TURING-GALLERY-FRONT

VERSION 1.0

CODE ANALYSIS

By: default

2020-02-10

CONTENT

INTRODUCTION

This document contains results of the code analysis of TURING-GALLERY-FRONT.

CONFIGURATION

- Quality Profiles
  - Names: Sonar way [CSS]; Sonar way [TypeScript]; Sonar way [HTML];
  - Files: AW-vB-6R6MX5uumeTY8S.json; AW-vB_h16MX5uumeTZp4.json; AW-vB_Ue6MX5uumeTZfB.json;
- Quality Gate
  - Name: Sonar way
  - File: Sonar way.xml

SYNTHESIS

Quality Gate	Reliability	Security	Maintainability	Coverage	Duplication
ERROR	D	A	A	0.0 %	2.5 %

METRICS

	Cyclomatic Complexity	Cognitive Complexity	Lines of code per file	Comment density (%)	Coverage	Duplication (%)
Min	0.0	0.0	0.0	0.0	0.0	0.0
Max	262.0	60.0	3324.0	80.0	0.0	96.2

TURING-GALLERY-FRONT

VOLUME

Language	Number
CSS	1256
TypeScript	1298
HTML	770
Total	3324

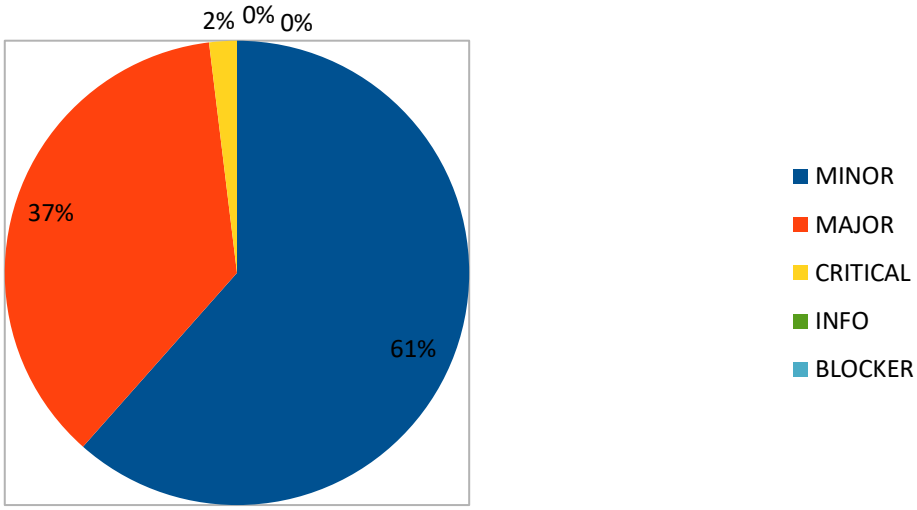
ISSUES COUNT BY SEVERITY AND TYPE

Type	Severity	Number
VULNERABILITY	BLOCKER	0
VULNERABILITY	CRITICAL	0
VULNERABILITY	MAJOR	0
VULNERABILITY	MINOR	0
VULNERABILITY	INFO	0
BUG	BLOCKER	0
BUG	CRITICAL	1
BUG	MAJOR	2
BUG	MINOR	29
BUG	INFO	0
CODE_SMELL	BLOCKER	0
CODE_SMELL	CRITICAL	0
CODE_SMELL	MAJOR	17
CODE_SMELL	MINOR	3

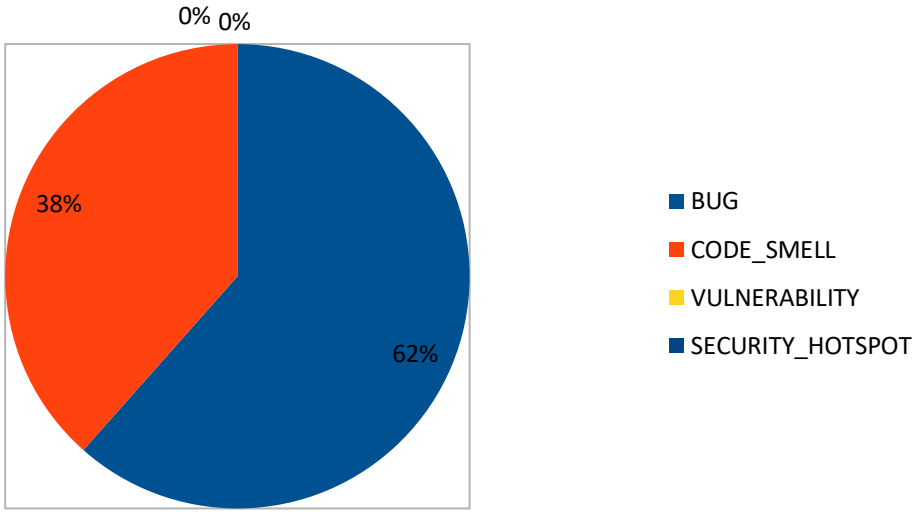
CODE_SMELL	INFO	0
SECURITY_HOTSPOT	BLOCKER	0
SECURITY_HOTSPOT	CRITICAL	0
SECURITY_HOTSPOT	MAJOR	0
SECURITY_HOTSPOT	MINOR	0
SECURITY_HOTSPOT	INFO	0

CHARTS

Number of issues by severity



Number of issues by type



ISSUES

Name	Description	Type	Severity	Number
Selectors should be known	HTML, SVG, and MathML define the selectors which can be used in a CSS. A selector that is not part of them is likely to be a typo or a misunderstanding of the CSS syntax. Noncompliant Code Example field {} ul list {} Compliant Solution input {} ul li {}	BUG	CRITICAL	1
"<th>" tags should have "id" or "scope"	Associating <table> headers, i.e. <th> elements, with their <td> cells enables screen readers to announce the header prior to the data. This considerably increases the accessibility of tables to visually impaired	BUG	MAJOR	2

attributes	users. There are two ways of doing it: Adding a scope attribute to <th> headers. Adding an id attribute to <th> headers and a headers attribute to every <td> element. It is recommended to add scope attributes to <th> headers whenever possible. Use <th id="..."> and <td headers="..."> only when <th scope="..."> is not capable of associating cells to their headers. This happens for very complex tables which have headers splitting the data in multiple subtables. See W3C WAI Web Accessibility Tutorials for more information. Note that complex tables can often be split into multiple smaller tables, which improves the user experience. This rule raises an issue when a <th> element has neither id nor scope attributes set. Noncompliant Code Example <table border="1"> <caption>Contact Information</caption> <tr> <td></td> <th>Name</th> <!-- Non-Compliant --> <th>Phone#</th> <!-- Non-Compliant --> <th>City</th> <!-- Non-Compliant --> </tr> <tr> <td>1.</td> <th>Joel Garner</th> <!-- Non-Compliant --> <td>412-212-5421</td> <td>Pittsburgh</td> </tr> <tr> <td>2.</td> <th>Clive Lloyd</th> <!-- Non-Compliant --> <td>410-306-1420</td> <td>Baltimore</td> </tr> </table> Compliant Solution <table border="1"> <caption>Contact Information</caption> <tr> <td></td> <th scope="col">Name</th> <!-- Compliant --> <th scope="col">Phone#</th> <!-- Compliant --> <th scope="col">City</th> <!-- Compliant --> </tr> <tr> <td>1.</td> <th scope="row">Joel Garner</th> <!-- Compliant --> <td>412-212-5421</td> <td>Pittsburgh</td> </tr> <tr> <td>2.</td> <th scope="row">Clive Lloyd</th> <!-- Compliant --> <td>410-306-1420</td> <td>Baltimore</td> </tr> </table> or: <table border="1"> <caption>Contact Information</caption> <tr> <td></td> <th id="name">Name</th> <!-- Compliant --> <th id="phone">Phone#</th> <!-- Compliant --> <th id="city">City</th>

	<!-- Compliant --> </tr> <tr> <td>1.</td> <th id="person1" headers="name">Joel Garner</th> <!-- Compliant --> <td headers="phone person1">412-212-5421</td> <td headers="city person1">Pittsburgh</td> </tr> <tr> <td>2.</td> <th id="person2" headers="name">Clive Lloyd</th> <!-- Compliant --> <td headers="phone person2">410-306-1420</td> <td headers="city person2">Baltimore</td> </tr> </table> See * WCAG2, 1.3.1 - Info and Relationships WCAG2, H43 - Using id and headers attributes to associate data cells with header cells in data tables			
"" and "" tags should be used	The / and /<i> tags have exactly the same effect in most web browsers, but there is a fundamental difference between them: and have a semantic meaning whereas and <i> only convey styling information like CSS. While can have simply no effect on a some devices with limited display or when a screen reader software is used by a blind person, will: Display the text bold in normal browsers Speak with lower tone when using a screen reader such as Jaws Consequently: in order to convey semantics, the and <i> tags shall never be used, in order to convey styling information, the and <i> should be avoided and CSS should be used instead. Noncompliant Code Example <i>car</i> <!-- Noncompliant --> train <!-- Noncompliant --> Compliant Solution car train Exceptions This rule is relaxed in case of icon fonts usage. <i class="..." aria-hidden="true" /> <!-- Compliant icon fonts usage -->	BUG	MINOR	26
Image, area and button with image tags should have an "alt" attribute	The alt attribute provides a textual alternative to an image. It is used whenever the actual image cannot be rendered. Common reasons for that include: The image can no longer be found Visually impaired users using a screen reader software Images loading is disabled, to reduce data consumption on mobile phones It is also very important to not set an alt attribute to a non-informative value. For example <img ... alt="logo"> is useless as it doesn't give any information to the user. In this case, as for any other decorative image, it is better to use a CSS background	BUG	MINOR	1

image instead of an <img> tag. If using CSS background-image is not possible, an empty alt="" is tolerated. See Exceptions bellow. This rule raises an issue when an <input type="image"> tag or an <area> tag have no alt attribute or their alt attribute has an empty string value. an <img> tag has no alt attribute. Noncompliant Code Example <!-- Noncompliant --> <input type="image" src="bar.png" /> <!-- Noncompliant --> <input type="image" src="bar.png" alt="" /> <!-- Noncompliant --> <map id="map1" name="map1"> <area shape="rect" coords="0,0,42,42" href="bedroom.html"/> <!-- Noncompliant --> <area shape="rect" coords="0,0,21,21" href="lounge.html" alt=""/> <!-- Noncompliant --> </map> Compliant Solution <input type="image" src="bar.png" alt="Textual description of bar.png" /> <map id="map1" name="map1"> <area shape="rect" coords="0,0,42,42" href="bedroom.html" alt="Bedroom" /> <area shape="rect" coords="0,0,21,21" href="lounge.html" alt="Lounge"/> </map> Exceptions <img> tags with empty string alt="" attributes won't raise any issue. However this technic should be used in two cases only: When the image is decorative and it is not possible to use a CSS background image. For example, when the decorative <img> is generated via javascript with a source image coming from a database, it is better to use an <img alt=""> tag rather than generate CSS code. <li *ngFor="let image of images"> <img [src]="image" alt=""> When the image is not decorative but it's alt text would repeat a nearby text. For example, images contained in links should not duplicate the link's text in their alt attribute, as it would make the screen reader repeat the text twice.   A blooming tulip In all other cases you should use CSS background images. See W3C WAI Web Accessibility Tutorials for more information. See WCAG2, H24 - Providing text alternatives for the area elements of image maps WCAG2, H36 - Using alt attributes on images used as submit buttons WCAG2, H37 - Using alt attributes on img elements WCAG2, H67 - Using null alt text and no title attribute on img elements for images that AT should ignore WCAG2, H2 - Combining adjacent image and text links for the same resource WCAG2, 1.1.1 - Non-text Content

	WCAG2, 2.4.4 - Link Purpose (In Context) WCAG2, 2.4.9 - Link Purpose (Link Only)			
"<table>" tags should have a description	In order to be accessible to visually impaired users, it is important that tables provides a description of its content before the data is accessed. The simplest way to do it, and also the one recommended by WCAG2 is to add a <caption> element inside the <table>. Other technics this rule accepts are: referencing the description element with an aria-describedby attribute in the <table>. embedding the <table> inside a <figure> which also contains a <figcaption>. adding a summary attribute to the <table> tag. However note that this attribute has been deprecated in HTML5. See W3C WAI Web Accessibility Tutorials for more information. This rule raises an issue when a <table> has neither of the previously mentioned description mechanisms. Noncompliant Code Example <table> <!-- Noncompliant --> ... <table> Compliant Solution Adding a <caption> element. <table> <caption>New York City Marathon Results 2013</caption> ... </table> Adding an aria-describedby attribute. <p id="mydesc">New York City Marathon Results 2013</p> <table aria-describedby="mydesc"> ... </table> Embedding the table in a <figure> which also contains a <figcaption>. <figure> <figcaption>New York City Marathon Results 2013</figcaption> <table> ... </table> </figure> Adding a summary attribute. <table summary="New York City Marathon Results 2013"> ... </table> Exceptions No issue will be raised on <table> used for layout purpose, i.e. when it contains a role attribute set to "presentation" or "none". Note that using <table> for layout purpose is a bad practice. No issue will be raised either on <table> containing an aria-hidden attribute set to "true". See WCAG2, 1.3.1 - Info and Relationships WCAG2, H39 - Using caption elements to associate data table captions with data tables	BUG	MINOR	2
Sections of code should not be commented out	Programmers should not comment out code as it bloats programs and reduces readability. Unused code should be deleted and can be retrieved from source control history if required.	CODE_SMELL	MAJOR	1
Selectors should not be	Duplication of selectors might indicate a copy-paste mistake. The rule detects the following kinds of duplications: within a list of selectors in a single rule set for duplicated	CODE_SMELL	MAJOR	5

duplicated	selectors in different rule sets within a single stylesheet. Noncompliant Code Example .foo, .bar, .foo { ... } /* Noncompliant */ .class1 { ... } .class1 { ... } /* Noncompliant */ Compliant Solution .foo, .bar { ... } .class1 { ... } .class2 { ... }			
CSS files should not be empty	This rule raises an issue when a CSS file is empty (ie: containing only spaces).	CODE_SMELL	MAJOR	11
Extra semicolons should be removed	Extra semicolons (;) are usually introduced by mistake, for example because: It was meant to be replaced by an actual statement, but this was forgotten. There was a typo which lead the semicolon to be doubled, i.e. ;;. There was a misunderstanding about where semicolons are required or useful. Noncompliant Code Example var x = 1;; // Noncompliant function foo() { }; // Noncompliant Compliant Solution var x = 1; function foo() { } See CERT, MSC12-C. - Detect and remove code that has no effect or is never executed CERT, MSC51-J. - Do not place a semicolon immediately following an if, for, or while condition CERT, EXP15-C. - Do not place a semicolon on the same line as an if, for, or while statement	CODE_SMELL	MINOR	2
Boolean checks should not be inverted	It is needlessly complex to invert the result of a boolean comparison. The opposite comparison should be made instead. Note that this rule requires Node.js to be available during analysis. Noncompliant Code Example if (!(a === 2)) { ... } // Noncompliant Compliant Solution if (a !== 2) { ... }	CODE_SMELL	MINOR	1