Issues Report Plugin

> **Deprecated**
>
> This plugin is deprecated since SonarQube 5.1. See SonarLint for in-IDE support instead.

Description / Features

The Issues Report plugin provides the ability to generate HTML report on issues.

The main use case is for developers to check the code they have added or changed before pushing it back to the SCM.

Here's an example of HTML report (you can download the full report here):

Issues can then be filtered by severity, new issues only, etc.

> **SonarQube Eclipse Plugin**
>
> Note that while the Issues Report plugin supports every language, there is also support for certain languages within the IDE. See SonarLint for Eclipse, IntelliJ, and VisualStudio.

Installation

1. Install the plugin through the Marketplace or download it into the *SONARQUBE_HOME/extensions/plugins* directory
2. Restart the SonarQube server

Usage

As the main use case is for developers to check the code they have added or changed before pushing it back to the SCM, the Issues Report plugin is usually used in preview mode. This is this usage that is detailed below.

Install your favorite scanner (SonarQube Scanner, SonarQube Scanner for Maven or SonarQube Scanner for Ant) on your local machine. You only have to set the `sonar.host.url` property to point to your remote SonarQube server (connection settings to the remote database do not have to be provided for a preview analysis as no data is pushed to the database). Note that you don't need to install any SonarQube server on your local machine.

Copy the configuration file (*sonar-project.properties*, *pom.xml*, etc.) that is used to analyze the project on the remote server to your local machine. Make sure that the `sonar.sources` property refers to the directory containing the source code on your local machine (or update it accordingly). The tree structure of the source code on your local machine must match the tree structure that has been remotely analyzed by SonarQube.

To get an HTML report, set the `sonar.issuesReport.html.enable` property to `true`. To define its location, set the `sonar.issuesReport.html.location` property to an absolute or relative path to the destination folder for the HTML report. The default value is *.sonar/issues-report/* for the SonarQube Runner and Ant, and *target/sonar/issues-report/* for Maven. By default 2 html reports are generated:

- The full report (default name is issues-report.html)
- The light report (default name is issues-report-light.html) that will only contains new issues.

The light report is useful when working on legacy projects with a lot of many issues, since the full report may be hard to display in your web browser. You can skip full report generation using property `sonar.issuesReport.lightModeOnly`.

You can also configure the filename of the generated html reports using property `sonar.issuesReport.html.name`.

To display a short report in the console, set the `sonar.issuesReport.console.enable` property to true:

Finally, run a preview analysis that generates an HTML report:

```
# Since SonarQube 4.0
sonar-runner -Dsonar.analysis.mode=preview -Dsonar.issuesReport.html.enable=true

# Prior to SonarQube 4.0
sonar-runner -Dsonar.dryRun=true -Dsonar.issuesReport.html.enable=true
```

Even more useful, you can limit the scope of the analysis to the files that have been recently created or modified (i.e. incremental mode):

```
# Since SonarQube 4.0
sonar-runner -Dsonar.analysis.mode=incremental -Dsonar.issuesReport.html.enable=true

# Prior to SonarQube 4.0, the files have to be manually listed
sonar-runner -Dsonar.dryRun=true -Dsonar.issuesReport.html.enable=true -Dsonar.inclusions=myCobolProgram.cbl,...
```