SECUCHECK

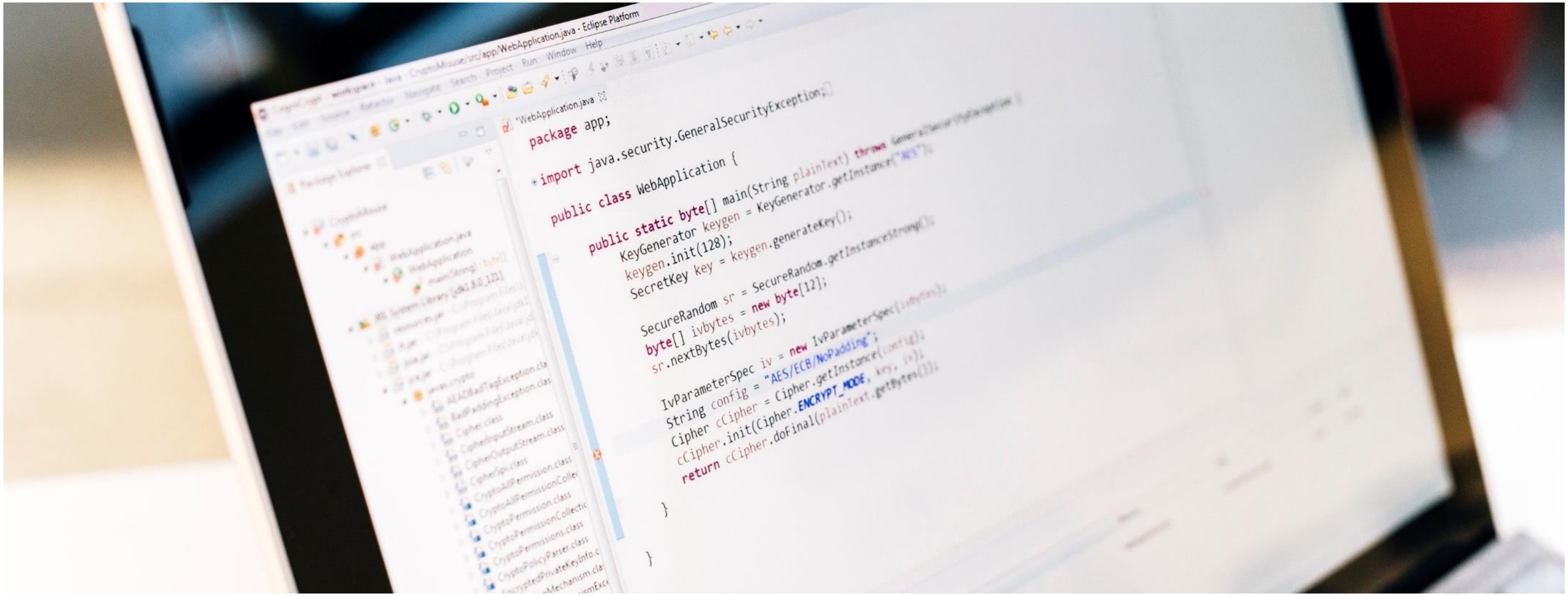
https://secucheck.github.io

@secucheck_swc

Interviews with Software Developers – Overview

Goran Piskachev, Fraunhofer IEM
Oshando Johnson, Fraunhofer IEM

30. June 2019



```
package app;

import java.security.GeneralSecurityException;

public class WebApplication {

    public static byte[] main(String plainText) throws GeneralSecurityException {
        KeyGenerator keygen = KeyGenerator.getInstance("AES");
        keygen.init(128);
        SecretKey key = keygen.generateKey();

        SecureRandom sr = SecureRandom.getInstanceStrong();
        byte[] ivbytes = new byte[12];
        sr.nextBytes(ivbytes);

        IvParameterSpec iv = new IvParameterSpec(ivbytes);
        String config = "AES/ECB/NoPadding";
        Cipher cCipher = Cipher.getInstance(config);
        cCipher.init(Cipher.ENCRYPT_MODE, key, iv);
        return cCipher.doFinal(plainText.getBytes());

    }

}
```

HEINZ NIXDORF INSTITUT
UNIVERSITÄT PADERBORN

Fraunhofer
IEM

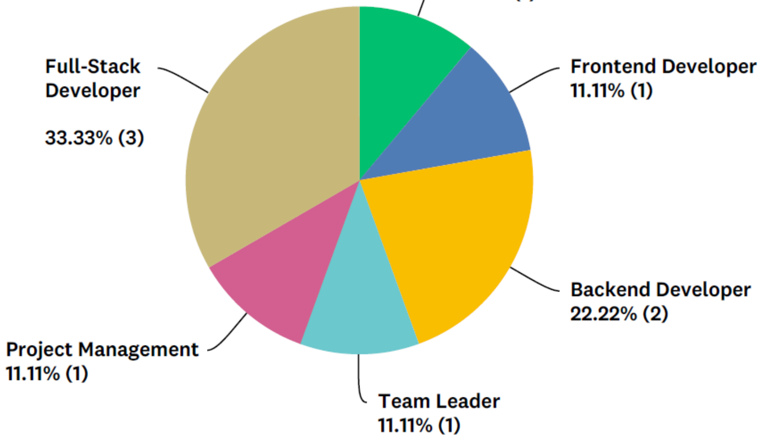
Interview Process

- **Period**: 14.02.2019 – 08.03.2019
- **Participants**: 9 Developers
  - Location of Interviewees: Germany, United Kingdom, India
  - From 5 companies
- **Duration**: 45 Minutes – 1 Hour
- **Tools**
  - Data Collection and Analysis – SurveyMonkey
  - Meeting – Skype for Business
  - Recording – QuickTime Player (with Soundflower Kernel) and Skype for Business

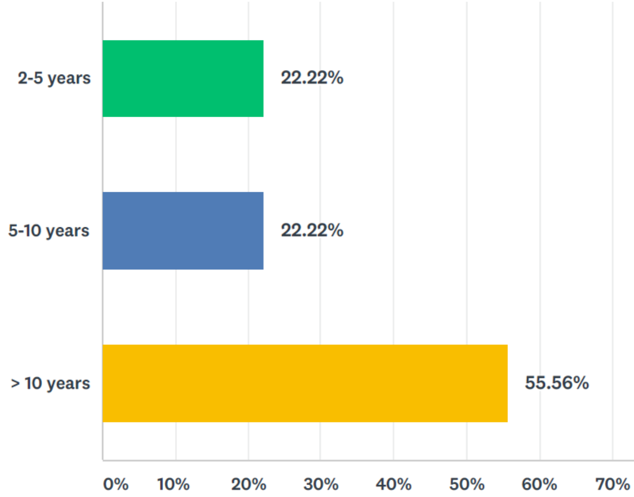
Interview Outline

- Sections
  - **General** (12 Questions) – day-to-day workflow of developers
  - **Static Analysis Tools** (13 Questions) – experience with static analysis tools
  - **Security Awareness** (11 Questions)– security awareness, practices and policies followed by developers
  - **Architecture and Design** (4 Questions) – user expectation and evaluation of mock-ups and prototype
  - **Query Language** (3 Questions) – evaluation of prototype and language approaches
- Following answers are based on 9 responses. When less, this is indicated.
- Note: some questions are multiple choice.

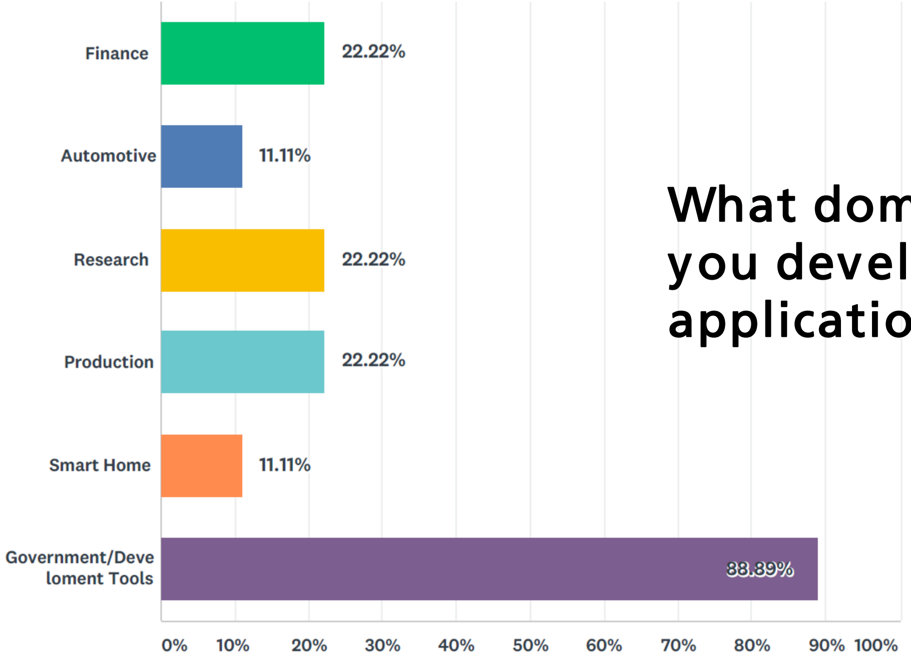
Roles

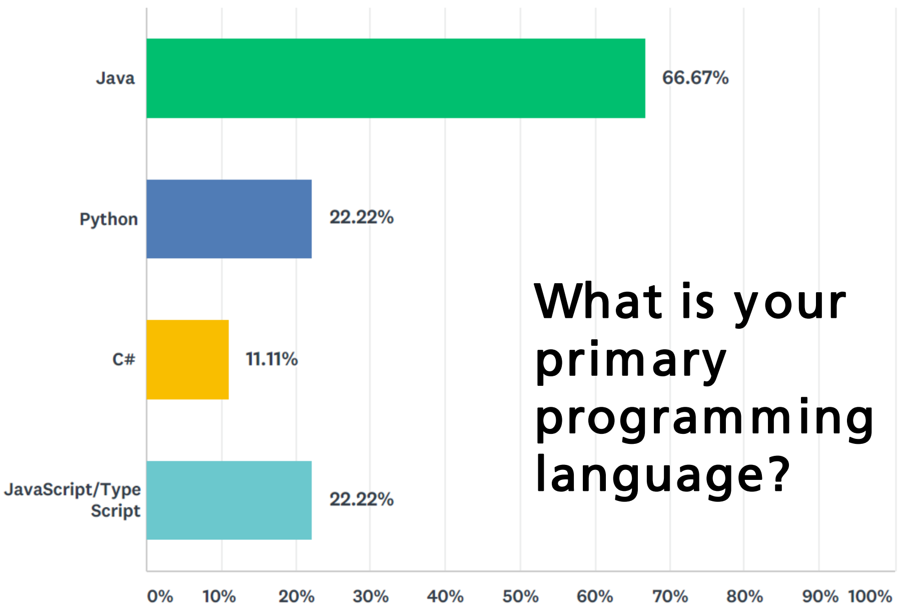


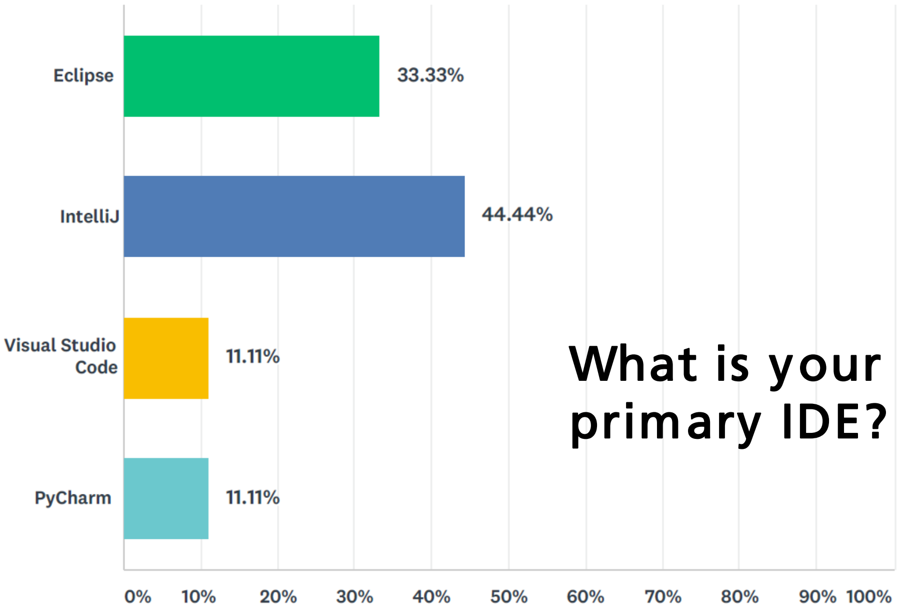
Years of experience



What domains did you develop application(s) for?

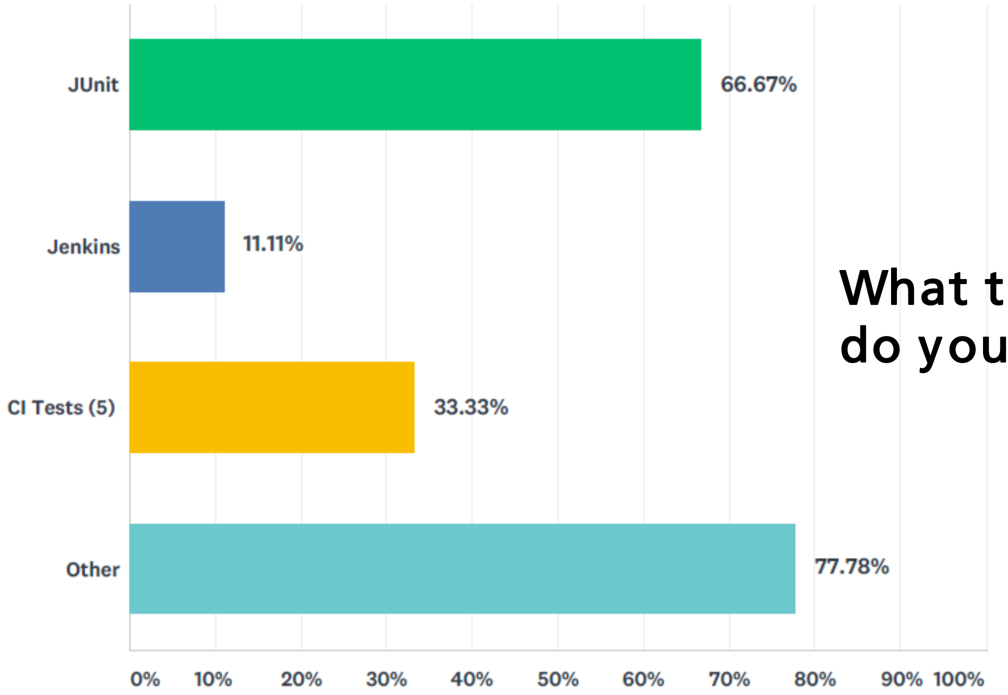



What is your primary programming language?

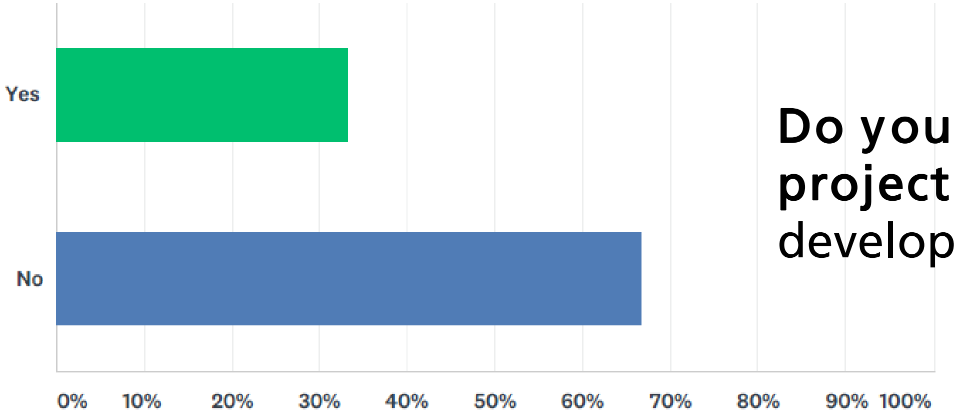

What is your primary IDE?

What frameworks/platforms/libraries do you use in your applications? Examples: Frameworks (Spring, Struts, Maven, Junit, etc.); Libraries (Apache Commons, Guava, Log4j/Slf4j, etc.) – 6 responses

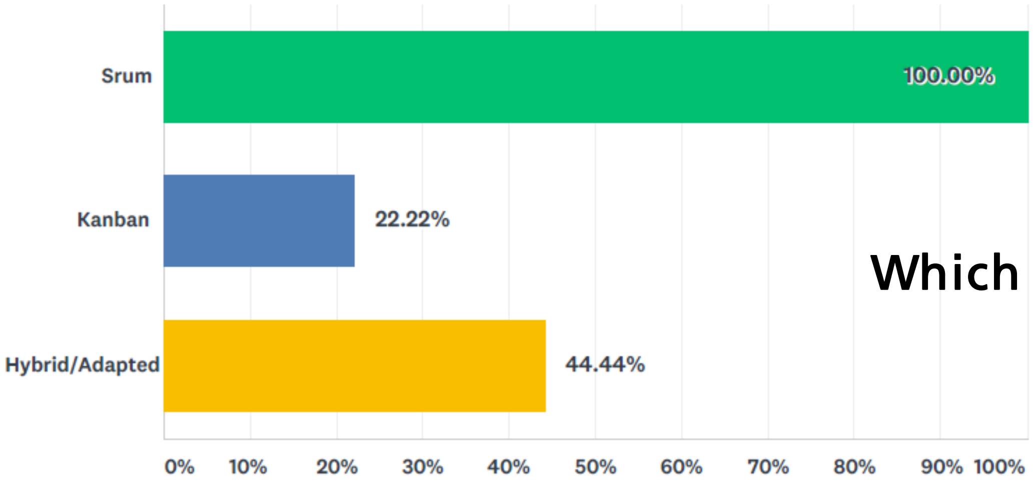
1. Low level application development so not many dependencies\libraries are used; Spring Boot is used.
2. OSGI, Jackson, Apache Commons, Software AG frontend tools, Angular
3. Python - Panda (data science), Flask (web framework) and Express.js
4. JWT framework; Web services stack with JAX-RS specification, Spring, Maven
5. Log4j, Jersey, H2, JDOM, Apache Common Compress and IO, JSON, NodeJS
6. Json.NET, Entitiy Framework, XML.NET



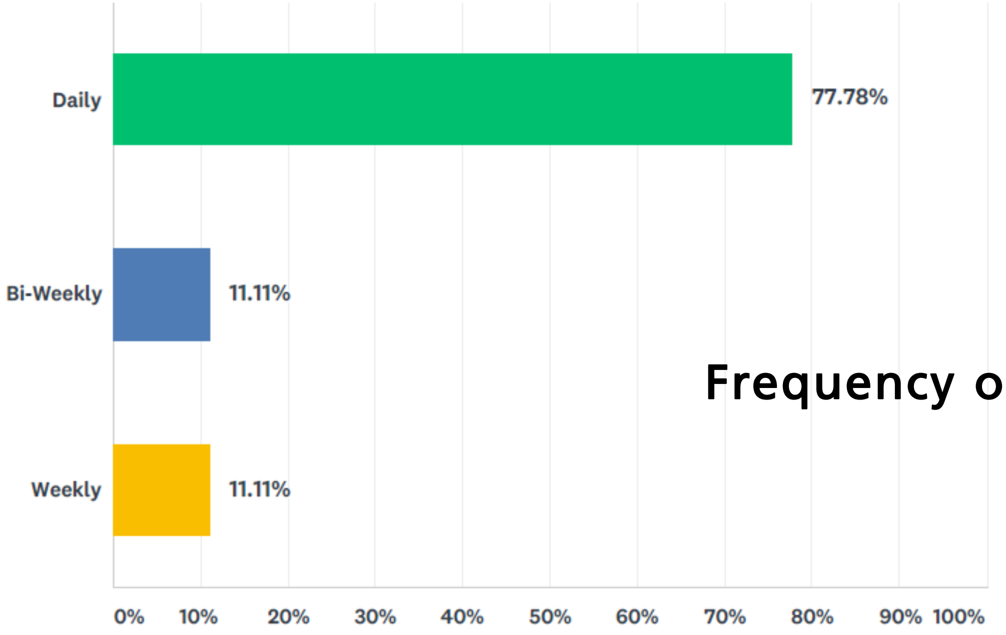
What testing framework do you use?



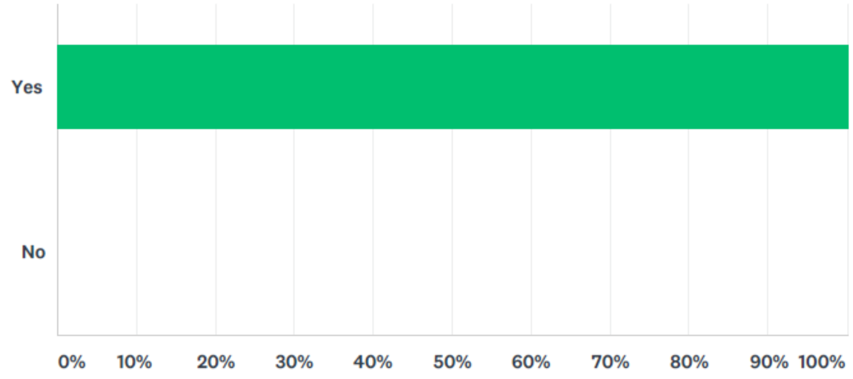
Do you change roles during project? (example testing and development)



Which agile model do you apply?

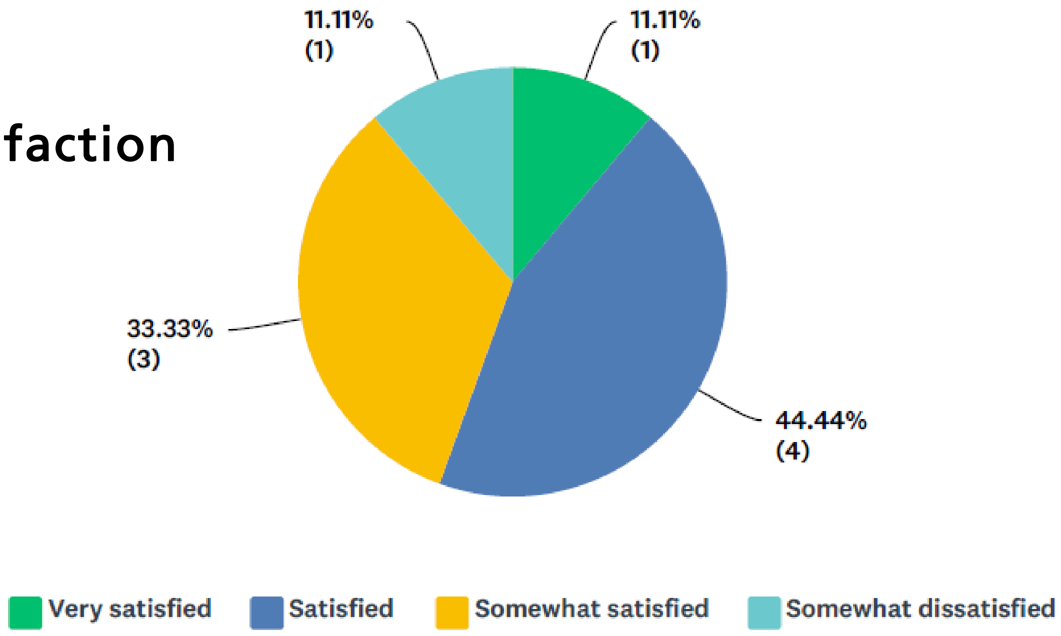


Frequency of meetings



Do you use static analysis tools while coding or to evaluate your code?

Tool Satisfaction

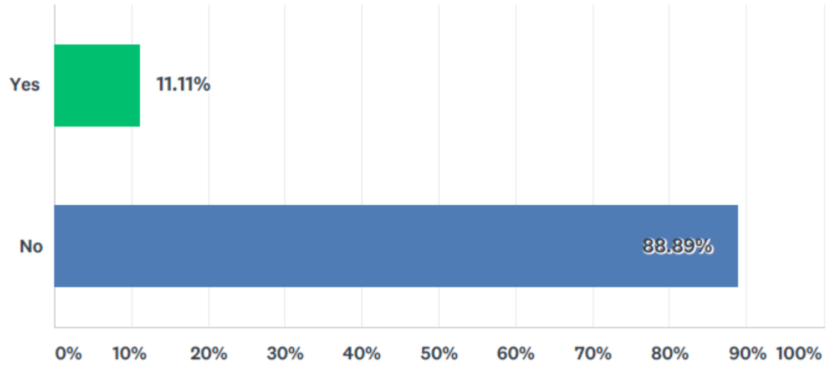


What features of the existing static analysis tools do you like? – 9 responses

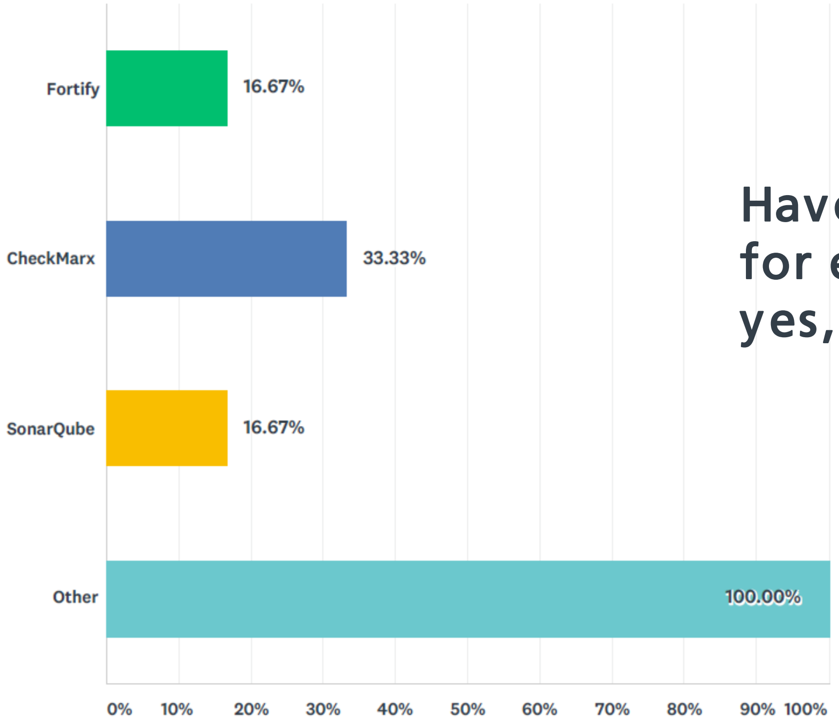
1. Shows how data flows in program; Click on the flow and see the code that corresponds to the flow; Flag methods as false positives and it remembers the flagged methods; Code segments can be be ignored
2. Help to improve coding style, identify application logic errors and other potential issues, ensures that coding conventions are followed and point out duplicated code. PyLint for example identifies if a class has too many or too few methods. Shows result both in the IDE and as reports. Pre-commit checks are executed and CI would catch other issues.
3. You can create stories for each product; CheckMarx allows you to click on the problem and you can get an explanation – identifies where sinks and sources are located in the application.
4. Integrated with IDE, fast to work with, no big setup or installation, quite accurate.
5. FindBugs is straight forward, has plugin, easy to use, developers are happy, zero threshold is used, uses standard ruleset. PMD also has a plugin. FindBugs and PMD give you feedback as you code. Checkmark provides excellent reports, integrating in build process, executes for change list (quickly runs and provide results), severity ratings, percentage of issues and graphical charts.
6. Results and GUI - lots of information is provided for reported issues, outlines exactly where problem is located and gives recommendations about how to fix problems. For example, it shows where a sanitizer should be used. Plugin is also available that shows data flow using diagrams as well as where problems are located.
7. SonarQube - Delivers everything PMD and Findbugs offers (rules were imported from these applications); Shows which issues are new or have been resolved and test coverage; A build breaker is also included which stops the build process if a critical issue is detected. Plugin obtains ruleset from server and plugin can also be customised. SonarQube offers a plugin for CheckMarx.
8. Gives immediate feedback by identifying code smells and bugs; Gives a score for code quality; Identified specific classes that may be problematic
9. Find Bugs - a small tool and it is easy to run. It also offers a plugin for Eclipse and IntelliJ. It is the main tool currently being used. Checkmarx - program the evaluation of the findings that the tool gives

What challenges or issues do you have when running the tools? Do you have problems understanding the reports? – 9 responses

1. Does not automatically detect areas that don't need to be scanned; Methods marked as false positives seem to disappear and have to be remarked - a complete list of methods marked would also be good to have; Lower/raise threshold to improve application's accuracy and reduce false positives; Sometimes hard to follow data flow - more intuitive representation would be effective, for example, hovering over a method; Too much code has to be read to understand the report; Classifying methods as sanitizers should also be possible; Software should intuitively pick up on different manifestations of the same problem/false positive; Annoying to check each project to find issues, it should be possible to see all defects;
2. PyLint sometimes report too many false positives and this may skew the application's true code quality. Application has to be customised correctly to obtain useful feedback. In rare situations, the reports are sometimes not very clear and further lookup has to be done to understand the problem.
3. Reports tend to be too large and could be more precise/specific
4. Issue with over approximation of issues, doesn't model entire lifecycle of the entire framework . - just uses code that it sees and operates on it.
5. Checkmarx does not have an IDE plugin so build has to be executed in order to see issues. PMD - a bit complex to understand
6. Speed of static analysis - some runs take more than one day. False positive rate is usually high when new projects are added. One file cannot be scanned, the analysis is also executed for the entire project.
7. CheckMarx requires a bit of deep diving to understand exactly what the issue is but it usually makes sense.
8. Application setup (server and database) was challenging.
9. HP Fortify - has a lot of false positives, not good for the team; A lot of work had to be done for little value. Xsanitizer - previously free but now it's a paid software and quite expensive; when the tool was evaluated, it was very difficult to setup - had to create entry points for all the frameworks that the company uses and the tool required a lot of configuration to get it to work. Because of the amount of work, it was decided to not use it anyone. FindBugs- doesn't offer a lot of results. Checkmarx - too expensive



Have you written static analysis on your own? for example in frameworks like Soot or WALA.



Have you written customized rules for existing static analysis tools? If yes, which tools? – 6 responses

How difficult is it to re-configure static analysis tools? For example enable/disable specific features or checks, customize the output view, etc. – 5 responses

1. It can sometimes be challenging because a very large configuration file is used and this needs to be copied to other projects.
2. Writing rules required a lot of effort and currently takes at least a day to write new rules. Language uses lists and other structures and is quite complex. Training was also done for the team. Language is powerful but requires a lot effort. Reconfiguration is not difficult.
3. Reconfiguration is not complicated. SonarQube server sends emails regarding duplicate or invalid rulesets. A web interface is also available to manage the server settings.
4. Initially it required some time, but it was eventually achievable.
5. Fortify - This was very hard to do especially for Fortify where an instance of Eclipse had to be started and then the abstractSyntaxTree had to be created along with other things. It was a very buggy process. FindBugs - easy to use and IDE integration is excellent.

Are you aware of static analysis features in your IDE? (name of features or examples)? Do you use them? Why or why not? – 8 responses

1. Built in default static analysis tool in PyCharm which detect errors and also suggest improvements and corrections.
2. Generally aware that static analysis is used by Eclipse to evaluate code and assist developers.
3. Auto-completion (feature is very handy), other background features, perform type checks
4. Yes, plugins in the IDE that show various violations in the code.
5. Find Security bugs, PMD; Build system also performs static analysis, SonarQube and internal tools.
6. Tools from JetBrains such as DotCover - these tools aren't used because they are paid and cannot be used on private projects
7. Not aware
8. Not aware

Is there a formal process for using static analyses and is it enforced?

1. Every project uses some type of static analysis tools and high priority issues are addressed. A report must be completed at the end of each project about how well the security process was followed and whether or not the project meets security requirements; It is not really enforced or drives development, but seems to have an occasional impact. Reports might be examined and decisions are made but doesn't influence individuals or teams.
2. A formal process does not exist. Developers are free to use whatever tools they think will be helpful
3. Formal process used only for CheckMarx - all new releases should have no high priority defects
4. No formal process. Developers can decide to use it.
5. A clear process from the company about how static analysis tools are to be used. Clear goals are created for each release and no project should be released that has high or critical Checkmarx issues. Dedicated team member reviews report and coordinates how the problems should be fixed. Each team is required to follow the process but no one will necessarily check. Jenkins is used to coordinate the resolution of the issue.
6. Security policy to be followed by all development teams; Static code analyzers have to be used and all critical problems must be resolved before release. Product owner must also do security release task in which he checks results from CheckMarx before releasing the product.
7. Every project must be checked by CheckMarx. This is also enforced as managers evaluate the report
8. Using static analysis tools (SonarQube) is part of the development life cycle; Code has to pass quality checks and can also be release when all issues are resolved.
9. Project dependent: Developers can use the tools if they want to.

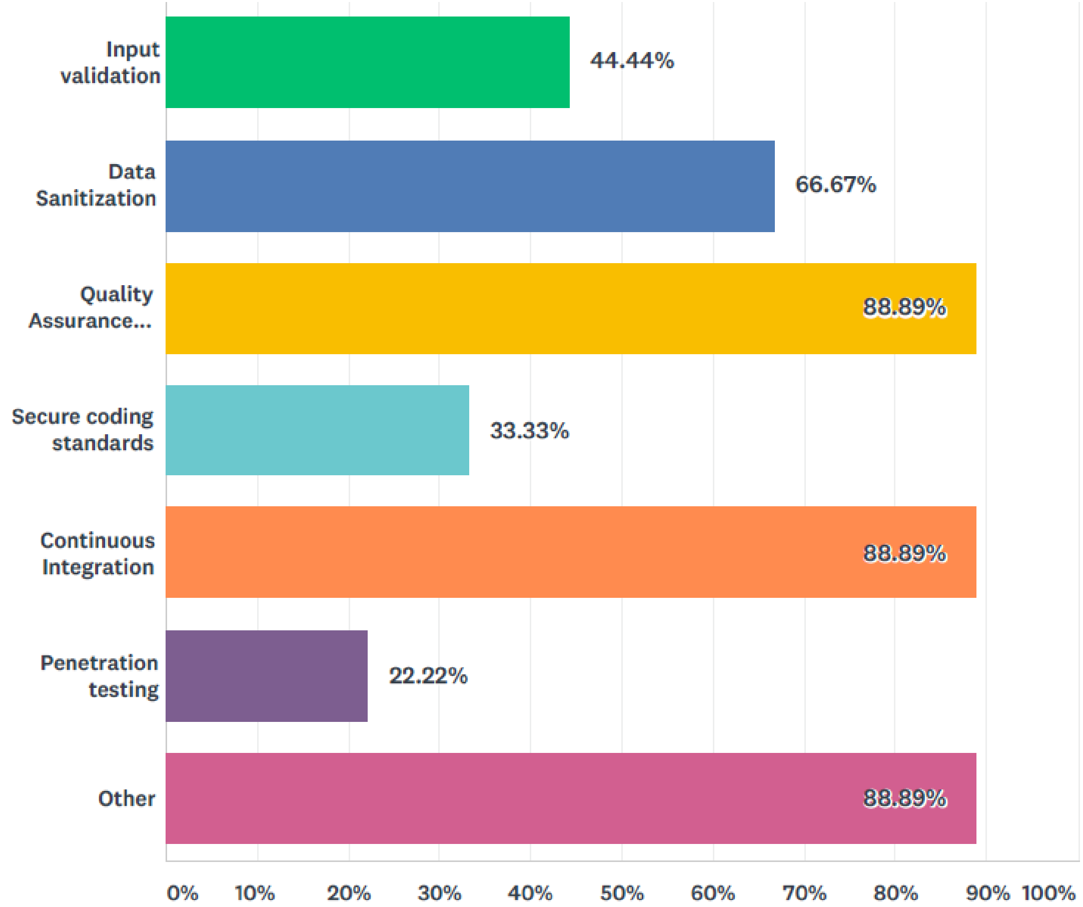
When gathering software requirements, designing systems and/or implementing new features, is security considered? If yes, what is done? Are components classified based on security vulnerability or impact?

1. Yes. Security reviews are supposed to be done by developers that are aware of security practices. However, this is often not done. Sometime dismissive towards security issues or possible vulnerabilities because applications are usually internal.
2. Yes. Security is considered by mostly ensuring that valid SSL certificates are being used in the application. The aim is to ensure the application in its entirety works as expected and not necessarily individual components.
3. Yes. Although important, security is not considered when designing new features. An internal tool with security related questions provides a report outlining security concerns, precautions and recommendations. This tool is actually used prior to releasing the software. The main reason is that security isn't considered and may be a result of an ineffective process. Some things are only remembered when release tasks are to be done. When small problems are found, they are usually fixed. However, for more tedious tasks, the product manager determine the priority which means that some issues are often not dealt with.
4. Yes. It's about the tools that you use. For example, using SQL it's a good idea to use an ORM to abstract SQL so you can avoid writing SQL statements that can open SQL injections - prepared statements are better. Development process determines library selection and can ensure basic security is available in the program. Try not to implement security - aim to use existing tools (for example, token generation). No categorisation of issues is done. Developer can decide.

When gathering software requirements, designing systems and/or implementing new features, is security considered? If yes, what is done? Are components classified based on security vulnerability or impact?

5. Yes. Security Analysis Framework developed by internal security team must be used by managers to answer questions relating to the software features. The tool will provide a risk analysis report that shows possible security vulnerabilities or concerns and also maps to security vulnerability database (CWE).
6. Yes. Security representatives are present in every team and the security team also supports and educate (security awareness training, online courses and online learning tools (XSS)) developers. Plan is to also schedule more training for developers. When designing or implementing new features, penetration tests must be planned for it.
7. Yes. However, not all the time. At times, there are different issues that are given more attention.
8. Yes. Ensure that certificates are valid, HTTPS is used, passwords are encrypted, etc.; Web APIs are also secured using authentication and authorization steps;
9. Yes. Security is considered and evaluated using penetration tests in which risk analysis of the projects are done. Whitebox testing as well as static analysis is also done if time is available. Educational/training programs are also done to help with improving code quality. Each project usually has a security lead/champion who is responsible for assessing security implications at various phases of the project. Aim to integrate security from the beginning for new projects -during requirements gathering, use cases are created and used to determine the proper frameworks which have good metrics in terms of security.

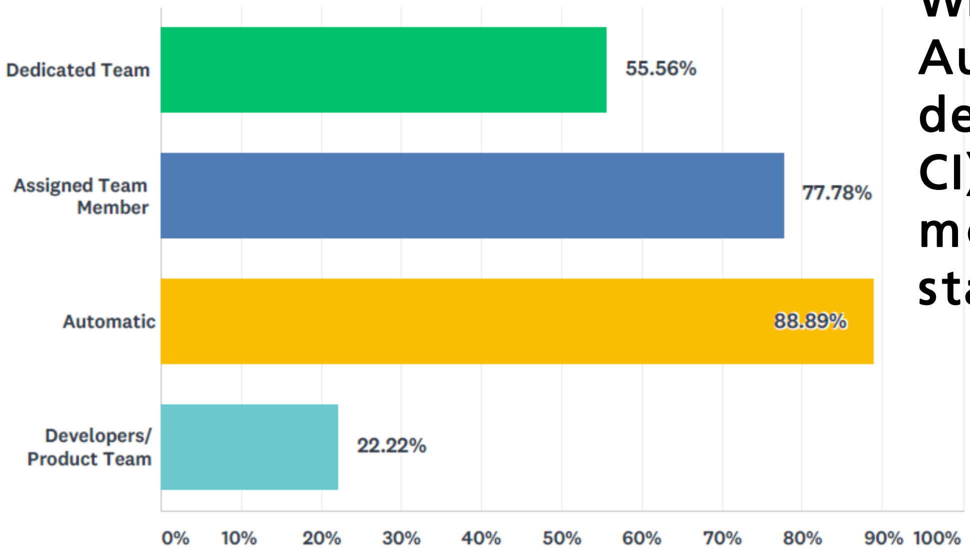
Do you have team and/or company secure coding standards/practices (standard of quality in development process)? If so, what are they?



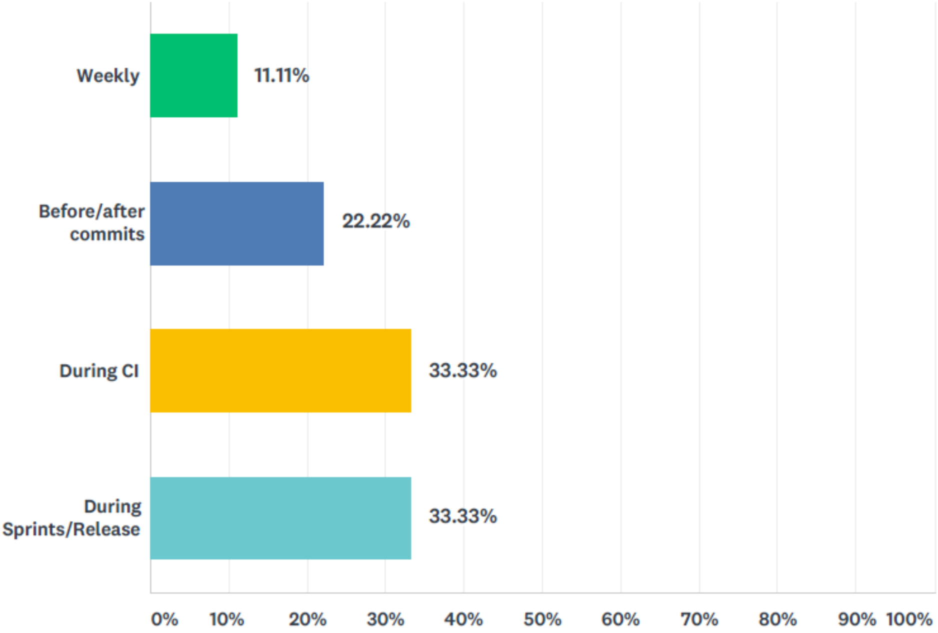
How are these standards enforced/monitored? What are the checks?

1. Often monitored during code reviews or CI.
2. Test cases are documented and must be passed
3. Enforced using static analysis and other continuous integration tools. A QA team also performs check and may open issues
4. Developers are usually responsible for enforcing standards. Team Leader sometimes perform checks but that is not always possible because team consists of freelancers.
5. Security experts will perform audits to ensure compliance as well as automatic checks in the CI process
6. At end the end of release phase, various security processes have to be completed. Monitoring is done by assessing security scan and teams are informed when high vulnerability rates are observed
7. Enforcement is mainly done by CheckMarx which is part of the CI
8. SonarQube enforces standards - developers can run it manually but it also part of the deployment/merging process; Quality Assurance engineer also performs tests
9. Enforcement is mostly done from an organizational. Usually someone on the team is assigned to take a look at things of interest for security. Continuous integration also helps but again this is driven by an assigned team member.

Who performs these checks? Automatic (integrated in development environment and/or CI), specific team or each team monitors and/or enforces the standards?



How often are these checks done?



What metrics or evaluation methods are used for software quality? – 8 responses

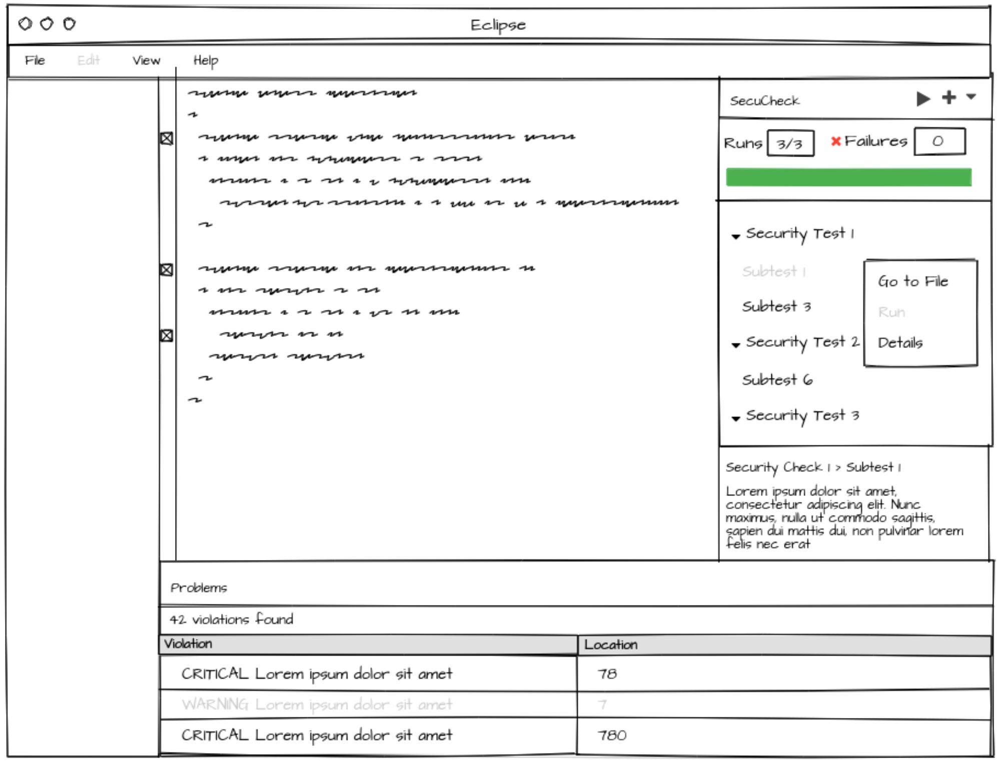
1. Results of tests is used to indicate software quality
2. Lines per function is sometimes used to prevent super complex functions or methods
3. Number of security risks/problems found by various tools; Security evaluation tool used prior to releasing software also assess software quality
4. Number of vulnerabilities reported by tools
5. CI Reports are used to assess software quality
6. 80% pass for checks done by SonarQube
7. Customer dependent: Some customers have their own metrics for example code coverage from unit tests; for new projects metrics are used to select frameworks; Also statistics about different libraries and projects.
8. This is mainly determined based on CI reports and the absence of high priority issues.

What design and architecture expectations would you have for a tool that allows developers to run various security checks on their code? Examples: Automatic/User triggered checks; dedicated view in Eclipse or gutter icons in the editor;

1. Not in Eclipse - tool should possibly also support IntelliJ; unobtrusive - screen space is limited; should be really fast - shouldn't take a minute
2. Clear and effective reports (maybe HTML reports); Tool should identify the specific line where a problem exists; Diagrams may also be helpful to explain issues; User interface should be easy to work with and should not be complicated to setup, configure and understand
3. (1) It should be automatic - run in the background. Shouldn't need to click on a button to start running the application. (2) Problems should be shown in the warning or error views. Plugin should work with all projects that are in the workspace. (3) Plugin should be easy to install (for example, install plugins from Jenkins top-chain). (4) Problem should be clearly explained. Descriptions and longer explanations as well as solutions should be available. Maybe even a link to webpage that provides more details.
4. Explain the problem clearly and give possible solutions or recommendations. Providing code code samples is also helpful and can help with educating developers. Suggest refactoring for code which can be applied at the click of a button. Errors should be clearly indicated on the left side of the editor and information should be given inline/with tooltips (information should be detailed). Reference to CWE database or other security lists. Run analysis on a file, method or entire project. A report that groups by type of security issues/pattern. Manager should be able to assign team member to fix issues. Impact analysis of identified security issues should also available - a report showing the list of affected files.

What design and architecture expectations would you have for a tool that allows developers to run various security checks on their code? Examples: Automatic/User triggered checks; dedicated view in Eclipse or gutter icons in the editor;

5. Not too many false positives or negatives, results in an appropriate time (no one wants to use a tool that takes too long), directly integrated in the IDE using popups, automatic execution and results
6. Run fast analyses on the code; Should be executable for a particular class or module and results should be provided within a feasible time. Can be automatic or manually initiated.
7. Plugin should work similar to SonarLint: tells security issue for active code in the editor; issues displayed in problem view or dedicated plugin problem view; run for of the source code
8. Performance - shouldn't take a lot of time to execute analyses, give clues which are correct, not too many fall positives (if there are too many false positives then developer might oversee something because they think that everything is wrong); shouldn't require too much effort to configure the tool. Dynamic feedback and recommendations given should be understandable for the programmer and it will be easier to fix and won't be ignored by the the developer.
9. Should be easy to use - doesn't require a lot of time to run manually or automatically; UI that provides more details the issue (explanation and impact on security); Give a score for code quality; Find repeated code; Options to customise rules and run for specific pushes/branches/components

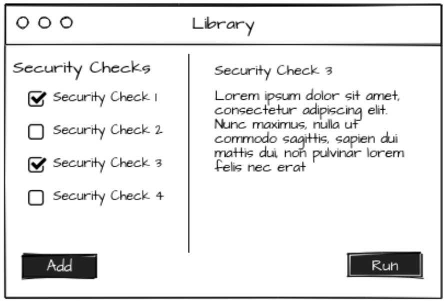


Security Check 1 > Subtest 1

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Nunc maximus, nulla ut commodo sagittis, sapien dui mattis dui, non pulvinar lorem felis nec erat

42 violations found

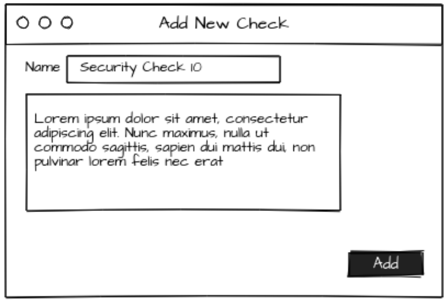
Violation	Location
CRITICAL Lorem ipsum dolor sit amet	78
WARNING Lorem ipsum dolor sit amet	7
CRITICAL Lorem ipsum dolor sit amet	780



- ☑ Security Check 1
- □ Security Check 2
- ☑ Security Check 3
- □ Security Check 4

Security Check 3

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Nunc maximus, nulla ut commodo sagittis, sapien dui mattis dui, non pulvinar lorem felis nec erat



Name: Security Check 10

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Nunc maximus, nulla ut commodo sagittis, sapien dui mattis dui, non pulvinar lorem felis nec erat

Given the mock-up. Which layout and design elements would you add or remove? Would you want to give feedback or provide examples/counter examples to the tool?- Would you like to add or edit queries?

1. Run for a class or method and not for the entire project; scan entire project and then process for individual project; important to give feedback
2. Looks logical and provides information required and expected. Modify sub-checks in list. Ignore warnings and it should be possible to view all issues that were ignored - shouldn't add anything to the code in order to ignore issues. The list of flagged issues should be accessible and applicable to other projects; Issues should also be grouped based on their categories. Pre commit hooks can be used to prevent committing until all issues are resolved. Would be interested in writing own queries, but only if there's no other way to do it with built in queries. As long as it doesn't require too much effort to learn the language.

Given the mock-up. Which layout and design elements would you add or remove?

Would you want to give feedback or provide examples/counter examples to the tool?- Would you like to add or edit queries?

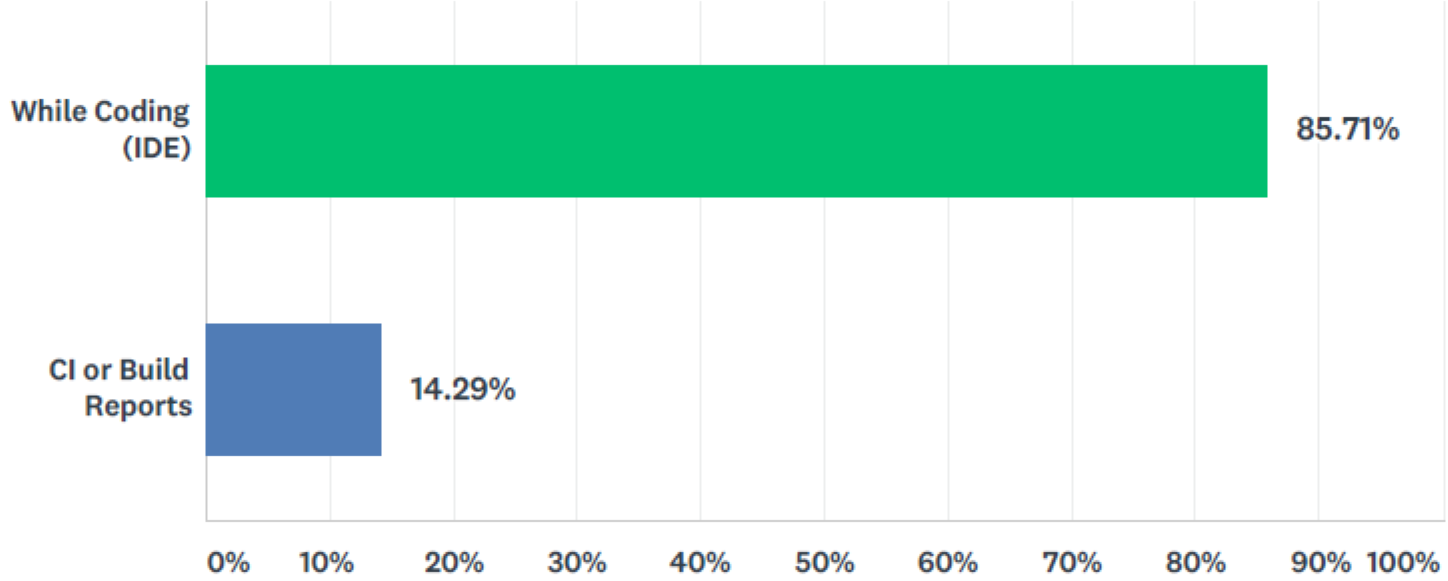
3. (1) Able to ignore problems - share this information and other settings with other developers working on the same project. (2) Editing queries might only be useful for the security team. Developer's would simply want an extensive set of security checks that they can run on their code. (3) Tool shouldn't be intrusive or affect the developer's workflow.
4. Good that it is integrated in Eclipse, it's nice that you can select the security checks that you want to execute. Not sure about the steps to set up new security checks. UX is good being that it is integrated. Would provide feedback about incorrectly flagged methods and the tool could possibly use that information with AI to do better analysis in the future. Should have a reward for saying that this is a problem. If nothing changes, there is not value for me doing it. Not focussed on static analysis and not much focus is placed on quality because it doesn't need to be maintained afterwards. Wouldn't have time to write own rules because of timespan and demands for the projects.
5. Should be possible to manually or automatically start an analysis. UI components should be clearly labelled; Checks should run based on active file in the editor and editor should also be able to manually execute check. If query language is easy to understand and write, would be interested to write own queries. Doesn't necessarily sees cases where queries need to be written especially if the tool already has a good predefined set of tools.
6. Line markers are effective; Mark false positives and it should be possible to define rules to prevent reoccurring false positives.

Given the mock-up. Which layout and design elements would you add or remove?

Would you want to give feedback or provide examples/counter examples to the tool?- Would you like to add or edit queries?

7. View to show list of affected files - effective way to navigate through violations; Show line number and preview of the line of code; Categorize violation - user can configure categories and determine impact in development mode versus release; Scan files or snippets of code; Should not take too much to execute; Integrate with source code control tools - stop build for example; Big data analysis on issues identified; Database - No SQL to read and normal relational database to write; Ignore methods and add notes to give an explanation;
8. Layout is good. The important thing is that the explanation text should be understandable. Should be able to see the way the source flows through the program - through the lines of code. Checkmarx provides a tree and it helps you to identify where a fix can be applied and that it would resolve other related issues. If artificial intelligence is uses the user feedback, then this can be beneficial. It's important for the developer and company to know exactly what information goes out. The company will always be afraid that it looses its intellectual property and that an external person sees where vulnerabilities in the software are located. It would be cool to update and/or add queries. For penetration testers, they can use this feature to write their own automated penetration tests.
9. Problems view is effective; SecuCheck view is a bit confusing with with rules and results being shown together. It's also a bit confusing being that it looks JUnit - it mixes up rules in SecuCheck and tests that would be in JUnit. Rule sets should be set in preferences and should be applicable to the entire project. Should be able to flag false positives. Not really interested in writing own queries as the required knowledge might be missing. Predefined rules should be sufficient.

When do you prefer to see the results from the tool?

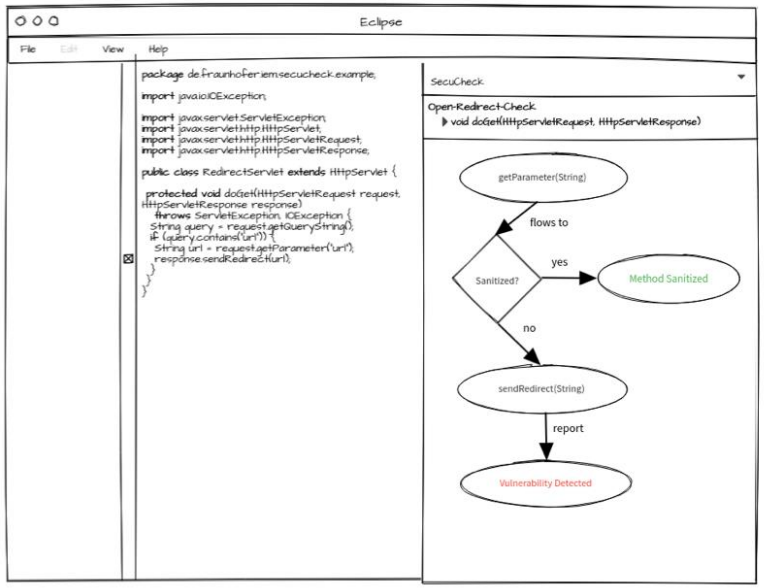


Compare the queries written with three different languages and evaluate the following properties: comprehensible, readable, usable and writeable.

1. Java Fluent Interface (internal)

```
TaintAnalysis openRedirect = new TaintAnalysis();
List < String > openRedirectSantizers = openRedirect.sanitizers().toList.get();
openRedirect.method("RedirectServlet.doGet(HttpServletRequest, HttpServletResponse)")
    .source("getParameter(String)")
    .flowsTo()
    .isSanitized(openRedirectSantizers)
    .sink("sendRedirect(String)")
    .report();
```

3. Graphical syntax



```
package de.fraunhofer.iem.secucheck.example;

import java.io.IOException;

import javax.servlet.ServletException;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;

public class RedirectServlet extends HttpServlet {

 protected void doGet(HttpServletRequest request,
HttpServletResponse response)
   throws ServletException, IOException {
  String query = request.getQueryString();
  if (query.contains("url")) {
   String url = request.getParameter("url");
   response.sendRedirect(url);
  }
 }
}
```

2. Domain Specific Language (external)

```
entry method void
       RedirectServlet.doGet(in HttpServletRequest request, out HttpServletResponse response);
query "open-redirect-check" {
    var a = doGet((in HttpServletRequest request = doGet));
    return "Vulnerability detected: URL Redirection to Untrusted Site ('Open Redirect') [CWE-601]"
}
```

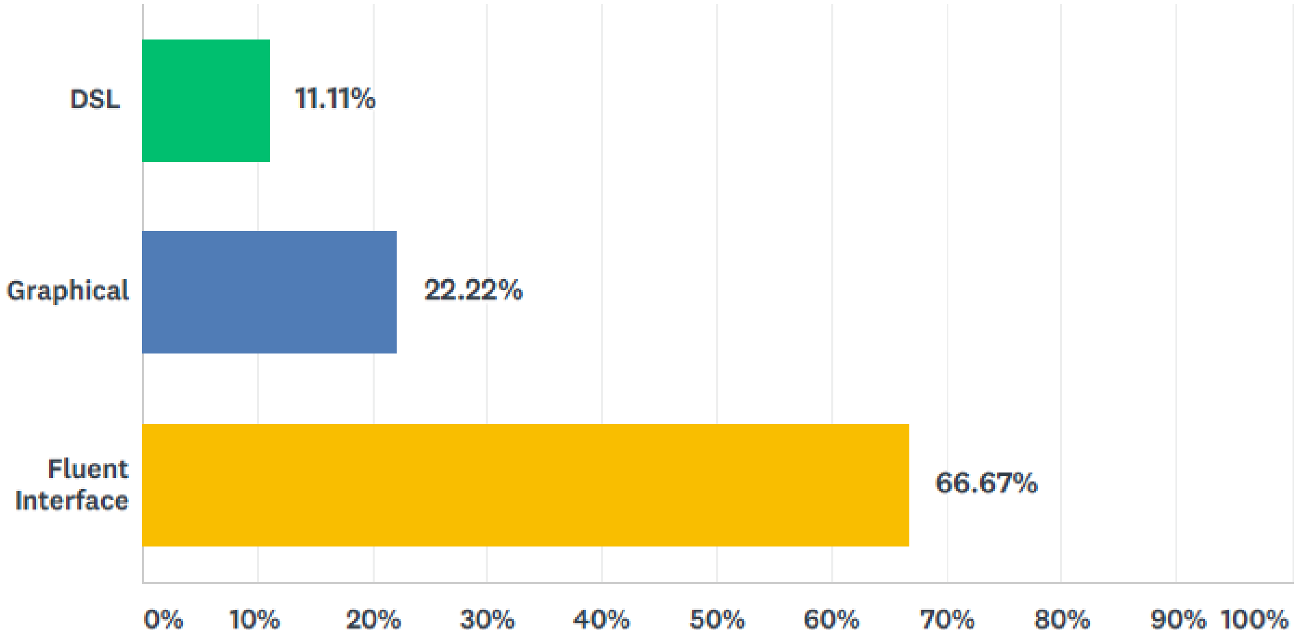
Compare the queries written with three different languages and evaluate the following properties: comprehensible, readable, usable and writeable.

1. The DSL is not very easy to understand - keywords that are used as well as the general structure of the syntax. The developer would not be interested in writing queries. Graphical is more understandable because it's a high level. Concerns about the versioning and maintenance of the graphical language. Maybe it's not necessary to represent the query graphically. Fluent Interface is much easier to understand. Specifying the method as a string and specifying the methods in general (this would be problematic when working with fully qualified names). "flowsto" is not needed as it is already implied.
2. Fluent interface integrate with the language that you already know, quite readable and can be read through the methods. DSL requires that you learn it first but it is still quite readable and specific to the particular use case. Graphical language is not that readable - probably easier to see it but personally do not like to use it. For others it might be easier to use and understand.
3. Fluent interface example is obvious especially if developer already understands Servlet API. Gives a logical flow of the problem. Fluent interface is easy to understand for Java developers Writing queries with the DSL might be difficult especially if the language's context and syntax isn't well known/documented. DSL requires extra effort. Flow diagram helps to better visualize the queries and helps the developer to also better understand the source code. Would be ideal for novice programmers. Nothing needs to be know to understand.
4. Text representation is much easier to understand in comparison to graphical language. Graphical language can be helpful for new users but a more precise language is better for more advanced users.

Compare the queries written with three different languages and evaluate the following properties: comprehensible, readable, usable and writeable.

5. Fluent Interface syntax was not easily understood, graphical might become complicated for bigger representations. Graphical would work very well for beginners. Graphical is better than the DSL; combine graphical with fluent;
6. Fluent interface integrate with the language that you already know, quite readable and can be read through the methods. DSL requires that you learn it first but it is still quite readable and specific to the particular use case. Graphical language is not that readable - probably easier to see it but personally do not like to use it. For others it might be easier to use and understand.
7. Fluent interface is readable, easier to learn/use and it the general idea can be assumed based on the query. Graphical language is also easily understood; however, distinguishing symbols and when to use them might be problematic/ambiguous. Maybe a combination fluent interface and the graphical language may also be effective.
8. DSL is a bit confusing, graphical gives you the idea of how the data flows; Fluent interface shows logical flow
9. Most understandable is the graphical approach and can clearly see how different approaches affect the analysis. Fluent interface shows the methods sequentially and you can go to the methods and get more details about what they are doing. Fluent interface is better than the DSL because you have to learn the DSL first. Maybe if you learn the DSL, it will be easier to understand.

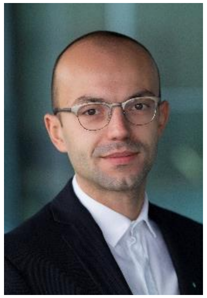
Which syntax do you prefer?



CONTACT

https://secucheck.github.io

@secucheck_swc



Goran Piskachev
Research Associate
Tel.: +49 5251 5465-168
goran.piskachev@iem.fraunhofer.de