Mariana ANDUJAR
mariana.andujar@univ-amu.fr

IBDM cnrs Aix*Marseille université

Mesurer la qualité du code en continu avec

06 juillet 2017

JDEV 2017

Plan

- **Qualité du code**
- Présentation de SonarQube
- Installation
- Configuration
- Utilisation

Qualité du code

- But
  - Fiabilité
  - Maintenabilité
  - Réutilisabilité

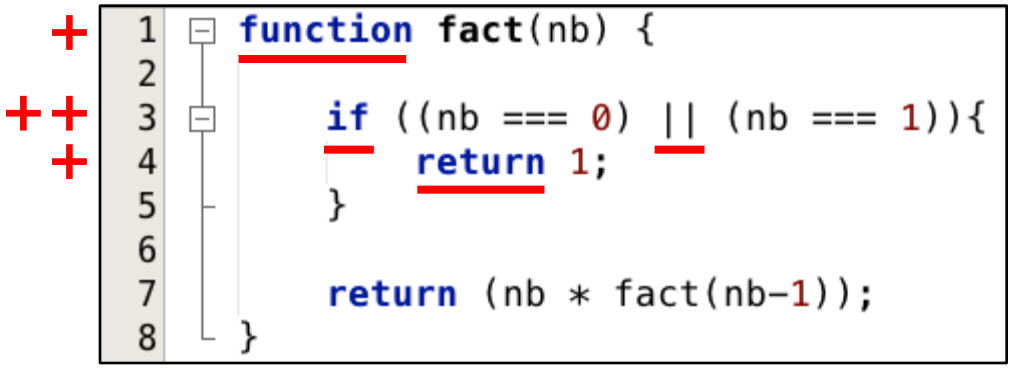
Critères	Métriques
Respect d'un standard de codage	
Documentation du code	Taux de documentation
Tests	Taux de couverture
Non duplication du code	Taux de duplication
Règles de responsabilité unique	Taille du code **Nombre de lignes de code par fichier** **Nombre de lignes de code par classe** **Nombre de méthodes par classe** Complexité cyclomatique

Qualité du code

- Complexité cyclomatique = nombre de chemins dans le code

Complexité	Basse	Modérée	Haute	Très haute
valeur	1 à 4	5 à 7	8 à 10	≥ 11

Calcul approximé par les outils : addition de certains mots clés et opérateurs



Qualité du code

- Complexité cyclomatique = nombre de chemins dans le code

Complexité	Basse	Modérée	Haute	Très haute
valeur	1 à 4	5 à 7	8 à 10	≥ 11

```
+  if (scope == SCOPE.GLOBAL) {
       //All non team members gets only READ Permission
+      if (permission.getImpliedBy() == Permission.READ) {
+          return true;
       }
       // Member of any of the team with JOB CREATE Permission can create Job
+      if (permission == Item.CREATE) {
+          for (Team userTeam : teamManager.findUserTeams(userName)) {
+              if (isTeamAwareSecurityRealm()) {
+                  return true; // for now give full permission to all team members
               }
               TeamMember member = userTeam.findMember(userName);
++            if ((member != null) && member.hasPermission(Item.CREATE)) {
+                 return true;
              }
          }
      }
      // Member of any of the team with View CREATE Permission can create View
+     if (permission == View.CREATE) {
+         for (Team userTeam : teamManager.findUserTeams(userName)) {
```

Qualité du code

- Problème de la complexité cyclomatique

```
int sumOfPrimes(int max) {                    // +1
  int total = 0;
  OUT: for (int i = 1; i <= max; ++i) {       // +1
    for (int j = 2; j < i; ++j) {             // +1
      if (i % j == 0) {                       // +1
        continue OUT;
      }
    }
    total += i;
  }
  return total;
}                    // Cyclomatic Complexity 4
```

```
String getWords(int number) { // +1
  switch (number) {
    case 1:                   // +1
      return "one";
    case 2:                   // +1
      return "a couple";
    default:                  // +1
      return "lots";
  }
}        // Cyclomatic Complexity 4
```

Source : https://blog.sonarsource.com/cognitive-complexity-because-testability-understandability/

Qualité du code

- Complexité cognitive (v6)
  - Basée sur la complexité cyclomatique
  - Mesure la difficulté de compréhension du code

```
                                // Cyclomatic Complexity    Cognitive Complexity
String getWords(int number) {   //          +1
  switch (number) {             //                                   +1
    case 1:                     //          +1
      return "one";
    case 2:                     //          +1
      return "a couple";
    default:                    //          +1
      return "lots";
  }
}                               //          =4                       =1
```

Source : https://blog.sonarsource.com/cognitive-complexity-because-testability-understandability/

Qualité du code

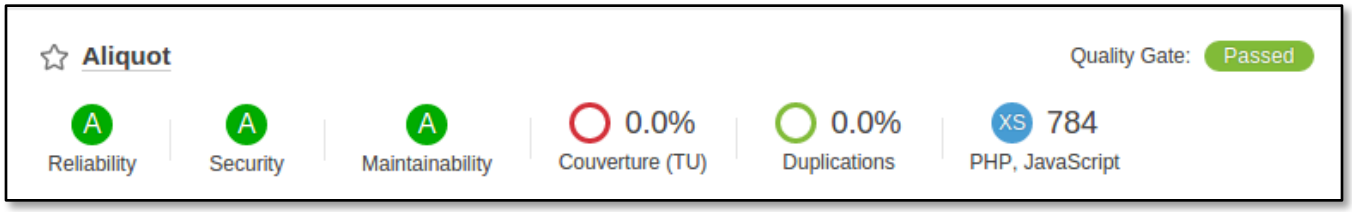
Critères	Métriques	Outils
Standard de codage		phpcs checkstyle Pmd
Documentation du code	Taux de documentation	Cloc
Tests	Taux de couverture	Junit, PHPUnit, Cobertura
Non duplication du code	Taux de duplication	CPD, Phpcpd
Règles de responsabilité unique	Taille du code : **Nombre de lignes par fichier** **Nombre de lignes par classe** **Nombre de méthodes par classe** Complexité cyclomatique	Cloc JDepend
Absence de bogue, sécurité		FindBugs Pmd

Plan

- Qualité du code
- **Présentation de SonarQube**
- Installation
- Configuration
- Utilisation

Présentation de SonarQube

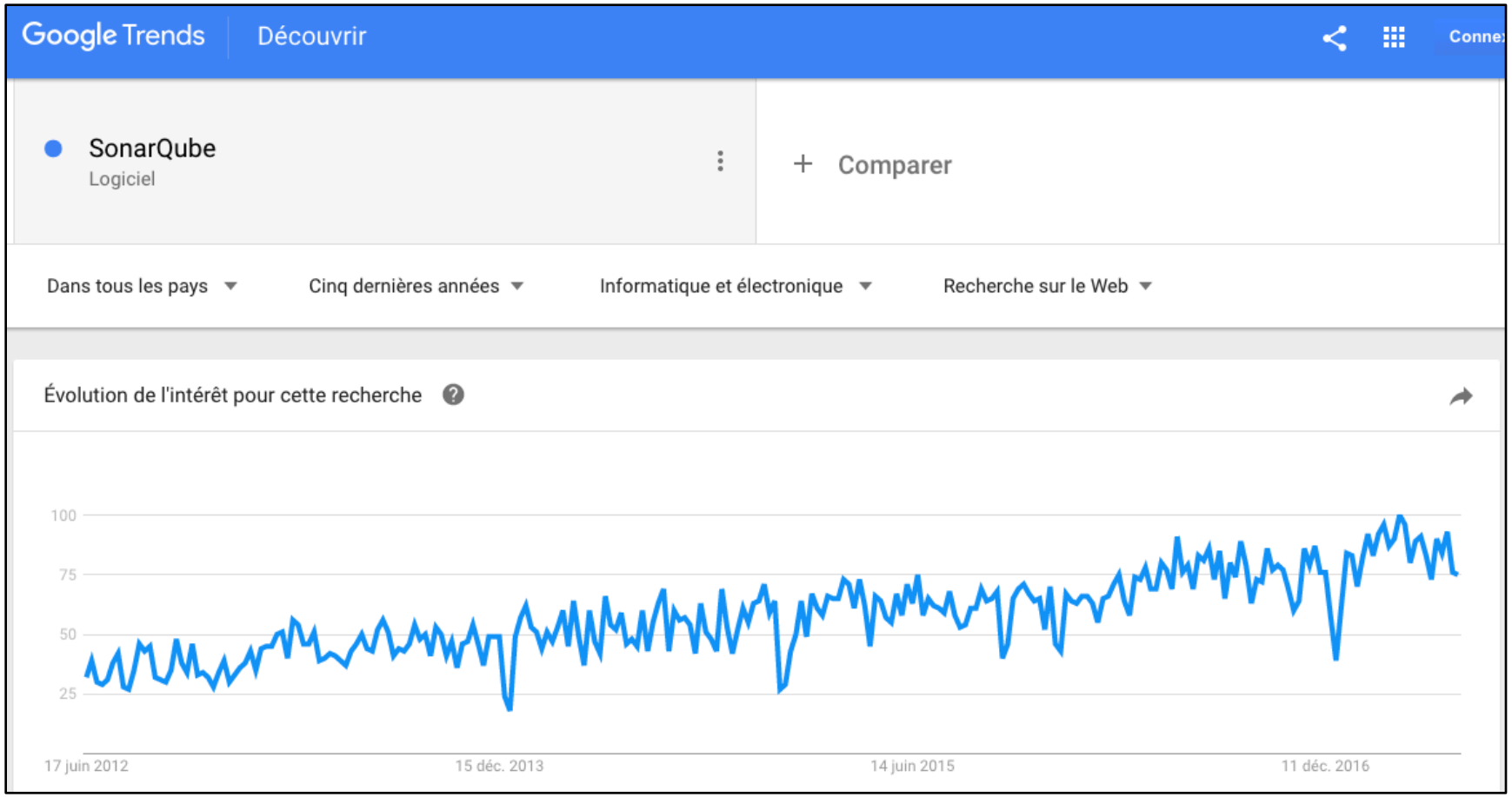
- Introduction
  - Mesure de la qualité du code source
  - ~ 20 langages : C, C++, PHP, Java, Python, JavaScript, XML, …
  - Open Source, licence GNU GPLv3
  - Développé depuis 2007
  - Périmètre : code source, design, tests unitaires



  - Clients SonarSource

DGA
AIRFRANCE
BNP PARIBAS
PayPal
SAMSUNG
MasterCard
amazon
cnrs
CISCO
orange
THALES
ebay

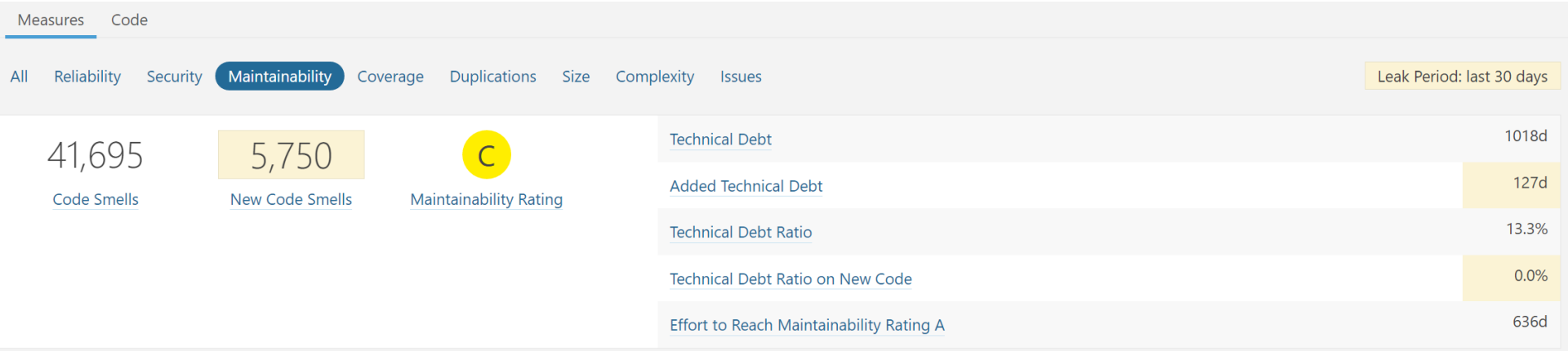
Présentation de SonarQube



https://trends.google.fr/trends/

Présentation de SonarQube

- Fonctionnalités
  - Mesure la qualité
    - Taille du projet
    - Densité des commentaires
    - Taux de couverture
    - Respect des conventions de nommage et de codage
    - Détection des bogues
    - Détection de code mort
    - Détection de code dupliqué
    - Complexité
    - Score de maintenabilité, fiabilité et sécurité
    - Dette technique



Présentation de SonarQube

- Fonctionnalités
  - Informations au niveau projet, fichier, classe, méthode
  - Historique des analyses
  - Résultat accessible via le web
  - Annotation du code source

Présentation de SonarQube

- Différents types de problèmes (issues)
  - Vulnerabilities

```
public function setPasswordSetsPassword()
{
    $password = 'f00Bar';
```

Remove this hard-coded password. ··· 2 years ago ▾ L65

Vulnerability Blocker Open Not assigned 30min effort cwe, owasp-a2, sans-top25-porous

```
    $this->subject->setPassword($password);
    $this->assertSame($password, $this->subject->getPassword());
}
```

  - Bug

```
henri…          // Skip any extra separator chars
                while (++i < end && ((c = path.charAt(i)) == '/' || c == '\\')) {
```

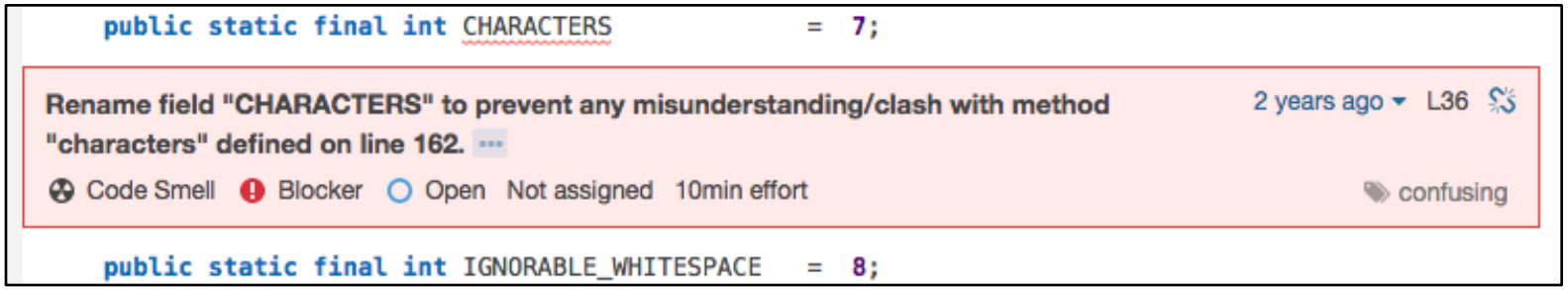
Either remove or fill this block of code. ··· 3 years ago ▾ L241

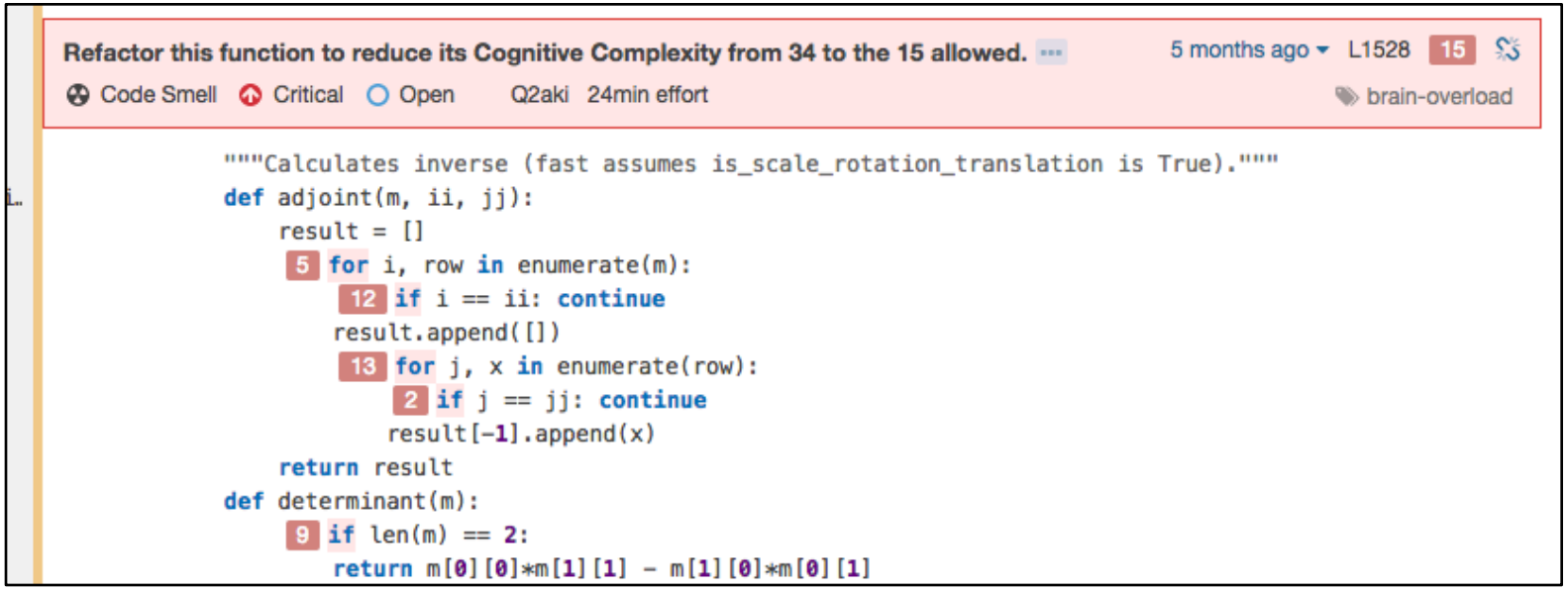
Bug Major Open Not assigned 5min effort No tags

```
                }
winst…          // Add token for separator unless we reached the end
```

Présentation de SonarQube

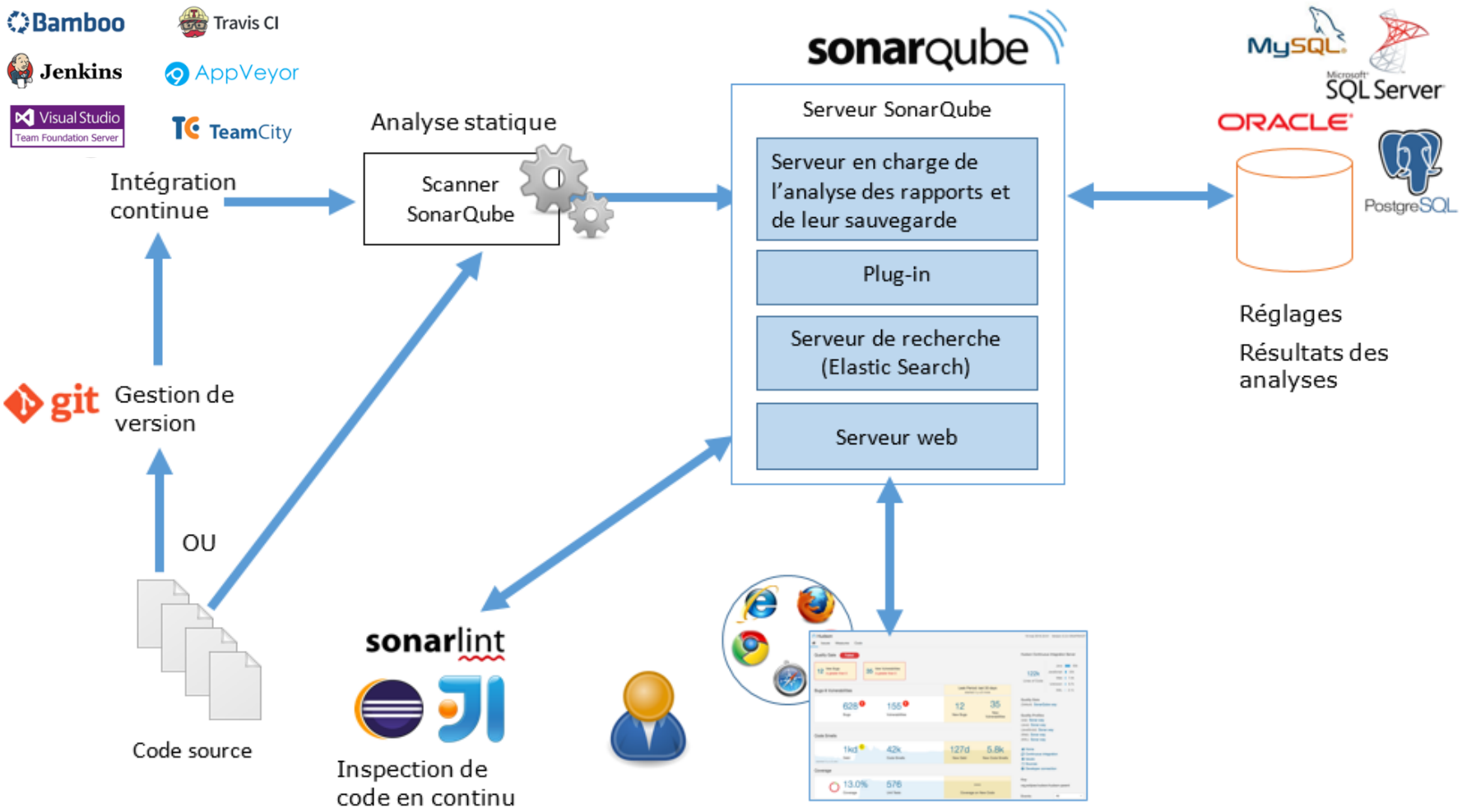
- Différents types de problèmes (issues)
  - Code smell





Présentation de SonarQube

- Architecture de la plateforme



Plan

- Qualité du code
- Présentation de SonarQube
- **Installation**
- Configuration
- Utilisation

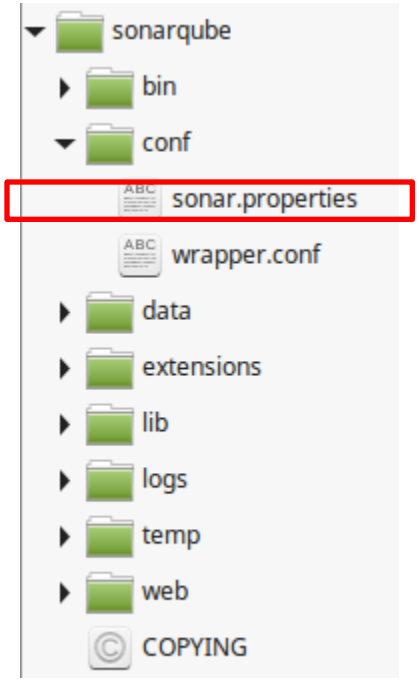
Installation

- Pré-requis (version 6.4)
  - Côté serveur
    - Java : Oracle JRE 8 ou OpenJDK 8 (conseillé pour mac OS X)
    - SGBD :
      - MySQL ≥ 5.6
      - Oracle ≥ 11G
      - PostgreSQL ≥ 8
      - Microsoft SQL Server ≥ 11.0
    - RAM : 2GO
  - Côté client
    - Navigateur web récent
    - JavaScript activé

Installation

- Télécharger Sonarqube
- Créer une base et un compte Sonar dans MySQL
- Configuration sonar.properties
  - URL d'accès http://localhost:7223

```
# 0.0.0.0 = n'importe quelle interface réseau
sonar.web.host= 0.0.0.0
# port (valeur par défaut 9000)
sonar.web.port=7223
# si /sonar/ => URL = http://localhost:7223/sonar/
sonar.web.context=/
```



  - Login et mot de passe de la base de données

```
sonar.jdbc.username=sonar
sonar.jdbc.password=sonar
```

  - MySQL

```
#----- MySQL 5.x
sonar.jdbc.url=jdbc:mysql://localhost:3306/sonar?useUnicode=true&character
Encoding=utf8&rewriteBatchedStatements=true&useConfigs=maxPerformance
```

Installation

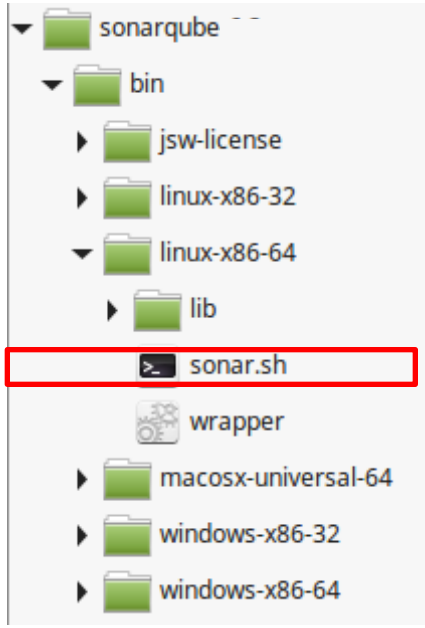
- Démarrer Sonarqube

  - Sous linux/mac os :

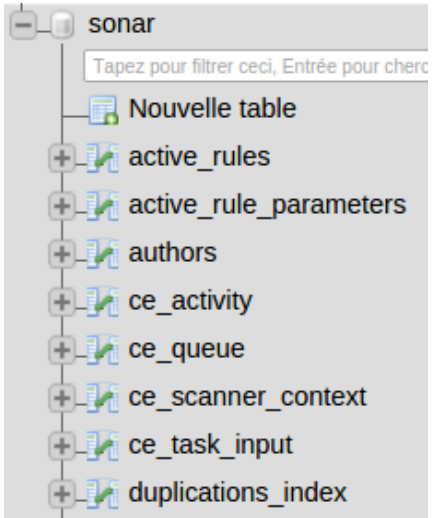
    ```
    sudo ./sonar.sh start
    ```

  - Sous Windows

    ```
    StartSonar.bat
    ```



  - Les tables sont créées dans la base sonar de MySQL



  - L'interface web de sonar est accessible sur le port 7223

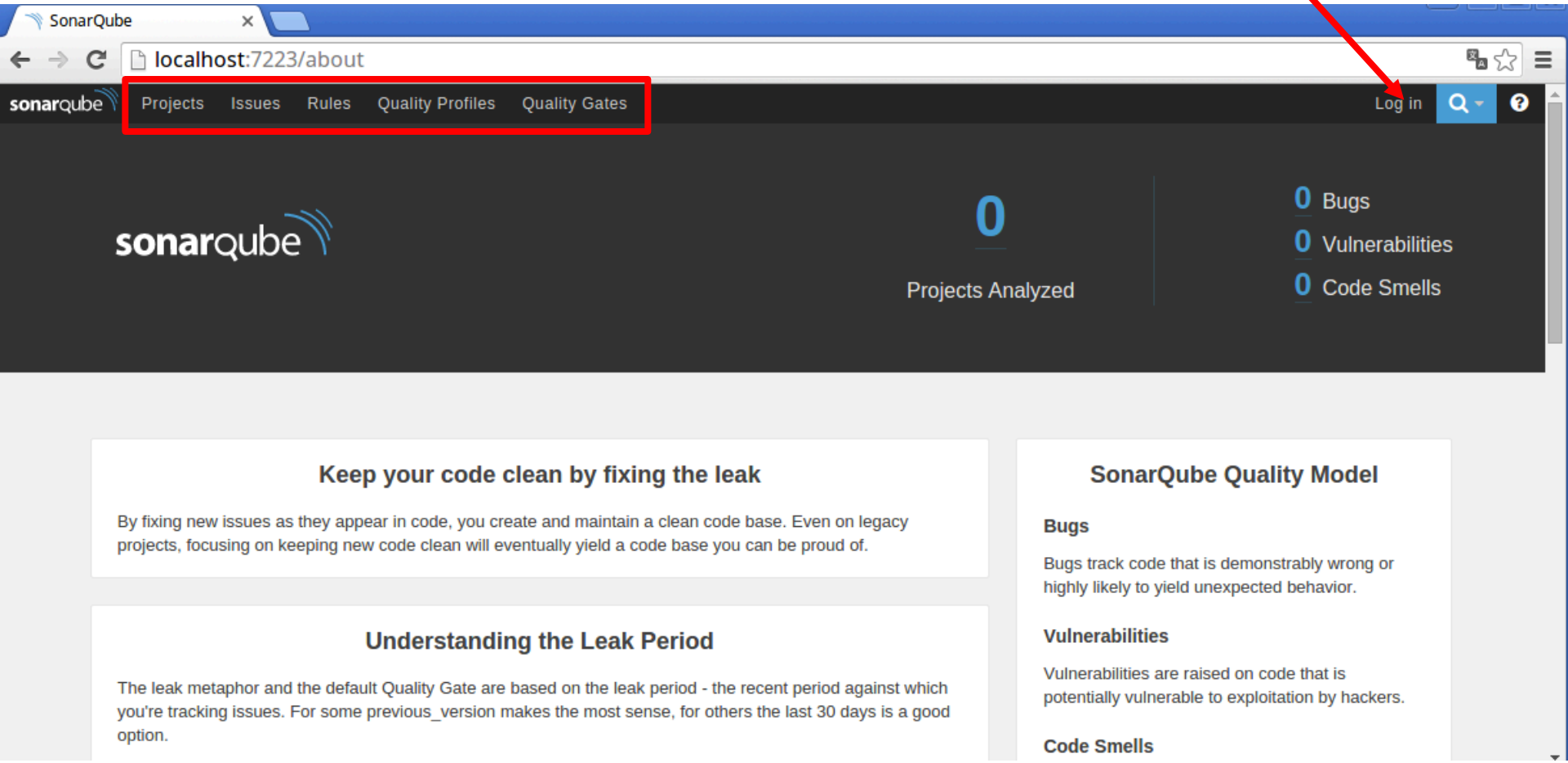
Plan

- Qualité du code
- Présentation de SonarQube
- Installation
- **Configuration**
- Utilisation

Configuration

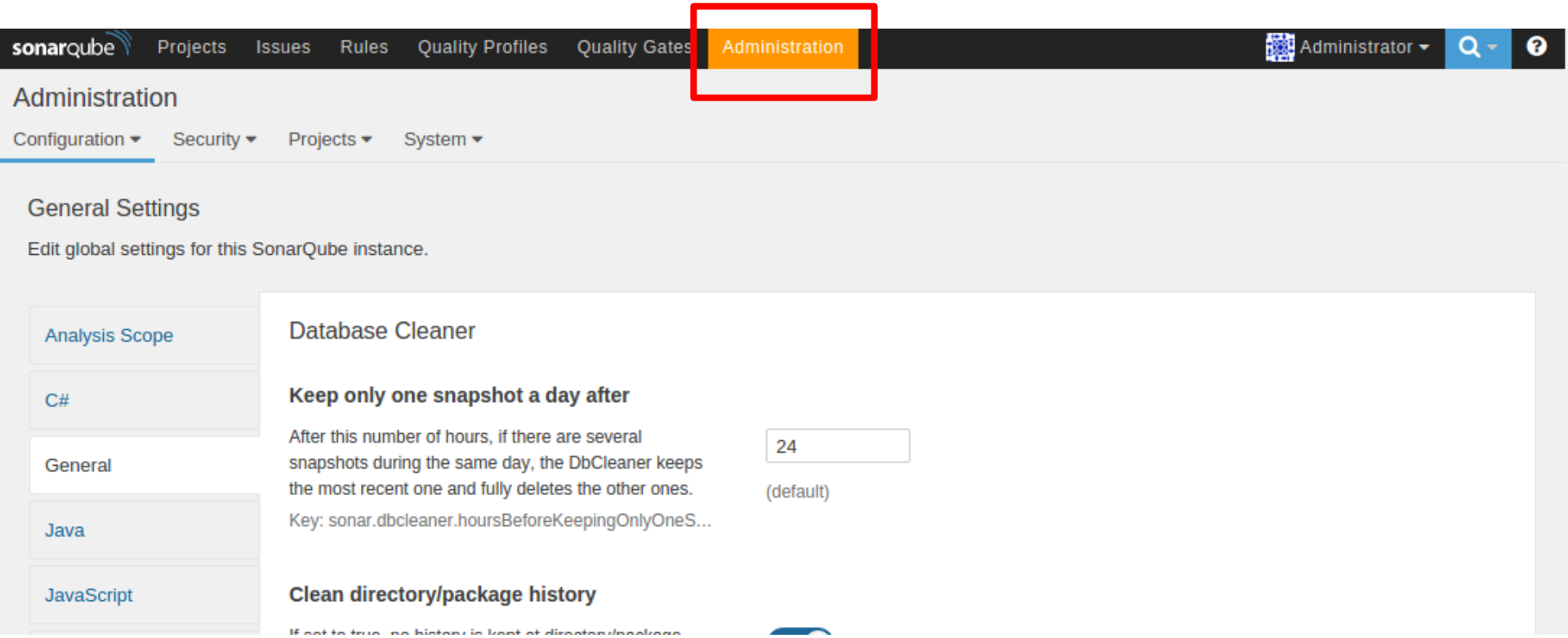
- Interface web de sonarQube
  - http://localhost:7223/

Login : admin
Mdp : admin



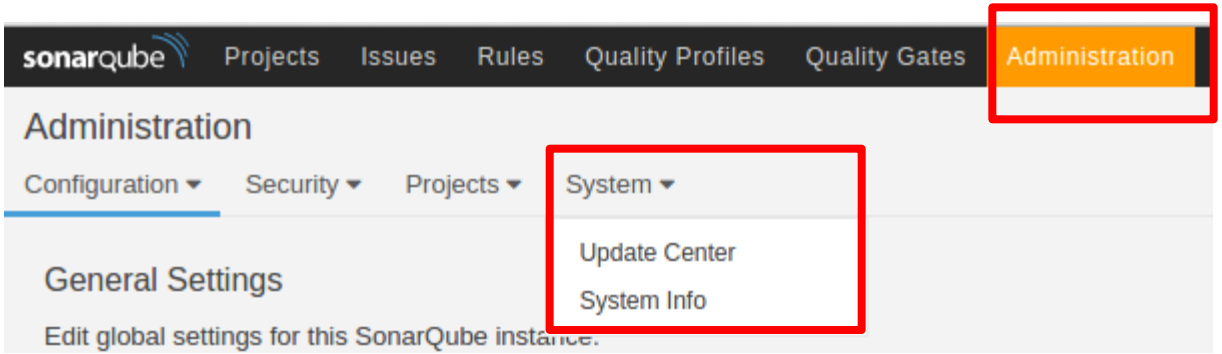
Configuration

- Interface d'administration

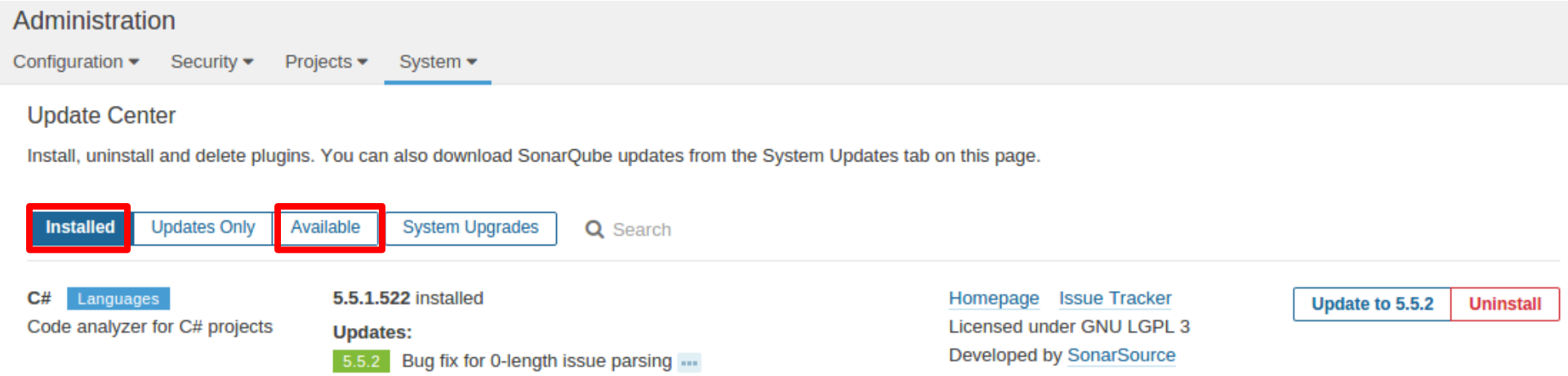


Configuration

- Interface d'administration
  - Centre de mises à jour

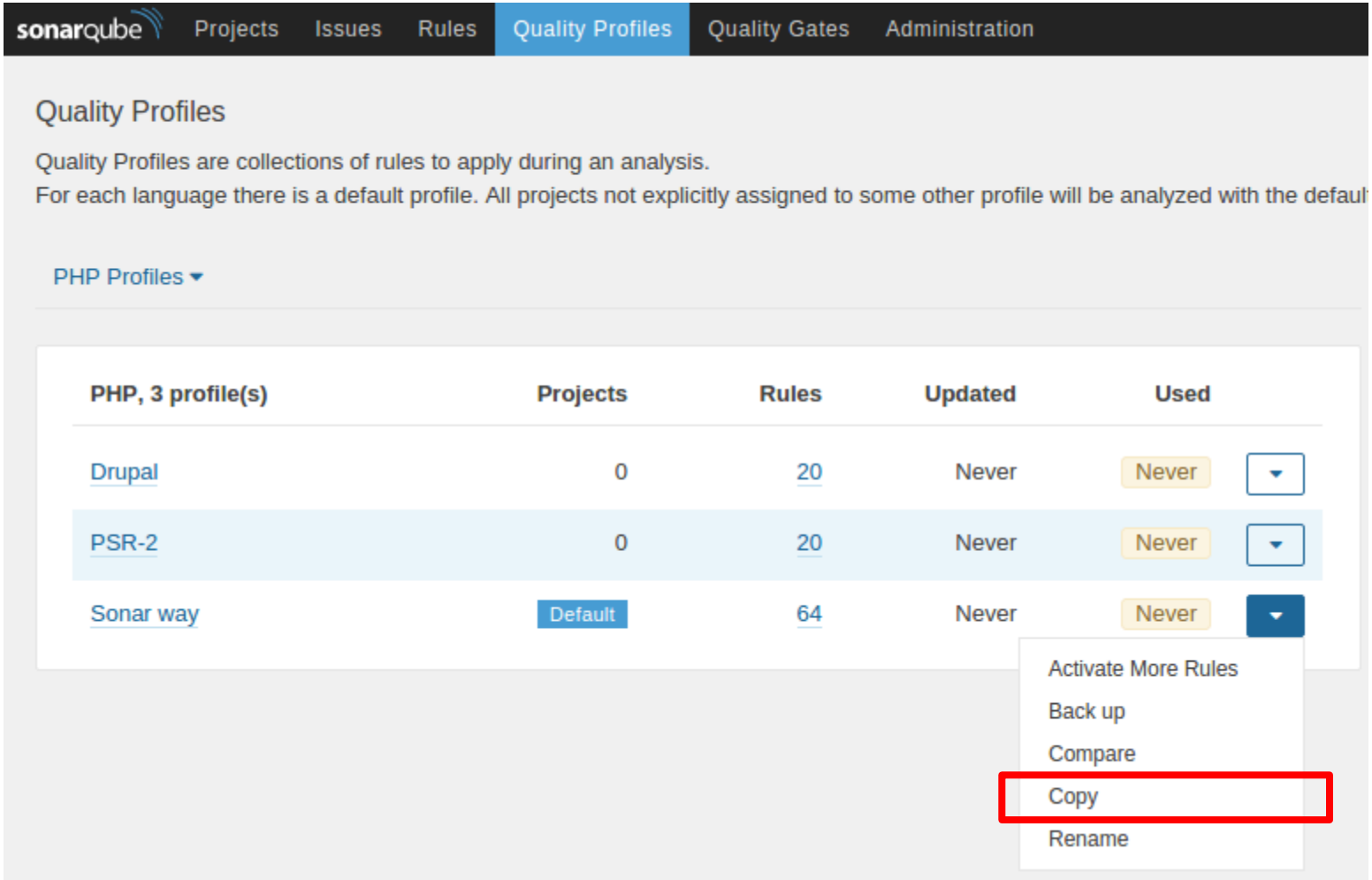


  - Listes des plugins



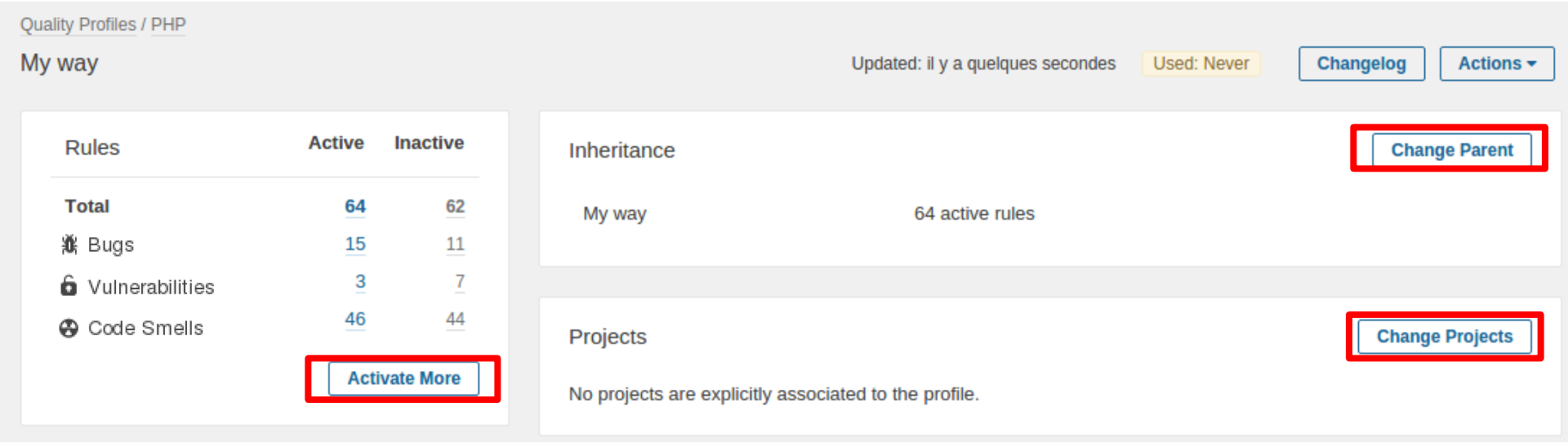
Configuration

- Profil Qualité
  - Création d'un nouveau profil en copiant celui de SonarWay



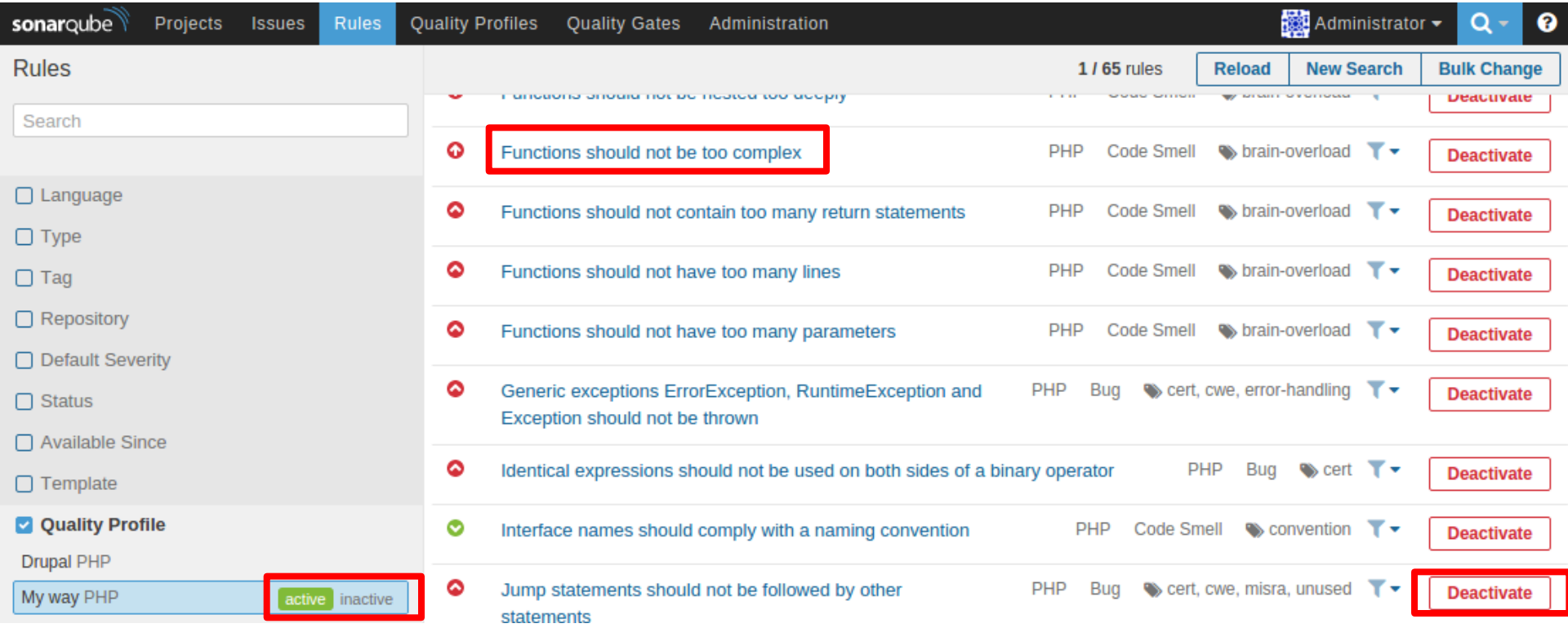
Configuration

- Profil Qualité
  - Configurer un profil



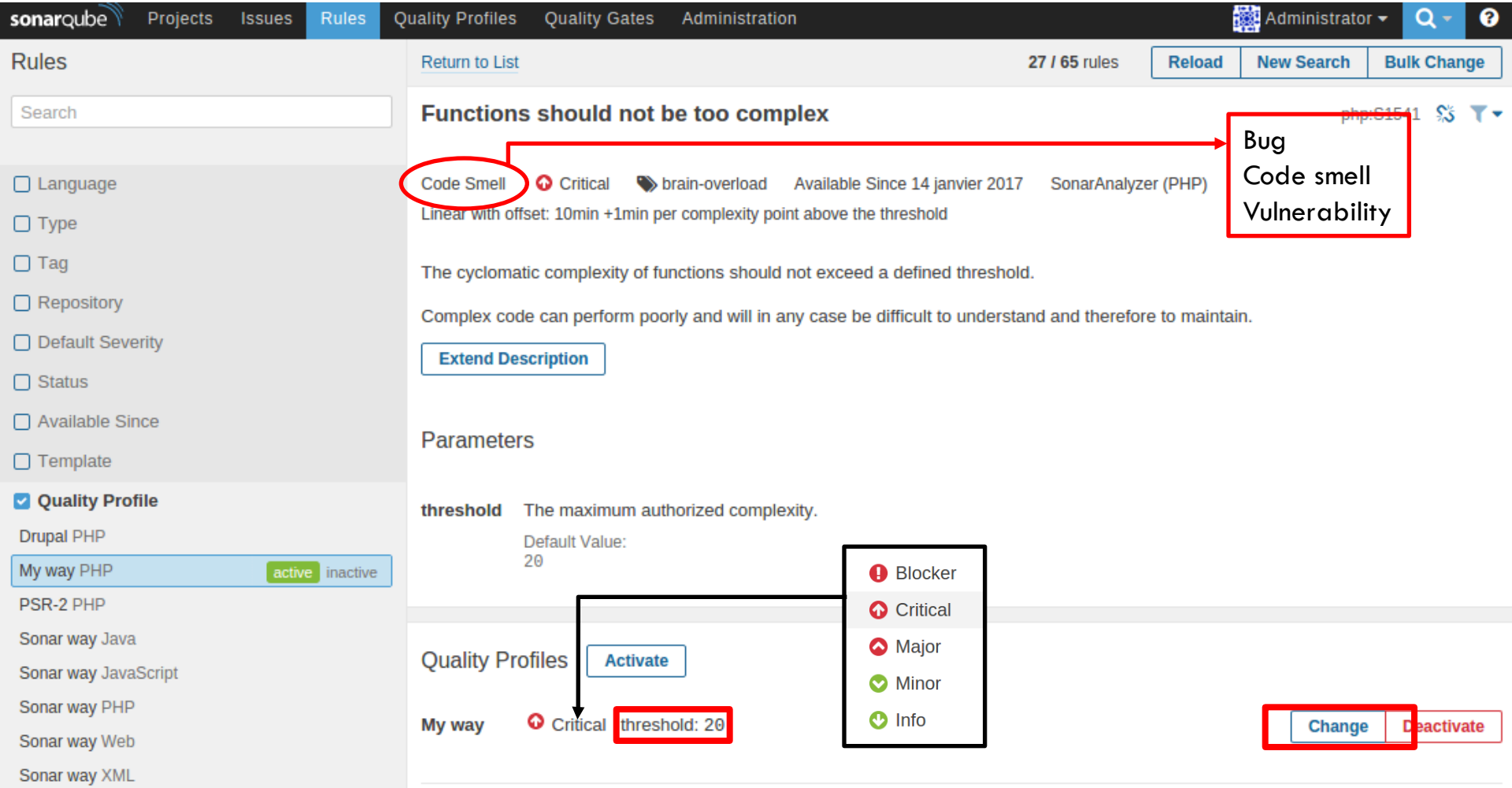
Configuration

- Profil Qualité
  - Ajout ou suppression de règles à un profil



Configuration

- Profil Qualité
  - Configuration d'une règle

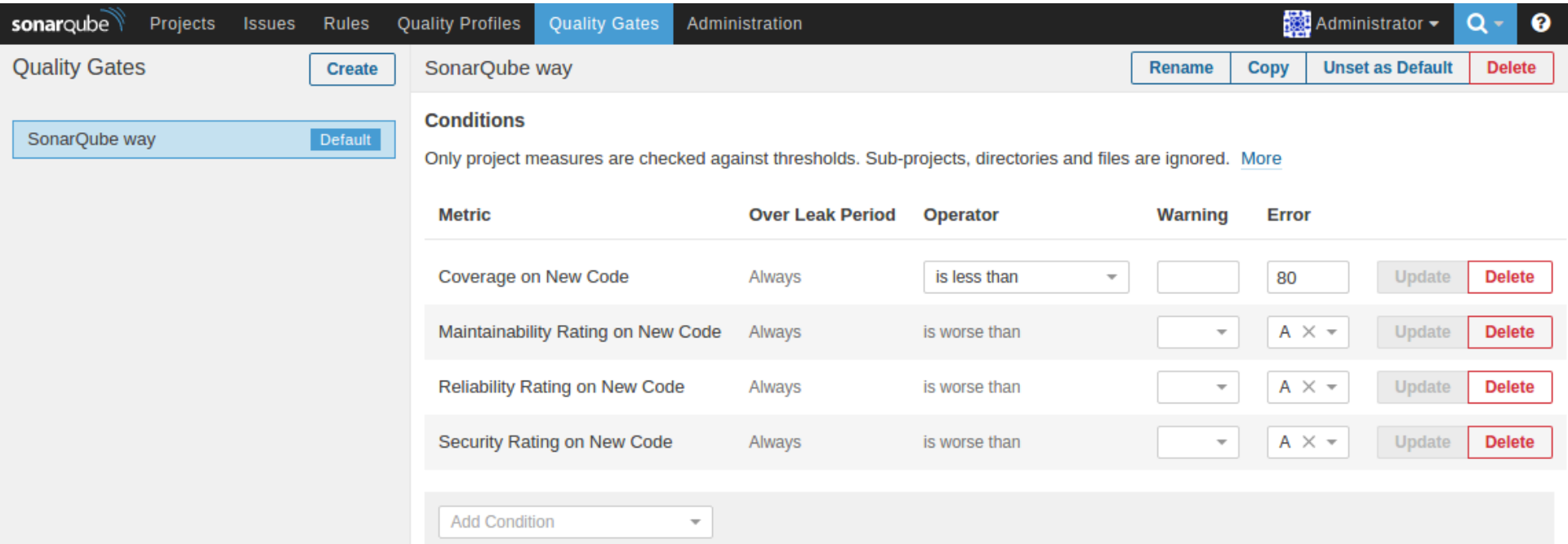


Configuration

- Barrière qualité
  - Profil par défaut
    - Taux de couverture tests > 80
    - Score de maintenabilité : A
    - Score de fiabilité : A
    - Score de sécurité : A

Quality Gate: Passed Quality Gate: Failed

Score	Fiabilité	Sécurité	Maintenabilité
A	0 bug	0 vulnérabilité	0.00 ≤ ratio dette technique ≤ 0.05
B	≥ 1 bug mineur	≥ 1 vulnérabilité mineure	0.06 ≤ ratio dette technique ≤ 0.1
C	≥1 bug majeur	≥ 1 vulnérabilité majeure	0.11 ≤ ratio dette technique ≤ 0.20
D	≥ 1 bug critique	≥ 1 vulnérabilité critique	0.21 ≤ ratio dette technique ≤ 0.5
E	≥ 1 bug bloquant	≥1 vulnérabilité bloquante	0.51 ≤ ratio dette technique ≤ 1
Issue	Bug	Vulnerability	Code smell



Plan

- Qualité du code
- Présentation de SonarQube
- Installation
- Configuration
- **Utilisation**

Utilisation

- Installation du scanner de sources
  - Télécharger SonarQube Scanner
  - Configurer le fichier de configuration conf/sonar-scanner.properties

```
sonar.host.url=http://localhost:7223
sonar.sourceEncoding=UTF-8
```

  - Ajout du chemin du répertoire *bin* de sonar-scanner au PATH

```
export PATH=$PATH:chemin vers sonar scanner
```

  - Création d'un fichier sonar-project.properties dans le projet

```
# cle unique
sonar.projectKey=IBDM:aliquot
# nom et version du projet affichés par l'interface graphique
sonar.projectName=Aliquot
sonar.projectVersion=1.0
# chemin relatif des fichiers sources, séparateur de chemins = virgule
sonar.sources=src/AppBundle,web/js
# encodage des fichiers sources
sonar.sourceEncoding=UTF-8
```

- Lancer sonar-scanner depuis le répertoire du projet

Utilisation

- Visualiser les résultats dans l'interface web

Démo

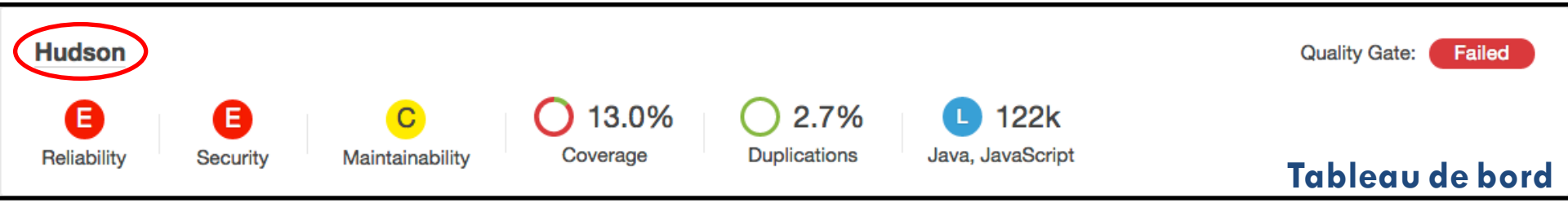
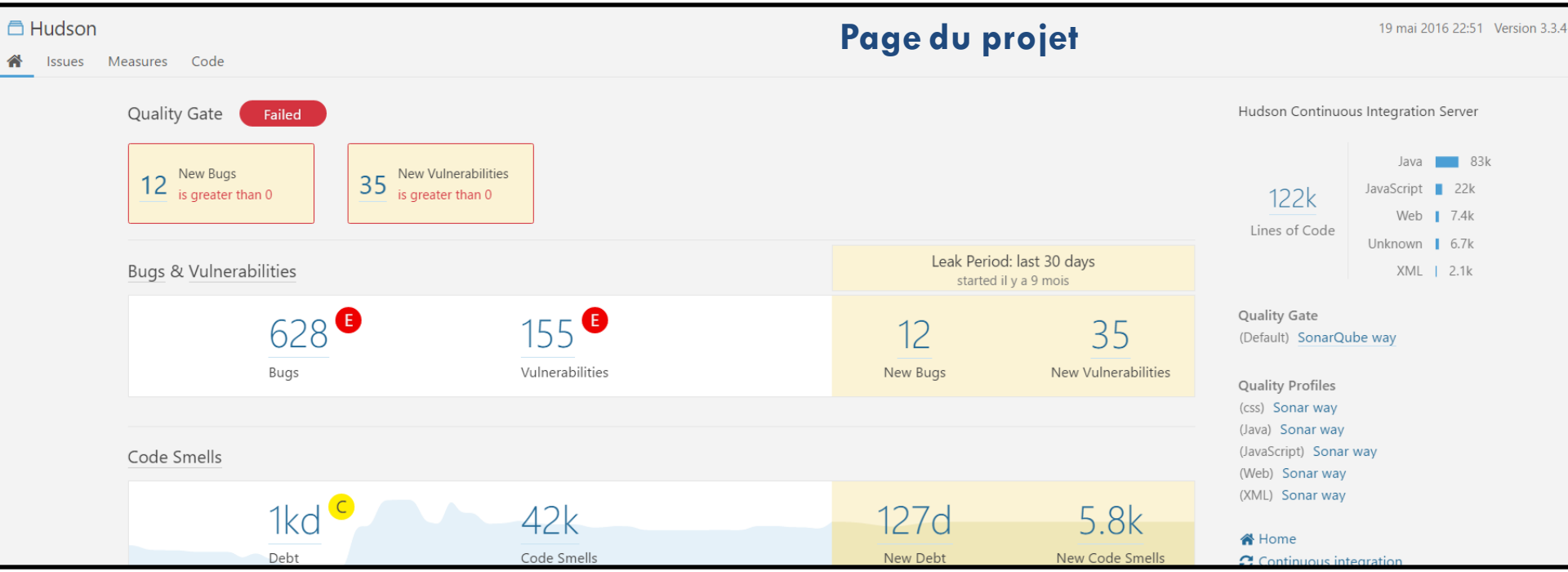


Tableau de bord



Page du projet

Utilisation

sonarlint = détection directement dans le code

```
170    public void setProduct(int id, String designation, SubCategory subCategory, Integer warningPeriod, double minQuantity,
171            double price, boolean visibility, String picture, double conditioning) {
172        setProductId(id);
```

Console | SonarLint On-The-Fly | SonarLint Rule Description

7 items

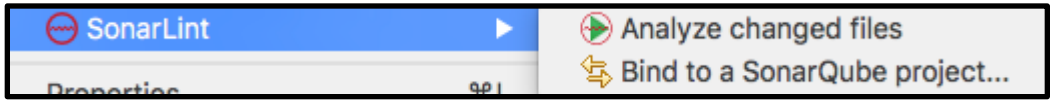
Date	Description	Resource
	Remove this empty statement.	Product.java
	Replace the type specification in this constructor call with the diamond operator ("<>").	Product.java
	Replace the type specification in this constructor call with the diamond operator ("<>").	Product.java
	Method has 9 parameters, which is greater than 7 authorized.	Product.java
	This block of commented-out lines of code should be removed.	Product.java
	Add a nested comment explaining why this method is empty, throw an UnsupportedOperationException or c...	Product.java
	Refactor this method to reduce its Cognitive Complexity from 29 to the 15 allowed.	Product.java

```
170    public void setProduct(int id, String designation, SubCategory subCategory, Integer warningPeriod, double minQuantity
171            double price, boolean visibility, String picture, double conditioning) {
172        setProductId(id);
```

Console | SonarLint On-The-Fly | SonarLint Rule Description

Methods should not have too many parameters (squid:S00107)

A long parameter list can indicate that a new structure should be created to wrap the numerous parameters or that the function is doing too many things.



Conclusion

- Intégrer la qualité au plus tôt dans le développement
  - Configurer les IDE pour respecter un standard de codage
  - Utiliser SonarLint
- Créer un profil qualité adapté
  - Choisir les règles
  - Paramétrer les seuils et les niveaux de criticité
- Forcer un développeur à respecter strictement les seuils peut mener à un comportement contre-productif
  - Taux de documentation => ajout de lignes de commentaires inutiles
  - Taille maximale d'une classe => code découpé sans logique
  - Complexité cyclomatique d'une fonction => code déplacé sans logique

Références

- SonarQube https://docs.sonarqube.org
- SonarLint http://www.sonarlint.org
- Plug-in https://docs.sonarqube.org/display/PLUG/Plugin+Library
- Inspection continue

  https://www.sonarsource.com/resources/white-papers/continuous-inspection.html
- https://blog.sonarsource.com/cognitive-complexity-because-testability-understandability/

Merci

Questions ?