BITS Pilani
Pilani | Dubai | Goa | Hyderabad

Introduction to DevOps

Sonika Rathi

Assistant Professor
BITS Pilani

Agenda

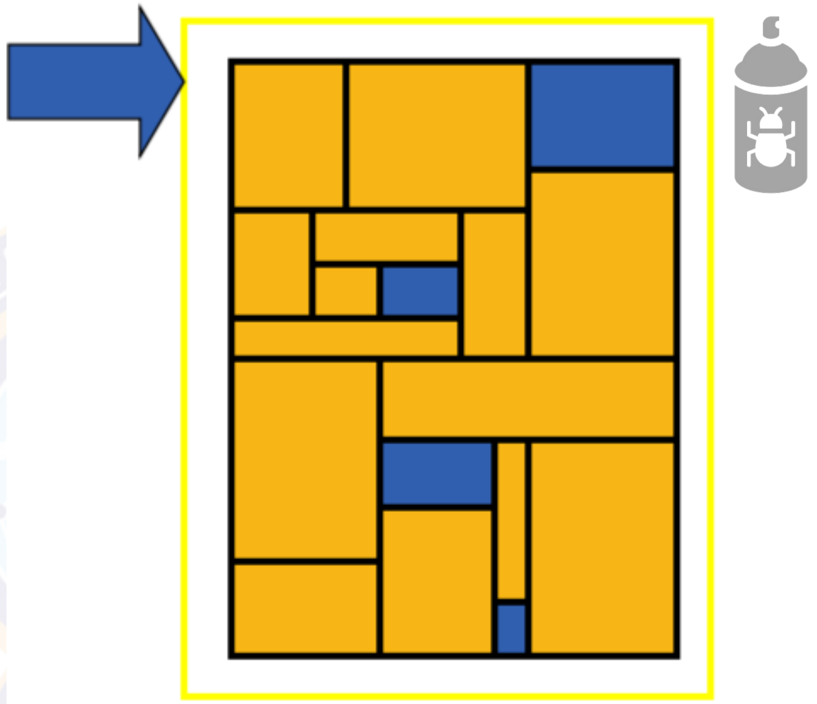
Automating Build Process

- Unit testing
- Automates Test Suite - Selenium
- Continuous Code Inspection
- Code Inspection Tools
    - Sonarqube

Unit Testing

Traditional Testing

- Test the system as a whole
- Errors go undetected
- Isolation of errors difficult to track down



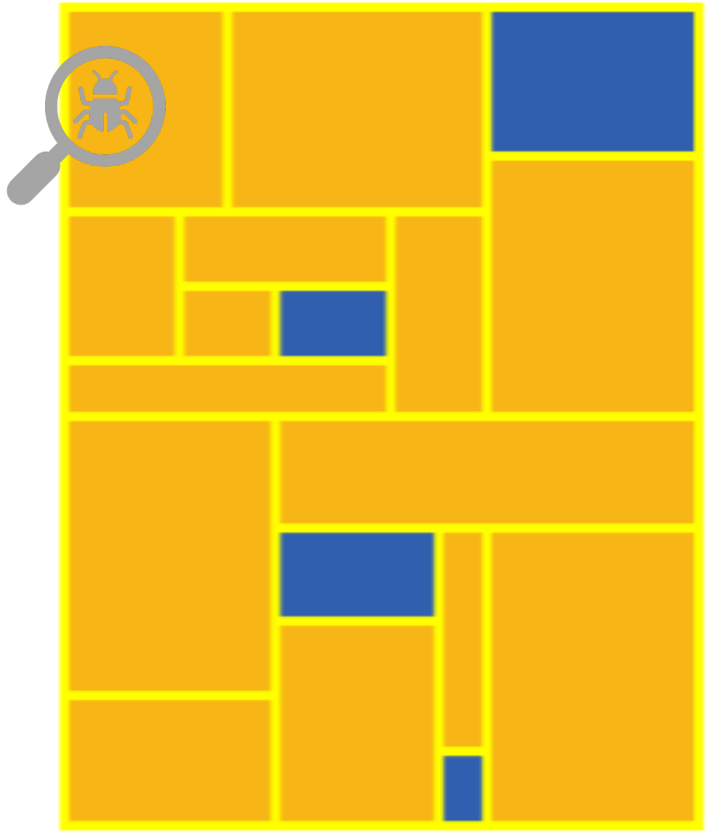
Traditional Testing Strategies

- Print Statements
- Use of Debugger
- Debugger Expressions
- Test Scripts

Unit Testing

What is Unit Testing

- Is a level of the software testing process where individual units/components of a software/system are tested
- Each part tested individually
- All components tested at least once
- Errors picked up earlier
- Scope is smaller, easier to fix errors
- Typically written and run by software developers
- Its goal is to isolate each part of the program and show that the individual parts are correct



Unit Testing

Why Unit Testing

Concerned with

- Functional correctness and completeness
- Error handling
- Checking input values (parameter)
- Correctness of output data (return values)
- Optimizing algorithm and performance

- Faster Debugging
- Faster Development
- Better Design
- Excellent Regression Tool
- Reduce Future Cost

Unit Testing

Benefits

- Unit testing allows the programmer to refactor code earlier and make sure the module works correctly
- By testing the parts of a program first and then testing the sum of its parts, i.e. integration testing becomes much easier
- Unit testing provides a sort of living documentation of the system

Unit Testing

Guidelines

- Keep unit tests small and fast
- Unit tests should be fully automated and non-interactive
- Make unit tests simple to run
- Measure the tests
- Fix failing tests immediately
- Keep testing at unit level
- Keep tests independent
- Name tests properly
- Prioritize testing

Test Automation

Selenium

- Preform an sort of interaction
- Selenium helps to automate web browser interaction
- Scripts perform the interactions

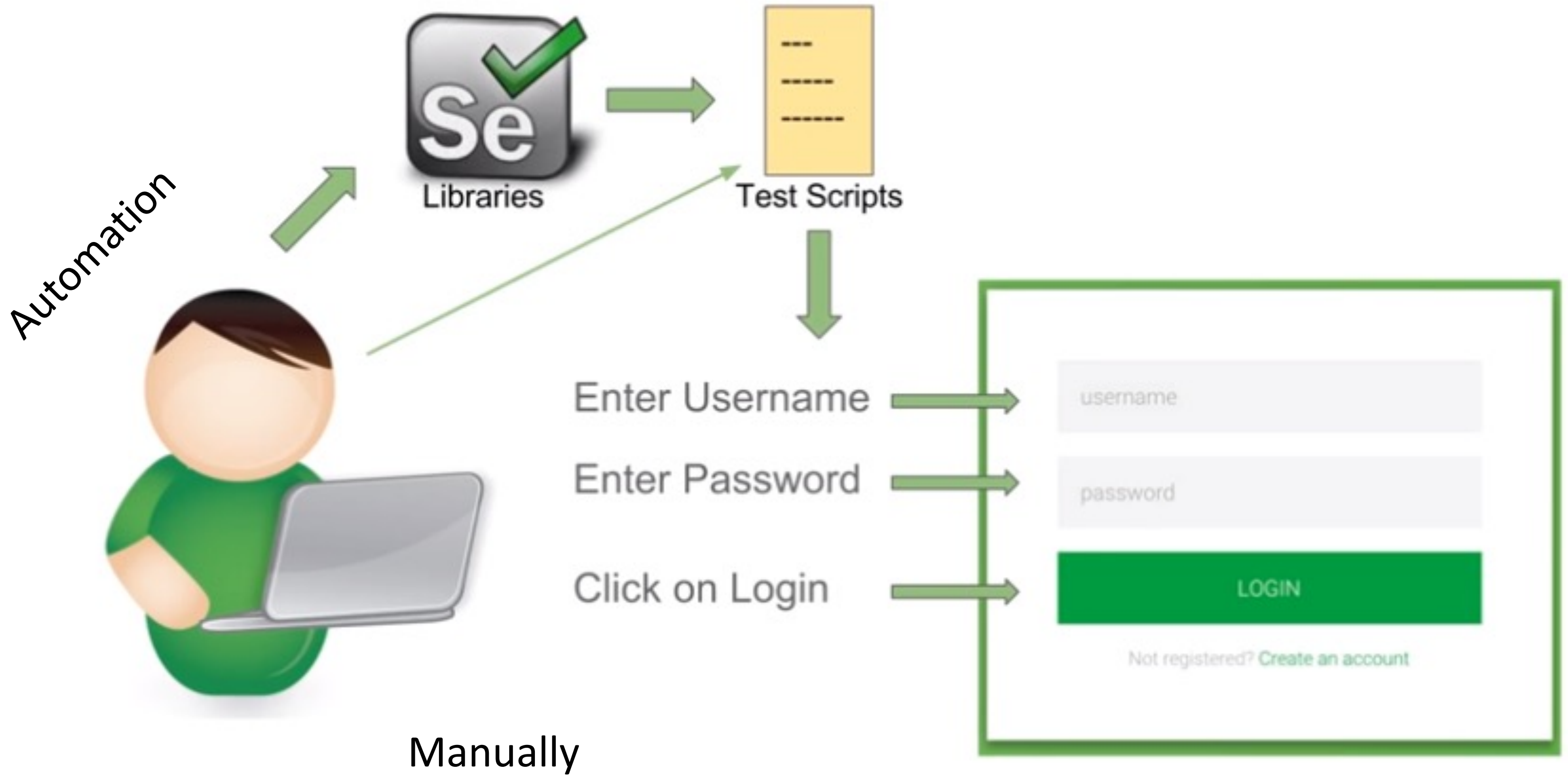
Selenium

Benefits

- Frequent regression testing
- Rapid feedback to developers
- Virtually unlimited iterations of test case execution
- Support for Agile and extreme development methodologies
- Disciplined documentation of test cases
- Customized defect reporting
- Finding defects missed by manual testing
- Reduced Business Expenses
- Reusability of Automated Tests
- Faster Time-to-Market

Selenium

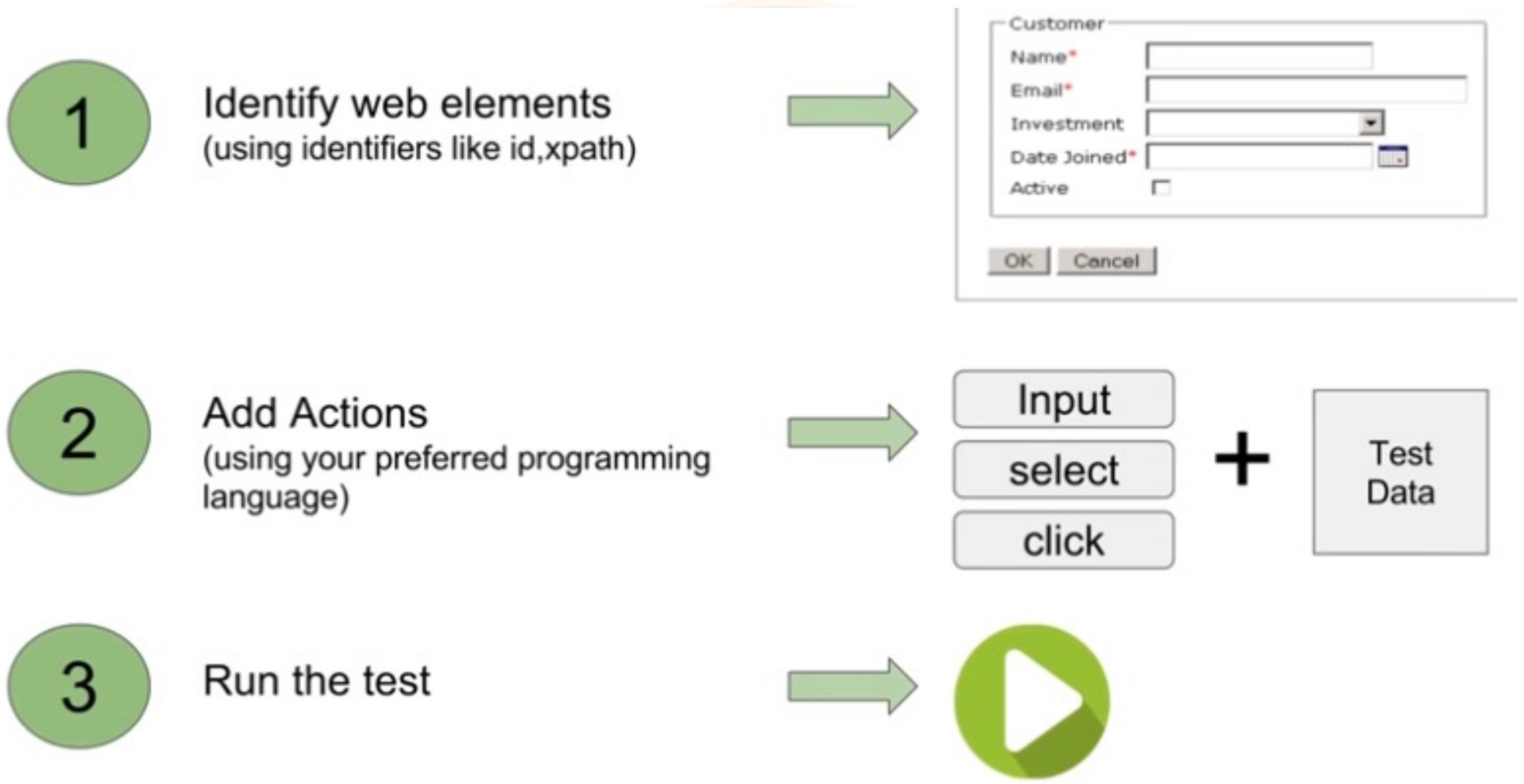
Lets say you want to test one login page



Selenium

Example

- At a high level you will be doing three things with Selenium



Selenium

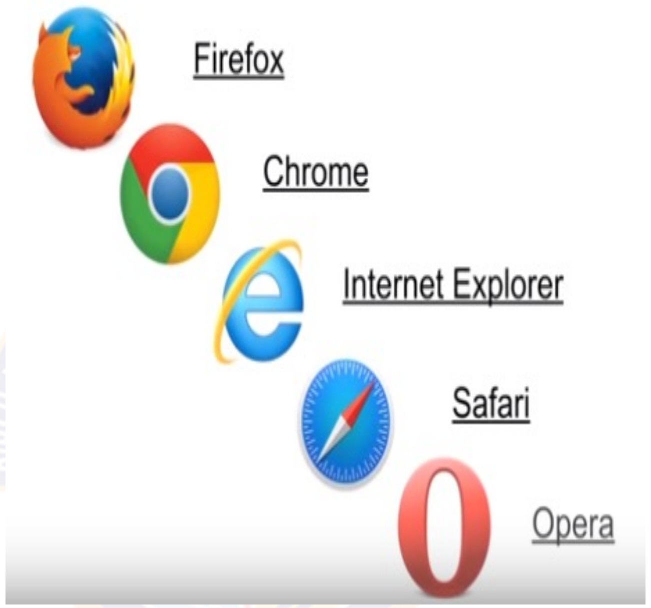
Components

- Selenium IDE
  - A Record and playback plugin for Firefox add-on
  - Prototype testing
- Selenium RC (Remote Control)
  - Also known as selenium 1
  - Used to execute scripts (written in any language) using Javascript
  - Now Selenium 1 is deprecated and is not actively supported
- WebDriver
  - Most actively used component today
  - An API used to interact directly with the web browser
  - Is a successor to Selenium 1 / Selenium RC
  - Selenium RC and WebDriver are merged to form Selenium 2
- Selenium Grid
  - A tool to run tests in parallel across different machines and different browser simultaneously
  - Used to minimize the execution time

Selenium

Supports

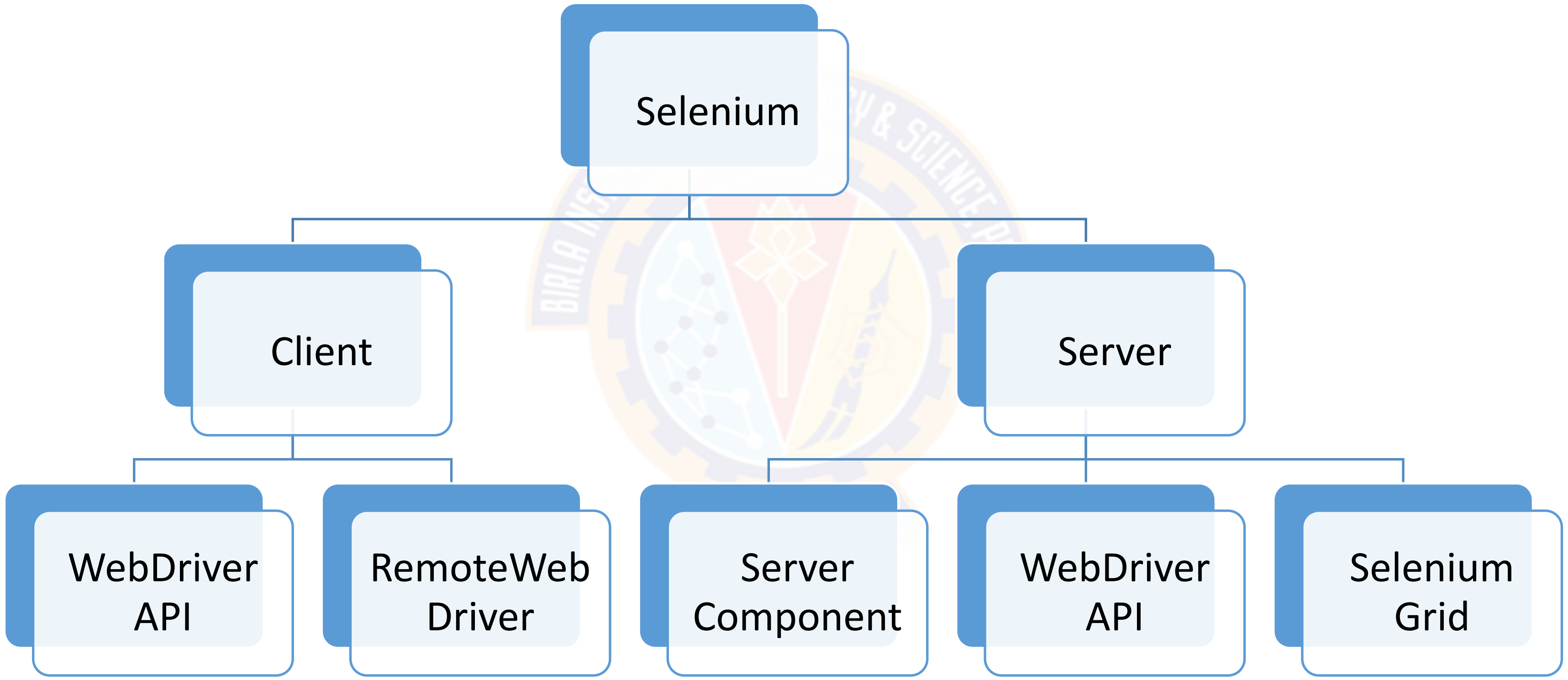
- Browsers
- OS
- Language



Windows
Mac OS
Linux
ORACLE SOLARIS
Solaris
Java
Perl
php
C#
python
Ruby
Se

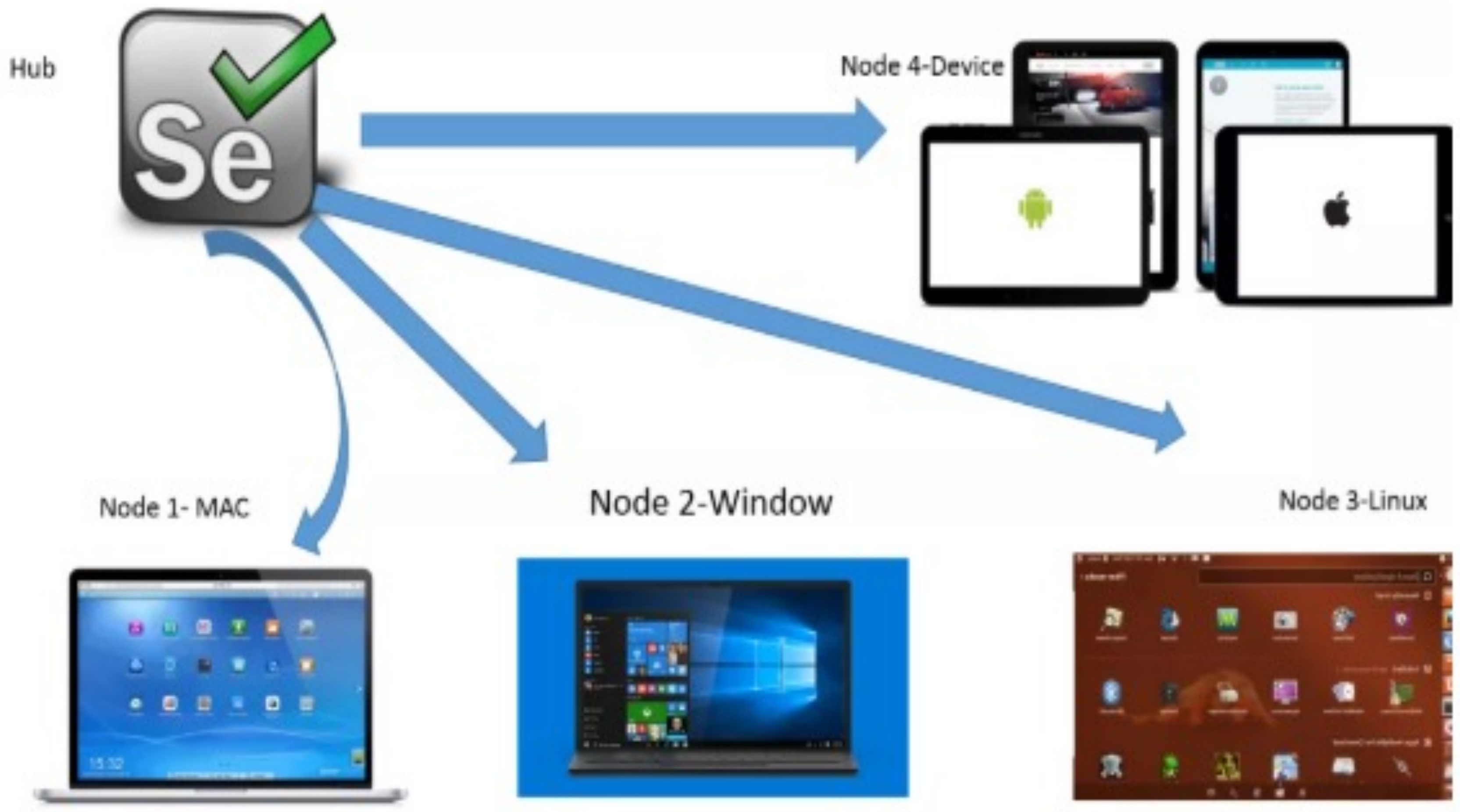
Selenium

Architecture



Selenium

Selenium Grid

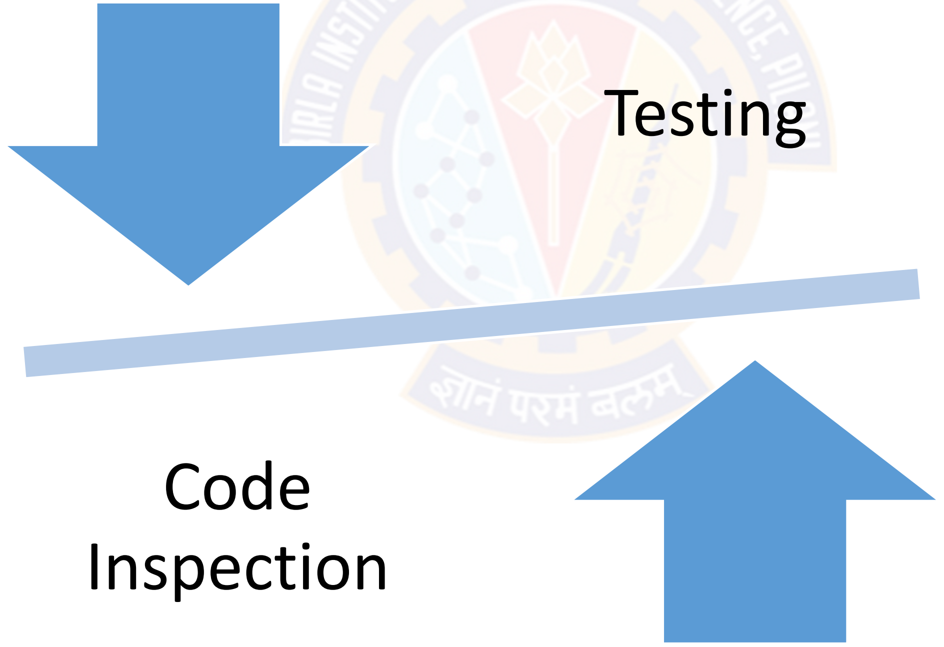


Continuous Code Inspection

Continuous code inspection = Constantly scanning code

- Identify if any defects
- It is a process of code review
- Its been proved 90% of defects can be addressed using code inspections tools

- **Find defects at the code level**
- **Minimizing defects during code inspections,** -> **testing efforts more efficient**



- **Verifies functionality** and **improves software quality**
- **Expensive** if you have to go through it over and over again

Note: Even with automated testing, it takes time to verify functionality;
by resolving defects at the code level, you'll be able to test functionality faster

Continuous Code Inspection

Code Inspection Measures

- Code inspections must be well-defined as per requirements:
  - Functional requirements : User Needs : Cosmetic
  - Structural requirements : System Needs : Re-engineering
- Run Time Defects:
  - Identify run time errors before program run
  - Examples: Initialization (using the value of unset data), Arithmetic Operations (operations on signed data resulting in overflow) & Array and pointers (array out of bounds, dereferencing NULL pointers), etc.,
- Preventative Practices:
  - This help you avoid error-prone or confusing code
  - Example: Declarations (function default arguments, access protection), Code Structure (analysis of switch statements) & Safe Typing (warnings on type casting, assignments, operations), etc.,
- Style:
  - In-house coding standards are often just style, layout, or naming rules and guidelines
  - Instead using a proven coding standard is better for improving quality

Continuous Code Inspection

Improve Your Code Inspection Process:

- Involve Stakeholders
  - Developer, Management & Customer
- Collaborate
  - Collaboration — both in coding and in code inspections
- Recognize Exceptions
  - Sometimes there are exceptions to the rule
  - In an ideal world, code is 100% compliant to every rule in a coding standard
  - The reality is different
- Document Traceability
  - Traceability is important for audits
  - Capture the history of software quality

- **What to Look For in Code Inspection Tools**
  - Automated inspection
  - Collaboration system

Continuous Code Inspection Tool

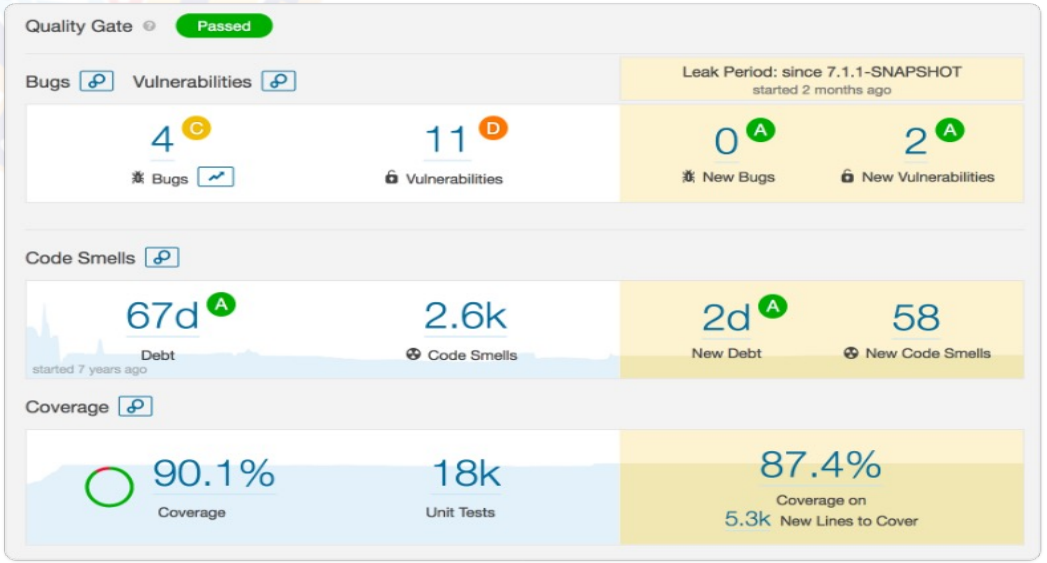
SonarQube

SonarQube

capability to show health of an application

Highlight issues newly introduced

Quality Gate, you can fix the leak and therefore improve code quality systematically


Quality Gate Passed

Bugs Vulnerabilities

Leak Period: since 7.1.1-SNAPSHOT
started 2 months ago

4 C	11 D	0 A	2 A
Bugs	Vulnerabilities	New Bugs	New Vulnerabilities

Code Smells

67d A	2.6k	2d A	58
Debt	Code Smells	New Debt	New Code Smells

started 7 years ago

Coverage

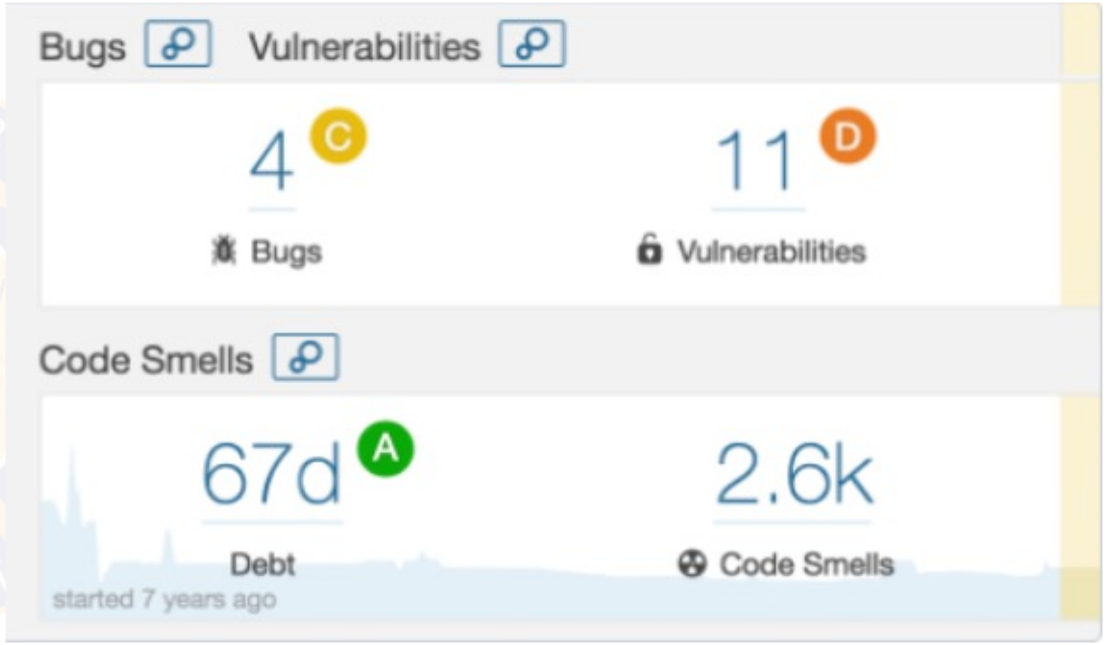
90.1%	18k	87.4%
Coverage	Unit Tests	Coverage on 5.3k New Lines to Cover

sonarqube

SonarQube

Overall health

- Bug:
  - An issue that represents something wrong in the code
  - If this has not broken yet, it will, and probably at the worst possible moment
- Code Smell:
  - A maintainability-related issue in the code
  - Examples: Dead Code, Duplicate code, Comments, Long method, Long parameter list, Long class etc.,
- Vulnerability:
  - A security-related issue which represents a backdoor for attackers



SonarQube

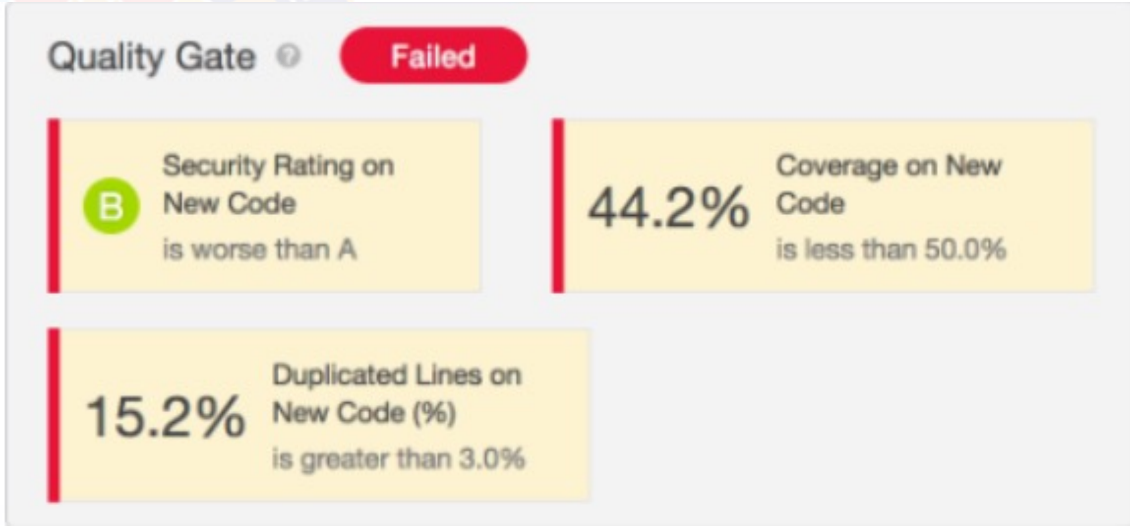
Enforce Quality Gate

- To fully enforce a code quality practice across all teams, need a Quality Gate
- A set of requirements that tells whether or not a new version of a project can go into production
- SonarQube’s default Quality Gate checks what happened on the Leak period and fails if your new code got worse in this period

A quality gate is the best way to enforce a quality policy in your organization

Define a set of Boolean conditions based on measure thresholds against which projects are measured

It supports multiple quality gate definitions



Quality Gate Failed

B Security Rating on New Code is worse than A

44.2% Coverage on New Code is less than 50.0%

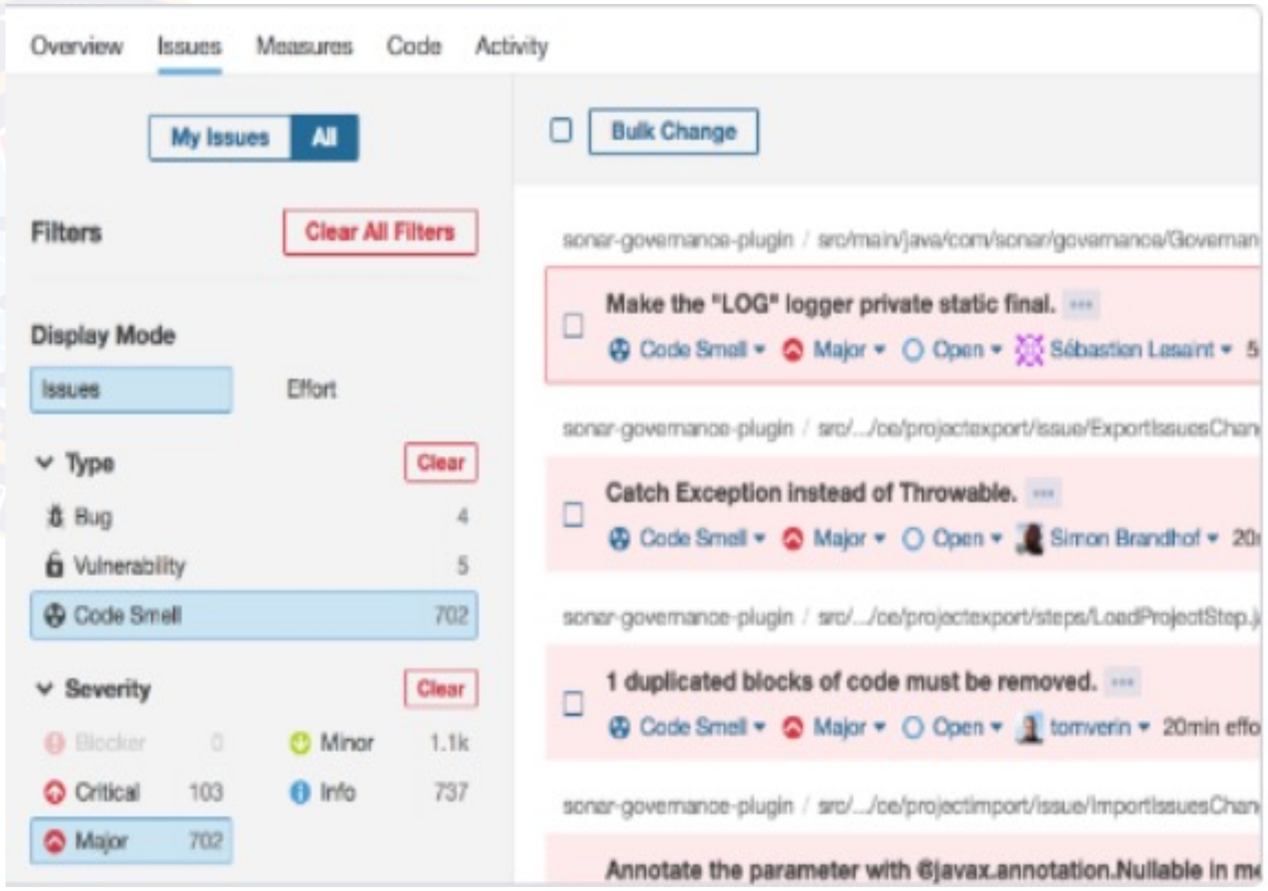
15.2% Duplicated Lines on New Code (%) is greater than 3.0%

Example: Failed Project

SonarQube

Dig into issues

- The “Issues” page of your project gives you full power to analyze in detail
- What the main issues are?
- Where they are located?
- When they were added to your code base?
- And who originally introduced them?



sonar-governance-plugin / src/main/java/com/sonar/governance/Governan

Make the "LOG" logger private static final.

Code Smell ▾ Major ▾ Open ▾ Sébastien Lesaint ▾ 5

sonar-governance-plugin / src/.../ce/projectexport/issue/ExportIssuesChan

Catch Exception instead of Throwable.

Code Smell ▾ Major ▾ Open ▾ Simon Brandhof ▾ 20

sonar-governance-plugin / src/.../ce/projectexport/steps/LoadProjectStep.j

1 duplicated blocks of code must be removed.

Code Smell ▾ Major ▾ Open ▾ tomverin ▾ 20min effo

sonar-governance-plugin / src/.../ce/projectimport/issue/ImportIssuesChan

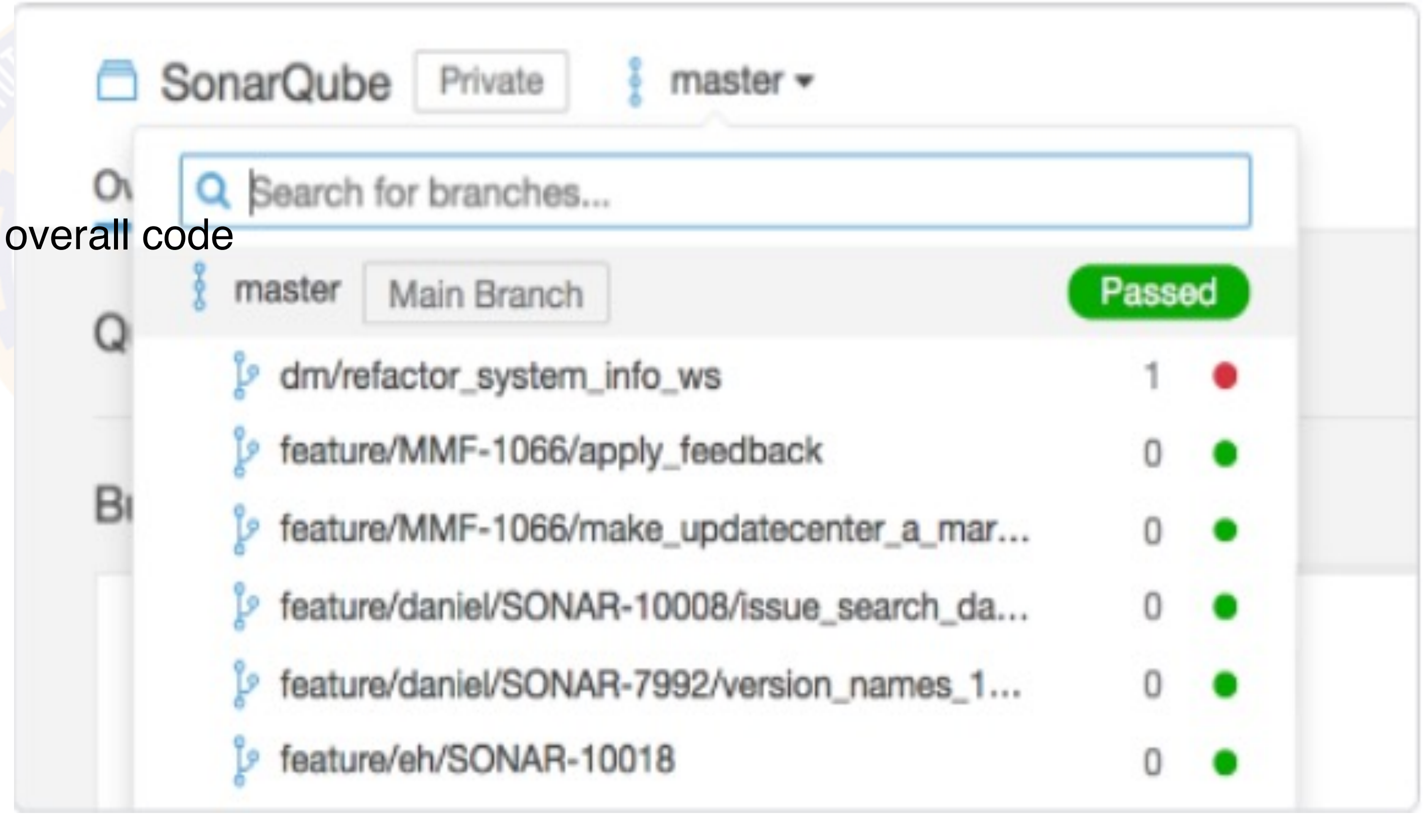
Annotate the parameter with @javax.annotation.Nullable in me

SonarQube

Analyzing Source Code

- Analyze pull requests
  - Focuses on new code – The Pull Request quality gate only uses your project's quality gate conditions that apply to "on New Code" metrics.

- Branch Analysis
  - Each branch has a quality gate that:
  - Applies on conditions on New Code and overall code
  - Assigns a status (Passed or Failed)



dm/refactor_system_info_ws	1	●
feature/MMF-1066/apply_feedback	0	●
feature/MMF-1066/make_updatecenter_a_mar...	0	●
feature/daniel/SONAR-10008/issue_search_da...	0	●
feature/daniel/SONAR-7992/version_names_1...	0	●
feature/eh/SONAR-10018	0	●

SonarQube

Integration for DevOps

maven

<APACHE ANT>

Gradle

Makefile

MSBuild

Bamboo

Travis CI

Jenkins

AppVeyor

Azure DevOps

TeamCity

sonarqube

Q&A

Thank You!

In our next session: