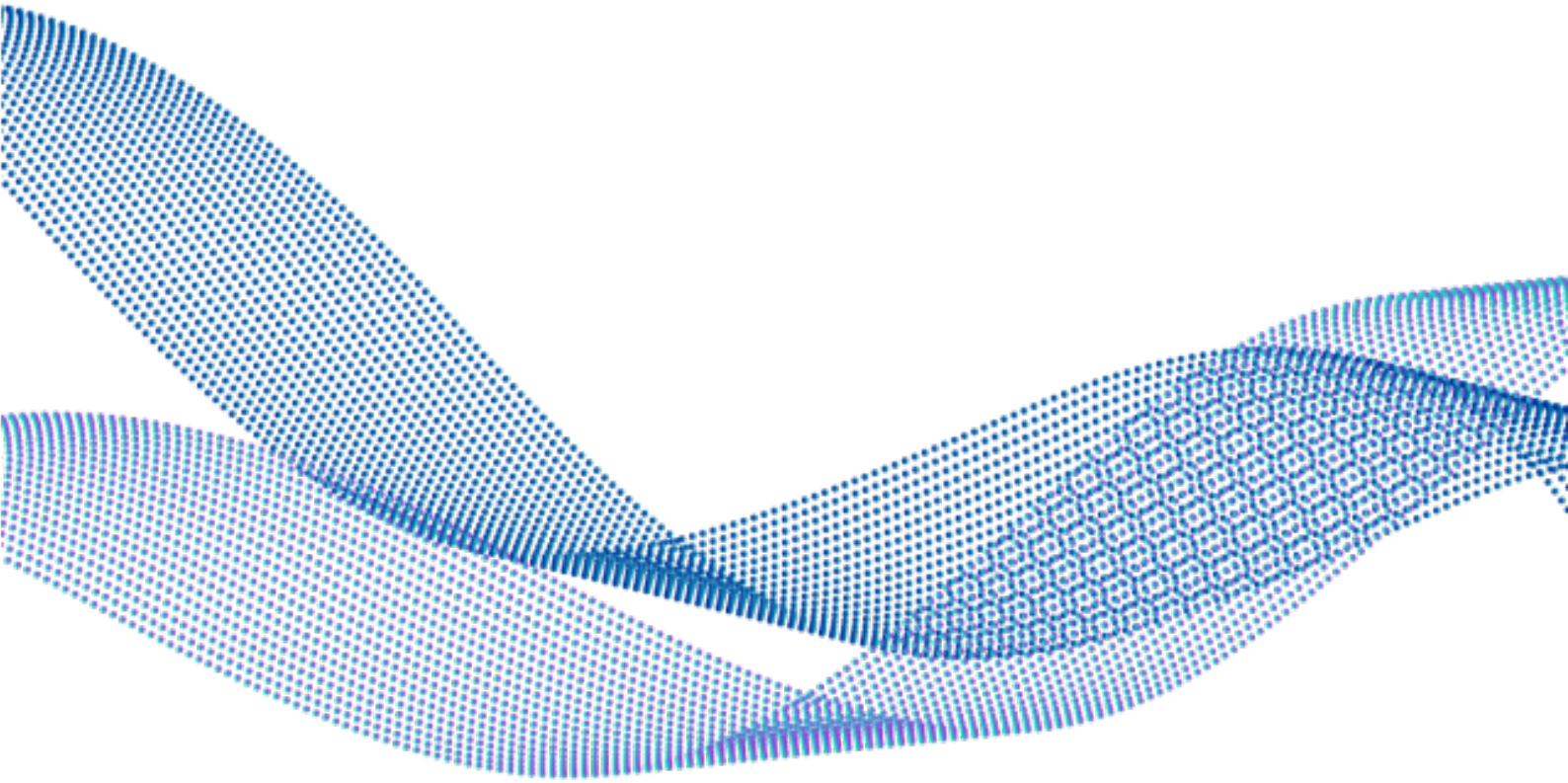
Project Issues Report

Sonar Tools



SONAR TOOLS

Project Overview

Name: Valentia Cheng Le Xuan
Date: 2021-04-24

Last Scanned	Hotspots Reviewed	Coverage	Duplication	No. of lines of code
2021-01-25	E 0.0%	0.0%	0.0%	4239

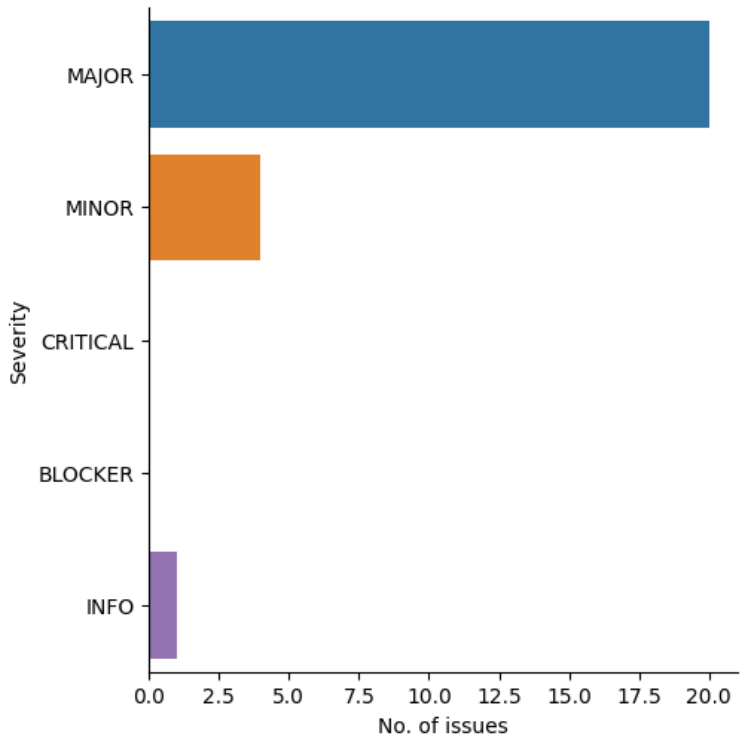
Project Name: CodeJam

Quality Gate Status
Passed

Bugs	Vulnerabilties	Code Smells
0	0	25

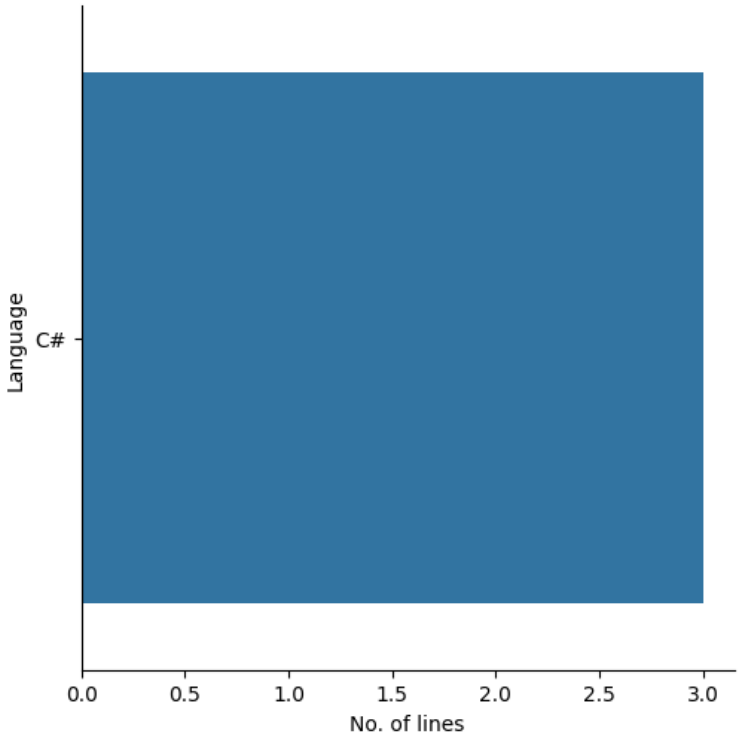
Number of issues for each severity level

Major	Minor	Critical	Blocker	Info
20	4	0	0	1



Number of lines for each language

Languages	No. of lines
C#	3



Project Issues

File Name: CodeJam:Services/FileBlogService.cs

Split this method into two, one handling parameters check and the other handling the asynchronous code.

2020-03-02 150

CODE_SMELL MAJOR OPEN 15min effort

What is the issue?

```
public static async Task SkipLinesAsync(this TextReader reader, int linesToSkip) // Noncompliant
{
   if (reader == null) { throw new ArgumentNullException(nameof(reader)); }
   if (linesToSkip < 0) { throw new ArgumentOutOfRangeException(nameof(linesToSkip)); }

   for (var i = 0; i < linesToSkip; ++i)
   {
       var line = await reader.ReadLineAsync().ConfigureAwait(false);
       if (line == null) { break; }
   }
}
```

Split this method into two, one handling parameters check and the other handling the asynchronous code.

2020-02-25 176

CODE_SMELL MAJOR OPEN 15min effort

What is the issue?

```
public static async Task SkipLinesAsync(this TextReader reader, int linesToSkip) // Noncompliant
{
   if (reader == null) { throw new ArgumentNullException(nameof(reader)); }
   if (linesToSkip < 0) { throw new ArgumentOutOfRangeException(nameof(linesToSkip)); }

   for (var i = 0; i < linesToSkip; ++i)
   {
       var line = await reader.ReadLineAsync().ConfigureAwait(false);
       if (line == null) { break; }
   }
}
```

Complete the task associated to this 'TODO' comment. 2020-02-25 237

CODE_SMELL INFO OPEN 0min effort

What is the issue?

```
private void DoSomething()
{
 // TODO
}
```

File Name: CodeJam:Controllers/BlogController.cs

Split this method into two, one handling parameters check and the other handling the asynchronous code. 2020-02-25 187

CODE_SMELL MAJOR OPEN 15min effort

What is the issue?

```
public static async Task SkipLinesAsync(this TextReader reader, int linesToSkip) // Noncompliant
{
   if (reader == null) { throw new ArgumentNullException(nameof(reader)); }
   if (linesToSkip < 0) { throw new ArgumentOutOfRangeException(nameof(linesToSkip)); }

   for (var i = 0; i < linesToSkip; ++i)
   {
       var line = await reader.ReadLineAsync().ConfigureAwait(false);
       if (line == null) { break; }
   }
}
```

File Name: CodeJam:Controllers/RobotsController.cs

Refactor your code not to use hardcoded absolute paths or URIs. 2020-02-25 166

CODE_SMELL MINOR OPEN 20min effort

What is the issue?

Hardcoding a URI makes it difficult to test a program: path literals are not always portable across operating systems, a given absolute path may not exist on a specific test environment, a specified Internet URL may not be available when executing the tests,

production environment filesystems usually differ from the development environment, ...etc. For all those reasons, a URI should never be hardcoded. Instead, it should be replaced by customizable parameter.

Further even if the elements of a URI are obtained dynamically, portability can still be limited if the path-delimiters are hardcoded.

This rule raises an issue when URI's or path delimiters are hardcoded.

Exceptions

This rule does not raise an issue when an ASP.NET virtual path is passed as an argument to one of the following:

- methods: `System.Web.HttpServerUtilityBase.MapPath()`, `System.Web.HttpRequestBase.MapPath()`, `System.Web.HttpResponseBase.ApplyAppPathModifier()`, `System.Web.Mvc.UrlHelper.Content()`
- all methods of: `System.Web.VirtualPathUtility`
- constructors of: `Microsoft.AspNetCore.Mvc.VirtualFileResult`, `Microsoft.AspNetCore.Routing.VirtualPathData`

See

- CERT, MSC03-J. - Never hard code sensitive information

File Name: CodeJam:Services/MetaWeblogService.cs

Split this method into two, one handling parameters check and the other handling the asynchronous code.

2020-02-25 50

CODE_SMELL MAJOR OPEN 15min effort

What is the issue?

```
public static async Task SkipLinesAsync(this TextReader reader, int linesToSkip) // Noncompliant
{
   if (reader == null) { throw new ArgumentNullException(nameof(reader)); }
   if (linesToSkip < 0) { throw new ArgumentOutOfRangeException(nameof(linesToSkip)); }

   for (var i = 0; i < linesToSkip; ++i)
   {
       var line = await reader.ReadLineAsync().ConfigureAwait(false);
       if (line == null) { break; }
   }
}
```

Split this method into two, one handling parameters check and the other handling the asynchronous code.

2020-02-25 205

CODE_SMELL MAJOR OPEN 15min effort

What is the issue?

```
public static async Task SkipLinesAsync(this TextReader reader, int linesToSkip) // Noncompliant
{
   if (reader == null) { throw new ArgumentNullException(nameof(reader)); }
   if (linesToSkip < 0) { throw new ArgumentOutOfRangeException(nameof(linesToSkip)); }

   for (var i = 0; i < linesToSkip; ++i)
   {
       var line = await reader.ReadLineAsync().ConfigureAwait(false);
       if (line == null) { break; }
   }
}
```

Remove the unnecessary Boolean literal(s). 2020-02-25 240

CODE_SMELL MINOR OPEN 5min effort

What is the issue?

```
if (booleanMethod() == true) { /* ... */ }
if (booleanMethod() == false) { /* ... */ }
if (booleanMethod() || false) { /* ... */ }
doSomething(!false);
doSomething(booleanMethod() == true);

booleanVariable = booleanMethod() ? true : false;
booleanVariable = booleanMethod() ? true : exp;
booleanVariable = booleanMethod() ? false : exp;
booleanVariable = booleanMethod() ? exp : true;
booleanVariable = booleanMethod() ? exp : false;

for (var x = 0; true; x++)
{
...
}
```

File Name: CodeJam:wwwroot/lib/prism/prism.js

Unnecessary semicolon. 2019-01-16 661

CODE_SMELL MINOR OPEN 2min effort

What is the issue?

```
var x = 1;; // Noncompliant
```

```
function foo() {
};  // Noncompliant
```

'i' is already defined.

2019-01-16 1420

CODE_SMELL MAJOR OPEN 20min effort

What is the issue?

```
var a = 'foo';
function a() {}   // Noncompliant
console.log(a);   // prints "foo"

function myFunc(arg) {
 var arg = "event"; // Noncompliant, argument value is lost
}

fun(); // prints "bar"

function fun() {
 console.log("foo");
}

fun(); // prints "bar"

function fun() {  // Noncompliant
 console.log("bar");
}

fun(); // prints "bar"
```

'clone' is already defined.

2018-08-13 73

CODE_SMELL MAJOR OPEN 20min effort

What is the issue?

```
var a = 'foo';
function a() {}   // Noncompliant
console.log(a);   // prints "foo"

function myFunc(arg) {
 var arg = "event"; // Noncompliant, argument value is lost
}
```

```
fun(); // prints "bar"

function fun() {
 console.log("foo");
}

fun(); // prints "bar"

function fun() {  // Noncompliant
 console.log("bar");
}

fun(); // prints "bar"
```

'lang' is already declared in the upper scope. 2018-08-13 89

CODE_SMELL MAJOR OPEN 5min effort

What is the issue?

Overriding or shadowing a variable declared in an outer scope can strongly impact the readability, and therefore the maintainability, of a piece of code. Further, it could lead maintainers to introduce bugs because they think they're using one variable but are really using another.

See

- CERT, DCL01-C. - Do not reuse variable names in subscopes
- CERT, DCL51-J. - Do not shadow or obscure identifiers in subscopes

Unexpected comma in middle of array. 2018-08-13 192

CODE_SMELL MAJOR OPEN 5min effort

What is the issue?

```
let a = [1, , 3, 6, 9];  // Noncompliant
```

'Token' is already declared in the upper scope. 2018-08-13 280

CODE_SMELL MAJOR OPEN 5min effort

What is the issue?

Overriding or shadowing a variable declared in an outer scope can strongly impact the readability, and therefore the maintainability, of a piece of code. Further, it could lead maintainers to introduce bugs because they think they're using one variable but are really using another.

See

- CERT, DCL01-C. - Do not reuse variable names in subscopes
- CERT, DCL51-J. - Do not shadow or obscure identifiers in subscopes

'match' is already defined. 2018-08-13 357

CODE_SMELL MAJOR OPEN 20min effort

What is the issue?

```
var a = 'foo';
function a() {}   // Noncompliant
console.log(a);   // prints "foo"

function myFunc(arg) {
 var arg = "event"; // Noncompliant, argument value is lost
}

fun(); // prints "bar"

function fun() {
 console.log("foo");
}

fun(); // prints "bar"

function fun() {  // Noncompliant
 console.log("bar");
}

fun(); // prints "bar"
```

'from' is already defined. 2018-08-13 373

CODE_SMELL MAJOR OPEN 20min effort

What is the issue?

```
var a = 'foo';
function a() {}   // Noncompliant
console.log(a);   // prints "foo"

function myFunc(arg) {
```

```
 var arg = "event"; // Noncompliant, argument value is lost
}

fun(); // prints "bar"

function fun() {
 console.log("foo");
}

fun(); // prints "bar"

function fun() {  // Noncompliant
 console.log("bar");
}

fun(); // prints "bar"
```

'match' is already defined.

2018-08-13 374

CODE_SMELL MAJOR OPEN 20min effort

What is the issue?

```
var a = 'foo';
function a() {}   // Noncompliant
console.log(a);   // prints "foo"

function myFunc(arg) {
 var arg = "event"; // Noncompliant, argument value is lost
}

fun(); // prints "bar"

function fun() {
 console.log("foo");
}

fun(); // prints "bar"

function fun() {  // Noncompliant
 console.log("bar");
}

fun(); // prints "bar"
```

'to' is already defined.

2018-08-13 375

CODE_SMELL MAJOR OPEN 20min effort

What is the issue?

```
var a = 'foo';
function a() {}   // Noncompliant
console.log(a);   // prints "foo"

function myFunc(arg) {
 var arg = "event"; // Noncompliant, argument value is lost
}

fun(); // prints "bar"

function fun() {
 console.log("foo");
}

fun(); // prints "bar"

function fun() {  // Noncompliant
 console.log("bar");
}

fun(); // prints "bar"
```

'lang' is already declared in the upper scope.

2018-08-13 505

CODE_SMELL MAJOR OPEN 5min effort

What is the issue?

Overriding or shadowing a variable declared in an outer scope can strongly impact the readability, and therefore the maintainability, of a piece of code. Further, it could lead maintainers to introduce bugs because they think they're using one variable but are really using another.

See

- CERT, DCL01-C. - Do not reuse variable names in subscopes
- CERT, DCL51-J. - Do not shadow or obscure identifiers in subscopes

Unnecessary semicolon.

2018-08-13 551

CODE_SMELL MINOR OPEN 2min effort

What is the issue?

```
var x = 1;; // Noncompliant

function foo() {
};  // Noncompliant
```

File Name: CodeJam:wwwroot/js/admin.js

'i' is already declared in the upper scope. 2018-08-21 9

CODE_SMELL MAJOR OPEN 5min effort

What is the issue?

Overriding or shadowing a variable declared in an outer scope can strongly impact the readability, and therefore the maintainability, of a piece of code. Further, it could lead maintainers to introduce bugs because they think they're using one variable but are really using another.

See

- CERT, DCL01-C. - Do not reuse variable names in subscopes
- CERT, DCL51-J. - Do not shadow or obscure identifiers in subscopes

Define this function outside of a loop. 2018-08-21 17

CODE_SMELL MAJOR OPEN 30min effort

What is the issue?

```
var funs = [];
for (var i = 0; i < 13; i++) {
 funs[i] = function() { // Non-Compliant
   return i;
 };
}
console.log(funs[0]()); // 13 instead of 0
console.log(funs[1]()); // 13 instead of 1
console.log(funs[2]()); // 13 instead of 2
console.log(funs[3]()); // 13 instead of 3
...
```

'event' is already declared in the upper scope. 2018-08-21 17

CODE_SMELL MAJOR OPEN 5min effort

What is the issue?

Overriding or shadowing a variable declared in an outer scope can strongly impact the readability, and therefore the maintainability, of a piece of code. Further, it could lead maintainers to introduce bugs because they think they're using one variable but are really using another.

See

- CERT, DCL01-C. - Do not reuse variable names in subscopes
- CERT, DCL51-J. - Do not shadow or obscure identifiers in subscopes

Define this function outside of a loop. 2018-08-21 20

CODE_SMELL MAJOR OPEN 30min effort

What is the issue?

```
var funs = [];
for (var i = 0; i < 13; i++) {
 funs[i] = function() { // Non-Compliant
   return i;
 };
}
console.log(funs[0]()); // 13 instead of 0
console.log(funs[1]()); // 13 instead of 1
console.log(funs[2]()); // 13 instead of 2
console.log(funs[3]()); // 13 instead of 3
...
```

'i' is already declared in the upper scope. 2018-08-21 143

CODE_SMELL MAJOR OPEN 5min effort

What is the issue?

Overriding or shadowing a variable declared in an outer scope can strongly impact the readability, and therefore the maintainability, of a piece of code. Further, it could lead maintainers to introduce bugs because they think they're using one variable but are really using another.

See

- CERT, DCL01-C. - Do not reuse variable names in subscopes

- CERT, DCL51-J. - Do not shadow or obscure identifiers in subscopes