



Integração Contínua – Prática - Etapa C

Prof. Jean Carlo Rossa Hauck
Prof. Osmar de Oliveira Braz Junior
Prof. Richard Henrique de Souza

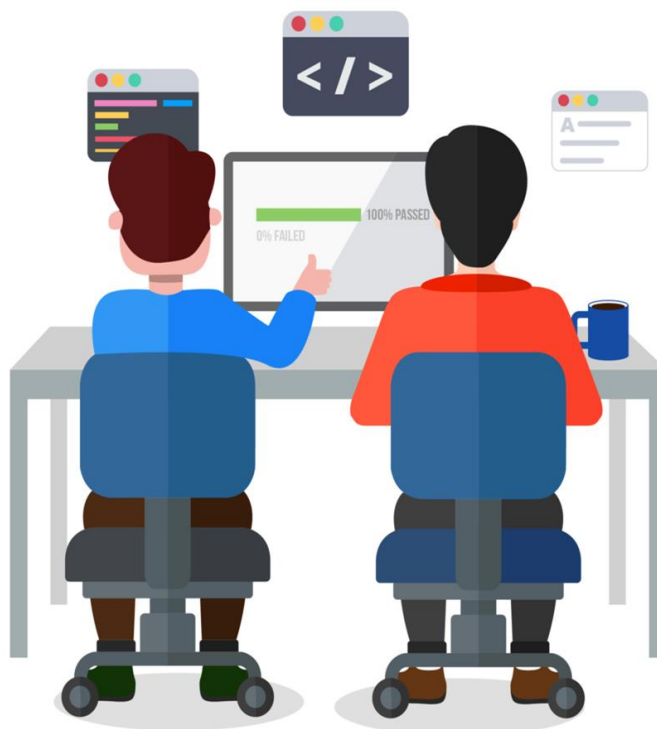
Objetivos

- Realizar um exemplo de **integração contínua** da **análise** até **cobertura** do código.
- A **integração** continua será realizada em 3 ambientes distintos com tarefas distintas.
- A **análise** irá considerar diversas métricas de qualidade de software como confiabilidade, manutibilidade, segurança, **corbertura** e duplicação de código.
- Nesta etapa realizaremos a **análise** do código.

Atividade em Grupo

Para esta atividade crie grupos de 2 alunos, para desenvolver a atividade segundo ***Pair Programming***.

Navegador



Piloto

Pair Programming

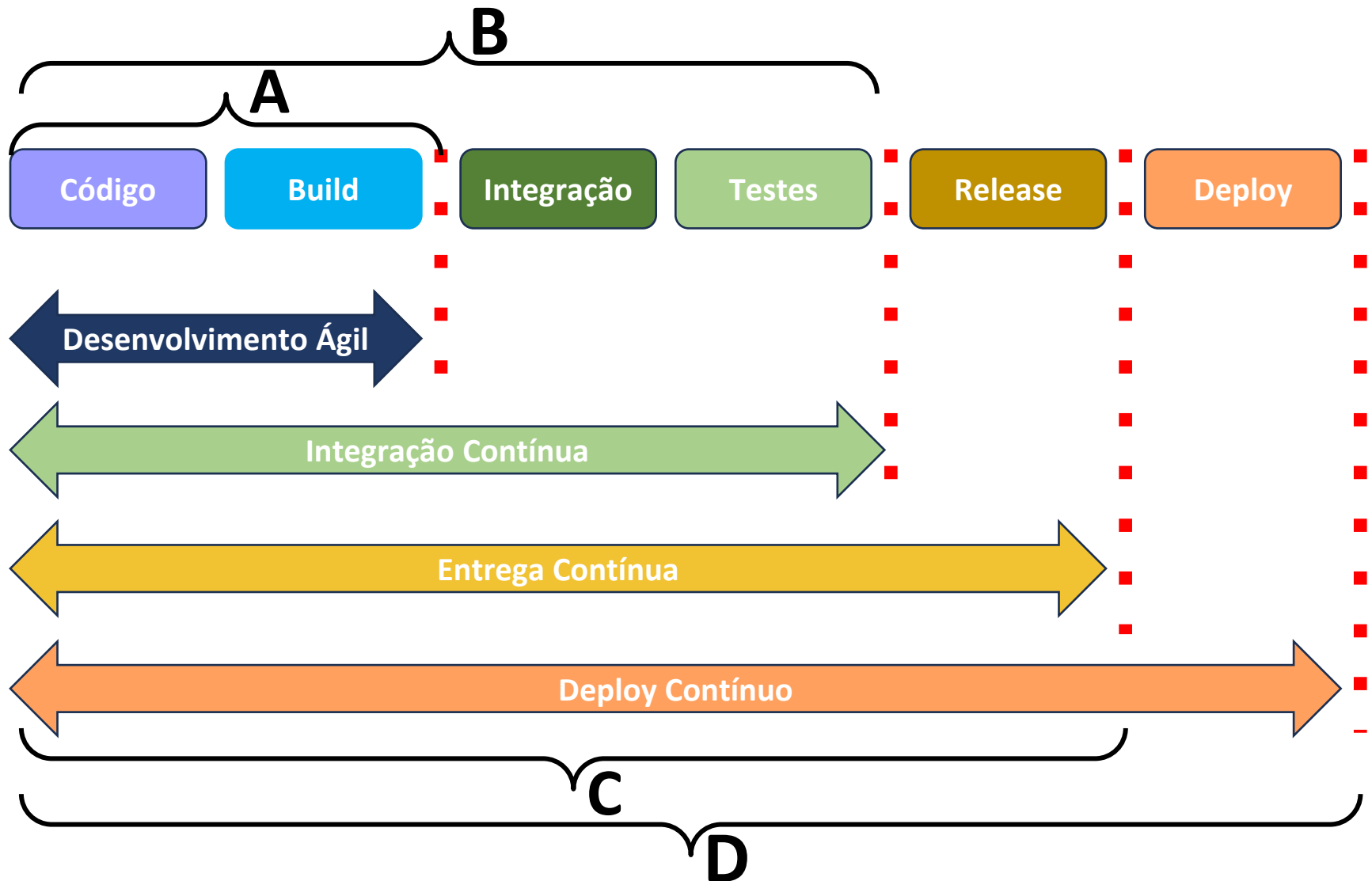
- Um é o **piloto**, responsável por escrever o código, o outro o navegador, acompanha a escrita de código e verificar se está de acordo com os **padrões do projeto** e de encontro à solução necessária.
- A intenção desta técnica é **evitar** erros de lógica, e ter um código mais confiável e melhor estruturado, utilizando-se para isso a máxima de que “**duas cabeças pensam melhor do que uma**”.

Integração, Entrega e Implantação Contínua

■ Abstração do Pipeline



Integração, Entrega e Implantação Contínua



Ferramentas utilizadas

- IDE com suporte
 - Apache Maven
 - JUnit 4
 - Github
- Github
 - Github Actions
 - Sonarcloud
 - JaCoCo

maven

JUnit



sonarcloud 

JACOCO
Java Code Coverage

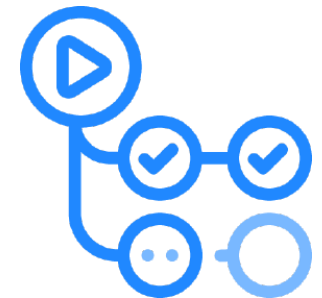
Atividades práticas

- A - Criação de Projeto e testes unitários
 - Criar projeto na IDE
 - Automatizado com Apache Maven
 - Criar testes unitários com JUnit 4
 - Armazenar projeto no Github
- B - Integração Continua
 - Github Actions
 - JUnit 4
- C - Análise do Código
 - Sonarcloud
 - Integração com Github Actions
- D - Cobertura do código
 - Jacoco
 - Integração com Github Actions
 - Integração com Maven e Sonarcloud

C - Análise do Código

- Sonarcloud
- Integração com Github Actions

sonarcloud 



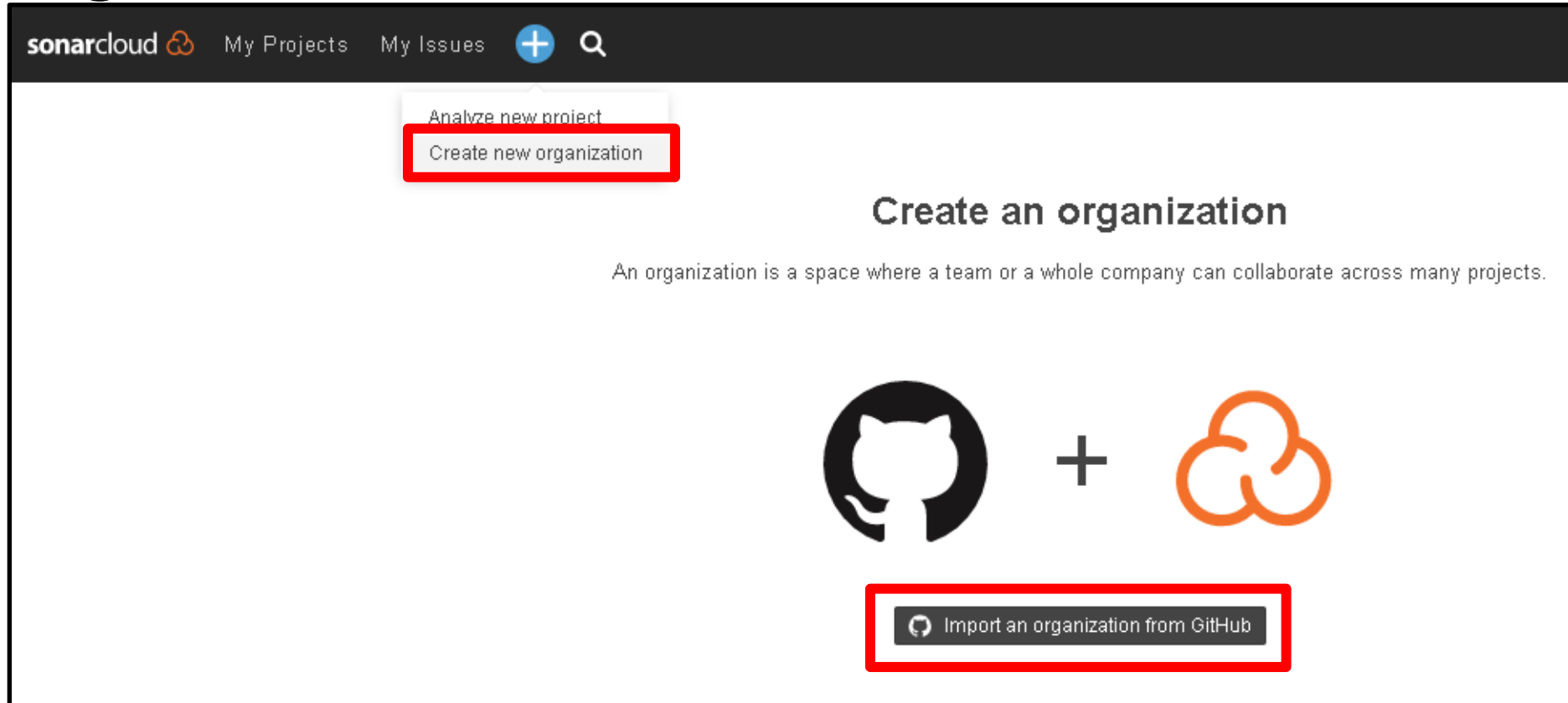
Passo 1

Crie uma conta no sonarcloud.

<https://sonarcloud.io/>

Passo 2


Criar uma organização no **sonarcloud** vinculado ao **github**.



The screenshot shows the SonarCloud web interface. The top navigation bar includes the SonarCloud logo, 'My Projects', 'My Issues', a plus icon, and a search icon. A dropdown menu is open from the plus icon, showing 'Analyze new project' and 'Create new organization', with the latter highlighted by a red rectangle. The main content area is titled 'Create an organization' and includes a descriptive sentence: 'An organization is a space where a team or a whole company can collaborate across many projects.' Below this, there is a visual representation of the integration: the GitHub logo (a black octocat) followed by a plus sign and the SonarCloud logo (three orange interlocking circles). At the bottom of the page, a button labeled 'Import an organization from GitHub' is highlighted with a red rectangle.

Passo 3

Coloque a senha do **github** para que o sonar possa recuperar os repositórios.



Confirm access

Password

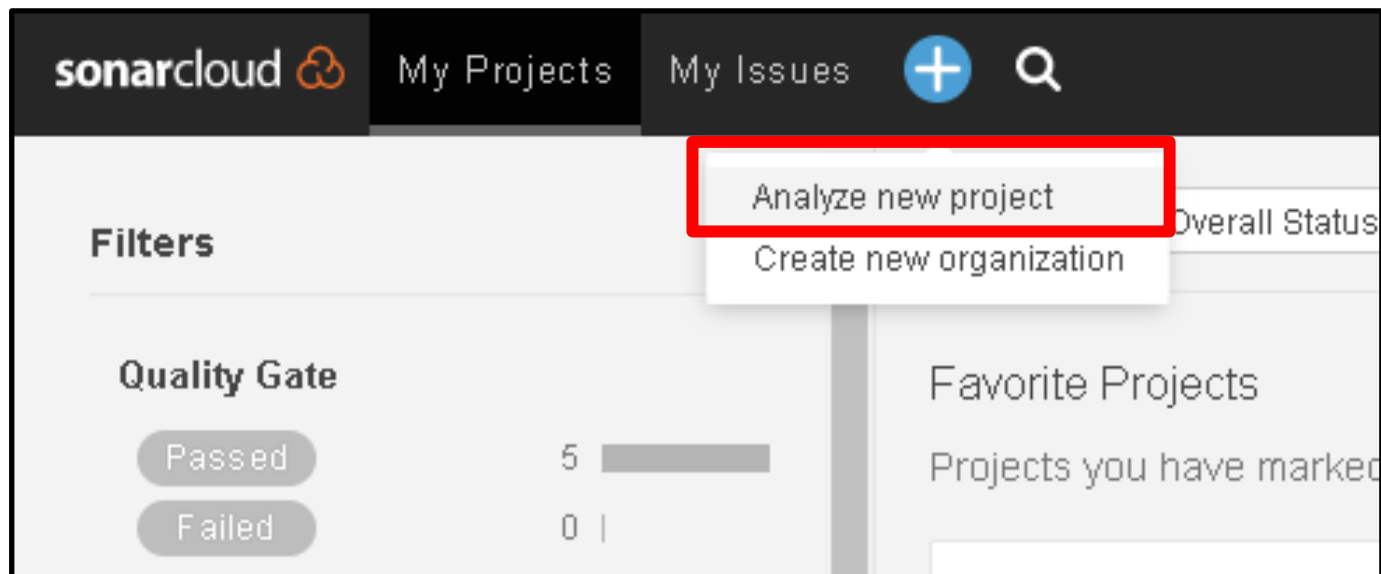
[Forgot password?](#)

[Confirm password](#)

Tip: You are entering [sudo mode](#). We won't ask for your password again for a few hours.

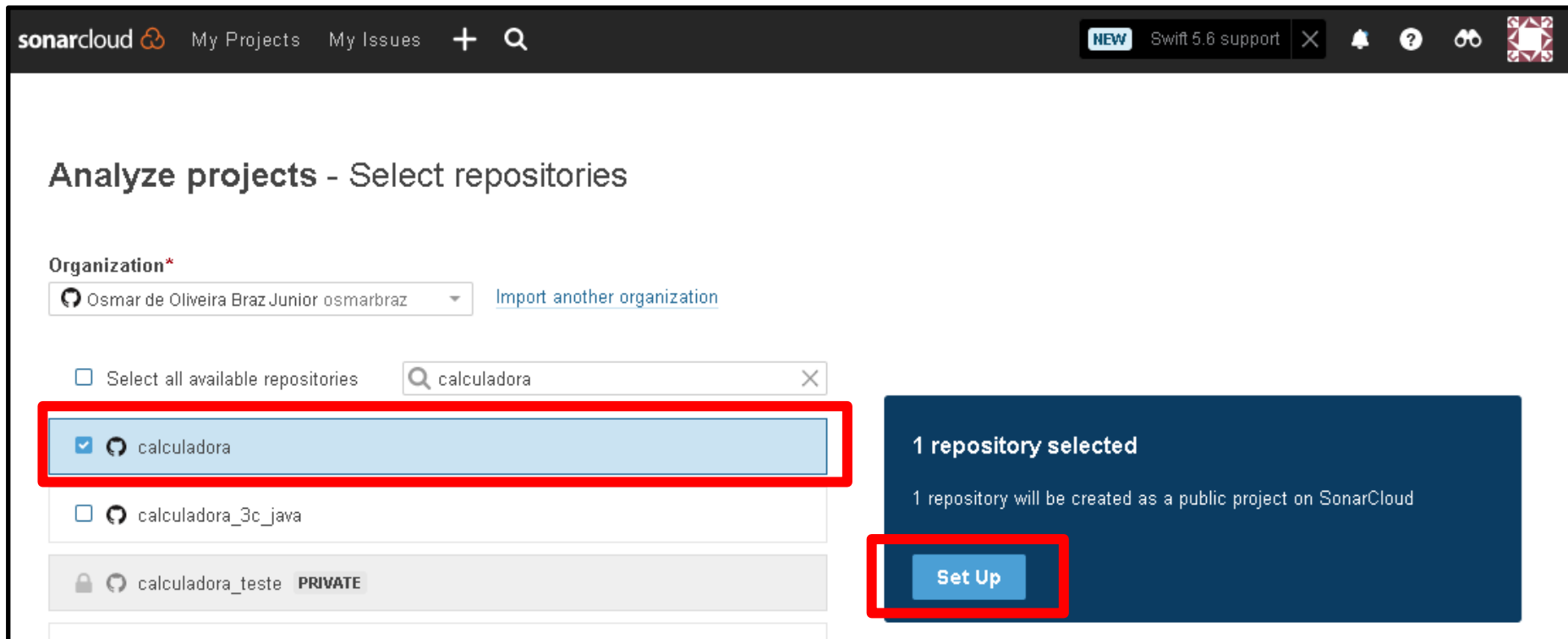
Passo 4


Crie um projeto no sonar.



Passo 5

Selecione o projeto no **github** a ser vinculado ao projeto no sonar.






sonarcloud  My Projects My Issues + Q

NEW Swift 5.6 support X ?

Analyze projects - Select repositories

Organization*
Osmar de Oliveira Braz Junior osmarbraz [Import another organization](#)

☐ Select all available repositories

- ☒  calculadora
- ☐  calculadora_3c_java
- ☐  calculadora_teste PRIVATE

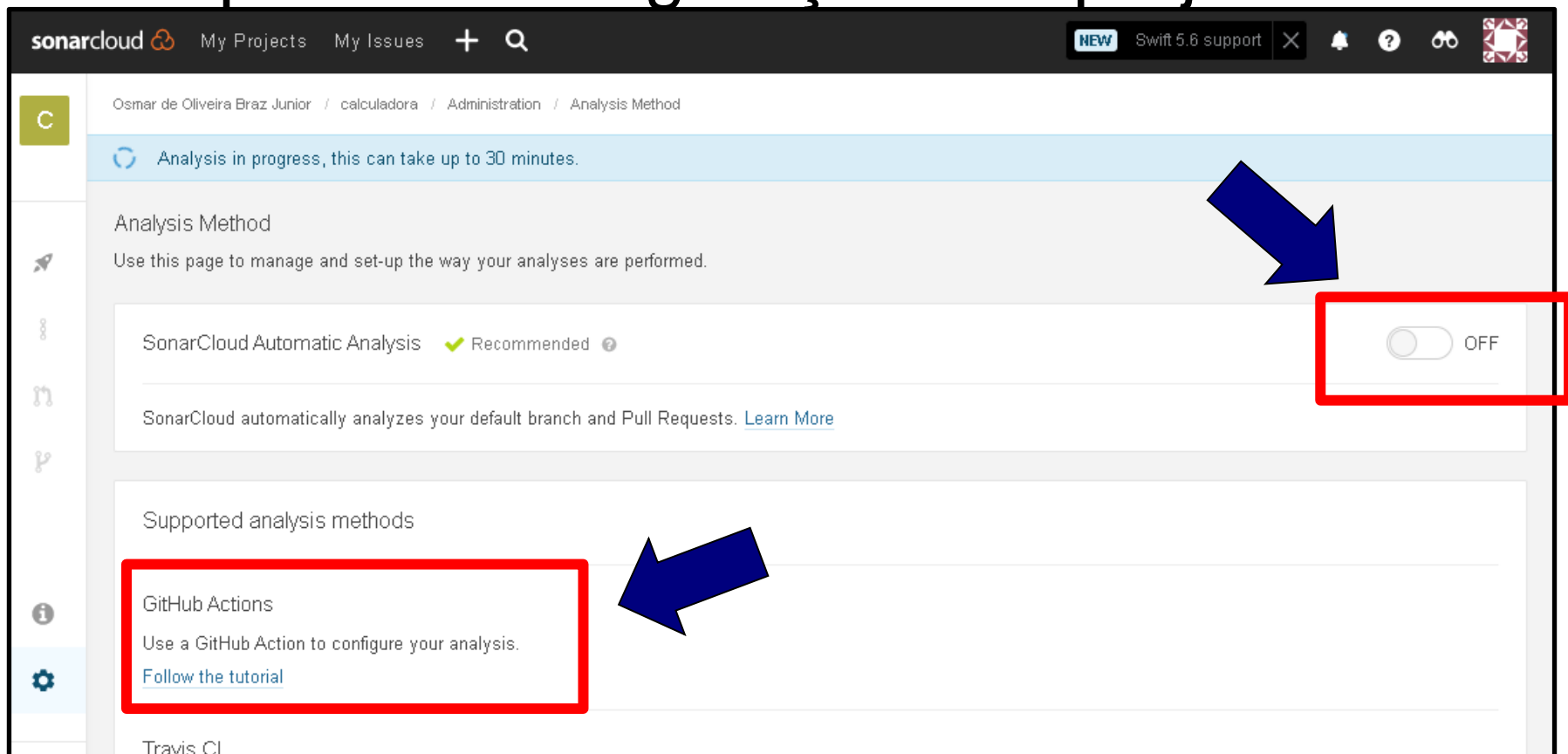
1 repository selected


1 repository will be created as a public project on SonarCloud

Set Up

Passo 6

Recupere as configurações do projeto.




sonarcloud  My Projects My Issues + Q NEW Swift 5.6 support X

Osmar de Oliveira Braz Junior / calculadora / Administration / Analysis Method

Analysis in progress, this can take up to 30 minutes.

Analysis Method

Use this page to manage and set-up the way your analyses are performed.

SonarCloud Automatic Analysis  Recommended ⓘ

SonarCloud automatically analyzes your default branch and Pull Requests. [Learn More](#)

Supported analysis methods

GitHub Actions

Use a GitHub Action to configure your analysis.

[Follow the tutorial](#)

Travis CI

Administration

→ Analysis Method


Passo 7



Copie o token e o valor secreto do sonar para o **github** e adicione em *secrets key*.

Analyze with a GitHub Action

1 Create a GitHub Secret

In your GitHub repository, go to [Settings > Secrets](#) and create a new secret with the following details:

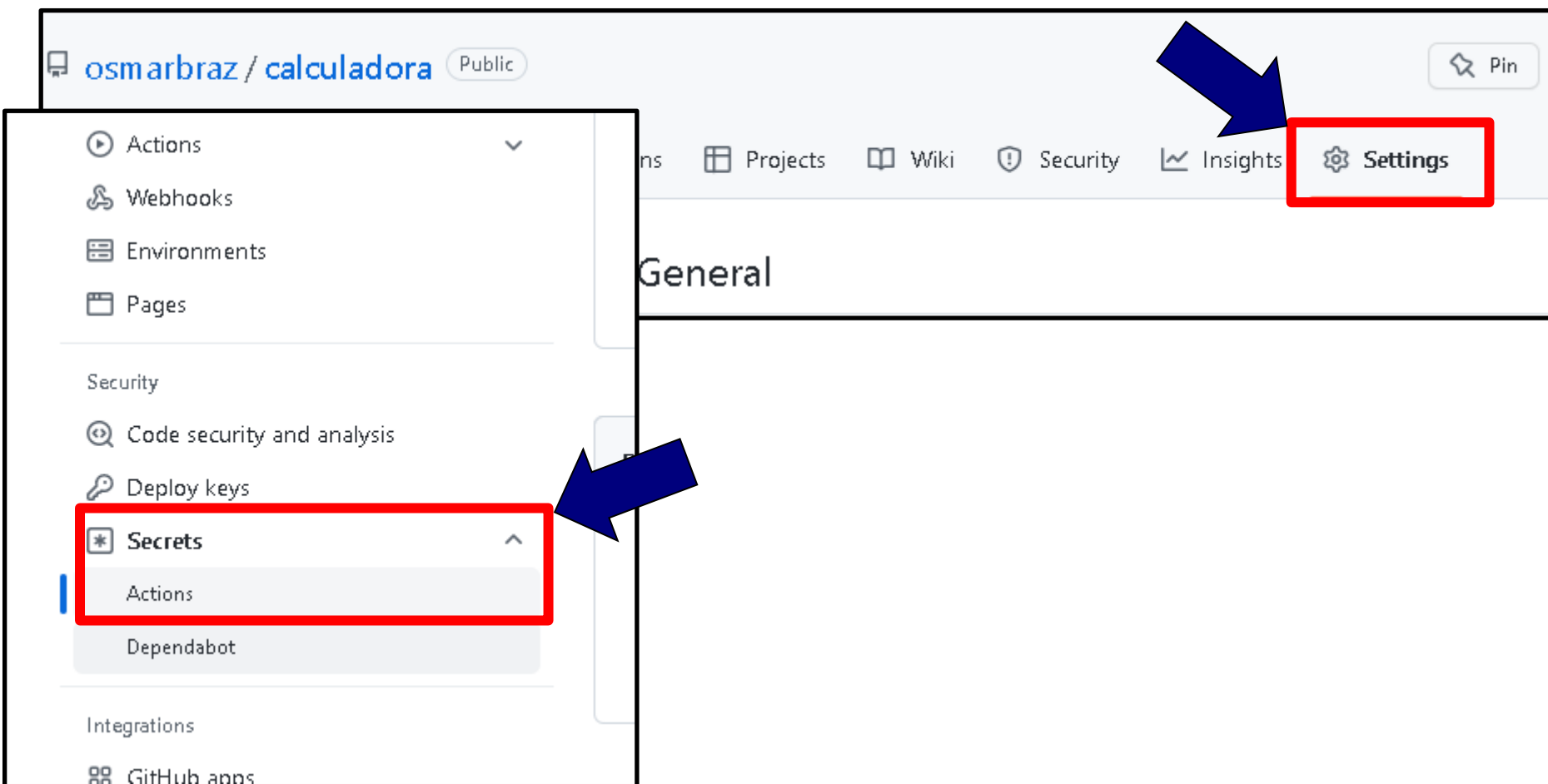
1 In the Name field, enter `SONAR_TOKEN` 

2 In the Value field, enter `574910f2909e43c96c98cc36468031fb8ab5306b`  

[Continue](#)

Passo 8

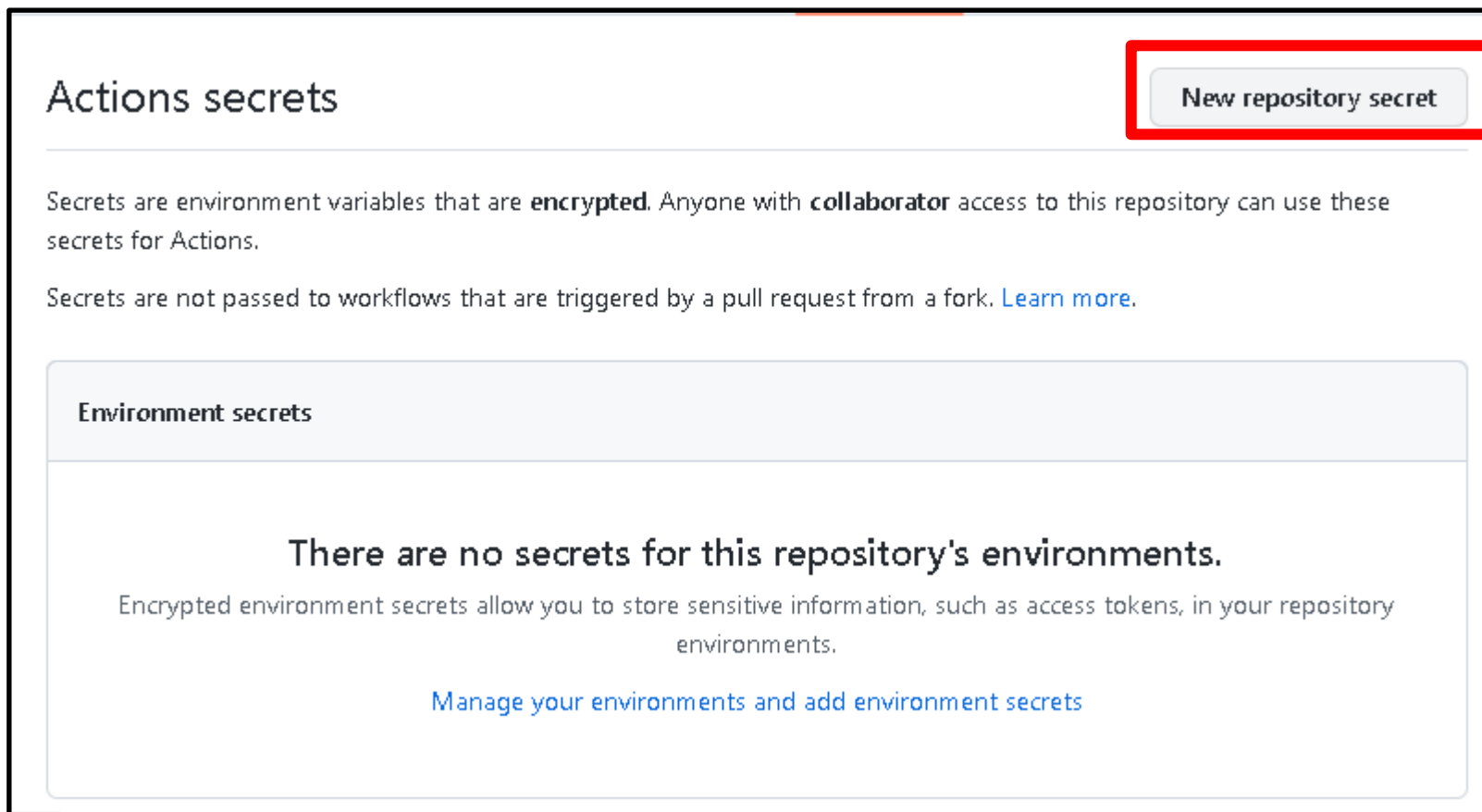
Insira o token e o valor no **github**.



The screenshot shows the GitHub repository settings page for the repository 'osmarbraz/calculadora'. The 'Settings' tab is selected and highlighted with a red box. A blue arrow points to the 'Settings' tab. The left sidebar shows the 'Secrets' option highlighted with a red box, with a blue arrow pointing to it. The 'Secrets' section is expanded, showing 'Actions' and 'Dependabot' sub-sections.

Passo 8 - Continuação

Cria uma novo token secreto no **github**.



Actions secrets

New repository secret

Secrets are environment variables that are **encrypted**. Anyone with **collaborator** access to this repository can use these secrets for Actions.

Secrets are not passed to workflows that are triggered by a pull request from a fork. [Learn more](#).

Environment secrets

There are no secrets for this repository's environments.

Encrypted environment secrets allow you to store sensitive information, such as access tokens, in your repository environments.

[Manage your environments and add environment secrets](#)

Passo 8 - Continuação

Insira o valor secreto do token do **Sonarcloud** no **github** na variável 'SONAR_TOKEN'.

Actions secrets / New secret

Name

SONAR_TOKEN

Value

574910f2909e43c96c98cc36468031fb8ab5306b|

Add secret

Passo 9

Visualizando as alterações do **pom.xml** e **maven.yml**

Analyze with a GitHub Action

- 1 Create a GitHub Secret

In your GitHub repository, go to [Settings > Secrets](#) and create a new secret with the following details:

- 1 In the Name field, enter `SONAR_TOKEN`
- 2 In the Value field, enter `574910F2909e43c96c98cc36468031fb8ab5306b`

Continue

2 Create or update a `.github/workflows/build.yml` file

What option best describes your build?

Maven | Gradle | C, C++ or ObjC | .NET | Other (for JS, TS, Go, Python, PHP, ...)

Passo 10

Dados do sonarcloud para o arquivo **pom.xml**

Update your `pom.xml` file with the following properties:

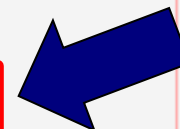
```
<properties>
  <sonar.organization>osmarbraz</sonar.organization>
  <sonar.host.url>https://sonarcloud.io</sonar.host.url>
</properties>
```

Passo 11

Alterando o arquivo pom.xml

```
        <artifactId>junit</artifactId>
        <version>4.13.2</version>
        <scope>test</scope>
    </dependency>
</dependencies>
<properties>
    <project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>
    <maven.compiler.source>1.8</maven.compiler.source>
    <maven.compiler.target>1.8</maven.compiler.target>
    <sonar.organization>osmarbraz</sonar.organization>
    <sonar.host.url>https://sonarcloud.io</sonar.host.url>
</properties>
<build>
    <plugins>
        <plugin>
            <groupId>org.apache.maven.plugins</groupId>
            <artifactId>maven-assembly-plugin</artifactId>
            <executions>
                <execution>
                    <phase>package</phase>
                </execution>
            </executions>
        </plugin>
    </plugins>
</build>
```

Já existia a tag ***properties***, portanto adicionamos as duas linhas logo em seguida.





Passo 12

Dados para o **maven.yml**.

Vamos modificar o arquivo existente inserindo as alterações.

Create or update your `.github/workflows/build.yml`

Here is a base configuration to run a SonarCloud analysis on your master branch and Pull Requests. If you already have some GitHub Actions, you might want to just add some of these new steps to an existing one.

```
name: Build
on:
  push:
    branches:
      - master
  pull_request:
    types: [opened, synchronize, reopened]
jobs:
  build:
    name: Build
    runs-on: ubuntu-latest
    steps:
      - uses: actions/checkout@v2
        with:
          fetch-depth: 0 # Shallow clones should be disabled for a better relevancy of analysis
      - name: Set up JDK 11
        uses: actions/setup-java@v1
        with:
          java-version: 11
      - name: Cache SonarCloud packages
        uses: actions/cache@v1
        with:
          path: ~/.sonar/cache
          key: ${{ runner.os }}-sonar
          restore-keys: ${{ runner.os }}-sonar
      - name: Cache Maven packages
        uses: actions/cache@v1
        with:
          path: ~/.m2
          key: ${{ runner.os }}-m2-${{ hashFiles('**/pom.xml') }}
          restore-keys: ${{ runner.os }}-m2
      - name: Build and analyze
        env:
          GITHUB_TOKEN: ${{ secrets.GITHUB_TOKEN }} # Needed to get PR information, if any
          SONAR_TOKEN: ${{ secrets.SONAR_TOKEN }}
        run: mvn -B verify org.sonarsource.scanner.maven:sonar-maven-plugin:sonar -Dsonar.projectKey=smartrexx_calculadora
```



Passo 12 - Continuação

```
...  
- name: Testa o projeto com Maven  
  run: mvn -B test --file pom.xml
```

Steps de homologação!

```
# Executa os passos da análise do código com o sonarcube  
- name: Cache dos pacotes do SonarCloud  
  uses: actions/cache@v3  
  with:  
    path: ~/.sonar/cache  
    key: ${{ runner.os }}-sonar  
    restore-keys: ${{ runner.os }}-sonar  
- name: Cache dos pacotes do Maven  
  uses: actions/cache@v3  
  with:  
    path: ~/.m2  
    key: ${{ runner.os }}-m2-${{ hashFiles('**/pom.xml') }}  
    restore-keys: ${{ runner.os }}-m2  
- name: Verifica o projeto com Maven e Jacoco  
  env:  
    GITHUB_TOKEN: ${{ secrets.GITHUB_TOKEN }}  
    SONAR_TOKEN: ${{ secrets.SONAR_TOKEN }}  
  run: mvn -B verify org.sonarsource.scanner.maven:sonar-maven-plugin:sonar  
      -Dsonar.projectKey=osmarbraz_calculadora
```

Novas linhas!

**Colocar em uma
única linha!**

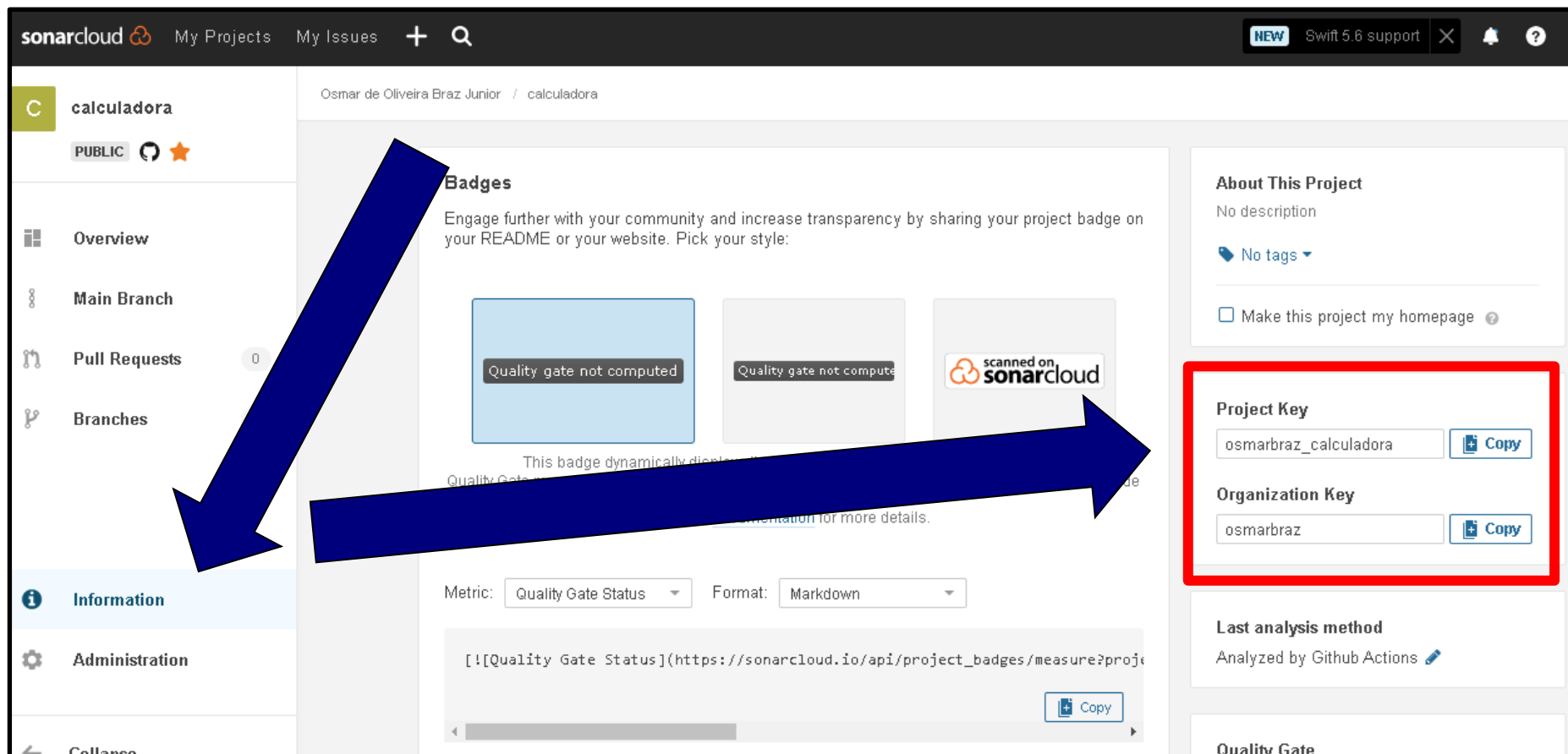
```
# Jobs do ambiente de produção  
build-prd:  
....
```

Steps de produção!

maven.yml

Passo 13

Conferindo as chaves do projeto (*Project key*).



The screenshot shows the SonarCloud interface for a project named 'calculadora'. The left sidebar contains navigation links: Overview, Main Branch, Pull Requests, Branches, Information, and Administration. The main content area displays 'Badges' with three options: 'Quality gate not computed' (blue), 'Quality gate not computed' (grey), and 'scanned on sonarcloud' (orange). Below the badges, there are dropdown menus for 'Metric: Quality Gate Status' and 'Format: Markdown'. The right sidebar, titled 'About This Project', contains the 'Project Key' (osmarbraz_calculadora) and 'Organization Key' (osmarbraz), both with 'Copy' buttons. A red rectangle highlights these keys. Two large blue arrows point from the 'Badges' section to the 'Project Key' and 'Organization Key' fields.

sonarcloud My Projects My Issues + Q NEW Swift 5.6 support X ?

calculadora
PUBLIC

Overview
Main Branch
Pull Requests
Branches
Information
Administration

Badges

Engage further with your community and increase transparency by sharing your project badge on your README or your website. Pick your style:

Quality gate not computed Quality gate not computed scanned on sonarcloud

This badge dynamically displays the project's quality gate status. For more details, see the documentation.

Metric: Quality Gate Status Format: Markdown

[[Quality Gate Status](https://sonarcloud.io/api/project_badges/measure?project=calculadora)]

Copy

About This Project

No description

No tags

Make this project my homepage

Project Key

osmarbraz_calculadora Copy

Organization Key

osmarbraz Copy

Last analysis method

Analyzed by Github Actions

Quality Gate

Passo 13

Tudo pronto!



E você está feito!

Você deve ver a página se atualizar em alguns momentos com os resultados da análise se tudo for executado com sucesso.

- ✓ Você obterá uma primeira análise do seu branch padrão
- ✓ Cada novo push que você fizer em um branch ou Pull Request acionará automaticamente uma nova análise
- ✓ Cada novo Pull Request que você criar será analisado automaticamente

Verifique estes links úteis enquanto espera: [O que é um Quality Gate?](#) , [Configure seus perfis de qualidade](#)

Passo 14

Submeta ao repositório **github** as alterações dos arquivos **pom.xml** e **maven.yml**

Passo 15

Verifique se o *workflow* da integração continua foi executada com sucesso.

osmarbraz / calculadora Public

Pin

Unwatch 1

Fork 0

Star 0

<> Code Issues Pull requests **Actions** Projects Wiki Security Insights Settings

✓ Início do projeto Integração continua de Java com Maven #6

Re-run all jobs

Summary

Jobs

- ✓ build-dev
- ✓ build-hmg
- ✓ build-prd

Triggered via push 7 minutes ago

osmarbraz pushed · d496e4e master

Status

Success

Total duration

1m 43s

Artifacts

1

maven.yml
on: push

✓ build-dev

14s

✓ build-hmg

42s

✓ build-prd

18s

Passo 16

Acessando a análise do código do repositório no sonarcloud a primeira vez.

Passo 16

Acessando a análise do código do repositório no sonarcloud a primeira vez.

New Code

Setting up a new code definition helps to focus attention on the most relevant changes to your project, enabling you to use the Clean as You Code methodology effectively. Learn more in the [documentation](#).

The New Code for this project will be based on:

☒ Previous version

All code that has changed since the previous version bump is considered new code

☐ Specific version


All code that has changed since the specified version bump is considered new code

☐ Number of days

All code that has changed in the last x days is considered new code

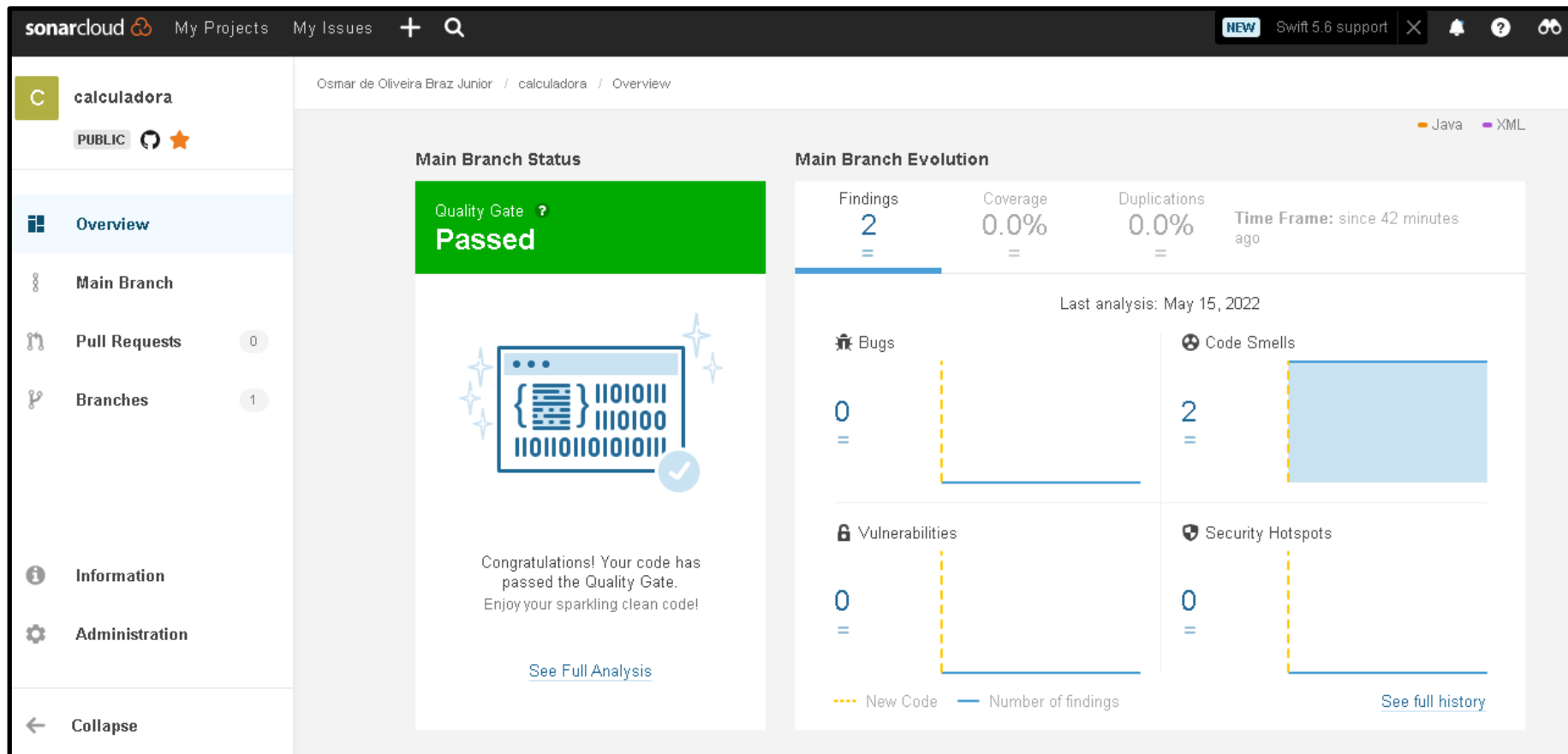
☐ Specific date

All code that has changed since the specified date is considered new code

 Changes will take effect after the next analysis

Passo 16

Acessando a análise do código do repositório no sonarcloud.



The screenshot displays the SonarCloud web interface for a project named 'calculadora'. The left sidebar shows the project overview with tabs for Overview, Main Branch, Pull Requests (0), and Branches (1). The main content area is divided into two sections: 'Main Branch Status' and 'Main Branch Evolution'.

Main Branch Status: A green box indicates 'Quality Gate Passed'. Below this, a graphic shows a code editor with binary code and a checkmark. The text reads: 'Congratulations! Your code has passed the Quality Gate. Enjoy your sparkling clean code!'. A link 'See Full Analysis' is provided.

Main Branch Evolution: This section shows metrics for the last analysis (May 15, 2022). The metrics are:

- Findings: 2
- Coverage: 0.0%
- Duplications: 0.0%
- Time Frame: since 42 minutes ago

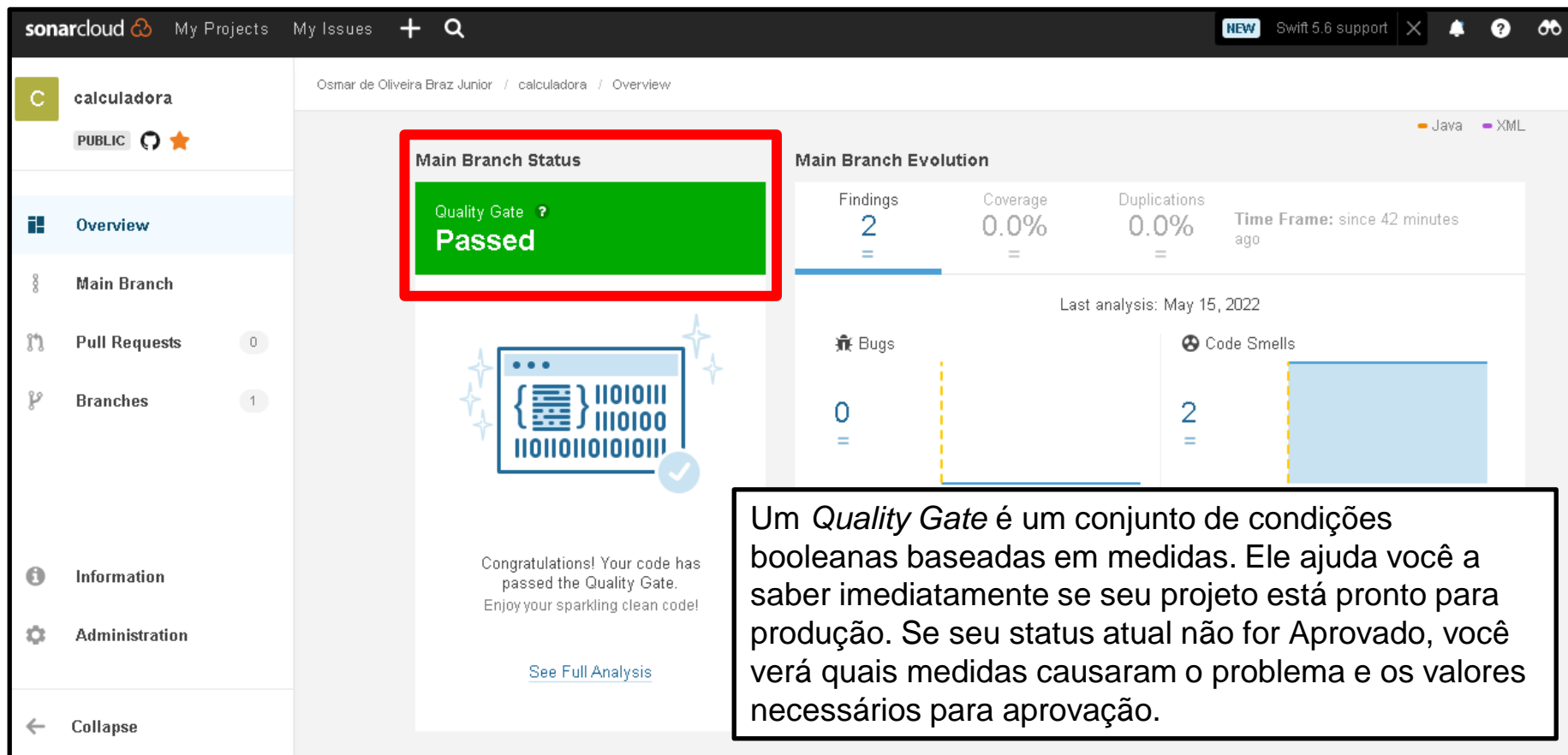
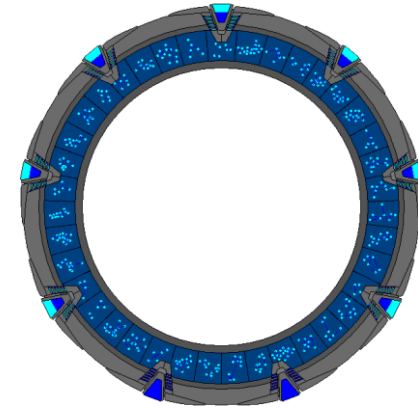
Below these metrics, four charts are displayed:

- Bugs:** 0 findings.
- Code Smells:** 2 findings.
- Vulnerabilities:** 0 findings.
- Security Hotspots:** 0 findings.

A legend at the bottom indicates that the orange dashed line represents 'New Code' and the blue solid line represents 'Number of findings'.

Passo 16

Portão da Qualidade (*Quality Gate*).



sonarcloud My Projects My Issues + Q NEW Swift 5.6 support

calculadora PUBLIC

Overview

Main Branch

Pull Requests 0

Branches 1

Information

Administration

Collapse

Osmar de Oliveira Braz Junior / calculadora / Overview

Main Branch Status

Quality Gate ?
Passed

Main Branch Evolution

Findings 2
Coverage 0.0%
Duplications 0.0%

Time Frame: since 42 minutes ago

Last analysis: May 15, 2022

Bugs 0

Code Smells 2

Congratulations! Your code has passed the Quality Gate.
Enjoy your sparkling clean code!

[See Full Analysis](#)

Um *Quality Gate* é um conjunto de condições booleanas baseadas em medidas. Ele ajuda você a saber imediatamente se seu projeto está pronto para produção. Se seu status atual não for Aprovado, você verá quais medidas causaram o problema e os valores necessários para aprovação.

Passo 17

Troque o nome do usuário e o nome do seu projeto no github.

Trocar o nome do workflow no arquivo maven.yml para: "Integração contínua de Java com Maven"

Criando o arquivo **README.md** na raiz do projeto.

```
[[Github Actions Status for
osmarbraz/calculadora](https://github.com/osmarbraz/calculadora/workflows/Integra%C3%A7%C3%A3o%20continua%20de%20Java%20com%20Maven/badge.svg)](https://github.com/osmarbraz/calculadora/actions)
[[Quality Gate
Status](https://sonarcloud.io/api/project_badges/measure?project=osmarbraz_calculadora&metric=alert_status)](https://sonarcloud.io/summary/new_code?id=osmarbraz_calculadora)
[[Coverage](https://sonarcloud.io/api/project_badges/measure?project=osmarbraz_calculadora&metric=coverage)](https://sonarcloud.io/component_measures?id=osmarbraz_calculadora&metric=coverage)]
```

Calculadora com CI.
Utiliza 3 ambientes:
- dev - Desenvolvimento
- hmg - Homologação
- prd - Produção

Pipeline
- dev - Compilação
- hmg - Compilação, Testes, Análise Código, Cobertura Código
- prd - Empacotamento

- Utiliza o Apache Maven para a automatização da construção.

- A pasta test contém os testes unitários do projeto utilizando JUnit 4.

- A cobertura do código é realizada através do JaCoCo.

Troque o nome do usuário e o nome do seu projeto no sonarcloud.

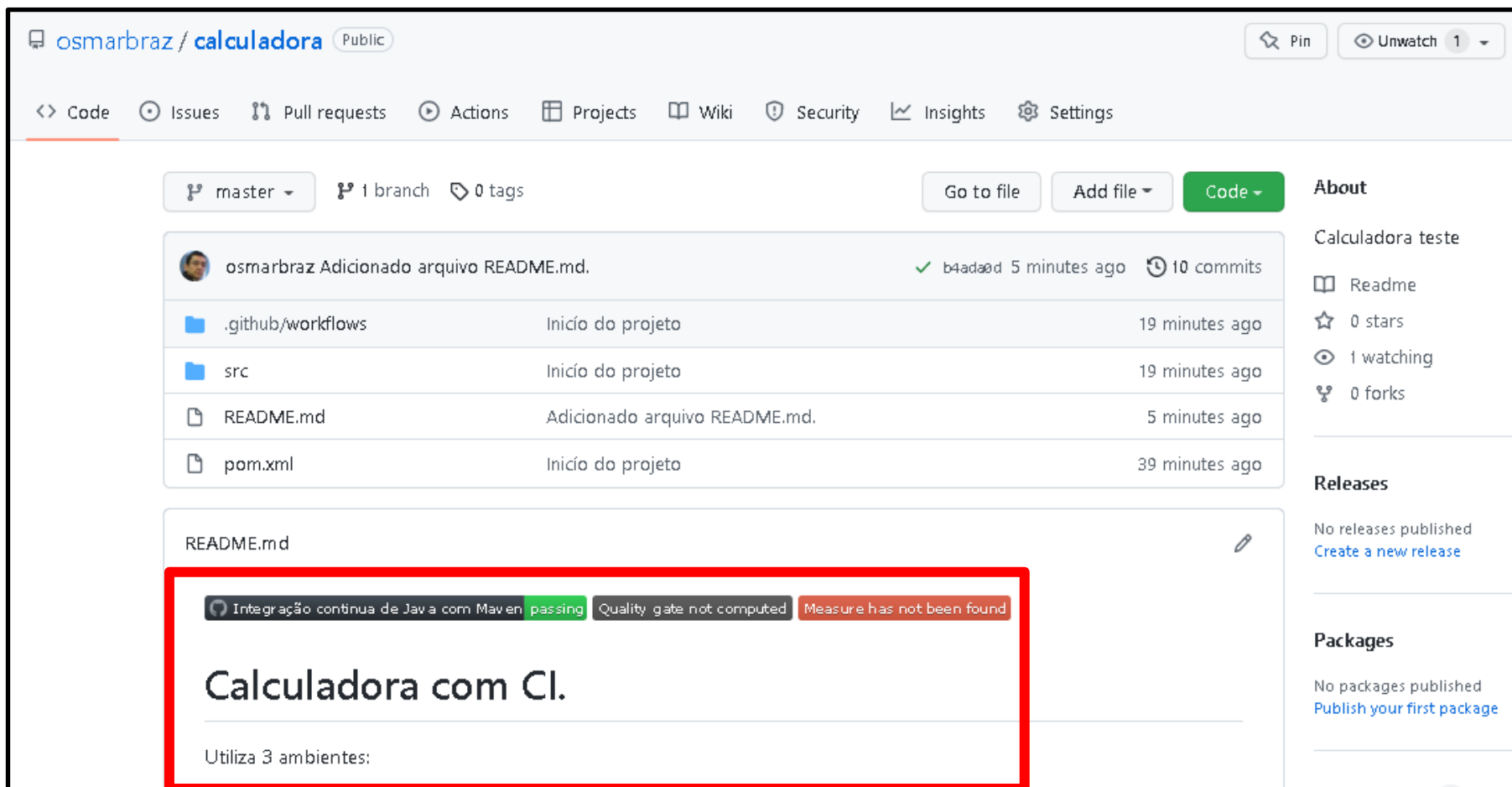
Passo 18

Submeta ao repositório **github** o arquivo **README.md**.

Outros badges podem ser criados em **<https://shields.io/>**

Passo 19

Acesse o repositório no **github**.



The screenshot shows the GitHub interface for the repository 'osmarbraz/calculadora'. The repository is public and has 1 branch (master) and 0 tags. The commit history shows a recent commit by 'osmarbraz' adding a README.md file. The file list includes .github/workflows, src, README.md, and pom.xml. The README.md file is selected, showing its content: 'Calculadora com CI.' and 'Utiliza 3 ambientes:'. The CI status bar at the top of the README content shows 'Integração contínua de Java com Maven' as 'passing', 'Quality gate not computed', and 'Measure has not been found'.

osmarbraz / **calculadora** Public

Pin Unwatch 1

<> Code Issues Pull requests Actions Projects Wiki Security Insights Settings

master 1 branch 0 tags Go to file Add file Code

osmarbraz Adicionado arquivo README.md. ✓ b4adad 5 minutes ago 10 commits

File	Commit Message	Time
.github/workflows	Início do projeto	19 minutes ago
src	Início do projeto	19 minutes ago
README.md	Adicionado arquivo README.md.	5 minutes ago
pom.xml	Início do projeto	39 minutes ago

README.md

Integração contínua de Java com Maven passing Quality gate not computed Measure has not been found

Calculadora com CI.

Utiliza 3 ambientes:

About
Calculadora teste
Readme
0 stars
1 watching
0 forks

Releases
No releases published
[Create a new release](#)

Packages
No packages published
[Publish your first package](#)

Conclusão

- A Integração Continua é um processo essencial a qualquer software que deseja manter vivo por um período de tempo mais longo.
- Conhecer e dominar as ferramentas é um ponto crítico para garantir agilidade no processo de distribuição do software.
- Nesta etapa realizamos a análise do código com o Sonar Cloud.

Referências

- PRESSMAN, Roger; MAXIM, Bruce. Engenharia de software: uma abordagem profissional. 8.ed. Bookman, 2016.
- SOMMERVILLE, Ian. Engenharia de software. 9. ed. São Paulo: Pearson Prentice Hall, 2011.
- LARMAN, Craig. Utilizando UML e padrões: uma introdução à análise e ao projeto orientados a objetos e desenvolvimento iterativo. 3. ed Porto Alegre: Bookman, 2007



Fim