Narrowing the Focus

Table of Contents

If SonarQube's results aren't relevant, developers will push back on using it. That's why configuring precisely what to analyze for each project is a very important step. Doing so allows you to remove noise, like the issues and duplications marked on generated code, or the issues from rules that aren't relevant for certain types of objects.

SonarQube gives you several options for configuring exactly what will be analyzed. You can

- completely ignore some files or directories
- exclude files/directories from Issue detection (specific rules or all of them) but analyze all other aspects
- exclude files/directories from Duplication detection but analyze all other aspects
- exclude files/directories from Coverage calculations but analyze all other aspects

You can make these changes globally or at a project level. At both levels, the navigation path is the same: **Administration > General Settings > Analysis Scope**.

Ignore Files

We recommend that you exclude generated code, source code from libraries, etc. There are four different ways to narrow your analysis to the source code that will be relevant to the development team. You can combine them all together to tune your analysis scope.

Source Directories

Set the `sonar.sources` property to limit the scope of the analysis to certain directories.

File Suffixes

Most language plugins offer a way to restrict the scope of analysis to files matching a set of extensions. Go to **Administration > General Settings > [Language]** to set the *File suffixes* property.

Choosing Files

To specify which files are are and are not included in an analysis go to **Administration > General Settings > Analysis Scope > Files.**

Use exclusion to analyze *everything but* the specified files:

- `sonar.exclusions` - to exclude source code files
- `sonar.test.exclusions` - to exclude unit test files

Use inclusion to analyzes *only* the specified files:

- `sonar.inclusions`
- `sonar.test.inclusions`

You can set these properties at the project and global levels.

See the Patterns section for more details on the syntax to use in these inputs.

See Also: Ignore Issues

Ignore Issues

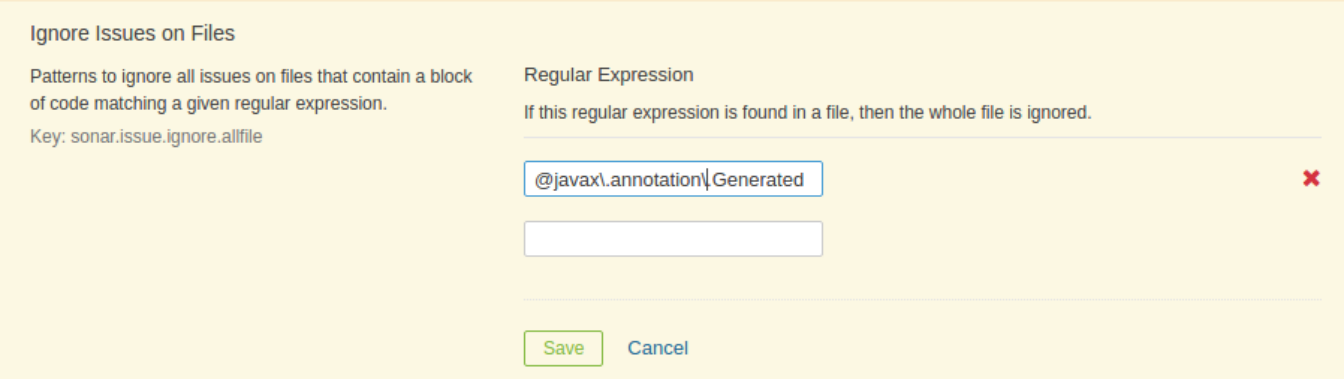
You can have SonarQube ignore issues on certain components and against certain coding rules. Go to **Administration > General Settings > Analysis Scope > Issues**.

Note that the properties below can only be set through the web interface because they are multi-valued.

Ignore Issues on Files

You can ignore all issues on files that contain a block of code matching a given regular expression.

Example: Ignore all issues on files containing `@javax.annotation.Generated`:



Ignore Issues in Blocks

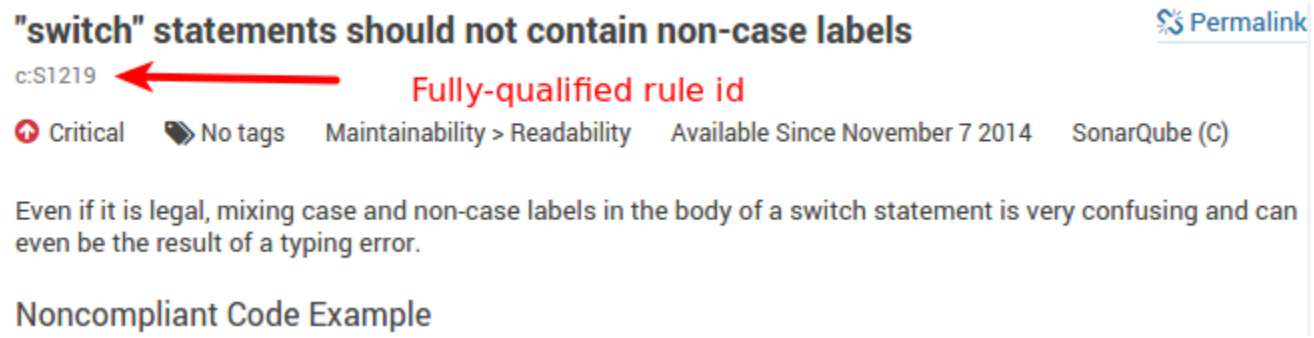
You can ignore all issues on specific blocks of code, while continuing to scan and mark issues on the remainder of the file. Blocks to be ignored are delimited by start and end strings which can be specified by regular expressions (or plain strings).

Notes:

- If the first regular expression is found but not the second one, the end of the file is considered to be the end of the block.
- Regular expressions are not matched on a multi-line basis.

Ignore Issues on Multiple Criteria

You can ignore issues on certain components and for certain coding rules. To list a specific rule, use the fully qualified rule id shown in the rule detail:



Examples:

- I want to ignore all issues on all files => key: `*; path: **/*`
- I want to ignore all issues on COBOL program *bank/ZTR00021.cbl* => key: `*; path: bank/ZTR00021.cbl`
- I want to ignore all issues on classes located directly in the Java package `com.foo`, but not in its sub-packages => key: `*;path: com/foo/*`
- I want to ignore all issues against coding rule "cpp:Union" on files in the directory *object* and its sub-directories => key: `cpp:Union; path: object/**/*`

Restrict Scope of Coding Rules

You can restrict the application of a rule to only certain components, ignoring all others.

Examples:

- I only want to check the rule *Magic Number* on *Bean* objects and not on anything else => key: `checkstyle:com.puppycrawl.tools.checkstyle.checks.coding.MagicNumberCheck;path: **/*Bean.java`
- I only want to check the rule *Prevent GO TO statement from transferring control outside current module* on COBOL programs located in the directories *bank/creditcard* and *bank/bankcard* => this one requires two criteria to define it:
  - `key: cobol:COBOL.GotoTransferControlOutsideCurrentModuleCheck; path: bank/creditcard/**/*`
  - `key: cobol:COBOL.GotoTransferControlOutsideCurrentModuleCheck; path: bank/bankcard/**/*`

Ignore Duplications

You can prevent some files from being checked for duplications.

To do so, go to **Administration > General Settings > Analysis Scope > Duplications** and set the *Duplication Exclusions* property. See the Patterns section for more details on the syntax.

Ignore Code Coverage

You can prevent some files from being taken into account for code coverage by unit tests.

To do so, go to **Administration > General Settings > Analysis Scope > Code Coverage** and set the *Coverage Exclusions* property. See the Patterns section for more details on the syntax.

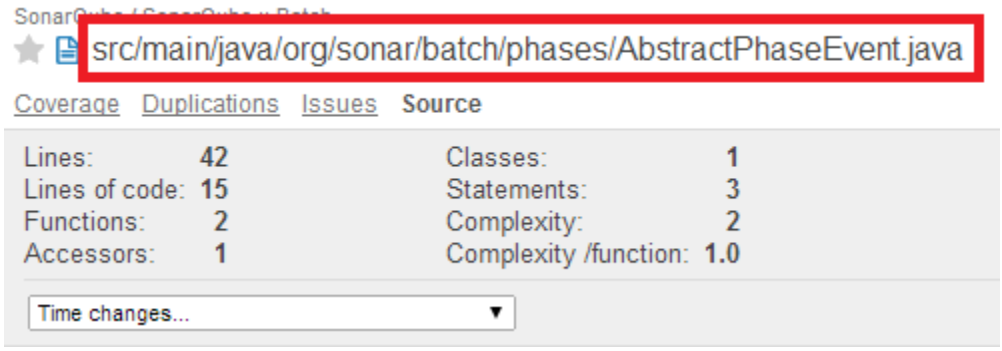
Patterns

Paths are relative to the project base directory.

The following wildcards can be used:

Wildcard	Matches
*	zero or more characters
**	zero or more directories
?	a single character

Relative paths are based on the fully qualified name of the component (like the one displayed in the red frame below):



.

Examples

```
# Exclude all classes ending by "Bean"
# Matches org/sonar.api/MyBean.java, org/sonar/util/MyOtherBean.java, org/sonar/util/MyDTO.java, etc.
sonar.exclusions=**/*Bean.java,**/*DTO.java

# Exclude all classes in the "src/main/java/org/sonar" directory
# Matches src/main/java/org/sonar/MyClass.java, src/main/java/org/sonar/MyOtherClass.java
# But does not match src/main/java/org/sonar/util/MyClassUtil.java
sonar.exclusions=src/main/java/org/sonar/*

# Exclude all COBOL programs in the "bank" directory and its sub-directories
# Matches bank/ZTR00021.cbl, bank/data/CBR00354.cbl, bank/data/REM012345.cob
sonar.exclusions=bank/**/*

# Exclude all COBOL programs in the "bank" directory and its sub-directories whose extension is .cbl
# Matches bank/ZTR00021.cbl, bank/data/CBR00354.cbl
sonar.exclusions=bank/**/*.cbl
```