Analyzing Source Code

Table of Contents

Once the SonarQube platform has been installed, you're ready to install an analyzer and begin creating projects. A project is created in the platform automatically on its first analysis. However, if you need to set some configuration on your project before its first analysis, you have the option of provisioning it.

Scope of Analysis: Types of Files and Data

SonarQube can perform analysis on 20+ different languages. The outcome of this analysis will be quality measures and issues (instances where coding rules were broken). However, what gets analyzed will vary depending on the language:

- On all languages, "blame" data will automatically be imported from supported SCM providers. Git and SVN are supported automatically. Other providers require additional plugins.
- On all languages, a static analysis of source code is performed (Java files, COBOL programs, etc.)
- A static analysis of compiled code can be performed for certain languages (*.class* files in Java, *.dll* files in C#, etc.)
- A dynamic analysis of code can be performed on certain languages.

Unrecognized files

By default, only files that are recognized by a language plugin are loaded into the project during analysis. For example if your SonarQube instance has the Java and JavaScript plugins on board, all *.java* and *.js* files will be loaded, but *.xml* files will be ignored. However, it is possible to import all text files in the analysis encoding in a project by setting **Settings > Exclusions > Files >** *Import unknown files* to true.

Scope of Analysis: Which Files, and What's Saved

There are 2 different paradigms for SonarQube analysis. You switch among the two modes using the `sonar.analysis.mode` analysis parameter with one of these 2 values:

- **`publish`** - this is the default. This mode analyzes everything that's analyze-able for the languages in question and pushes the results to the server for processing.
- **`issues`** `(previously called` **`preview`**`)` - is typically used to determine whether code changes are good enough to move forward with, E.G. merge into the Git master. Used for example to perform pull request analysis. This is not intended to be used by developers. They should prefer using SonarLint.

Publish Mode

Publish mode performs a full analysis on the entire code base and sends it to the server, which will process it and save the results to the database. Assuming code changes, you will ideally analyze once a day in this mode - typically over night when all changes have been made for the day. Use this mode to update the central server and keep the team at large abreast of your current code quality.

If you're using continuous integration, you don't want to add full analysis to your CI job. Doing so would cause needless churn in the SonarQube metrics and make the "Since previous analysis" differential nearly useless for most people. Instead, you'll want to set up a separate analysis job that polls your SCM on a nightly basis.

If you're *not* using continuous integration, but only building intermittently or at most daily, then by all means include SonarQube analysis in your regular build job.

Preview Mode

Preview mode performs a full analysis on the entire code base and *does not* send the results to the database. Typically this mode is used when you want a quick feedback about the issues that you might have introduced before pushing your modifications to the main code base. See all the possibilities in the "Pre-commit Check" documentation.

During Analysis

During analysis, data is requested from the server, the files provided to the analysis are analyzed, and the resulting data is sent back to the server at the end in the form of a report, which is then analyzed asynchronously server-side.

Analysis reports are queued, and processed sequentially, so it is quite possible that for a brief period after your analysis log shows completion, the updated values are not visible in your SonarQube project. However, you will be able to tell what's going on because an icon will be added next to the project name. Mouse over it for more detail (and links if you're logged in with the proper permissions.)

The icon goes away once processing is complete, but if analysis report processing fails for some reason, the icon will change:

For more detail on analysis report processing, see Background Tasks.

Running Analysis

First, you should install the plugin(s) for the language(s) of the project to be analyzed, either by a direct download or through the update center.

Then, you need to choose an analysis method. The following are available:

- SonarQube Scanner: Launch analysis from the command line
- SonarQube Scanner for MSBuild: Launch analysis of .Net projects
- SonarQube Scanner for Ant: Launch analysis from Ant
- SonarQube Scanner for Maven: Launch analysis from Maven with minimal configuration
- SonarQube Scanner for Gradle: Launch Gradle analysis
- SonarQube Scanner For Jenkins: Launch analysis from Jenkins

Note that we do not recommend running an antivirus on the machine where a SonarQube analysis runs, it could result in unpredictable behavior.

For more information

See also:

- Analysis Parameters
- Project Examples