

SonarQube Training

Developer Session

A journey in the land of code quality and security



Agenda

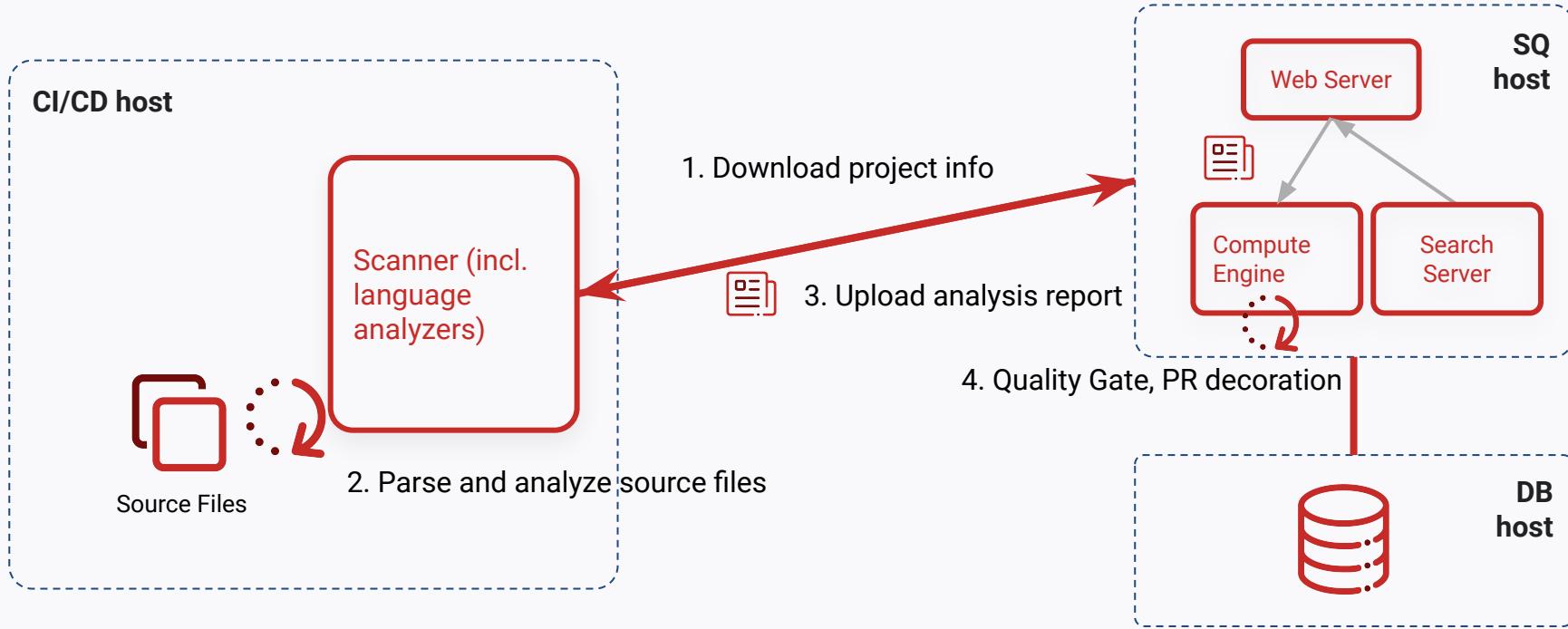
Developer Session

- Scanner integration
- Key metrics
- Rules and Quality Profiles
- Quality Gates
- Configuring Clean as you Code and the New Code Period
- Issue Management
- Working practically with SonarQube SAST results
- Branch Analysis / Pull Request decoration (Merge Request)
- SonarLint
- Q&A

Scanners

Scanning is simple and complex at the same time

Scanner workflow





Java-ish build systems

- Making it easy to scan using the same tool you use to build
- Each plugin provides a SonarQube scanning target
- Java analysis requires compile-time artefacts (retrieved from your config)
- Command line options added using `-D<option>=<value>`

Maven

Maven

```
mvn sonar:sonar [options]
```

Gradle



```
gradlew sonarqube [options]
```



.NET scanning

- Special scanner must be used for .NET solutions
- Integrates with MSBuild and the Roslyn compiler
- Requires a full, clean build of your solution
- Command line options added using /d:"<option>=<value>"



.NET scanning

.NET Framework 4.6+ (Windows)

```
SonarQube.Scanner.MSBuild.exe begin /k:"projectKey" /d:"sonar.login=<AuthToken>" [optional parameters]
```

```
MSBuild.exe <path to solution.sln> /t:rebuild
```

```
SonarQube.Scanner.MSBuild.exe end /d:"sonar.login=<AuthToken>"
```



.NET Core Global Tool (Linux)

```
dotnet sonarscanner begin /k:"project-key" /d:"sonar.login=<AuthToken>" [optional parameters]
```

```
dotnet build <path to solution.sln>
```

```
dotnet sonarscanner end /d:"sonar.login=<AuthToken>"
```

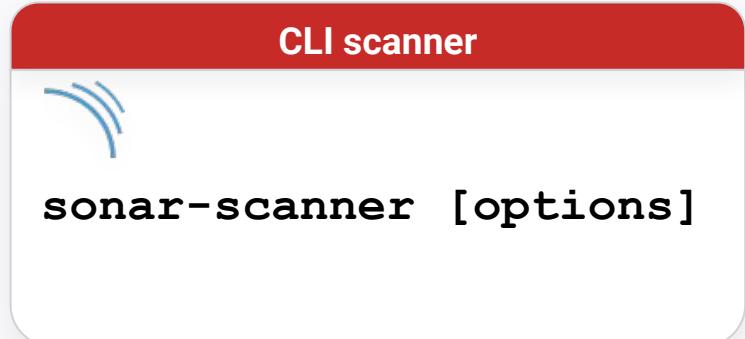


Full parameter list: <https://docs.sonarqube.org/latest/analysis/scan/sonarscanner-for-msbuild/>



Other types of projects: CLI scanner

- Invoked as a command line utility by your DevOps platform (or standalone)
- Used for languages where SonarQube doesn't need compile-time artefacts
 - Everything except Java and .NET
- Also used for C, C++, Objective-C (see later)
- Used in conjunction with **sonar-project.properties** config file or/and...
- ...Command line options added using `-D<option>=<value>`





C, C++, Objective-C

Build Wrapper

```
build-wrapper --out-dir <some_dir> <build command>  
  
sonar-scanner -Dsonar.cfamily.build-wrapper-output=<some_dir>
```

Compilation database

```
<build command that generates compile_commands.json>  
  
sonar-scanner -Dsonar.cfamily.compile-commands=compile_commands.json
```

Compilation DB: [Compilation database | SonarSource blog](#)

Parameters: [C/C++/Objective-C | SonarQube Docs](#)

Example repos: <https://github.com/sonarsource-cfamily-examples>

Note: multi-threading and cached analysis options available



DevOps pipeline integrations

Dedicated tasks and pipeline support:

- GitHub Actions
- BitBucket Cloud Pipelines
- GitLab CI/CD Pipelines
- Azure DevOps extension
- Jenkins plugin



Azure DevOps Extension

1. Define your service endpoints and credentials centrally

Add SonarQube service connection

Connection name: ACME Enterprise SonarQube

Server Url: https://sonarqube.acme.com

Token: (redacted)

OK Close

2. Easy tasks to create your build jobs

Agent job 1 Run on agent +

- Use NuGet 4.4.1 NuGet Tool Installer
- NuGet restore NuGet
- Prepare analysis on SonarQube Prepare Analysis Configuration
- Build solution ***.sln Visual Studio Build
- VsTest - testAssemblies Visual Studio Test
- Run Code Analysis Run Code Analysis
- Publish Quality Gate Result Publish Quality Gate Result



Azure DevOps Extension

Prepare Analysis Task to select MSBuild, Maven, Gradle or Standalone scanner and specify analysis properties

Optional Publish Quality Gate Task to report QG in Azure DevOps

The screenshot shows the Azure DevOps interface. At the top, there's a green checkmark icon and the text "#20181127.1: Add project files." Below this, the pipeline progression is shown:

- Deployments:** 0 deployments were found for this build.
- Build pipeline succeeded:** 0 errors(s) / 4 warning(s)
- Manually queued:** Olivier Korach requested 27 nov 2018
- Associated changes:** 1 commit(s)
Add project files: okorach authored 5 062d42b 27 nov 2018

At the bottom, there's a red box highlighting the "SonarQube Analysis Report" section. It shows a "Quality Gate" status of "Passed". A link to "Detailed SonarQube report" is also present.

The screenshot shows the configuration of a "Prepare analysis" task in Azure DevOps. The task is named "Prepare analysis on SonarQube". The "SonarQube Server Endpoint" is set to "ACME Enterprise SonarQube". The "Choose the way to run the analysis" section is highlighted with a red box and contains the following options:

- Integrate with MSBuild
- Integrate with Maven or Gradle
- Use standalone scanner

The "Project Key" is set to "training:dot-net-sample". The "Project Name" is "Training: .Net Example" and the "Project Version" is "1.0".

In the "Advanced" section, the "Additional Properties" field is highlighted with a red box and contains the following code:

```
# Additional properties that will be passed to the scanner,  
# Put one key=value per line, example:  
  
sonar.exclusions=**/*DAO.cs,**/*gen/**/
```

The "Control Options" and "Output Variables" sections are also visible at the bottom.



Jenkins plugin

1. Install Scanners

Global Tool Configuration

SonarQube Scanner

SonarQube Scanner installations

Add SonarQube Scanner

SonarQube Scanner

Name: SonarQube Scanner CLI

Install automatically

Install from Maven Central

Version: SonarQube Scanner 4.2.0.1873

SonarScanner for MSBuild

SonarScanner for MSBuild installations

Add SonarScanner for MSBuild

SonarScanner for MSBuild

Name: SonarQube Scanner MSBuild

Install automatically

Install from GitHub

Version: SonarScanner for MSBuild 4.8.0.12008 - .NET Fwk 4.6

2. Configure Server

Configure System

SonarQube servers

Environment variables

Enable injection of SonarQube server configuration as build environment variables
If checked, job administrators will be able to inject a SonarQube server configuration as environment variables in the build.

SonarQube installations

Name: SonarQube PROD

Server URL: https://prod.sonar.acme.com
Default is http://localhost:9000

Server authentication token: Migrated SonarQube authentication token

SonarQube authentication token: Mandatory when anonymous access is disabled.

Advanced...

Global properties

Disable deferred wipeout on this node

Environment variables

List of variables

Name: SONAR_SCANNER_OPTS

Value: -Xmx2G -XX:MaxPermSize=128m



Jenkins plugin

3. Define Step or Pipeline *Declarative pipeline example:*

```
pipeline {
    agent any
    stages {
        stage(' SCM') {
            steps {
                git url: ' https://github.com/SonarSource/sonar-java ', branch: ' master'
            }
        }
        stage(' Build') {
            steps {
                withMaven() {
                    sh ' mvn clean org.jacoco:jacoco-maven-plugin:prepare-agent verify '
                }
            }
        }
        stage('SonarQube analysis') {
            steps {
                withMaven() {
                    withSonarQubeEnv('SonarQube PROD') {
                        sh 'mvn sonar:sonar'
                    }
                }
            }
        }
        stage('Quality Gate') {
            steps {
                timeout(time: 1, unit: 'HOURS') {
                    waitForQualityGate false
                }
            }
        }
    }
}
```



Scanning w/ GitLab - Scanner CLI

```
sonarqube-check:  
  image:  
    name: sonarsource/sonar-scanner-cli:latest  
  variables:  
    GIT_DEPTH: "0"  
  cache:  
    key: "${CI_JOB_NAME}"  
    paths:  
      - .sonar/cache  
  script:  
    - sonar-scanner -Dsonar.qualitygate.wait=true  
  allow_failure: false  
  only:  
    - merge_requests  
    - master  
    - develop
```

SonarSource official scanner docker image embedding all latest scanner requirements (OS/JVM/Node.JS)

Force full depth clone
Shallow clone can have side effects

Define scanner plugin cache to not re-download for each pipeline execution

Fail pipeline when Quality Gate fails

Trigger on Merge Requests and key branches



GitHub Actions & docker scanner alternative

```
scan-cli:  
  steps:  
    - name: Checkout code  
      uses: actions/checkout@v2  
      with:
```

```
        fetch-depth: 0
```

Force full depth clone
Shallow clone can have side effects

```
    - name: Cache SonarQube plugins  
      uses: actions/cache@v1  
      with:
```

```
        path: ~/.sonar/cache
```

Define scanner plugin cache to not
re-download for each pipeline execution

```
        key: ${{ runner.os }}-sonar  
        restore-keys: ${{ runner.os }}-sonar
```

```
    - name: Analyze CLI comp
```

```
      uses: docker://sonarsource/sonar-scanner-cli:latest
```

Run official scanner docker image
embedding all latest scanner
requirements (OS/JVM/Node.JS)

```
      env:
```

```
        GITHUB_TOKEN: ${{ secrets.GITHUB_TOKEN }}
```

```
        SONAR_TOKEN: ${{ secrets.SONAR_TOKEN }}
```

```
        SONAR_HOST_URL: ${{ secrets.SONAR_HOST_URL }}
```

Secrets passed to scanner for
analysis submission

```
      with:
```

```
        sonar.qualitygate.wait: false
```

No need to wait for QG
Decoration is asynchronous



Scanning w/ GitHub Actions

```
name: SonarQube scan
jobs:
  scan-cli:
    runs-on: ubuntu-latest
    steps:
      - uses: actions/checkout@v2
        with:
          fetch-depth: 0
      - name: SonarQube Scan
        uses: sonarsource/sonarqube-scan-action@master
        env:
          SONAR_TOKEN: ${{ secrets.SONAR_TOKEN }}
          SONAR_HOST_URL: ${{ secrets.SONAR_HOST_URL }}
```

Force full depth clone

Shallow clone can have side effects

Run scanner CLI GitHub Action

Pass GitHub secrets as env vars for
scanner analysis submission

See: <https://github.com/marketplace/actions/official-sonarqube-scan>



Scanning w/ AWS CodeBuild

```
version: 0.2

phases:
  build:
    commands:
      - mvn sonar:sonar -Dsonar.login=$SONAR_LOGIN -Dsonar.host.url=$SONAR_HOST
```



Branch and PR/MR auto-detection

Simpler pipelines without need for conditional logic:

- GitHub
- GitLab
- Azure DevOps
- BitBucket Pipes
- Jenkins (using multi-branch plugin)
- CodeMagic



Failing pipelines

Your decision whether to fail pipelines:

- Jenkins: webhooks (lightweight, no polling)
- GitHub Actions: SonarQube Quality Gate Check
- BitBucket Pipelines: SonarQube Quality Gate Check
- Others: `sonar.qualitygate.wait=true`

Scanning Example





Narrowing the Focus

Generated and third-party source code

- Usually not under your control
- Often has many issues that 'pollute' your SonarQube projects
- General rule: don't scan this code
- Or scan once in a separate project

Excluding code - *sonar.exclusions, sonar.test.exclusions*

Including only some code - *sonar.inclusions, sonar.test.inclusions*

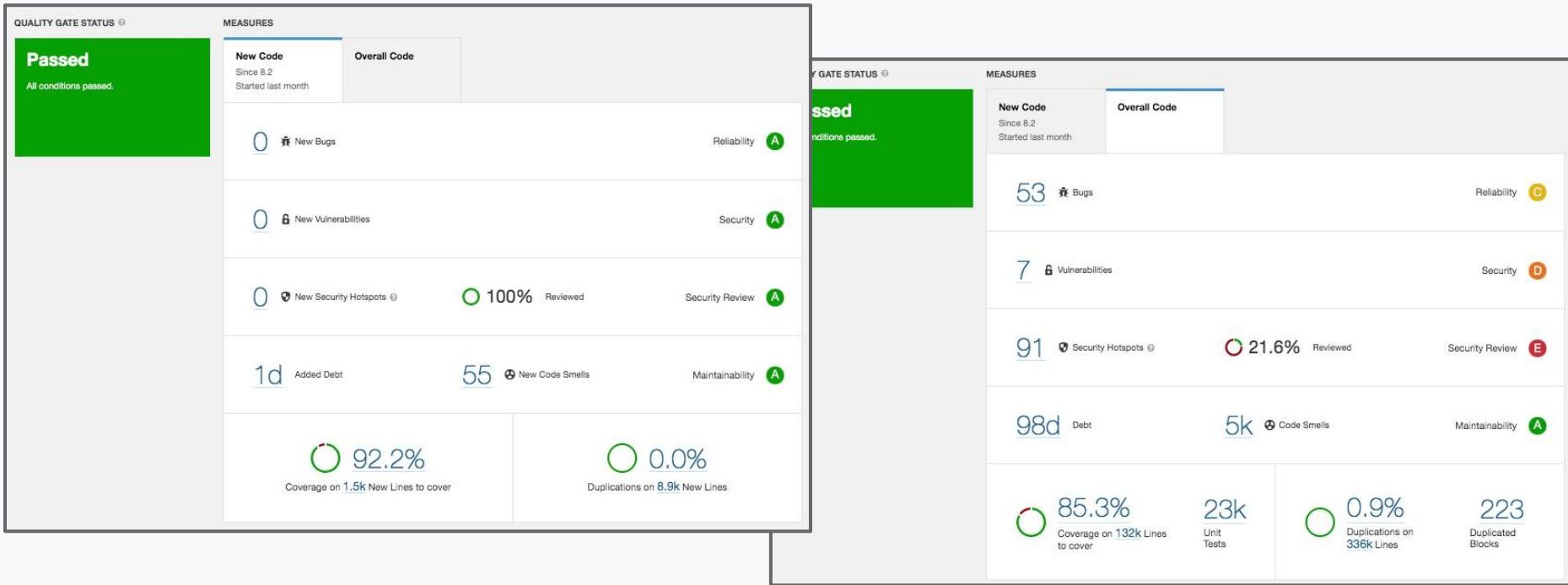
Key metrics

Focus on what matters first



Project key metrics

Focus on what matters first





Security and Reliability Ratings

A matter of severity

Rating represents the single highest Severity Vulnerability or Bug in the code period

| Severity | Rating |
|---------------------|--------|
| Blocker | E |
| Critical | D |
| Major | C |
| Minor | B |
| Info (or No issues) | A |



Tech Debt ratio and Maintainability Rating

A matter of code smell density

- Tech Debt (TD): Total effort to fix all CODE SMELL issues
- TD Ratio: Tech Debt / Effort to entirely rewrite the code
- Example: project with

- 118 code smells → 1,120 minutes of Technical Debt,
- 1,500 LoC → Total rewrite effort = $1500 \times 30 \text{ min} = 45,000 \text{ min}$

Tech Debt Ratio Maintainability Rating

0 - 5%

A

5 - 10%

B

10 - 20%

C

20 - 50%

D

> 50%

E

• **Tech Debt Ratio** = $1120 / 45000 = 2.5\%$

• **Maintainability Rating:**

A



Security Review Rating

How are you doing on review?

SonarQube 8.1+

Rating represents the percentage of hotspots reviewed

Separate ratings for overall code and New Code

| Reviewed | Rating |
|----------|--------|
| < 30% | E |
| 30 - 50% | D |
| 50 - 70% | C |
| 70 - 80% | B |
| >= 80% | A |



Portfolio key metrics

Releasability [?](#)

A

Metric trend

was B 6 months ago

Lowest rated projects

3 projects

Failed

[Measures](#)

[Activity](#)

Reliability [?](#)

B

Metric trend

was A 10 months ago

Lowest rated projects

3 projects in E

[Measures](#)

[Activity](#)

Security Vulnerabilities [?](#)

A

Metric trend

was B 7 months ago

Security Review [?](#)

A

Metric trend

has always been A

Maintainability [?](#)

A

Metric trend

has always been A

% of projects Passed

- > 80% => A
- > 60% => B
- > 40% => C
- > 20% => D
- <= 20% => E

Average of underlying projects ratings

- Unweighted
- All projects under the portfolio at any depth (not portfolios and projects only 1 level below)



Complexity

▼ Complexity ?

| | |
|-----------------------|-------|
| Cyclomatic Complexity | 3,996 |
| Cognitive Complexity | 1,841 |

- **Metric** exists on projects but barely useful
- What matters is to spot methods with excessive complexity
- Better achieved through complexity **rules**: An issue is raised when complexity is above threshold, pointing at the problematic method



Complexity

Cyclomatic = Testability

Cognitive = Understandability
(ie Maintainability too)

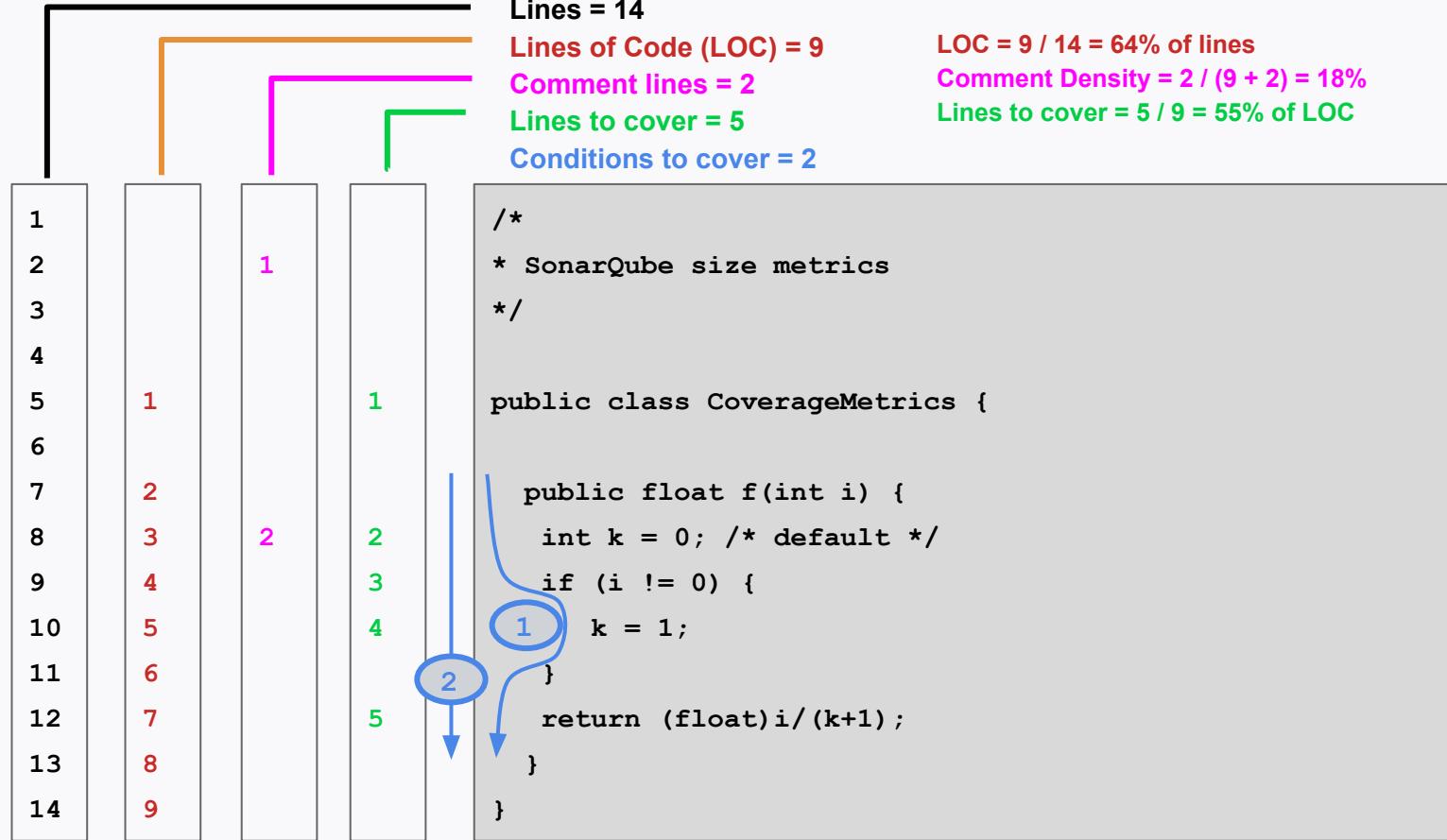
```
public static void highCycloLowCognitive(int i)
{
    switch (i) {
        case 1:
            doSomething();
            break;
        case 2:
            doSomethingElse();
            break;
        case 3:
            forgetIt();
            break;
        case 4:
            forgetItAgain();
            break;
        case 5:
            yetAnotherCase();
            break;
        case 6:
            soBoring();
            break;
        case 7:
            timeForVacations();
            break;
        case 8:
            eightCase();
            break;
        case 9:
            cantSeeTheEndOfIt();
            break;
        case 10:
            tenThisIsIt();
            break;
        default:
            doNothing();
            break;
    }
}
```

Cyclomatic Complexity = 11
Cognitive Complexity = 2

```
public static void highCycloVeryHighCognitive(int i)
{
    if (i <= 5) {
        if (i <= 2) {
            if (i == 1) {
                doSomething();
            } else {
                doSomethingElse();
            }
        } else {
            if (i > 3) {
                forgetItAgain();
            } else {
                forgetIt();
            }
        }
    } else {
        if (i <= 7) {
            if (i == 6) {
                soBoring();
            } else if (i == 7) {
                timeForVacations();
            } else {
                // Never reached
                doNothing();
            }
        } else {
            if (i == 8) {
                eightCase();
            } else if (i == 9) {
                cantSeeTheEndOfIt();
            } else if (i == 10) {
                tenThisIsIt();
            } else {
                doNothing();
            }
        }
    }
}
```

Cyclomatic Complexity = 11
Cognitive Complexity = 27

Line and coverage metrics



Coverage

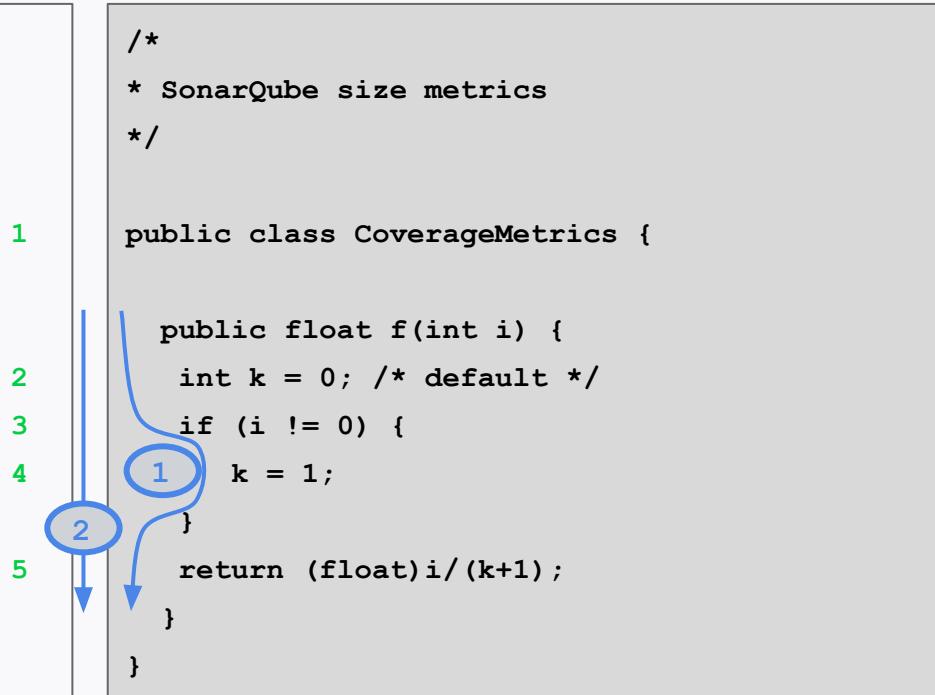
Line coverage = Lines covered / Lines to cover

Condition coverage = (Cond true + cond false) / 2 * branches

(Overall) coverage = (Cond true + cond false + covered lines) / (2 * branches + exec lines)

Lines to cover = 5

Conditions to cover = 2



Unit Test 1: assertEquals(0.5, c.f(1), 0.0);

Line coverage = $5 / 5 = 100\%$

Cond coverage = $(1 + 0) / 2 = 50\%$

Overall coverage = $(1+0+5) / (2+5) = 85.7\%$

Unit Test 2: assertEquals(0.0, c.f(0), 0.0);

Line coverage = $4 / 5 = 80\%$

Cond coverage = $(0 + 1) / 2 = 50\%$

Overall coverage = $(0+1+4) / (2+5) = 71.4\%$

UT1 + UT2

Line coverage = $5 / 5 = 100\%$

Cond coverage = $(1 + 1) / 2 = 100\%$

Overall coverage = $(1+1+5) / (2+5) = 100\%$

Rules and quality profiles

Defining your quality governance



Quality Profile best practices

- **One or a few QP per language only**
- Set an organizational baseline and use QP inheritance
- If using LATEST, revisit QP on a periodic basis



Quality Profile management

- Two key options
 - **Inherit** (from Sonar Way or your own baseline)
 - **Copy** (removes automatic inheritance)
- *Compare* function available to measure drift from baseline

Quality Gates

Defining your quality governance



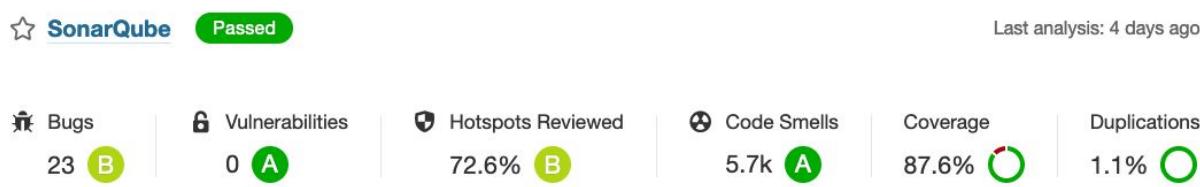
Quality Gate best practices

- One or a few Quality Gates only
- Use metrics on new code
- A reliability rating (0 bugs)
- A security rating (0 vulnerabilities)
- 100% hotspots reviewed
- Set thresholds that are challenging but reachable, then tighten criteria progressively



Quality Gates best practices

6 Key project metrics:



Conditions on New Code

Conditions on New Code apply to all branches and to Pull Requests.

| Metric | Operator | Value |
|----------------------------|-----------------|-------|
| Coverage | is less than | 80.0% |
| Duplicated Lines (%) | is greater than | 3.0% |
| Maintainability Rating | is worse than | A |
| Reliability Rating | is worse than | A |
| Security Hotspots Reviewed | is less than | 100% |
| Security Rating | is worse than | A |

6 Key Quality Gate criteria...
(at least)

... On new code, to adhere to CAYC paradigm



What ? No criteria on legacy ?

Yes you can, but only:

- When your *new code* is well under control
- On very specific metrics

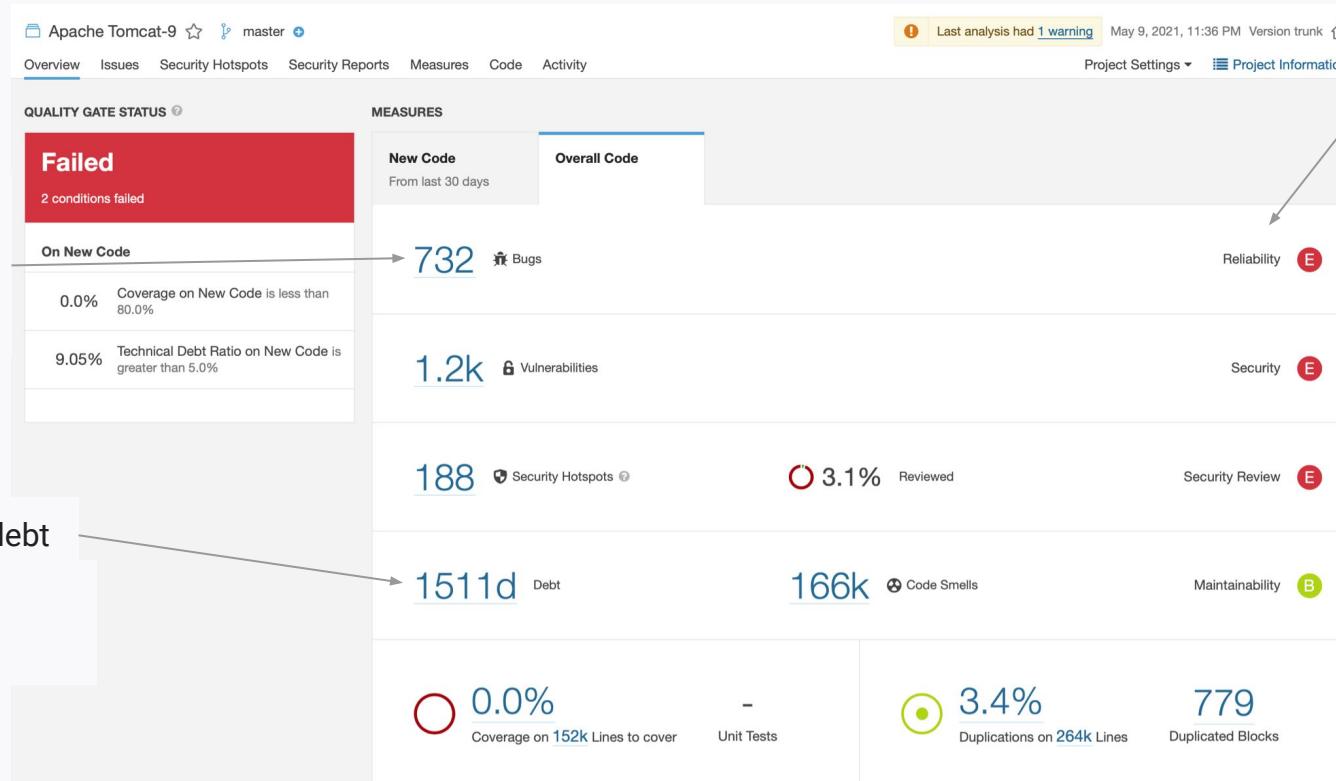
| Conditions on New Code | | | | | |
|--|-----------------|-------|------|--------|--|
| Conditions on New Code apply to all branches and to Pull Requests. | | | | | |
| Metric | Operator | Value | Edit | Delete | |
| Coverage | is less than | 80.0% | | | |
| Duplicated Lines (%) | is greater than | 3.0% | | | |
| Maintainability Rating | is worse than | A | | | |
| Reliability Rating | is worse than | A | | | |
| Security Hotspots Reviewed | is less than | 100% | | | |
| Security Rating | is worse than | A | | | |

| Conditions on Overall Code | | | | | |
|--|---------------|-------|------|--------|--|
| Conditions on Overall Code apply to branches only. | | | | | |
| Metric | Operator | Value | Edit | Delete | |
| Reliability Rating | is worse than | D | | | |
| Security Rating | is worse than | D | | | |

Clean As You Code

Configuring the new code period

Caveats of traditional approaches



Fix “easy” old issues, not “difficult” new ones

Huge legacy debt

Where to find budget?

Risk of functional regression

Developer pushback

- Not my code
- Too many issues
- Boring!

Resistance to evolution of Quality Profiles



What is best practice?

You take personal responsibility for your code





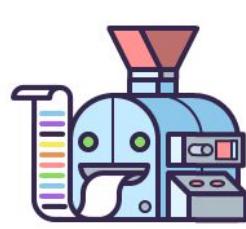
Setting the right new code period

Well tuned new code period leads to better code quality

New code period should be

- Long enough to have time to fix issues before they fall into the “legacy” period
- Short enough to enforce fixing issues before they accumulate too much
- Typically between 2 weeks and 3 months

New Code Example





But... what's the new code ?

4 different configurations possible

- **Previous Version**
 - Best option if compatible with your versioning scheme
 - Align New Code period with releases/sprints
 - Fully automatic
- **Reference Branch**
 - Specific branch chosen as baseline
 - Changes considered as New Code
- **Number of days**
 - Sliding window for new code
- **Specific Analysis**
 - Specific Analysis in a given branch chosen as baseline
 - Changes considered as New Code
 - Can be set up at branch level only



But... what's the new code ?

4 different configurations possible

Project
scans



Previous
version



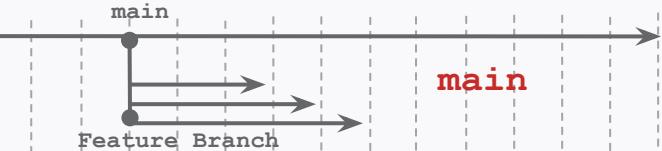
Nbr of Days



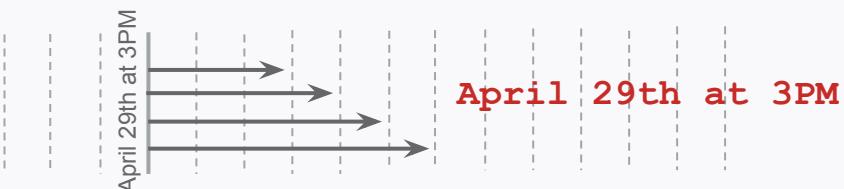
Favor **previous_version** whenever possible



Reference
Branch



Specific
Analysis





CAYC and project versioning

- For smooth CAYC application, project versioning should be properly defined
(sonar.projectVersion)
- Bad practices
 - Always provide same version
 - Change version for every build
 - Randomly change the version
 - Change version although the code fails the Quality Gate



CAYC corollary: Respect Quality Gates

- CAYC approach depends on strict enforcement of Quality Gate
- Don't reset New Code period on a failed quality gate



Issue Management

Because false positives happen

Issue management

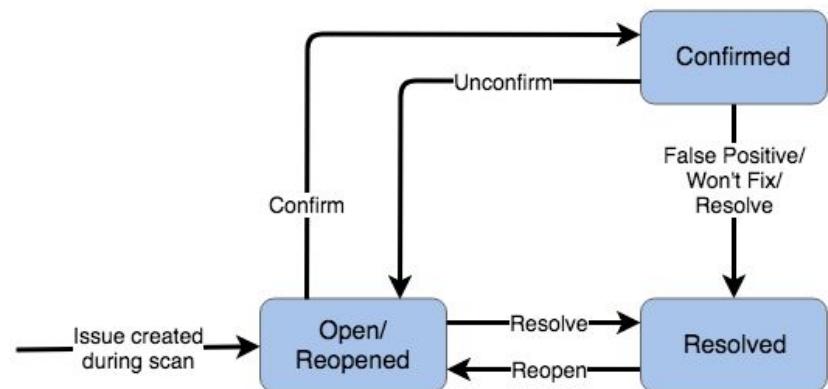
Resolving issues the right way

Automatic resolution

- Just fix the code 😊

Manual resolution

- *Administer issues* permission required
- False positive
- Won't fix
- Never reopened (in that location)



Multiple issue locations

Managing complex data flow issues

- SonarQube highlights data flow
- Cross-file analysis
- Helps you decide where and how to fix

The image shows the SonarQube interface. On the left, a sidebar displays '4 / 4 issues' related to 'Refactor this code to not construct SQL queries directly from tainted user-controlled data.' It lists four files: Servlet.java, BusinessThingsUtils.java, another Servlet.java, and SQLInjectionVulnerabilityCollectionM... Each file has several numbered annotations indicating tainted value propagation. On the right, a code editor shows a Java file named 'SQLInjectionVulnerabilityCollectionMultipleFiles.java'. The code contains several SonarQube annotations, including a 'Vulnerability' at line 15 and another at line 16. A tooltip for the line 16 annotation provides details about refactoring the code.

```
import java.sql.SQLException;
import java.sql.Statement;

public class SQLInjectionVulnerabilityCollectionMultipleFiles {
    Connection con = DatabaseHelper.getJDBCConnection();
    Statement stmt = null;

    try {
        stmt = con.createStatement();

        if (sql == null) {
            sql = "default";
        }

        stmt.execute(sql); // Noncompliant
    } catch (Exception e) {
        // ...
    } finally {
        if (stmt != null) {
            stmt.close();
        }
    }
}
```

Refactor this code to not construct SQL queries directly from tainted user-controlled data.
Why is this an issue?
Vulnerability • Blocker • Open • Alexandre Gigeux • 30min effort • Comment • cert, cwe, owasp-a1, sans-top25-inse...

Working with Security

Injections reporting UI and Security hotspots
workflow



Security

What we do and don't do

SAST

Primarily for
Developers

Auditors

DAST

SCA

In the
DevSecOps
pipeline

Specific
process

Fast!





Security: What we detect

OWASP Top 10 Security Risk Categories

| | | | |
|------------------------------------|--|---|---------------------------------------|
| A1 Injection | A2 Broken Auth | A3 Data Exposure | A4 XXE |
| A5 Broken Access Control | A6 Security Misconfiguration | A7 XSS | A8 Insecure Deserialization |
| | A9 Components with Vulnerabilities | A10 Insufficient Logging & Monitoring | |



Security: What we detect

OWASP Top 10 Security Risk Categories

| | Java | C# | PHP | Python | JS | TS | C/C++ |
|--|------|----|-----|--------|----|----|-------|
| A1 Injection (SQL, OS, LDAP...) | | | | | | | |
| A2 Broken Authentication | | | | | | | |
| A3 Sensitive Data Exposure | | | | | | | |
| A4 XML External Entities (XXE) | | | | | | | |
| A5 Broken Access Control | | | | | | | |
| A6 Security Misconfiguration | | | | | | | |
| A7 Cross-Site Scripting (XSS) | | | | | | | |
| A8 Insecure Deserialization | | | | | | | |
| A9 Using Components with Known Vulnerabilities | | | | | | | |
| A10 Insufficient Logging & Monitoring | | | | | | | |

🔓 Vulnerabilities

Fix Security Risks

The screenshot shows two code editor panes and a central analysis interface.

Top Editor: Shows Java code in `Servlet.java`. Lines 28-36 are highlighted with red boxes numbered 1 through 5, indicating tainted data flow. A tooltip box highlights issue 5: "Refactor this code to not construct SQL queries directly from tainted user-controlled data." It also lists 6 vulnerabilities for this file.

Bottom Editor: Shows Java code in `SQLInjectionVulnerability.java`. Line 18 is highlighted with a red box numbered 6, indicating a tainted query construction. A tooltip box highlights issue 6: "Refactor this code to not construct SQL queries directly from tainted user-controlled data." It also lists 6 vulnerabilities for this file.

Analysis Interface: A central panel displays the following information:

- Refactor this code to not construct SQL queries directly from tainted user-controlled data.**
- Vulnerability +6**
- Servlet.java**:
 - source: taint value is propagated
 - taint value is propagated
- SQLInjectionVulnerability.java**:
 - sink: taint value is propagated

A note at the bottom says: **alt + ↑ ↓ to navigate issue locations**.

At the bottom of the interface, there is a status bar with: **Vulnerability 1 Blocker 0 Open Not assigned 30min effort**, **See Rule**, **3 months ago L18 %**, and a link: **cert, cwe, owasp-a1, sans-top25-inse...**.



Security Hotspots

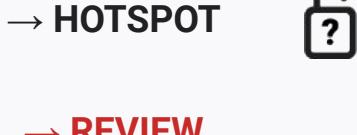
What are they ?

Analyzer has identified a security risk...



→ FIX

Developers should be aware they are writing security-sensitive code...



→ REVIEW

```
@PutMapping //assignment path is bounded to class so we use different http method :-
@ResponseBody
public AttackResult registerNewUser( 1 @RequestParam String 2 username_reg, @RequestParam String email_reg,
AttackResult attackResult = checkArguments(username_reg, email_reg, password_reg);

if (attackResult == null) {
    Connection connection = DatabaseUtilities.getConnection(webSession);
    checkDatabase(connection);

 4 String checkUserQuery = 3 "select userid from " + USERS_TABLE_NAME + " where userid = '" + username_r
Statement statement = connection.createStatement();
ResultSet resultSet = 5 statement.executeQuery(checkUserQuery);

Refactor this code to not construct SQL queries directly from tainted user-controlled data. 2 years ago ▾ L52 ⚙
See Rule
```

🔒 Vulnerability ▾ ❶ Blocker ▾ ○ Open ▾ Not assigned ▾ 30min effort Comment 🔗 cert, cwe, owasp-a1, sans-top25-inse... ▾

```
String token = Jwts.builder()
    .setClaims(claims)
    .signWith(io.jsonwebtoken.SignatureAlgorithm.HS512, JWT_PASSWORD)
    .compact();
Cookie cookie = new Cookie("access_token", token);

Make sure that this cookie is used safely. See Rule
last year ▾ L77 ⚙
🔗 Security Hotspot ○ In Review ▾ Not assigned ▾ Comment
cert, cwe, owasp-a3 ▾
```

```
response.addCookie(cookie);
response.setStatus(HttpStatus.OK.value());
response.setContentType(MediaType.APPLICATION_JSON_VALUE);
} else {
```



Hotspot lifecycle

The screenshot illustrates the SonarQube Security Hotspots interface, highlighting various features and the hotspot lifecycle process.

- Dedicated tab:** A red box highlights the "Security Hotspots" tab in the top navigation bar.
- Sorted by priority:** A red box highlights the list of hotspots on the left, which are sorted by priority: HIGH, MEDIUM, and LOW. The first hotspot listed is "Authentication" with a review priority of HIGH.
- Progress indicator:** A red box highlights the progress indicator in the top right corner, showing "Security Hotspots Reviewed" at 21.6%.
- New workflow:** A red box highlights the modal dialog for selecting a status, which includes options: To review, Fixed, or Safe.

Key UI Elements:

- Top Navigation:** Overview, Issues, Security Hotspots (highlighted), Security Reports, Measures, Code, Activity.
- Filters:** Assigned to me (All), Status (To review), Overall code.
- Project Settings:** Project information, Security Hotspots Reviewed (21.6%).
- Code Review Details:** Category (Authentication), Review priority (HIGH), Assignee (Not assigned). The code snippet shows a hard-coded password in an expression.
- Status Selection Modal:** "Select a status..." dialog with options: To review (selected), Fixed, or Safe.
- Comment Input:** "Add a comment (Optional)" input field with Markdown Help instructions.
- Code Snippet:** A Java code snippet from SonarQube showing a hard-coded password in an expression.
- Risk Summary:** "Because it is easy to extract strings from an application source code, this is particularly true for applications that are distributed or that are open-source."
- Vulnerability History:** "In the past, it has led to the following vulnerabilities:" with a link to CVE-2019-13466.



Security Rating

Overview of code security

Projects

Highest severity Vulnerability in the code period

Portfolios

Average of the Security Ratings of included Projects

| Severity | Rating |
|----------|--------|
| Blocker | E |
| Critical | D |
| Major | C |
| Minor | B |
| Info | A |



Security Review Rating

How are you doing on Security Hotspot review?

Projects

Percentage of hotspots reviewed in the code period

Portfolios

Average of the Security Review Ratings of included Projects

| Reviewed | Rating |
|----------|--------|
| < 30% | E |
| 30 - 50% | D |
| 50 - 70% | C |
| 70 - 80% | B |
| >= 80% | A |



Customization

Support your Proprietary Frameworks

Make the Security Engine aware of your own frameworks by adding:

SOURCES

SANITIZERS

PASS-THROUGHS

SINKS

Input data, vulnerable to
Injection Attacks (tainted
data)

**Removes malicious
content** from Input data

Keep track of data
passing through libraries
outside of your code

**Security-sensitive
functions** that accept
Input data



Compliance

OWASP/CWE Security Reports for Projects & Portfolios

| SonarSource | | OWASP Top 10 | CWE Top 25 | SANS Top 25 |
|---|--|--|---|--------------------------------|
| <input type="checkbox"/> Show CWE distribution <small>?</small> | | | | |
| Categories <small>?</small> | |  Security Vulnerabilities |  Security Hotspots | |
| Buffer Overflow | | 1 E | | 23 E |
| SQL Injection | | 0 A | | 0 A |
| Code Injection (RCE) | | 0 A | | 0 A |
| Object Injection | | 0 A | | 0 A |
| Command Injection | | 0 A | | 0 A |
| Path Traversal Injection | | 0 A | | 0 A |
| LDAP Injection | | 0 A | | 0 A |
| XPath Injection | | 0 A | | 0 A |
| Log Injection | | 0 A | | 0 A |
| XML External Entity (XXE) | | 0 A | | 0 A |
| Cross-Site Scripting (XSS) | | 0 A | | 0 A |
| Denial of Service (DoS) | | 0 A | | 1 E |
| Server-Side Request Forgery (SSRF) | | 0 A | | 0 A |
| Cross-Site Request Forgery (CSRF) | | 0 A | | 0 A |



Security in 8.9 LTS

Languages coverage and evolution

Web Apps

- Injections (taint) vulnerabilities
- Non injection vulnerabilities
- Security Hotspots



System & Embedded

- Buffer overflow Vulnerabilities
- Other non-injection Vulnerabilities
- Security Hotspots





Security best practices

Elevate security governance with SonarQube

- Include **Security Rating** in Quality Gates
- Incorporate Security **Hotspot reviews** into your development cycle
- Enforce Security Hotspot reviews using the **Security Review Rating** Quality Gate criterion (8.3+)

Branch and PR analysis

Shift left in your feature branches

Why Use Branch Analysis in SonarQube?

- Issue metadata shared across branches
- Focus on issues created within each branch
- Branch-specific New Code Periods
- Licensing

Scanner Configuration

- sonar.branch.name
- sonar.pullrequest.key
- sonar.pullrequest.branch
- sonar.pullrequest.base
- Automated in Jenkins, Azure Devops, GitLab and GitHub

Issue Behavior

- Issues metadata is synched
 - From parent branch at creation
 - From PR to target Branch after merge
 - To reference branch after merge
- Comment added when issue is synced

"The issue has been copied from branch 'xxx' to branch 'yyy'."

Integrating Branch Analysis

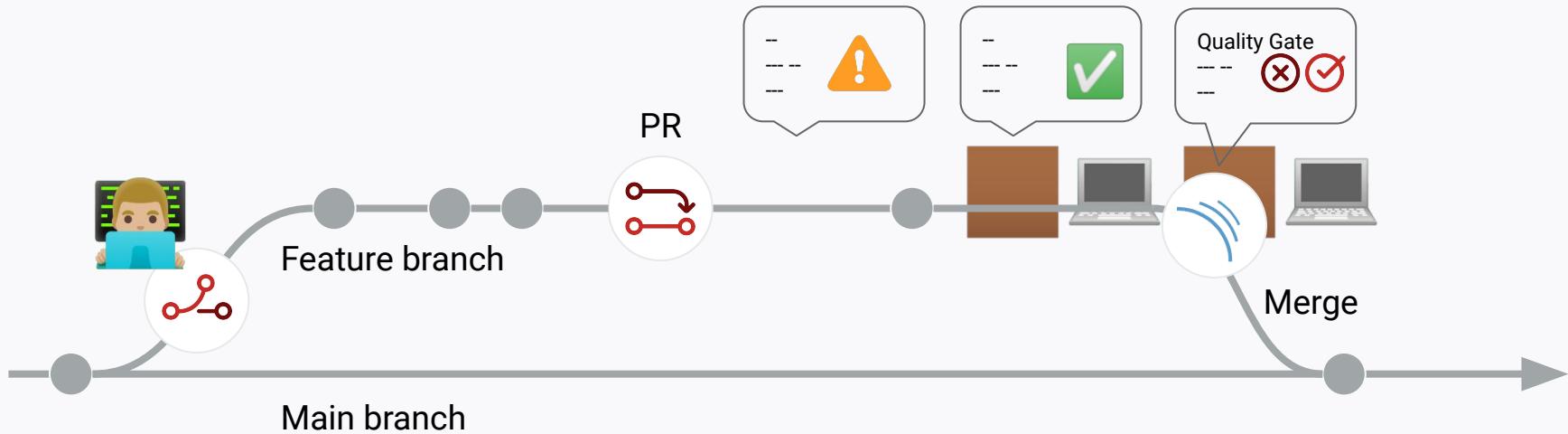
System setup

- Set global branch naming regex to organization standard
- Set lifespan of short-lived branches

Per-project setup

- Set up project scanning main ('master') branch if not already existing
- Set project branch naming regex if varies from global setting
- Add branch builds to CI/CD
 - Retrieve `sonar.branch.name` and `sonar.branch.target` from build trigger
- Build out branch structure ensuring parent branches scanned before children
- Set leak period for long-lived branches is different from main.

Pull request workflow



Thanks to **SonarQube PR decoration**

- Developers know about issues in their code **before merging**
- Information about issues is **pushed in the SCM**
- It is possible to **enforce a policy** that PR quality gate must be passed before pull request can be merged



Example: GitHub PR Decoration

Integration steps

Global Settings

1. Register App on GitHub
2. Add information to SonarQube
 - a. Private Key
 - b. API URL
 - c. App Name and ID
3. Install App on Organization

Project Settings

1. Add Repository identifier
2. Add pull request parameters to scan

(Or use SonarQube Onboarding UI)



GitHub PR Decoration

SonarQube ★ 4339 – Attempt to have a new servlet for merge into master from feature/bp/swagger-poc

June 23, 2021, 5:30 PM See the PR Failed

Overview Issues Security Hotspots Measures Code

QUALITY GATE STATUS Failed

FAILED CONDITIONS

Coverage on New Code 0.0% (Code is less than 85.0%)

MEASURES

| Condition | Count | Reliability |
|-----------------------------------|-------|-----------------------------|
| New Bugs | 0 | A |
| New Vulnerabilities | 0 | A |
| New Security Hotspots | 0 | A |
| New Code Smells | 1 | A |
| Coverage on 10 New Lines to cover | 0.0% | 87.4% Estimated after merge |
| Duplications on 105 New Lines | 0.0% | 1.0% Estimated after merge |

Pull requests 11 Actions Security Insights

Attempt to have a new servlet #4339

Closed belen-pruvost-s... wants to merge 3 commits into `master` from `feature/bp/swagger-poc`

Conversation 1 Commits 3 Checks 19 Files changed 5

not working thing 1d6778e ▾

SonarSource-Next

SonarQube Code Analysis Re-run

Cirrus CI

SLE trigger on: pull_request

SonarSource-Next / SonarQube Code Analysis failed 6 days ago in 10m 41s

Quality Gate failed

Failed

0.0% Coverage on New Code (is less than 85%)

See analysis details on SonarQube

Additional information

The following metrics might not affect the Quality Gate status but improving them will improve your project code quality and security.

1 Issue

- 0 Bugs
- 0 Vulnerabilities
- 0 Security Hotspots
- 1 Code Smell

Coverage and Duplications

- 0.0% Coverage (87.4% Estimated after merge)
- 0.0% Duplication (1.0% Estimated after merge)



...and other ALMs

Quality reports

SonarQube FAILED

SonarQube reported 1 week ago

+ Find integrations

SonarQube

Quality Gate failed

- E Reliability Rating on New Code (is worse than A)
- 68.2% Coverage on New Code (is less than 80%)
- 17.7% Duplicated Lines on New Code (is greater than 3%)

| Bugs | 1 | Code coverage | 68.2% |
|-----------------|-------|---------------|-------|
| Vulnerabilities | 0 | Duplication | 17.7% |
| Code Smells | 4 | | |
| Annotations | 3 1 1 | | |

Severity Message

- High Add some tests to this class.
- High Add an end condition to this loop.
- High Add a nested comment explaining why this m
- Medium Remove this useless assignment to local varia
- Low Remove this unused "primary" local variable.

Projects Groups Snippets Help

Show all activity

Discussion 1 Commits 7 Pipelines 7 Changes 5

SonarQube Bot @SonarQube-Bot · 3 days ago

SonarQube Code Analysis

Quality Gate failed

Failed

- ✗ D Security Rating on New Code (is worse than A)
- ✗ 0.0% Coverage on New Code (is less than 80%)

See analysis details on SonarQube

2 ACTIVE Adding a new API!

Jeff Zapotocny feature/newAPI into master

0/4 resolved Approve Set auto-complete ...

Overview Files Updates Commits

Description

Kicking off this project with a new candidate API method.

Show everything ▾

NB Add a comment...

C# MyAPIController.cs /Simple Web App/Controllers/MyAPIController.cs 5/8/2019

Policies

- Required
- ✗ Not all comments resolved
- ✓ JZ Branch Build Policy succeeded

Status

- ✗ Quality Gate failed

Work Items

No related work items

Sign in / Register

vers

None

None

0

2

Additional information

2 ACTIVE

Adding a new API!

JZ Jeff Zapotoczny

feature/newAPI into master

0/4 resolved

NB

Approve



Set auto-complete



Overview Files Updates Commits

Description

Kicking off this project with a new candidate API method.

Show everything ▾



Add a comment...



C# MyAPIController.cs

/Simple Web App/Controllers/MyAPIController.cs

5/8/2019

```
^ 19 +         connection.ConnectionString = "db info";
 20 +         connection.Open();
 21 +         var selectSql = string.Format("select from MyStuff where id='{0}';", i
 22 +         var selectCommand = new SqlCommand(selectSql, connection);
 23 +
 24 +
 25 +         var dataReader = selectCommand.ExecuteReader();
    return dataReader.GetString(0);
```



Jeff Zapotoczny 5/8/2019

🔒 Vulnerability: Refactor this code to not construct SQL queries directly from tainted user-controlled data. ([roslyn.sonaranalyzer.security.cs:S3649](#))

Active ▾

[See it in SonarCloud](#)

Write a reply...

Resolve

Policies

Required

✗ Not all comments resolved

✓ JZ Branch Build Policy succeeded

Status

✗ Quality Gate failed

Work Items

No related work items



Reviewers



No reviewers

Labels

[Add label](#)

SonarLint

The ultimate shift left

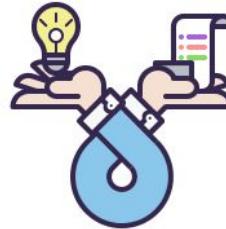
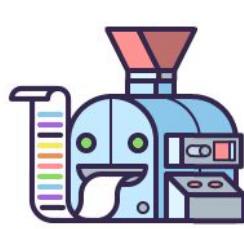


SonarLint key features

- Supports **five major IDEs**
 - Plus variants (e.g. Eclipse CDT, IDz/RDz, Topaz)
- **On-the-fly analysis** of code as you type
- Analysis of changed files or all files
- **Connected Mode:**
 - Analyze more languages
 - Align Quality Profiles with SonarQube
 - Suppress False Positives / Won't Fix
 - In-IDE notifications of key events in SonarQube
 - Hotspots and Vulnerabilities in the IDE



SonarLint Example





SonarLint key features

On-the-fly analysis

The screenshot shows the SonarLint interface integrated into an IDE. On the left, the project structure of 'flowable-bpmn-layout' is visible, with 'BpmnAutoLayout.java' selected. The main area displays the Java code for 'BpmnAutoLayout'. A red callout points to a specific line of code: 'startPoint.setY(closestPoint.getY());'. A tooltip from SonarLint states: 'SonarLint: closestPoint is dereferenced.' Another tooltip further down the code indicates: 'SonarLint: A "NullPointerException" could be thrown; "closestPoint" is nullable here.' A red callout on the right highlights these inline issue annotations.

Issues highlighted inline

```
mxPoint westPoint = new mxPoint(gatewayState.getX(), gatewayState.getY() + gatewayState.getHeight() / 2);

double closestDistance = Double.MAX_VALUE;
mxPoint closestPoint = null;
for (mxPoint rhombusPoint : Arrays.asList(northPoint, southPoint, eastPoint, westPoint)) {
    double distance = euclideanDistance(startPoint, rhombusPoint);
    if (distance < closestDistance) {
        closestDistance = distance;
        closestPoint = rhombusPoint;
    }
}
startPoint.setX(closestPoint.getX());
startPoint.setY(closestPoint.getY()); // SonarLint: closestPoint is dereferenced.

// We also need to // Since we know the layout is from left to right, this is not a
// problem
if (points.size() > 1) {
    mxPoint nextPoint = points.get(1);
    nextPoint.setY(closestPoint.getY());
}

createDiagramInterchangeInformation(handledFlowElements.get(sequenceFlowId), optimizeEdgePoints(points));
}

protected void generateAssociationDiagramInterchangeElements() {
    for (String associationId : generatedAssociationEdges.keySet()) {
        Object edge = generatedAssociationEdges.get(associationId);
        List<mxPoint> points = new ArrayList<mxPoint>();
        List<mxPoint> northPoints = new ArrayList<mxPoint>();
        List<mxPoint> southPoints = new ArrayList<mxPoint>();
        List<mxPoint> eastPoints = new ArrayList<mxPoint>();
        List<mxPoint> westPoints = new ArrayList<mxPoint>();
        ...
    }
}
```

SonarLint: Current file Report Log

Found 12 issues in 1 file

BpmnAutoLayout.java (12 issues)

- (247, 26) Define and throw a dedicated exception instead of using a generic one.
- (257, 8) Replace the synchronized class "Hashtable" by an unsynchronized one such as "HashMap".
- (264, 8) Replace the synchronized class "Hashtable" by an unsynchronized one such as "HashMap".
- (295, 8) Replace the synchronized class "Hashtable" by an unsynchronized one such as "HashMap".
- (302, 8) Replace the synchronized class "Hashtable" by an unsynchronized one such as "HashMap".
- (334, 12) Replace the synchronized class "Hashtable" by an unsynchronized one such as "HashMap".
- (347, 12) Replace the synchronized class "Hashtable" by an unsynchronized one such as "HashMap".
- (366, 8) Iterate over the "entrySet" instead of the "keySet".
- (383, 8) Iterate over the "entrySet" instead of the "keySet".
- ✗ (410, 32) A "NullPointerException" could be thrown; "closestPoint" is nullable here. (+2 locations)
- ✗ (428, 8) Iterate over the "entrySet" instead of the "keySet".
- ✗ (460, 205) This branch's code block is the same as the block for the branch on line 458. (+1 location)

Automatic analysis is enabled

Rule Locations

Null pointers should not be dereferenced

Bug Major squid:S2269

A reference to null should never be dereferenced/accessed. Doing so will cause a NullPointerException to be thrown. At best, such an exception will cause abrupt program termination. At worst, it could expose debugging information that would be useful to an attacker, or it could allow an attacker to bypass security measures.

Note that when they are present, this rule takes advantage of @CheckForNull and @NotNull annotations defined in JSR-308 to understand which values are and are not nullable except when @NotNull is used on the parameter to equals, which by contract should always work with null.

Noncompliant Code Example

Full rule documentation available



SonarLint connected mode

The screenshot illustrates the SonarLint connected mode interface across four panels:

- Left Panel:** Shows the SonarLint General Settings under Other Settings. A red callout box labeled "Bind to SonarQube" points to the "Bind to SonarQube / SonarCloud" section.
- Top Center Panel:** Shows the SonarLint General Settings with a red callout box labeled "Configure rules locally" pointing to the "Automatically trigger analysis" checkbox.
- Right Center Panel:** Shows the SonarLint Project Settings with a red callout box labeled "Align to project Quality Profile" pointing to the "Bind project to SonarQube / SonarCloud" checkbox.
- Bottom Panel:** Shows the SonarLint interface with an open Java file and a notification bar at the bottom stating "SonarQube event: Quality Gate of project 'Sonar.java' is now Red (was Green)...". A red callout box labeled "Manage advanced configuration" points to the "Configure the connection..." button in the Project binding section.



SonarLint language support

Which languages in which IDE?

- Main languages for each IDE
 - E.g. Java for Eclipse, C# for Visual Studio
- Plus supplementary languages where possible
- Plus commercial languages in Connected Mode
 - E.g. COBOL in Eclipse
- Up-to-date list on <https://www.sonarlint.org/>
 - Choose your IDE to see the list

Questions ?

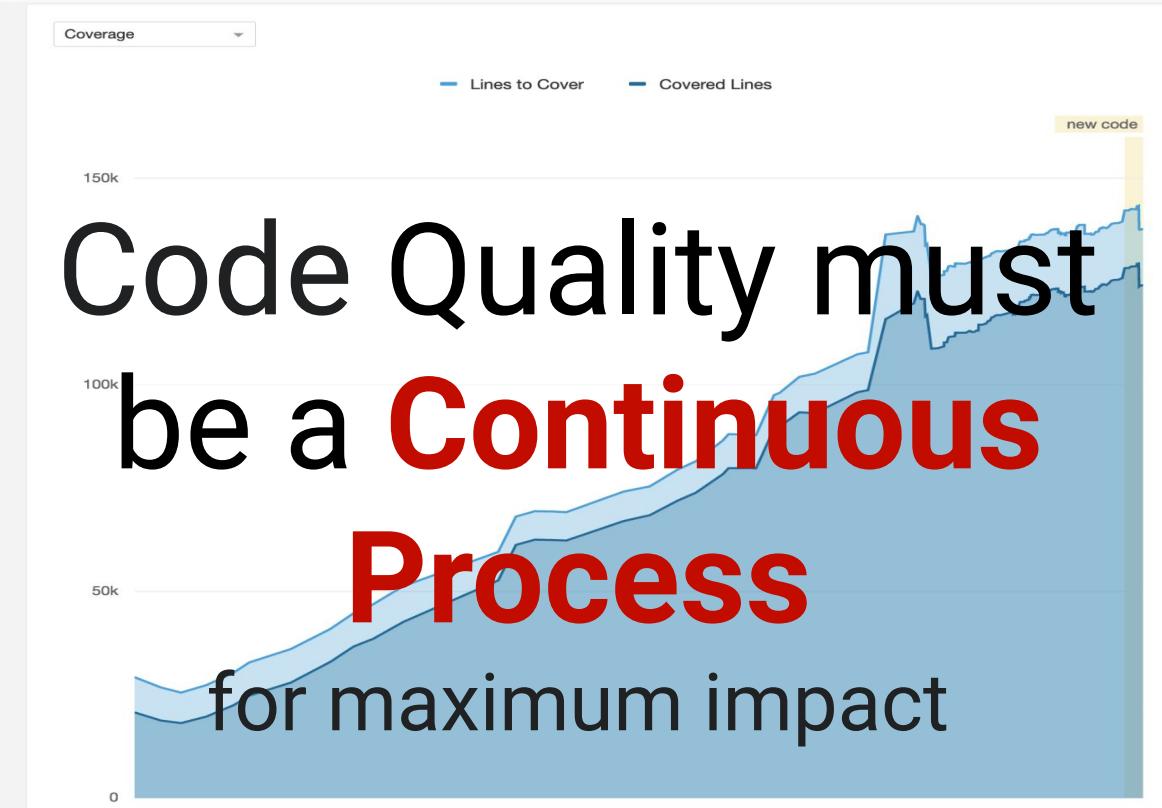




TAKE AWAY

If you only remember a few things...

| 9.0-SNAPSHOT | |
|--|---------------------------|
| June 29, 2021 | |
| 9:58 AM | Build string: 9.0.0.45353 |
| ◇ Version: 9.0-SNAPSHOT | |
| 9:46 AM | Build string: 9.0.0.45351 |
| 9:45 AM | Build string: 9.0.0.45350 |
| 9:35 AM | Build string: 9.0.0.45348 |
| June 28, 2021 | |
| 4:21 PM | Build string: 9.0.0.45328 |
| June 25, 2021 | |
| 10:24 PM | Build string: 9.0.0.45301 |
| June 24, 2021 | |
| 10:37 AM | Build string: 9.0.0.45236 |
| ◇ Quality Profile: Changes in 'For SQ Only' (Java) | |
| June 23, 2021 | |

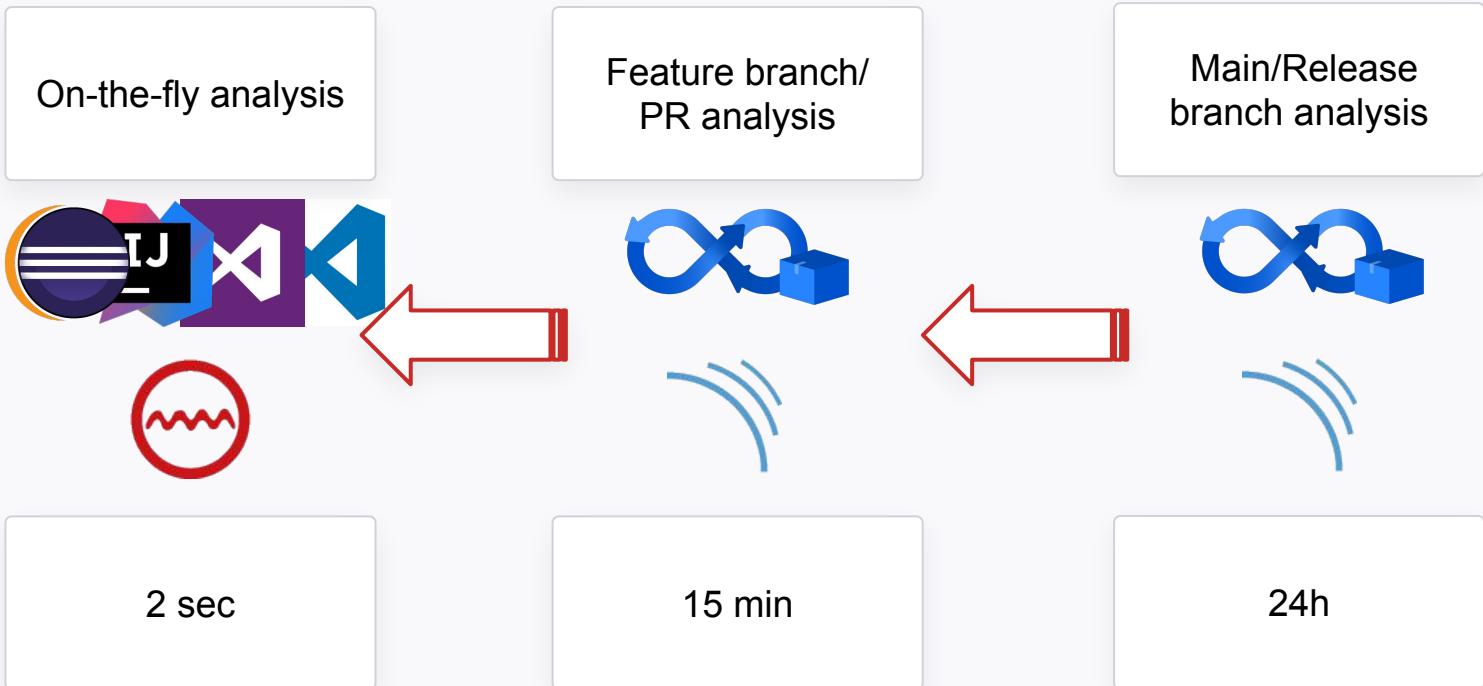




KEEP
CALM
AND
CLEAN AS YOU CODE

Shift Left

Shorten the feedback loop



Feedback
loop

Set and **Enforce** Quality Gates



**Update Quality Gates
(and optionally Profiles)
on a regular basis**



**KEEP
CALM
AND
RAISE
THE BAR**

Feedback is a gift ! Thank you



<http://tiny.cc/h21xlz>



Questions ?

