

Industrialisation et Continuous Delivery

GÉRER SES SOURCES ET LES LIER À SON INTÉGRATION CONTINUE
AVEC L'UN DES OUTILS LES PLUS POPULAIRES

SonarQube

* Le monde est tel que nous le façonnons.

The world is how we shape it*

sopra  steria

Vos intervenants



Benoit POIRIER

Architecte Solution

Sopra Steria

benoit.poirier@soprasteria.com



Fabrice ROULAND

Expert Technique

Sopra Steria

fabrice.rouland@soprasteria.com

Ce que nous allons aborder

Les bases de SonarQube

Les métriques

Les Quality Profiles

Les Quality Gates

Rappel

POURQUOI AVOIR UNE ANALYSE DE CODE ?

— Pourquoi une analyse de code ?

- └ Pour garantir la qualité par rapport à un niveau attendu
- └ Pour estimer la dette technique de la solution
- └ Pour respecter nos engagements contractuels (exigences spécifiques)
- └ Pour faire grandir nos collaborateurs

— La dette technique peut être

- └ Consciente :
 - On a pris un raccourci pour respecter un délai.
 - On a ajouté des fonctionnalités sans faire le refactoring nécessaire
- └ Inconsciente :
 - Non respect de règles

— La qualité logicielle aura un lien direct sur la maintenabilité :

- └ Stabilité de la solution
- └ Coûts des évolutions ou des corrections

01

Les bases de SonarQube

SE RECENTRER SUR L'ESSENTIEL : LA VALEUR MÉTIER

- SonarQube est un logiciel *open source* de mesure de la qualité du code source de projets de développement.
- Il est développé par SonarSource, distribué sous licence GNU GPLv3.
- Il permet d'obtenir des informations sur la qualité au niveau du projet, du fichier ou d'un module et donne des indications sur chaque problème de qualité détecté et le temps de remédiation.
- Il existe depuis 2007. Utilisé par la DSI du CNRS, le Ministère de la Défense, CISCO, Airbus, AirFrance, Boeing, BMW, Fiat, Renault, Volvo, SNCF, Banque Postale, BNP Paribas, Société Générale, MasterCard, Bouygues Telecom, Bosch, Amazon, ebay, PayPal, DHL, Oracle.

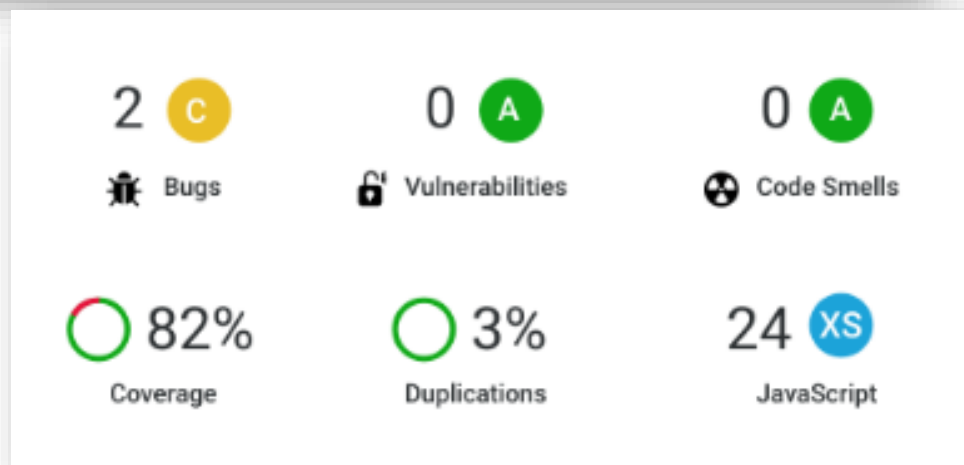
SonarQube

Ca sert à quoi ?

```
4 -
5 - public int getNumberOfPawsPlusOne(String animal) {
6 - | 2 String trimmedAnimal = 1 trim(animal);
7 -
8 - int paws;
9 - if ("dog".equals(trimmedAnimal)) {
10 -     paws = 4;
11 - } else if (3 trimmedAnimal.equals("Antonio")) { // NullPointerException raised
12 -
13 -     paws = 1;
14 - } else if ("Centipede".equals(trimmedAnimal)) {
15 -     paws = 100;
16 - } else {
17 -     throw new RuntimeException(String.format("Unknown Animal %s", trimmedAnimal));
18 - }
19 - // Quizz: What is the problem here?
20 - return paws++;
21 - }
```

A "NullPointerException" could be thrown; "trimmedAnimal" is nullable here. [See Rule](#) 2 months ago L11 [cert, cwe](#)

🐛 Bug 🚨 Major 🔓 Open Not assigned 10min effort



Pull Request Decoration

SonarQube Code Analysis

✓ Success

🕒 ran 8 days ago in 2 minutes

🔗 6c54458 by @andrea-guarino-sonarsource

🔗 SONARJAVA-3093

Quality Gate **Passed**

SonarQube

Ca sert à quoi ?

Couvrant tous les angles

Fiabilité

Évitez les bugs et les comportements indéfinis

Sécurité

Évitez les brèches ou les attaques

Maintenabilité

Facilitez les mises à jour de code et augmentez la vitesse des développeurs

Variables should not be self-assigned



Bug



Major



Cert

There is no reason to re-assign a variable to itself. Either this statement is redundant and should be removed, or the re-assignment is a mistake and some other value or variable was intended for the assignment instead.

Noncompliant Code Example

```
function setName(name) {  
    name = name;  
}
```

Compliant Solution

```
function setName(name) {  
    this.name = name;  
}
```

Classé par gravité

Mappé aux normes (cert, misra, cwe, sans, owasp, etc.)

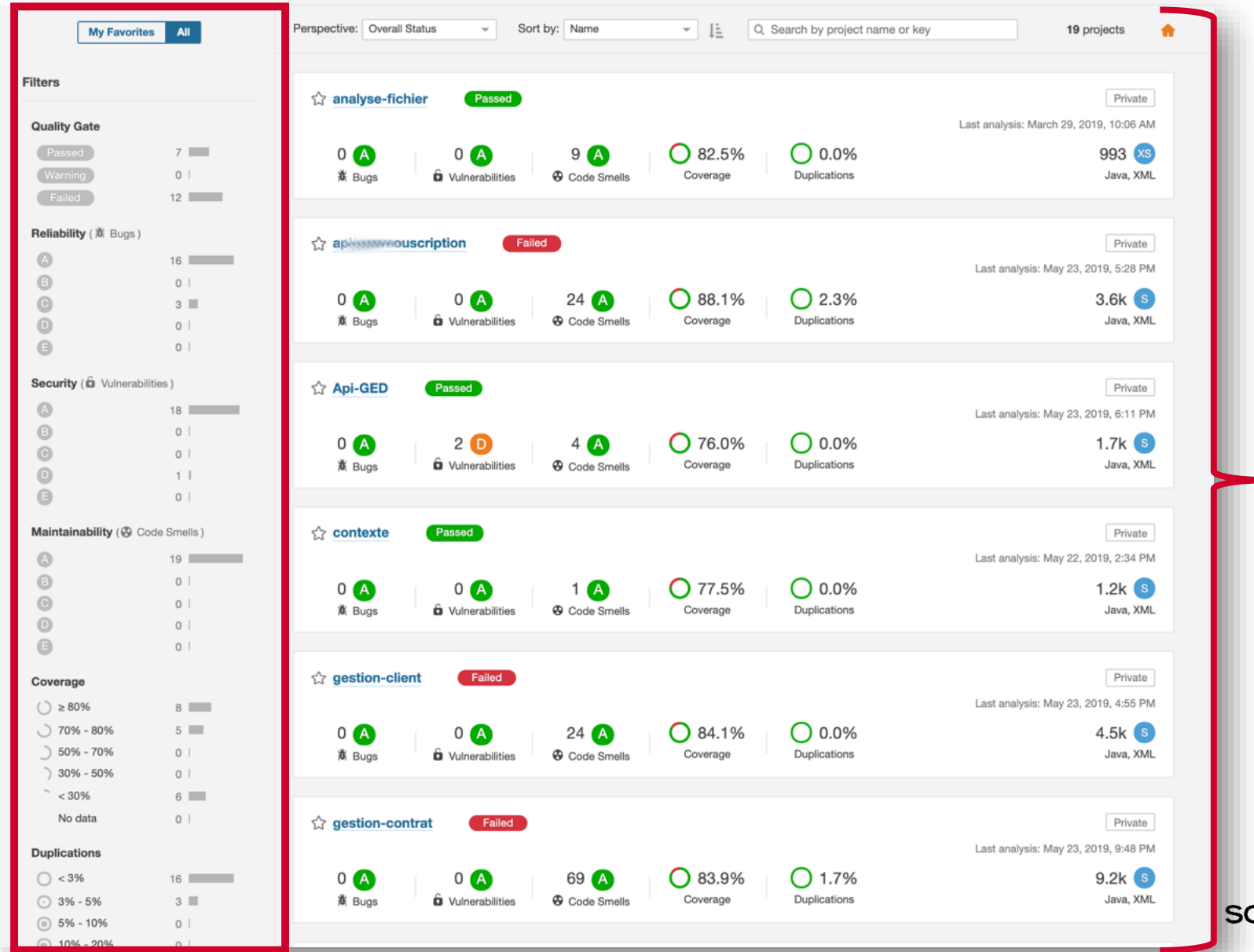
Entièrement documenté

Apprenez les meilleures pratiques et améliorez le codage

SonarQube

Ca sert à quoi ?

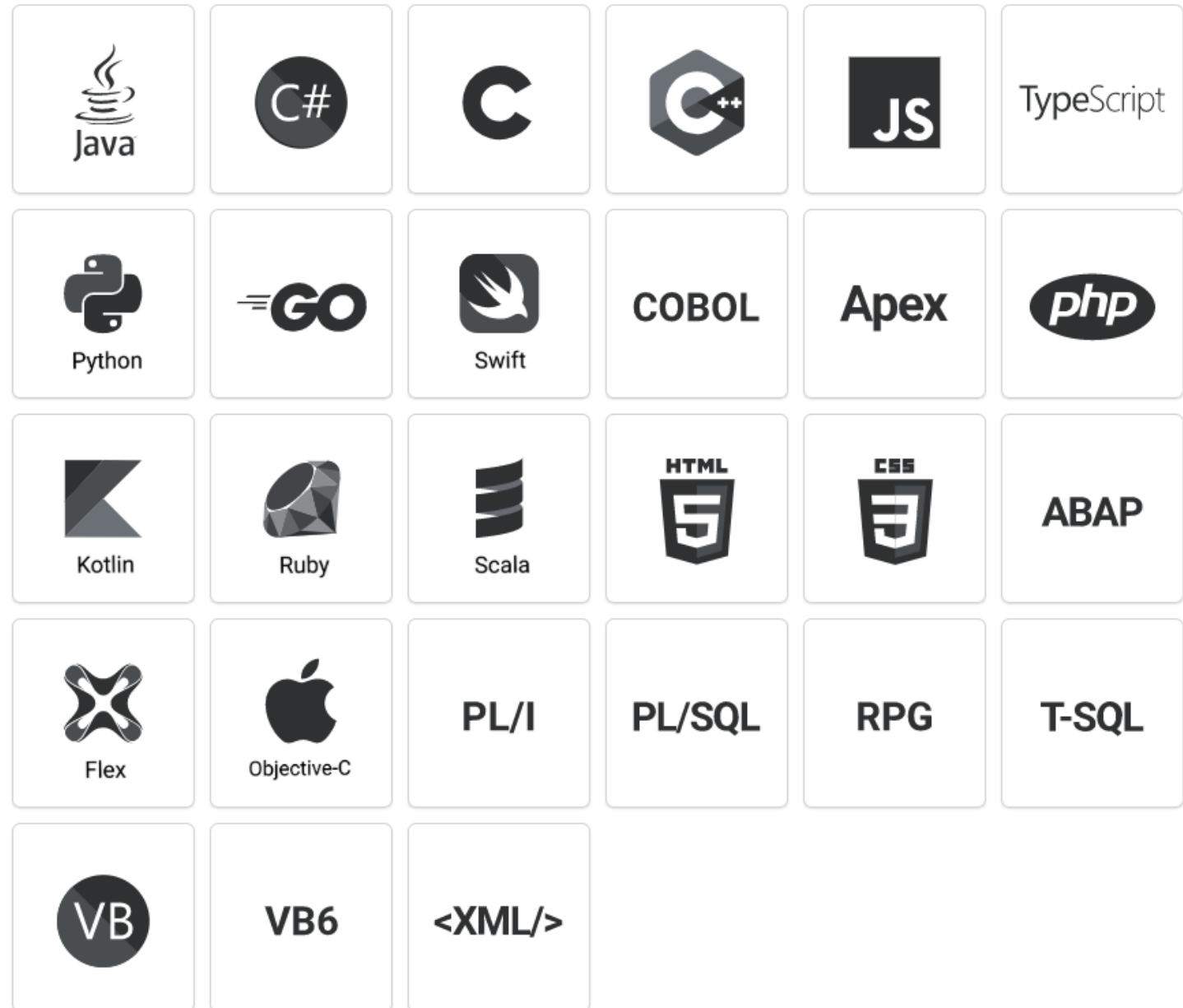
Statistiques sur
l'ensemble des projets
audités



Vue globale
des projets

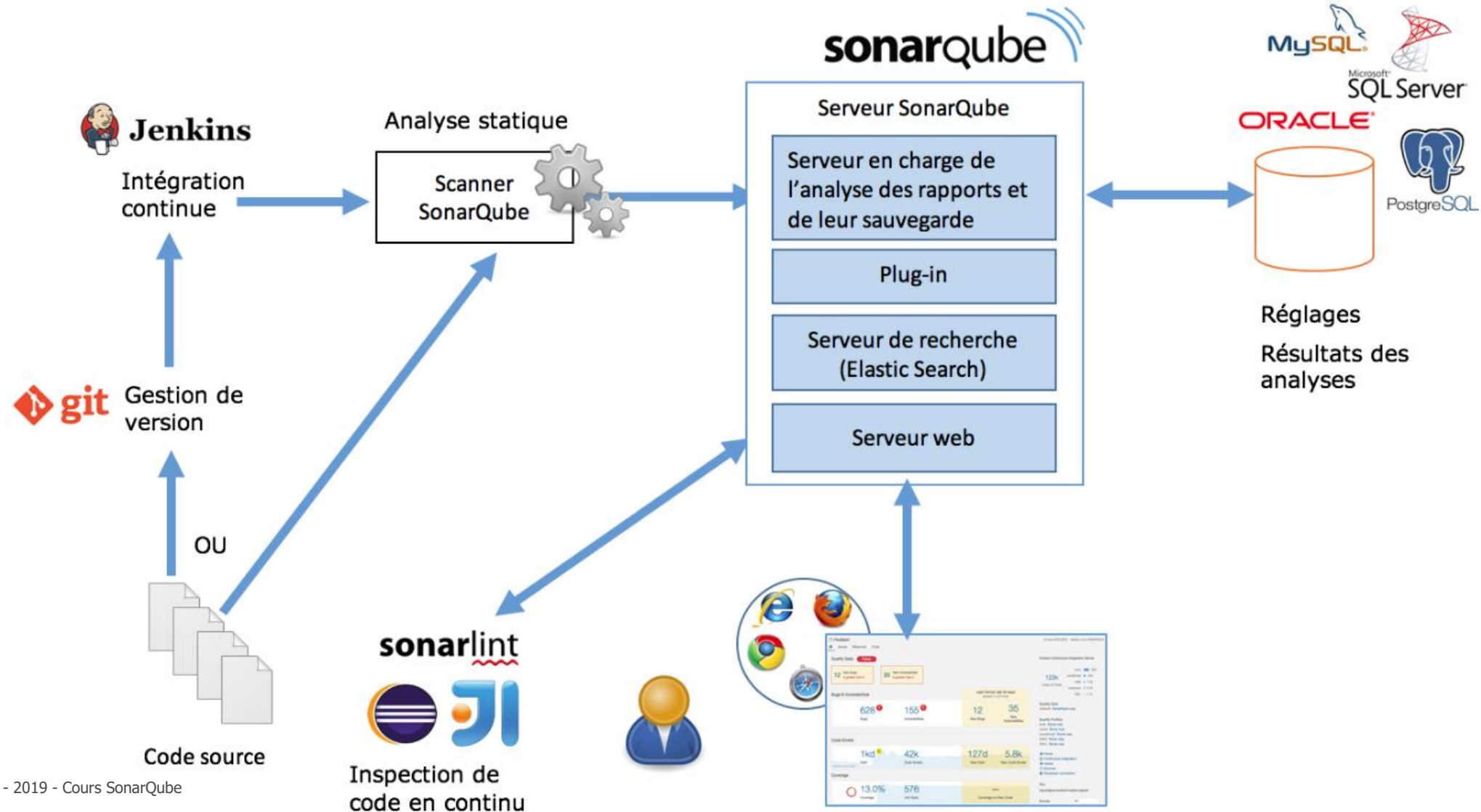
SonarQube

Il est multi-langues !



SonarQube

Une intégration native dans le pipeline d'intégration continue



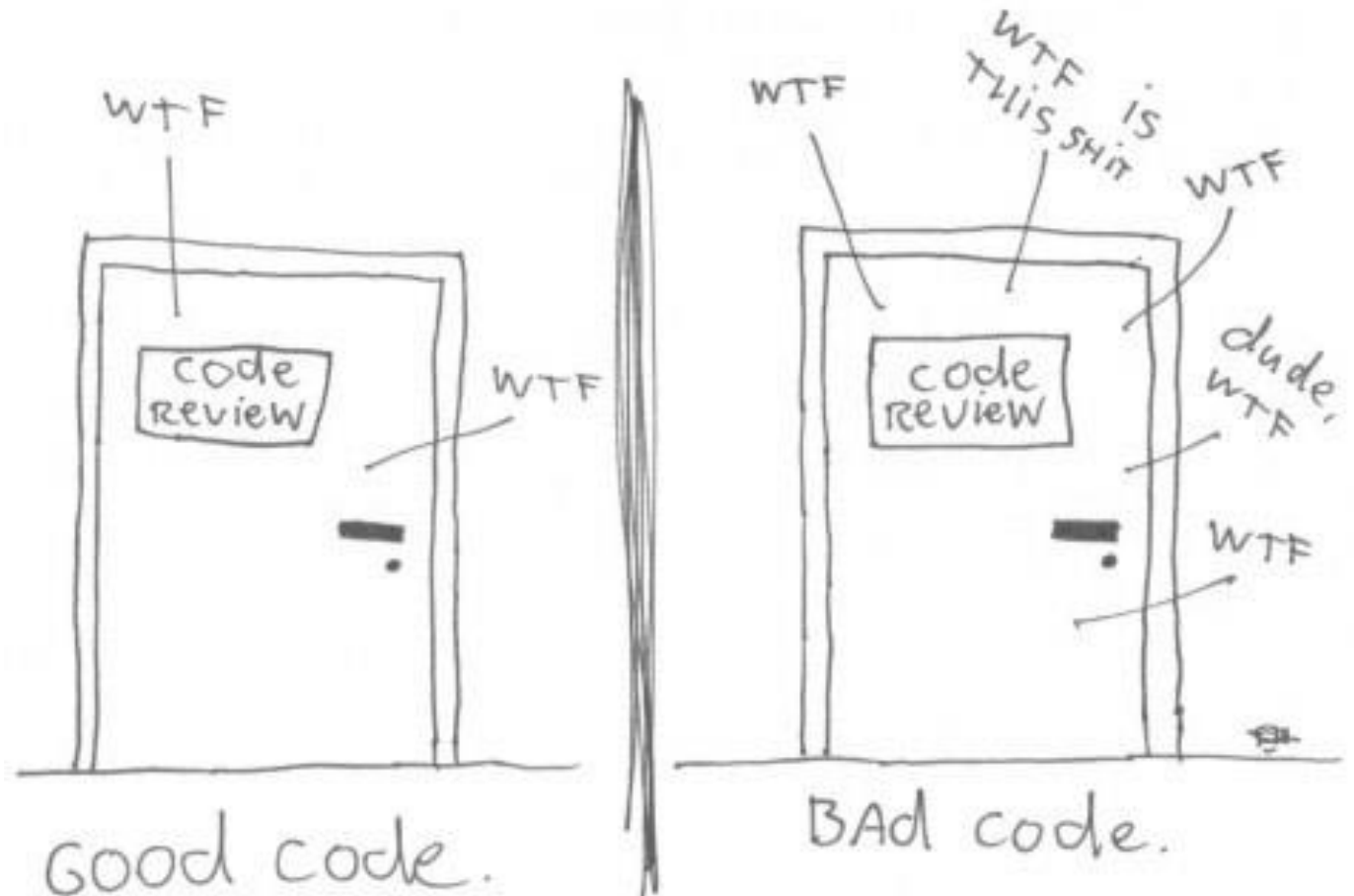
02

Les métriques

SonarQube

Quelles métriques ?

The ONLY valid MEASUREMENT
OF CODE QUALITY: WTFs/MINUTE



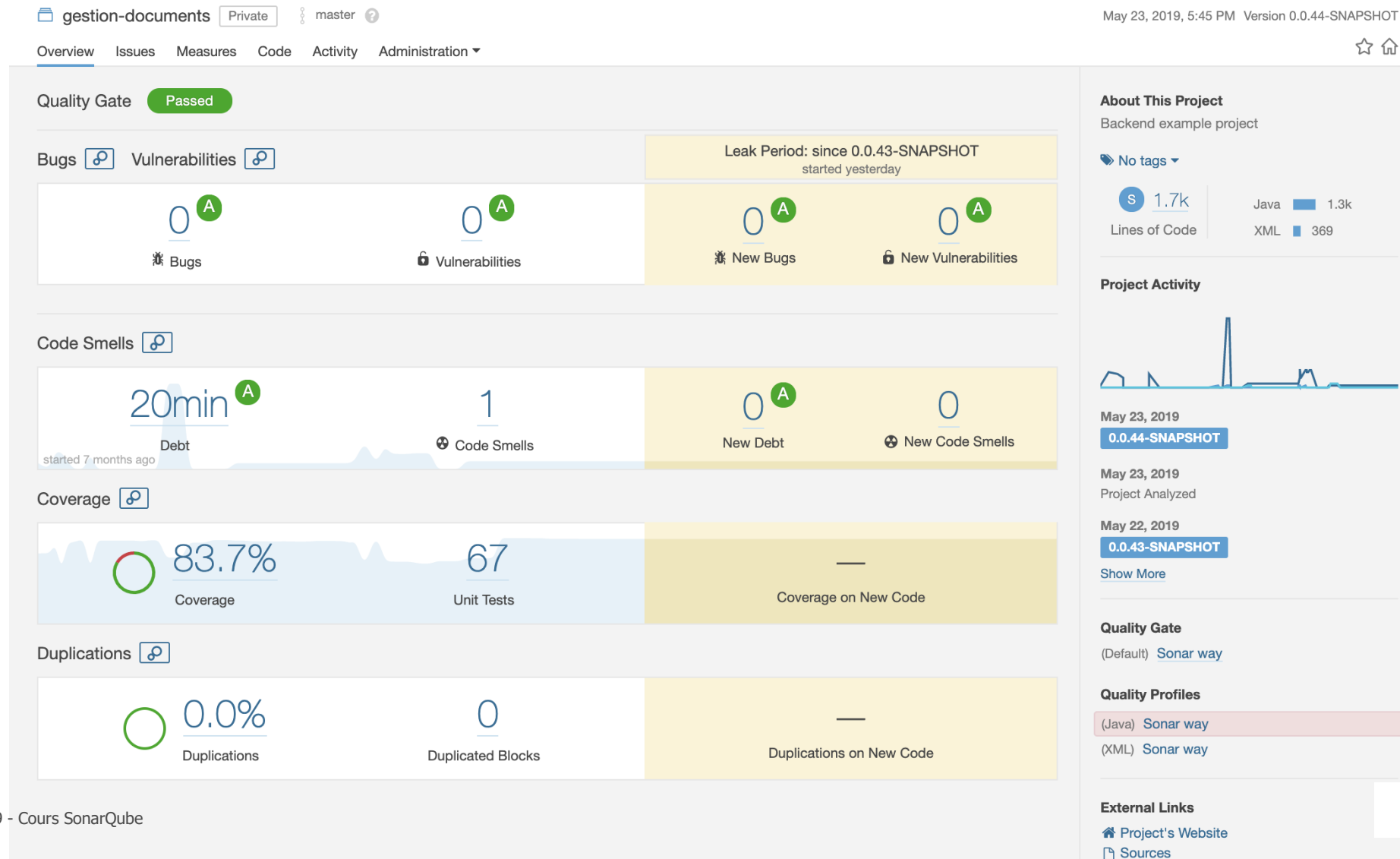
SonarQube

Les 7 axes de qualité



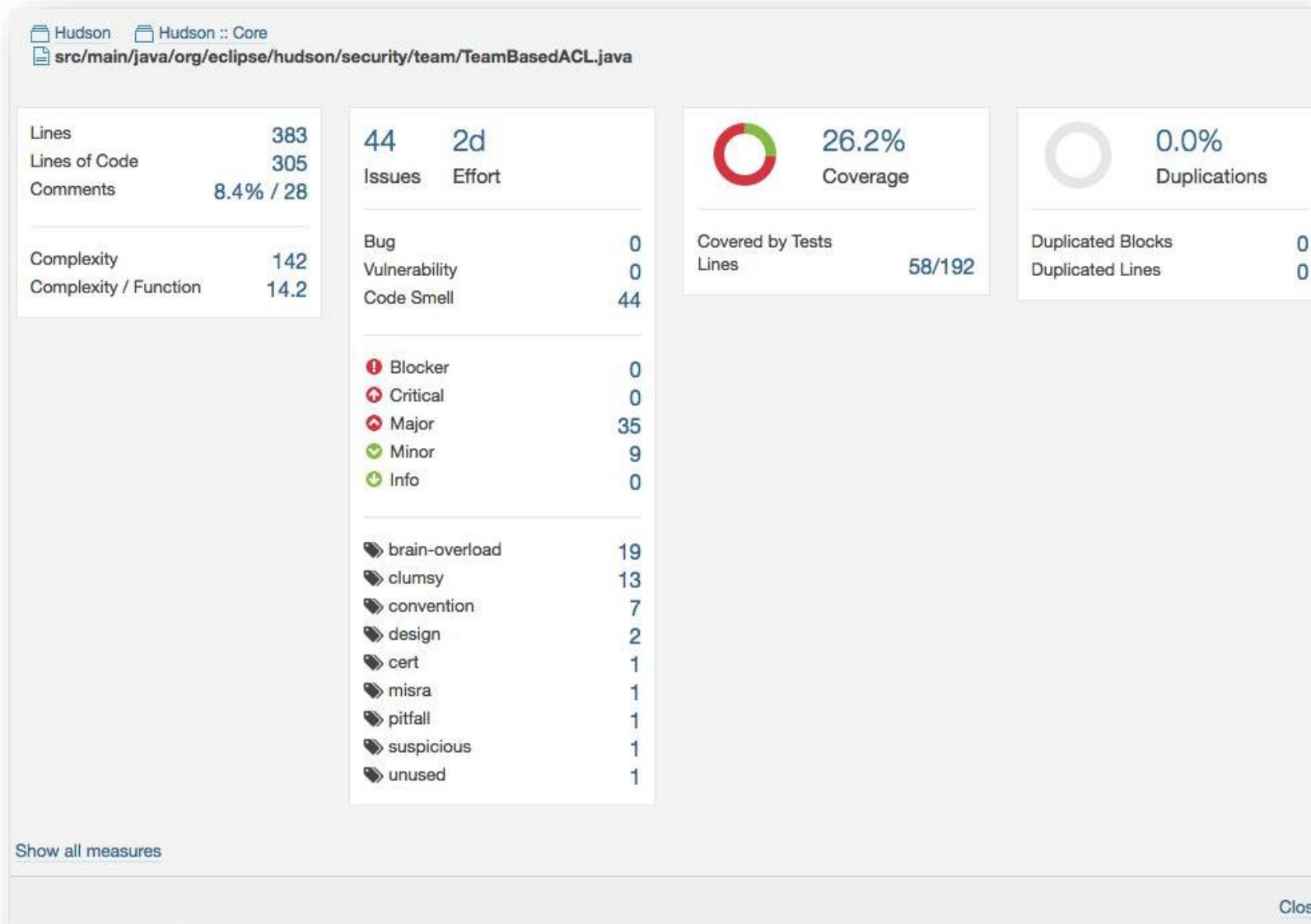
SonarQube

Les 7 axes de qualité : Vue projet / fonctionnalité



SonarQube

Les 7 axes de qualité : Vue fichier

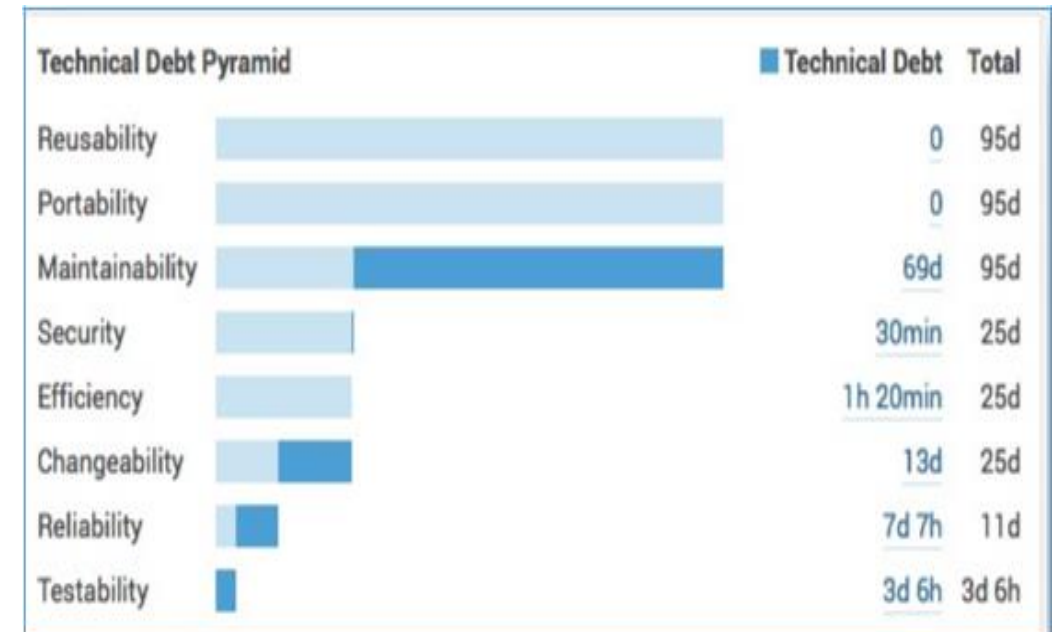


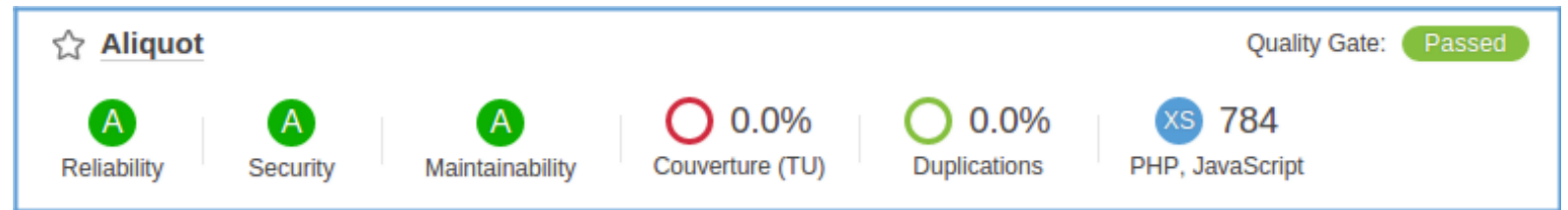
SonarQube

Dette technique

Lorsque le logiciel n'est pas basé sur les meilleurs pratiques possibles, une dette technique est introduite. Les intérêts ne cessent d'augmenter avec le temps (coûts supplémentaires dans le futur). Si le développeur code vite et mal, il contracte une dette technique qu'il faudra rembourser tout au long de la vie du projet (temps de développement de plus en plus longs, bugs de plus en plus fréquents).

La dette peut être non intentionnelle (non respect des règles de codages, il ne devrait jamais y avoir de telle dette) ou intentionnelle : la qualité augmente la charge de travail, pour livrer le projet dans les temps il faut parfois ne pas respecter une conception idéale. Il est conseillé dans ce cas de sortir une nouvelle version après la livraison, corrigeant au plus tôt la dette.





Un tableau de bord montre les scores de maintenabilité, fiabilité et sécurité, le taux de couverture des tests, le taux de duplications, la taille du projet ainsi que les langages des sources.

Score	Fiabilité	Sécurité	Maintenabilité (*)
	0 bug	0 vulnérabilité	$0.00 \leq \text{ratio dette technique} \leq 0.05$
	Au moins 1 bug mineur	Au moins 1 vulnérabilité mineure	$0.06 \leq \text{ratio dette technique} \leq 0.1$
	Au moins 1 bug majeur	Au moins 1 vulnérabilité majeure	$0.11 \leq \text{ratio dette technique} \leq 0.20$
	Au moins 1 bug critique	Au moins 1 vulnérabilité critique	$0.21 \leq \text{ratio dette technique} \leq 0.5$
	Au moins 1 bug bloquant	Au moins 1 vulnérabilité bloquante	$0.51 \leq \text{ratio dette technique} \leq 1$
<i>Issue</i>	<i>Bug</i>	<i>Vulnerability</i>	<i>Code smell</i>

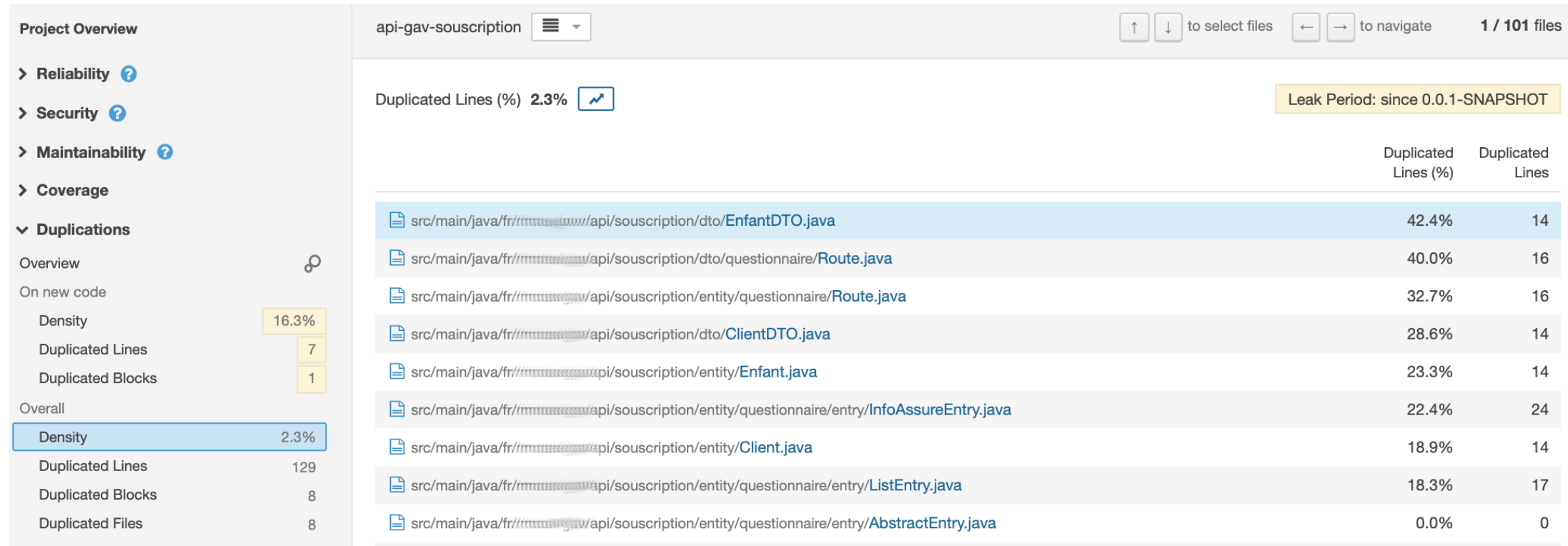
Le **ratio de la dette technique** est le ratio entre le coût pour remédier aux problèmes de type *code smell* et le coût de développement de l'application. La formule de calcul est la suivante :

coût total de remédiation des *issues* / (coût pour développer une ligne de code x nombre de lignes de code)

L'idée est de déterminer si réécrire l'application est plus rentable que corriger tous les problèmes. Lorsque le ratio est trop grand, il est préférable de réécrire l'application de zéro plutôt que d'essayer de réduire la dette en corrigeant les problèmes.

Duplication

- Forte probabilité de production d'anomalies
- Code inutile → Refactoring



Complexité

2 niveaux de complexité

- Complexité cyclomatique, appelée méthode de McCabe, représente le nombre de décision d'un algorithme en comptabilisant le nombre de chemin linéairement indépendant (nombre de if / else / case)
- Complexité cognitive, inventée en 2016 par SonarSource, s'affranchie de tout fondement mathématique. Le but est de qualifier l'effort nécessaire pour comprendre un algorithme.

Cyclomatic Complexity		262	
	src/main/java/fr/mimivideux/api/souscription/mapper/QuestionnaireFactory.java	14	
	src/main/java/fr/mimivideux/api/souscription/entity/questionnaire/Question.java	12	
	src/main/java/fr/mimivideux/api/souscription/service/QuestionnaireService.java	10	
	src/main/java/fr/mimivideux/api/souscription/mapper/DossierSouscriptionFullMapper.java	9	
	src/main/java/fr/mimivideux/api/souscription/entity/questionnaire/entry/InfoAssureEntry.java	8	
	src/main/java/fr/mimivideux/api/souscription/entity/questionnaire/entry/ListEntry.java	8	
	src/main/java/fr/mimivideux/api/souscription/entity/questionnaire/entry/QCMEntry.java	8	
	src/main/java/fr/mimivideux/api/souscription/entity/questionnaire/Section.java	8	
	src/main/java/fr/mimivideux/api/souscription/configuration/SwaggerConfig.java	8	
	src/main/java/fr/mimivideux/api/souscription/mapper/ClientMapper.java	7	
	src/main/java/fr/mimivideux/api/souscription/mapper/ProduitMapper.java	7	
	src/main/java/fr/mimivideux/api/souscription/controller/DossierSouscriptionController.java	6	
	src/main/java/fr/mimivideux/api/souscription/configuration/SecurityConfig.java	6	
	src/main/java/fr/mimivideux/api/souscription/entity/questionnaire/entry/AbstractEntry.java	5	
	src/main/java/fr/mimivideux/api/souscription/service/implementation/DossierSouscriptionService.java	5	
	src/main/java/fr/mimivideux/api/souscription/dto/questionnaire/entry/ListEntryDTO.java	5	
	src/main/java/fr/mimivideux/api/souscription/dto/questionnaire/entry/QCMEntryDTO.java	5	

Complexity ?

Cyclomatic Complexity	262
Cognitive Complexity	108

SonarQube

Couverture des tests unitaires

OverviewIssuesMeasuresCodeActivityAdministration

Project Overview

> Reliability

> Security

> Maintainability

Coverage

Overview

On new code

Lines to Cover0

Uncovered Lines0

Conditions to Cover0

Uncovered Conditions0

Overall

Coverage82.5%

Lines to Cover311

Uncovered Lines57

Line Coverage81.7%

Conditions to Cover78

Uncovered Conditions11

Condition Coverage85.9%

Tests

Unit Tests51

Errors0

Failures0

Skipped0

Success100%

Duration2s

analyse-fichier

↑↓to select files←→to navigate1 / 12 files

Coverage 82.5%

Leak Period: since 0.0.21

	Coverage	Uncovered Lines	Uncovered Conditions
src/main/java/com/.../api/analyse/dto/ResultatTraitement.java	0.0%	2	-
src/main/java/com/.../api/analyse/configuration/SecurityConfig.java	0.0%	23	-
src/main/java/com/.../api/analyse/ServletInitializer.java	0.0%	2	-
src/main/java/com/.../api/analyse/controller/ExtraireFichierController.java	50.0%	1	-
src/main/java/com/.../api/analyse/dto/College.java	56.3%	7	-
src/main/java/com/.../api/analyse/dto/Siret.java	60.7%	10	1
src/main/java/com/.../api/analyse/dto/CCN.java	63.6%	6	2
src/main/java/com/.../api/analyse/ApiAnalyseFichierApplication.java	66.7%	2	-
src/main/java/com/.../api/analyse/service/AnalyseSalariesService.java	92.7%	0	4
src/main/java/com/.../api/analyse/configuration/SwaggerConfig.java	95.8%	1	1
src/main/java/com/.../api/analyse/service/ExtraireFichierService.java	96.4%	2	0
src/main/java/com/.../api/analyse/service/ValiditeFichierService.java	96.9%	1	3

12 of 12 shown

Couverture de la classe

Lignes non couvertes

Conditions partiellement couvertes

SonarQube

Respect des standards

Temps
estimatif de
correction



Information
sur
l'anomalie

The detail view shows a code snippet with a highlighted line:

```
GetProduitResponse response = (GetProduitResponse) wsProduit.callWebService(request,  
    new SoapActionCallback("http://tempuri.org/IProduitService/GetProduit"));
```

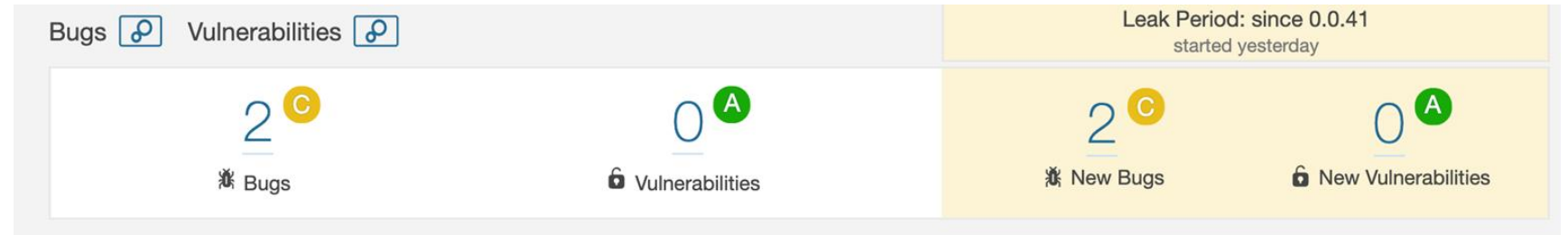
The issue is titled "Immediately return this expression instead of assigning it to the temporary variable 'response'." and is classified as a "Minor" Code Smell. It was reported 5 hours ago by user L227. The issue is currently "Open" and "Not assigned", with an estimated effort of 2min. A comment by user "clumsy" is visible.

Niveau de l'anomalie
parmi :

- Bloquant
- Critique
- Majeur
- Mineur
- Info

SonarQube

Anomalies



Anomalies pouvant entrainer des « plantages » ou des erreurs lors de l'exécution.

Exemple :

- Ressources non fermées
- Détection des boucles infinies
- Mauvais usages des classes (new BigDecimal(double))
- Création d'exception sans un throw

Filters Clear All Filters

Display Mode

Issues Effort

Type Clear

Bug 2

Vulnerability 0

src/.../api/referentiel/service/ProduitServiceImpl.java

☐ Call "Optional#isPresent()" before accessing the value. ... 5 hours ago L205 🔗 🔍 🔽

🐛 Bug 🔴 Major 🔵 Open 🔴 R rvz 10min effort Comment 🔗 🔍 🔽 cwe

☐ Call "Optional#isPresent()" before accessing the value. ... 5 hours ago L200 🔗 🔍 🔽

🐛 Bug 🔴 Major 🔵 Open 🔴 R rvz 10min effort Comment 🔗 🔍 🔽 cwe

2 of 2 shown

SonarQube

Au bout du compte

Est-ce que je peux livrer mon projet en production
aujourd'hui ?

PASSED

FAILED

En se basant

Absence d'anomalies bloquantes

Couverture de code supérieur à 80%

Et d'autres indicateurs liés à la Sécurité, Fiabilité et la maintenabilité

03

Les Quality Profiles

SonarQube

Le profil qualité

Un profil qualité comporte un jeu de règles qui seront utilisées pour l'analyse d'un projet.

Les profils permettent d'analyser des applications avec des exigences plus ou moins fortes.

L'onglet *Quality Profiles* permet de définir les règles à utiliser pour un profil.

Les profils sont listés par langage, il existe au moins un profil par langage (Sonar way).

Par exemple, trois profils sont définis pour le langage Java, celui utilisé par défaut est Sonar way, il comporte 381 règles.

sonarqube					
Projects Issues Rules Quality Profiles Quality Gates Administration					
C#, 1 profile(s)		Projects ?	Rules	Updated	Used
Sonar way	Built-in	Default	233	7 days ago	Never
CSS, 1 profile(s)		Projects ?	Rules	Updated	Used
Sonar way	Built-in	Default	23	7 days ago	Never
Flex, 1 profile(s)		Projects ?	Rules	Updated	Used
Sonar way	Built-in	Default	48	7 days ago	Never
Go, 1 profile(s)		Projects ?	Rules	Updated	Used
Sonar way	Built-in	Default	29	7 days ago	Never
Java, 1 profile(s)		Projects ?	Rules	Updated	Used
Sonar way	Built-in	Default	381	7 days ago	Never
JavaScript, 2 profile(s)		Projects ?	Rules	Updated	Used
Sonar way	Built-in	Default	101	7 days ago	Never
Sonar way Recommended	Built-in	0	141	7 days ago	Never




SonarQube

Le profil qualité

Possibilité d'étendre, copier ou comparer un profil qualité.

Etendre : Ajouter des règles à un profil sans pouvoir désactiver les règles existantes

Copier : Modifier, Supprimer ou Ajouter des règles d'un profil existant

Java, 1 profile(s)		Projects ?	Rules	Updated	Used	
Sonar way	Built-in	Default	381	7 days ago	Never	
JavaScript, 2 profile(s)		Projects ?	Rules	Updated		
Sonar way	Built-in	Default	101	7 days ago	Never	
Sonar way Recommended	Built-in	0	141	7 days ago	Never	

Compare
Copy
Extend

SonarQube

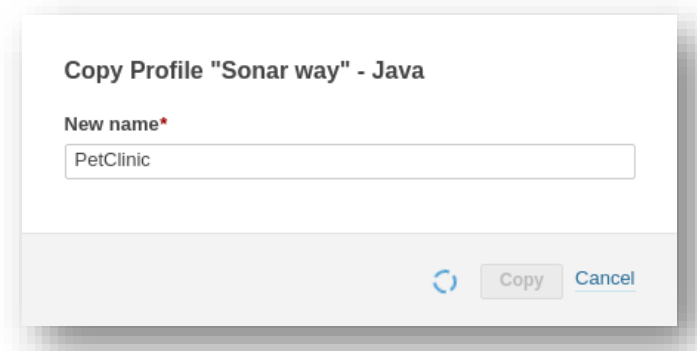
Le profil qualité

Une fois le profil qualité copié, il est possible de visualiser les règles du profil parent, et de les modifier ou d'en ajouter.

Le profil copié peut être affecté à des projets dans le bloc *Projects*.

Le bloc *Rules* indique le nombre de règles activées pour ce profil, classées par types.

Le bloc *Inheritance* permet d'hériter des règles d'un autre profil (par exemple le profil général utilisé par une entreprise).



Quality Profiles / Java

PetClinic

Updated: 2 minutes ago

Used: Never

Changelog



Rules

Active

Inactive

Total

381

156

Bugs

109

12

Vulnerabilities

36

8

Code Smells

206

134

Security Hotspots

30

2

Activate More

Inheritance

Change Parent

PetClinic

381 active rules

0 overridden rules

Projects

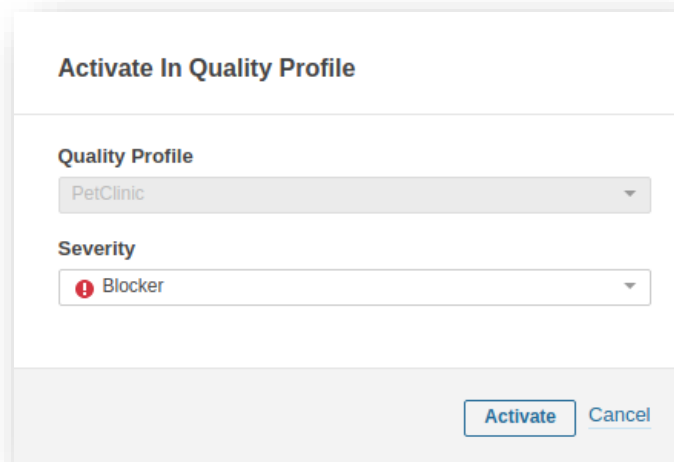
Change Projects

No projects are explicitly associated to the profile.

SonarQube

Activer de nouvelles règles

Un clic sur *Activate More* affiche la page de l'onglet *Rules* du menu d'administration, cette dernière permet d'ajouter ou supprimer des règles à un profil.



Quality Profiles Quality Gates Administration ? Search for projects and files... + A

Bulk Change ↑ ↓ to select rules ← → to navigate ↺ 1 / 156 rules

"==" and "!=" should not be used when "equals" is overridden	Java	Code Smell	cert, cwe, suspicious	▼	Activate
"@EnableAutoConfiguration" should be fine-tuned	Java	Code Smell	performance, spring	▼	Activate
"@Import"s should be preferred to "@ComponentScan"s	Java	Code Smell	performance, spring	▼	Activate
"action" mappings should not have too many "forward" entries	Java	Code Smell	brain-overload, struts	▼	Activate
"Bean Validation" (JSR 380) should be properly configured	Java	Code Smell		▼	Activate
"collect" should be used with "Streams" instead of "list::add"	Java	Code Smell	java8	▼	Activate
"deleteOnExit" should not be used	Java	Code Smell	performance	▼	Activate
"equals" methods should be symmetric and work for subclasses	Java	Bug	cert	▼	Activate
"Exception" should not be caught when not required by called methods	Java	Code Smell	cwe, error-handling	▼	Activate
"final" classes should not have "protected" members	Java	Code Smell	confusing	▼	Activate
"if ... else if" constructs should end with "else" clauses	Java	Code Smell	based-on-misra, cert	▼	Activate
"instanceof" operators that always return "true" or "false" should be removed	Deprecated	Java	Bug	▼	Activate
"java.lang.Error" should not be extended	Java	Code Smell	error-handling	▼	Activate
"java.time" classes should be used for dates and times	Java	Code Smell	java8, pitfall	▼	Activate
"main" should not "throw" anything	Java	Code Smell	error-handling	▼	Activate
"NullPointerException" should not be caught	Java	Code Smell	cert, cwe, error-handling	▼	Activate
"NullPointerException" should not be explicitly thrown	Java	Code Smell	error-handling, pitfall	▼	Activate

SonarQube

Désactiver des règles

	".equals()" should not be used to test the values of "Atomic" classes	Java Bug multi-threading	Deactivate
	"=+" should not be used instead of "+="	Java Bug	Deactivate
	"@CheckForNull" or "@Nullable" should not be used on primitive types	Java Code Smell	Deactivate
	"@Controller" classes that use "@SessionAttributes" must call "setComplete" on their "SessionStatus" objects	Java Bug spring	Deactivate
	"@Deprecated" code should not be used	Java Code Smell cert, cwe, obsolete	Deactivate
	"@NonNull" values should not be set to null	Java Bug cert, cwe	Deactivate
	"@Override" should be used on overriding and implementing methods	Java Code Smell bad-practice	Deactivate
	"@RequestMapping" methods should be "public"	Java Vulnerability owasp-a6, spring	Deactivate
	"@RequestMapping" methods should specify HTTP method	Java Vulnerability cwe, owasp-a6, sans-top25-insecure, ...	Deactivate
	"@SpringBootApplication" and "@ComponentScan" should not be used in the default package	Java Bug spring	Deactivate
	"Arrays.stream" should be used for primitive arrays	Java Code Smell performance	Deactivate
	"BigDecimal(double)" should not be used	Java Bug cert	Deactivate
	"catch" clauses should do more than rethrow	Java Code Smell clumsy, error-handling, finding, unused	Deactivate
	"Class.forName()" should not load JDBC 4.0+ drivers	Java Code Smell obsolete	Deactivate
	"clone" should not be overridden	Java Code Smell suspicious	Deactivate
	"Cloneables" should implement "clone"	Java Code Smell api-design, convention	Deactivate

Un clic sur la règle permet d'obtenir le détail de la règle et des exemples d'implémentation de code déclenchant ou non la règle.

"=+" should not be used instead of "+="

squid:S2757 🔗 🏠

🐛 Bug ⬆ Major 📄 Main sources 🏷 No tags Available Since Dec 04, 2019 SonarAnalyzer (Java) Constant/issue: 2min

The use of operators pairs (`=+` , `=-` or `=!`) where the reversed, single operator was meant (`+=` , `-=` or `!=`) will compile and run, but not produce the expected results.

This rule raises an issue when `=+` , `=-` , or `=!` is used without any spacing between the two operators and when there is at least one whitespace character after.

Noncompliant Code Example

```
int target = -5;
int num = 3;

target =- num; // Noncompliant; target = -3. Is that really what's meant?
target =+ num; // Noncompliant; target = 3
```

Compliant Solution

```
int target = -5;
int num = 3;

target = -num; // Compliant; intent to assign inverse value of num is clear
target += num;
```

[Extend Description](#)

Quality Profiles [Activate](#)

[Sonar way](#) Built-in ⬆ Major

[PetClinic](#) ⬆ Major

Permet de
changer la
gravité de la
règle

[Change](#) [Deactivate](#)

04

Les Quality Gates

SonarQube

Les quality gates ou barrières qualité

La page *Quality gates* permet de définir les exigences qu'un projet doit satisfaire pour être mis en production. Il est composé d'un ensemble de conditions booléennes.

SonarQube propose un profil par défaut. Les exigences pour ce profil sont :

- taux de couverture des tests ≥ 80
- score de maintenabilité : A
- score de fiabilité : A
- score de sécurité : A

Quality Gates ?

Create

Sonar way

Default

Built-in

Sonar way

Built-in

Copy

Conditions ?

Only project measures are checked against thresholds. Directories and files are ignored.

Metric ?	Operator	Error
Coverage on New Code	is less than	80.0%
Duplicated Lines on New Code	is greater than	3.0%
Maintainability Rating on New Code	is worse than	A
Reliability Rating on New Code	is worse than	A
Security Rating on New Code	is worse than	A

Projects ?

Every project not specifically associated to a quality gate will be associated to this one by default.

SonarQube

Les quality gates ou barrières qualité

Il est possible de créer ses exigences en ajoutant des conditions à une nouvelle quality gate, ou modifier les conditions copiées.

Quality Gates ?

Create

PetClinic

Sonar way

Default

Built-in

PetClinic

Rename

Copy

Set as Default

Delete

Conditions ?

Add Condition

Only project measures are checked against thresholds. Directories and files are ignored.

Metric ?

Coverage on New Code

Duplicated Lines on New Code

Maintainability Rating on New Code

Reliability Rating on New Code

Security Rating on New Code

Update Condition

Metric

Coverage on New Code

Operator

is less than

Error

80

Update Condition

Cancel

Operator	Error	
is less than	80.0%	
is greater than	3.0%	
is worse than	A	
is worse than	A	
is worse than	A	

Projects ?

With

Without

All

☒

 petclinic

05

Gestion des problèmes (issues)

SonarQube

Liste et détails des problèmes

☆ Aliquot

Issues Measures Code Administration

My Issues All

Display Mode
Issues Effort

Type

Bug 0

Vulnerability 0

Code Smell 2h

Resolution

Unresolved 2h Fixed 0

False Positive 0 Won't fix 0

Removed 0

ordered by creation date 1 / 12 issues

Reload New Search Bulk Change

Aliquot src/AppBundle/Form/Type/PersonneType.php

Define a constant instead of duplicating this literal "required" 5 times. ...

Code Smell Critical Open Not assigned 12min effort Comment

il y a une heure L21 design

Define a constant instead of duplicating this literal "label" 5 times. ...

Code Smell Critical Open Not assigned 12min effort Comment

il y a une heure L22 design

Aliquot src/AppBundle/Form/Type/SearchPersonneType.php

Define a constant instead of duplicating this literal "label" 3 times. ...

Code Smell Critical Open Not assigned 8min effort Comment


il y a une heure L18 design

Define a constant instead of duplicating this literal "required" 3 times. ...

Code Smell Critical Open Not assigned 8min effort Comment

il y a une heure L18 design

36 INSA - 2019 - Cours SonarQube

sopra  steria

SonarQube

Liste et détails des problèmes

Type de l'issue (modifiable) :

- **Bug** : code de case identiques dans un switch, boucle infinie, ...
- **vulnerability** : problème de sécurité (mot de passe en clair, cookie secure, ...)
- **code smell** : problème de conception ou de design (complexité cyclomatique, lignes de code commenté, ...)

Niveau de sévérité du problème (il peut être modifié) :

- **blocker** : bug qui a une forte probabilité d'impacter le comportement, à régler immédiatement
- **critical** : problème qui peut impacter le comportement ou problème de sécurité (injection SQL)
- **major** : problème impactant fortement la productivité du développeur : duplications, paramètres non utilisés, ...
- **minor** : problème impactant faiblement la productivité du développeur : lignes trop longues, ...
- **info.**



SonarQube

Suivi des problèmes

Il est possible de marquer chaque *issue* :

- **confirm** : confirmer qu'il y a un problème, le statut passe de Open à Confirmed
- **false positive** : ce n'est pas un vrai problème
- **won't fix** : c'est un problème valide mais qui ne nécessite pas d'être modifié
- **change severity** : c'est un problème à régler mais il n'a pas le niveau de sévérité indiqué
- **resolve** : le problème a été réglé, lors de la prochaine analyse, si c'est vraiment le cas l'issue prendra le statut Closed, sinon son statut sera ré-ouvert.

Le problème peut être affecté à un utilisateur enregistré de SonarQube, un commentaire peut être ajouté (Comment).

Lorsque beaucoup d'issues sont marquées *false positive* ou *won't fix*, il faut changer les règles du jeu utilisé car elles ne sont pas appropriées pour le projet (désactiver des règles dans le profil qualité ou modifier les paramètres).



SonarQube

Classification des problèmes

Exemples de Tag indiquant la classe du problème :

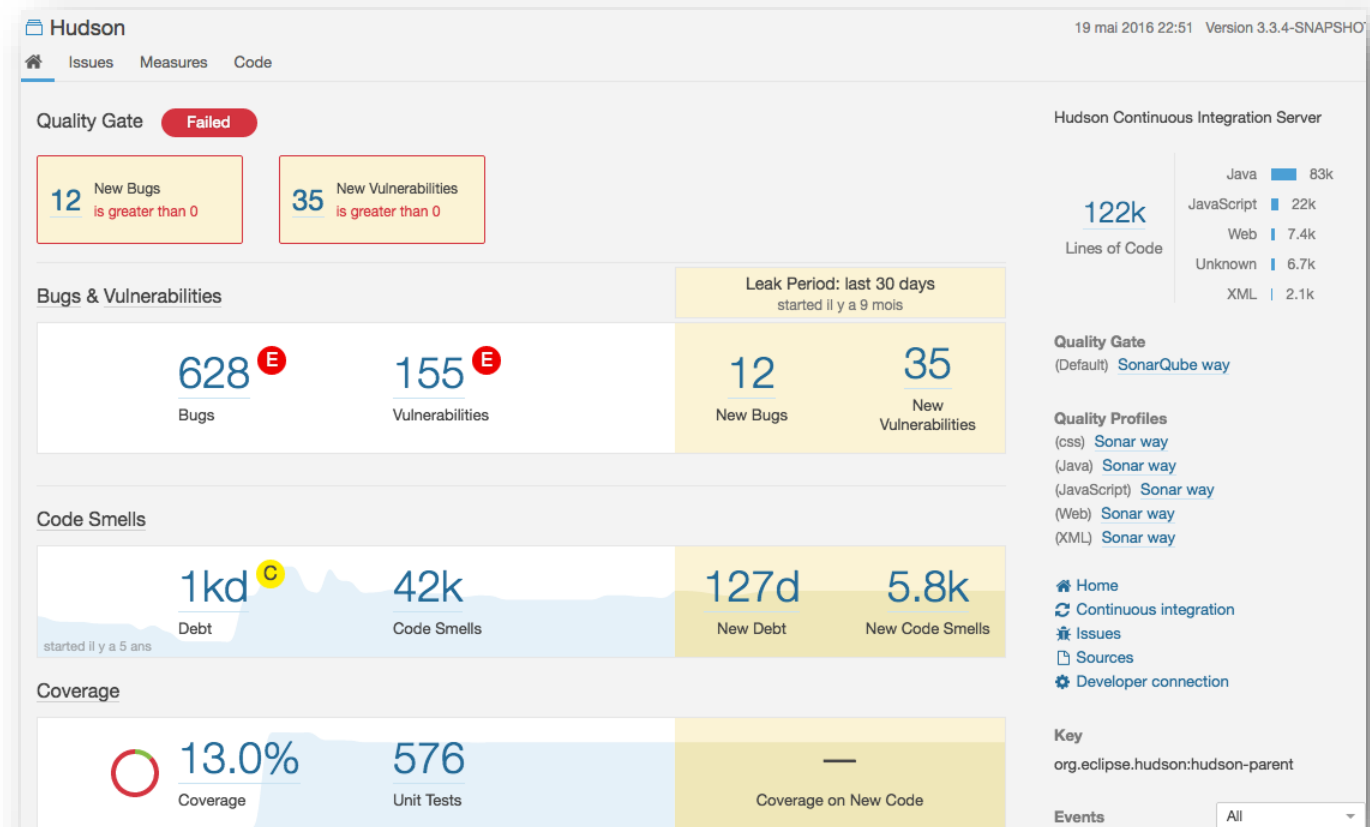
- └ brain overload : trop d'information à garder en mémoire pour comprendre le code
- └ cert : règle du standard CERT (bonnes pratiques de programmation)
- └ clumsy : pourrait être accompli de manière plus concise ou claire
- └ confusing : la compréhension prendra du temps
- └ convention : convention de codage (formatage, nom, espaces, ...)
- └ cwe : règle du CWE (Common Weakness Enumeration)
- └ misra : règle du standard MISRA (Motor Industry Software Reliability Association)
- └ owasp-* : règle de l'OWASP Top Ten
- └ psr2 : standard de codage PHP
- └ san-top25-* : SANS Top 25 coding errors
- └ security
- └ suspicious
- └ unpredictable : le code pourrait avoir un comportement imprévisible si les conditions changent
- └ unused : code non utilisé

SonarQube

Résolution des problèmes récents en priorité

Pour améliorer la qualité du projet la priorité est de **résoudre les nouveaux problèmes** (analogie de la fuite d'eau : fermer le robinet avant d'éponger, ici le but est de ne pas introduire de nouveaux problèmes de qualité, de plus il est plus facile d'intervenir sur du code frais que sur un ancien code).

Il faut corriger en priorité ce qui apparaît dans la partie droite sur fond jaune. Intervenir sur la duplication de code, puis corriger les bugs et code smells.



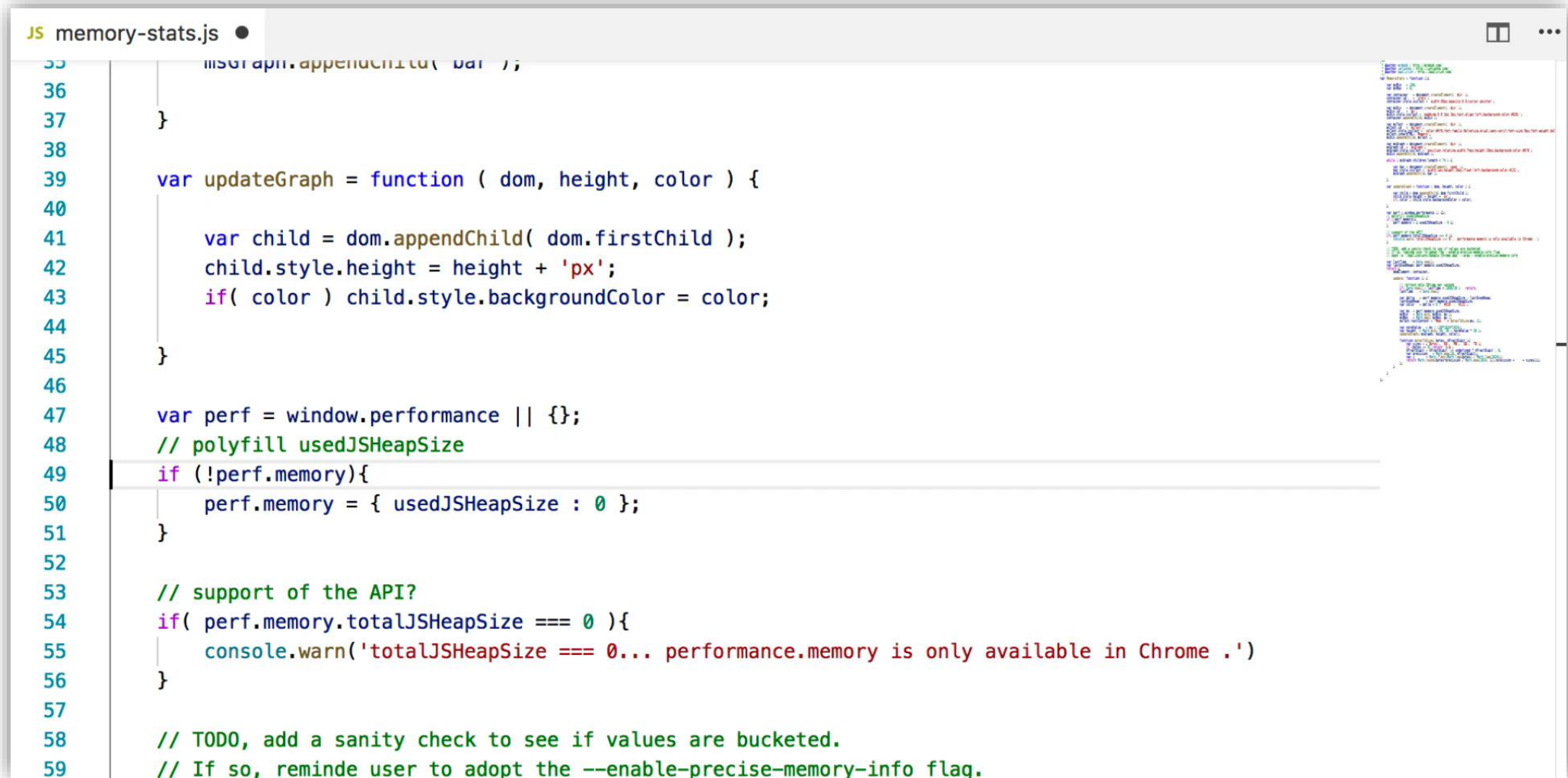
06

SonarLint

SONAR SUR SON POSTE DE TRAVAIL

SonarLint

Ou comment bénéficier de Sonar directement sur son IDE



```
JS memory-stats.js ●
35      msGraph.appendChild( bar );
36
37    }
38
39    var updateGraph = function ( dom, height, color ) {
40
41      var child = dom.appendChild( dom.firstChild );
42      child.style.height = height + 'px';
43      if( color ) child.style.backgroundColor = color;
44
45    }
46
47    var perf = window.performance || {};
48    // polyfill usedJSHeapSize
49    if (!perf.memory){
50      perf.memory = { usedJSHeapSize : 0 };
51    }
52
53    // support of the API?
54    if( perf.memory.totalJSHeapSize === 0 ){
55      console.warn('totalJSHeapSize === 0... performance.memory is only available in Chrome .')
56    }
57
58    // TODO, add a sanity check to see if values are bucketed.
59    // If so, remind user to adopt the --enable-precise-memory-info flag.
```