**SonarQube 3.6 代码质量管理实战**

SonarQube 是一个开源的代码质量管理系统，它可以通过使用插件机制与 eclipse 和 JIRA 等其他外部工具集成，从而实现了对代码的质量的全面自动化分析和管理。本文首先讨论了SonarQube 3.6 的基本工作原理与安装流程；其次分析了采用 SonarQube 3.6 的三种不同方法来对不同语言和架构的项目进行代码质量管理的基本流程；最后对笔者在实际使用过程中遇到的问题进行了总结，并给出了相应的解决方案。

**概述**

随着 IT 行业中软件产品的推陈出新，客户对于软件产品的要求也越来越高，因此如何高质量的管理软件代码，及时地对代码质量进行分析并给出合理的解决方案就成为了当下必须要解决的一个问题。与当今众多的代码质量管理工具相比，SonarQube 更具有特色和竞争力，其优势主要体现为：它是一个开源的代码质量管理系统，支持 25+ 种语言，可以通过使用插件机制与 eclipse 和 JIRA 等其他外部工具集成，从而实现了对代码的质量的全面自动化分析和管理。

本文首先讨论了 SonarQube3.6 的基本工作原理与安装流程；其次分析了使采用 SonarQube3.6 的三种不同方法来对不同语言和架构的项目进行代码质量管理的基本流程；最后对自己在实际使用过程中遇到的问题进行了总结，并给出了相应的解决方案。

本文主要讨论如下几个方面内容：

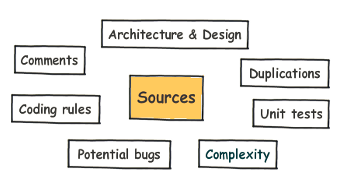
* SonarQube 的工作原理介绍
* SonarQube3.6 的安装
* SonarQube3.6 代码质量分析
* 代码质量分析结果解析
* SonarQube3.6 实际使用中遇到的问题及解决方案

**SonarQube 的工作原理介绍**

SonarQube 并不是简单地将各种质量检测工具的结果（例如 FindBugs，PMD 等）直接展现给客户，而是通过不同的插件算法来对这些结果进行再加工，最终以量化的方式来衡量代码质量，从而方便地对不同规模和种类的工程进行相应的代码质量管理。

SonarQube 在进行代码质量管理时，会从图 1 所示的七个纬度来分析项目的质量。

**图 1. 质量纬度**

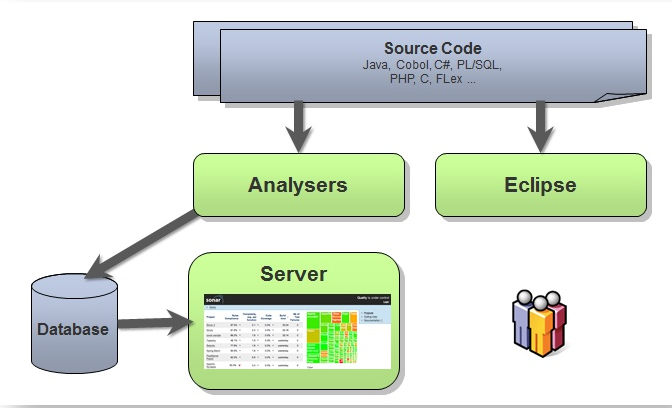


SonarQube 可以支持 25+ 种编程语言，针对不同的编程语言其所提供的分析方式也有所不同：

1. 对于所有支持的编程语言，SonarQube 都提供源了代码的静态分析功能；
2. 对于某些特定的编程语言，SonarQube 提供了对编译后代码的静态分析功能，比如 java 中的 class file 和 jar 和 C# 中的 dll file 等；
3. 对于某些特定的编程语言，SonarQube 还可以提供对于代码的动态分析功能，比如 java 和 C# 中的单元测试的执行等。

SonarQube 平台的整体架构如图 2 所示：

**图 2. SonarQube 的服务架构**



远程客户机可以通过各种不同的分析机制，从而将被分析的项目代码上传到 SonarQube server 并进行代码质量的管理和分析，SonarQube 还会通过 Web API 将分析的结果以可视化、可度量的方式展示给用户。

**SonarQube3.6 的安装**

***软件及硬件的要求***

SonarQube3.6 的安装通常需要满足一定的软硬件条件，具体要求如下所示：

1. Server 要求

Web server 最少需要 500MB 的内存空间，推荐内存空间大小 2GB。Sonar 在进行代码质量分析时，通常大约每 1 KLOC 需要存储 350KB 左右的数据，所以要尽量为 SonarQube3.6 的 web server 提供大的内存。

2. Database 要求

尽管 SonarQube3.6 本身自带嵌入的 Derby 数据库，但是由于 Derby 比较简单，所以在生产环境中强烈推荐安装相应的企业版数据库，SonarQube3.6 支持的数据库包括： MySQL 5.x+、Oracle10g+、PostgreSQL 9.x 和 MS SQLServer 2005 and 2008，推荐使用 MySQL。目前 SonarQube3.6 还不支持 DB2，但在后续版本中可能会提供对 DB2 的支持。

3. Browser 要求

SonarQube3.6 支持大多数的浏览器，包括 Firefox、Internet Explorer 7.x and 8.x and chromed 等，推荐使用 Firefox。

***安装步骤***

SonarQube3.6 的安装包可以从 SonarQube 的官方网站 (http://www.sonarqube.org/) 下载。总的来说，它的安装过程可以分为以下六个步骤：

1. 获取公司的法律授权

本步骤主要取决于公司的规定，如果公司对于在本公司机器上安装的软件有法律要求，那么尽管 SonarQube 是开源的，但是也还是要获得公司相关法律部门的授权，然后再进行安装。

2. 创建数据库

创建数据库时，按照相应数据库的安装说明进行安装即可，安装成功后，需要手动创建一个名为“sonarqube”的用户和一个 empty schema --“sonarqube”，并赋予该用户在相应 schema 上的 create、update 和 delete 对象的权限，同时数据库的字符集设定为“UTF-8”，示例的 sql 语句如下：

[点击查看代码清单](http://www.ibm.com/developerworks/cn/opensource/os-sonarqube/index.html#N100AF)

CREATE DATABASE sonarqube CHARACTER SET utf8 COLLATE utf8\_general\_ci; CREATE USER 'sonarqube 'IDENTIFIED BY 'sonarqube' ; GRANT ALL ON sonarqube.\* TO 'sonarqube'@'%' IDENTIFIED BY 'sonarqube'; GRANT ALL ON sonarqube.\* TO 'sonarqube'@'localhost' IDENTIFIED BY 'sonarqube'; FLUSH PRIVILEGES;

3. 安装 Sonar 的服务器

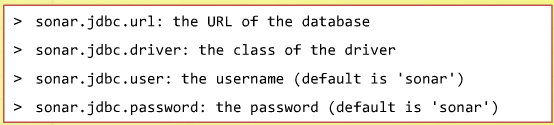
将所下载的 SonarQube3.6 压缩包解压即可，注意要与所在的操作系统平台匹配。

4. 配置数据库

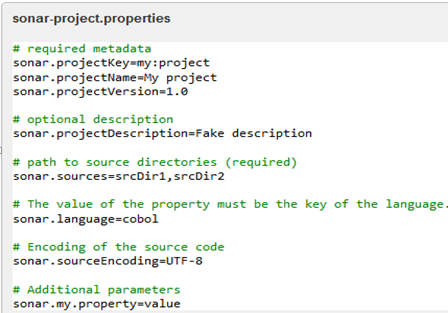
如果没有使用 sonarQube3.6 自带的 derby 数据库，则此时需要编辑 sonarqube3.6 解压目录 \conf\sonar.properties 文件，对数据库的访问参数进行配置。

数据库配置项和配置示例如图 3 和图 4 所示：

**图 3. Sonar.properties 文件中的数据库配置项**



**图 4. Sonar.properties 文件中的 MySQL 数据库配置**



如果是 Oracle 数据库，则需要将 JDBC 的 driver 拷贝到 sonarqube3.6 解压目录 \extensions\jdbc-driver\[your database]。此外，对于其他所支持的数据库的 driver，sonarqube 也已经提供。

5. 启动 SonarQube3.6 服务器

启动 SonarQube3.6 服务器的方法取决于所使用的安装平台和 webserver。

常见的启动方式有图 5 所示的几种：

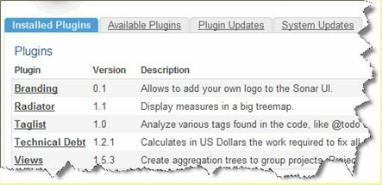
**图 5. SonarQube 启动方式**



6. 插件下载与安装

SonarQube3.6 服务器成功启动后，登陆默认的 webserver:◊ IP:9000/ ，其中 IP 为 sonarqube3.6 安装的服务器网络地址，用户名和密码为默认的 admin/admin。登陆成功后点击 configuration 菜单按钮，然后选择 update center 进行插件的下载与安装，如图 6 所示：

**图 6. Plugin 下载与安装**



注意：如果 sonarqube server 所在的机器不能访问外网，那么此时就需要先在另一台能够访问外网的机器上下载所需的所有插件，然后将下载的插件拷贝到 sonarqube server 所在机器的 sonarqube3.6 解压目录 \extensions\plugins 目录内，然后重启 Sonarqube3.6。下载 Sonarqube 插件的官方地址为：<http://docs.codehaus.org/display/SONAR/Plugin+Library/>，下载时需注意插件板本与 sonarqube3.6 的兼容问题。

**SonarQube3.6 代码质量分析**

当使用 SonarQube3.6 对某一个项目进行代码质量管理时，通常可以采用三种不同的方法来发起代码分析，分别为 Analyzing with SonarQube Runner、，Analyzing with SonarQube Ant Task 和 Analyzing with Maven 方法。这三种方法的主要区别是 , 它们分别适用于不同架构和组织形式的项目，并且其相应的配置文件的写法也不尽相同。

下面将分别介绍这三种方式。

***Analyzing with SonarQube Runner***

通常我们会推荐使用 SonarQube Runner 这种方式，因为该方法适用于所有不同架构的项目，包括没有使用任何源代码管理工具的项目形式，以及使用各种不同代码管理工具（SVN、Git、ClearCase 等）和编译工具（ant，maven）的项目形式，它都能够适用。在使用该方式进行分析时步骤如下：

1. 首先需要下载与 SonarQube3.6 Server 兼容的 sonar-runner 客户端并对其进行解压；下载地址为 <http://repo1.maven.org/maven2/org/codehaus/sonar/runner/sonar-runner-dist/2.3/sonar-runner-dist-2.3.zip>

2. 在被分析项目代码的根目录下，创建并配置 sonar-project.properties，内容如图 7 所示：

**图 7. Sonar-project.properties 文件示例**

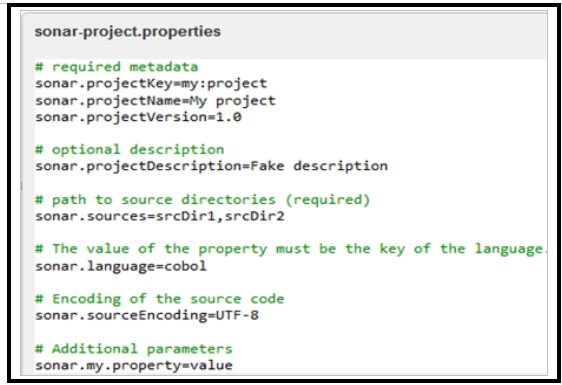
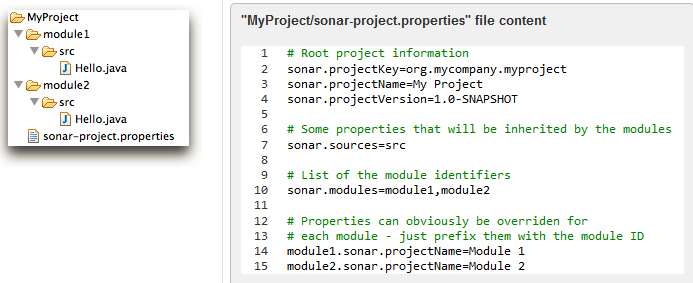


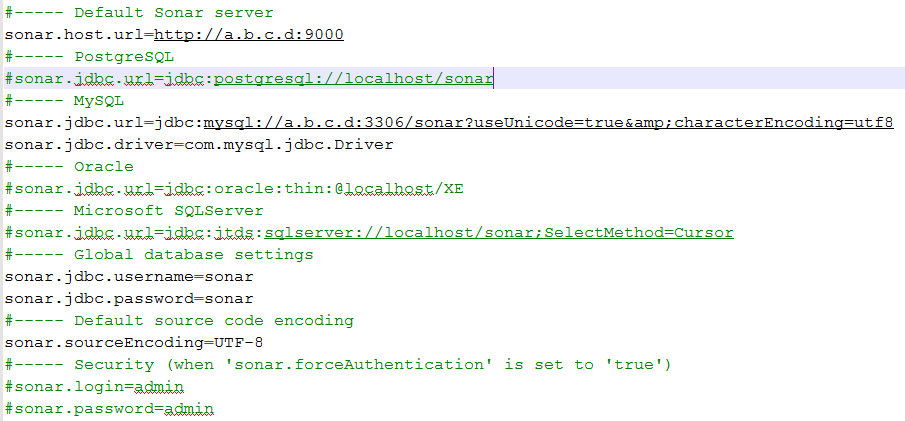
图 7 中的 sonar.projectName 为项目在 sonar server 上显示的名称，其值要求唯一。sonar.source 是用于指定待分析的代码位置的。sonar.language 用于指定待分析的项目代码语言。如果项目包含多个模块，则该配置文件应按照图 8 所示编辑：

**图 8. 多模块项目的 Sonar-project.properties 示例**



3. 在解压的 soanr-runner 客户端的 sonar-runner\config 目录下，编辑 sonar-runner.properties 文件 , 示例如图 9 所示：

**图 9. Sonar-runner.properties 示例**



请注意，图 7、图 8 以及图 9 中的相关配置项 ( 如 host.url 和 jdbc.url 等 ) 都要与图 4 的 sonar.properties 文件中的相关配置项保持一致。

4. 打开命令行，在被分析的项目目录下 , 运行 sonar-runner 命理分析项目代码。

***Analyzing with SonarQube Ant Task***

本方法主要适用于使用 Ant 工具进行构建、编译、打包和测试的项目，使用 Ant Task 分析代码时，步骤如下：

1. 下载 SonarQube Ant Task 并安装，下载地址如下： <http://repository.codehaus.org/org/codehaus/sonar-plugins/sonar-ant-task/2.1/sonar-ant-task-2.1.jar>

2. 在被分析项目代码的根目录下，创建并配置 build.xml 文件。示例如图 10 所示：

**图 10. Build.xml 配置示例**



Build.xml 中的相关配置项 ( 如 jdbc.username 和 jdbc.url 等 ) 要与图 4 所示的 sonar.properties 文件中的相关配置项保持一致；同时可以看出，在 build.xml 文件中既包含了被分析项目代码的相关信息，也包含了其所连接的对应 SonarQube Server 的信息。

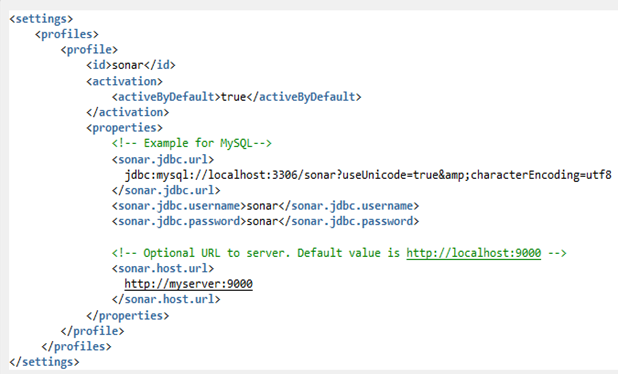
3. 在命令行中，切换到代码根目录，使用命令 “ant sonar”进行代码分析。

***Analyzing with Maven***

当项目使用 Maven 工具管理和构建时，则该项目就适用于本方法。使用 Maven 分析代码时，步骤如下：

1. 下载、安装并配置 Maven，配置文件为 setting.xml，配置如图 11 所示：

**图 11. Setting.xml 配置示例**



Setting.xml 中的相关配置项 ( 如 host.url 和 jdbc.url 等 ) 要与图 4 所示的 sonar.properties 文件中的相关配置项保持一致，主要用于配置其所连接的 SonarQube 服务器的信息。

2. 在待分析项目代码的根目录下，创建并配置 prom.xml 文件，示例如图 12 所示，该配置文件用于设定带分析代码的相关信息，包括代码目录等。

**图 12. Prom.xml配置示例**



3. 打开命令行窗口，切换到项目的根目录，然后执行“mvn sonar:sonar”来分析项目代码的质量。

***小结***

总的来说这三种方式有不同的针对性，具体使用哪一种方式要根据项目的实际情况来灵活的判断。

**SonarQube3.6 代码质量分析结果解析**

当项目代码被成功分析完成后，SonarQube Server 会将分析的结果存储在相应的数据库中，并以报告的形式将分析结果在 webserver 中展示给用户。

下面将以一个真实项目代码的分析结果为例，对其各项分析指标进行简要介绍。

**图 13.分析结果示例-01**

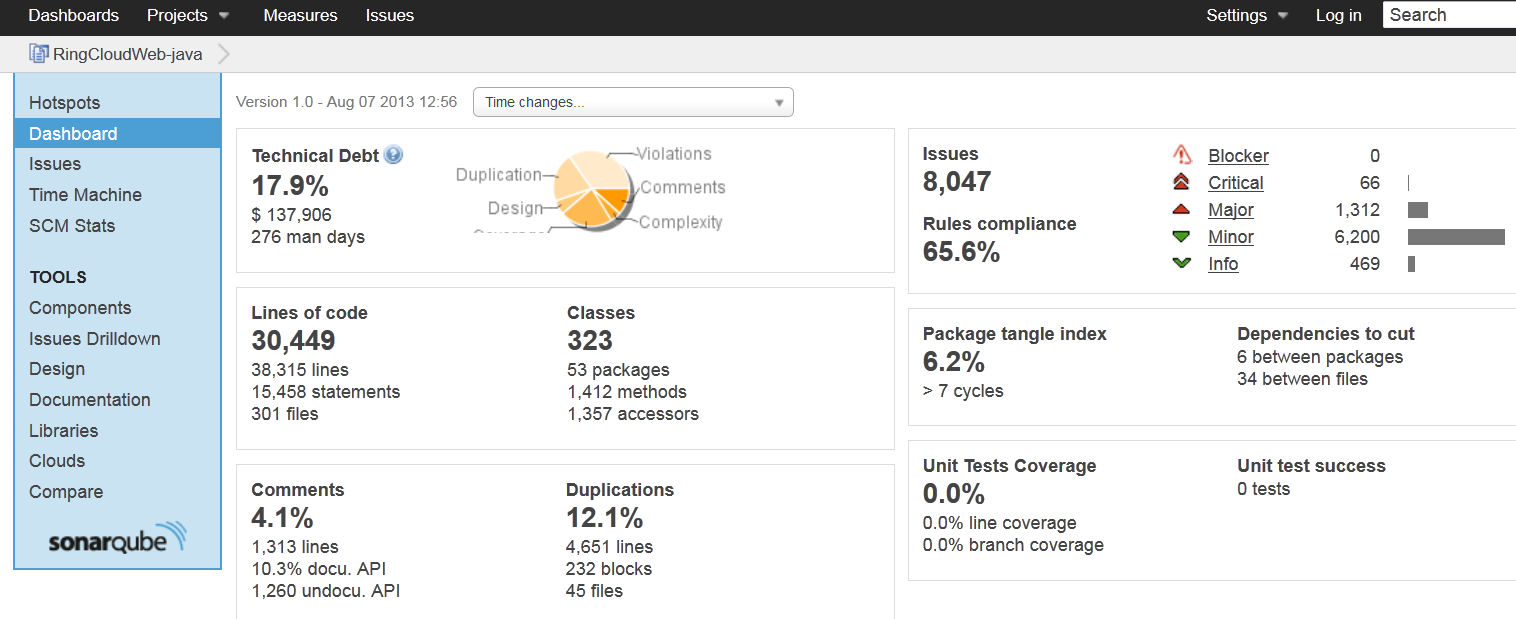


图 13 为 SonarQube3.6 的 Project Dashboard，它是查看任意项目分析结果的入口，展示了该项目各项指标总的分析数据结果，其中包括项目的重复率，注释的比例，单元测试覆盖率以及 Issue 的比例等。通过 Dashboard 就可以对项目总体的质量情况有一个全局性的认识。

选择图 13 中的各项分析指标，就可以具体的去查看该项目在这一指标上的详细情况。

下面以重复率和 Issue 为例，对其分析结果进行解析。

1. 选择图 13 中的重复率指标，将会得到图 14 所示的关于重复代码的详细分析结果：

**图 14. 分析结果示例-02**

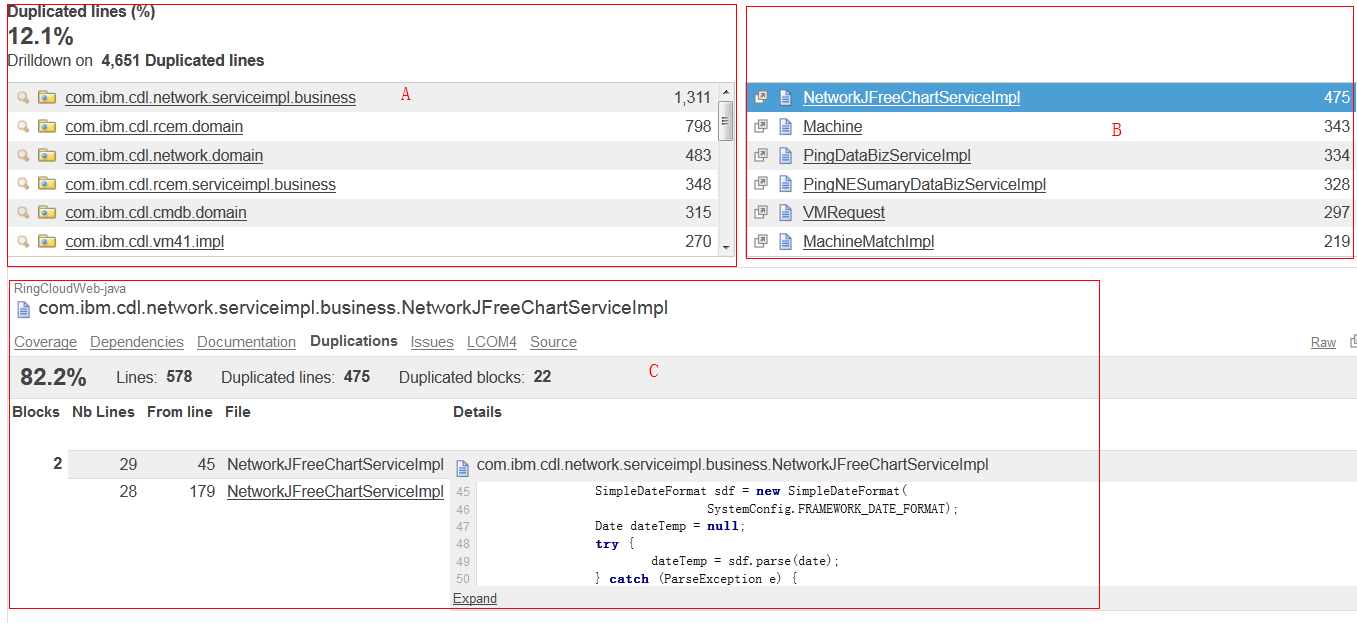


图 14 中的区域 A 展示了整个项目中产生代码重复的所有模块及其重复的代码行数。选择区域 A 中的任一某块，在区域 B 中则将显示该模块下产生代码重复的所有文件；选择区域 B 中的某一文件，则在下方区域 C 中会显示项目中与该文件产生代码重复的所有文件，同时还会显示该文件相应的重复代码的内容。这样的分析结果，可以有效地指导项目组成员进行项目代码优化并降低重复率。

2. 选择图 13 中的 Issues 指标，会得到图 15 所示的关于项目 Issue 的详细分析结果：

**图 15. 分析结果示例-03**

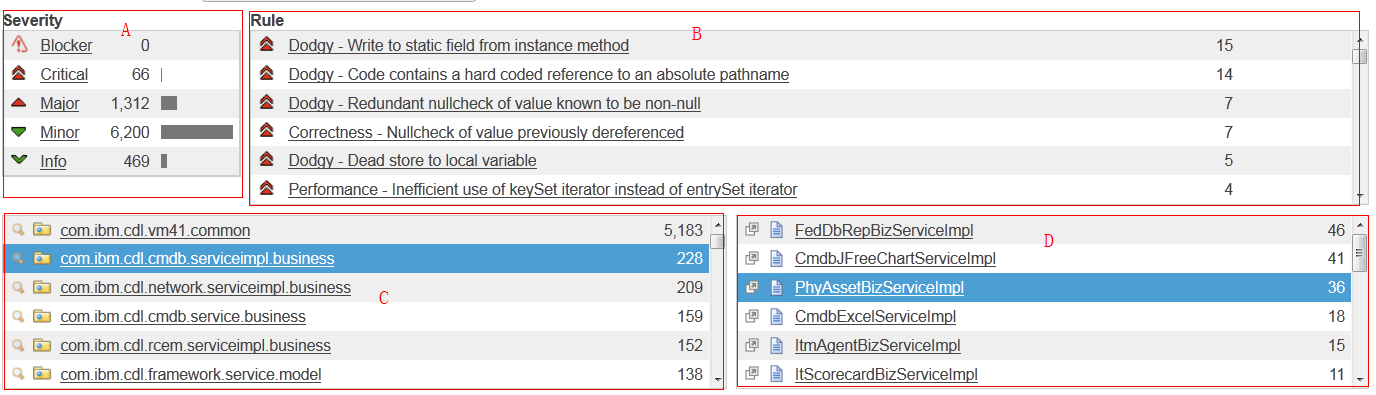


图 15 中的区域 A 展示了整个项目中不同严重程度的 Issues 数量；区域 B 则显示了项目代码具体所违反的各种编码规则以及违反规则的次数；点击区域 B 中相应的规则，在区域 C 中就会显示该项目中违反相应规则的所有模块及其违反规则的次数；选择区域 C 中的某一模块，在区域 D 中就会显示该模块内违反规则的所有文件；如果要查看代码文件到底如何违反了某一规则，只需在区域 D 中点击相应的文件，则会显示如图 16 所示的详细情况。

在图 16 中，可以看到违反相应规则的具体代码，用红色标出；点击所违反的具体规则，SonarQube 则会解释代码违反规则的具体原因，并给出相应的解决方案以供参考，这对于项目组成员定位问题和解决问题都有很大的帮助。

**图 16. 分析结果示例-04**



其实 SonarQube 还提供了很多其他方面的质量维度分析，如复杂度，测试覆盖率等，这里就不再讨论其具体的参数细节了。本部分旨在从整体上对代码的质量分析结果进行概括，并介绍如何查看相应的质量分析结果，而关于具体如何利用各个分析参数的值来维护、改善代码质量，会在后续系列中与大家分享。

**SonarQube3.6 实际使用中遇到的问题及解决方案**

在使用 SonarQube3.6 分析代码质量时，通常会遇到以下三个问题，在实际使用中要注意：

1. find-bug.jar 版本问题

由于在 sonarQube3.3.1 之前， 其所默认使用的 find-bug 的版本为 2.0.0，该版本会导致代码分析时出现时间字符串的格式解析错误，因此在安装 sonarQube3.6 后，务必要确保其所安装的插件 find-bug.jar 的版本为 2.0.1。

2. 认证授权错误

由于在 sonarqube3.4 之前的版本中，连接 sonar server 是不需要 authorized，而在 3.6 版本中则需要 authorized，授权问题可以通过两种方法解决：

* 在 sonar.properties 和 sonar-runner.properties 中同时指定相同的用户名和密码。（sonar.login= sonar.password=）
* 在 sonar.properties 和 sonar-runner.Properties 分别强制设定不需要进行访问授权，即 sonar.forceAuthentication=false

3. 源代码的编码格式错误

由于 find-bug.jar 2.0.1 和 sonarQube3.6 server 的默认编码格式为 GBK，因此必须认真地处理源代码的编码格式（即 sonar.sourceEncoding=UTF-8/GBK/......），避免因编码格式不兼容而导致的问题。

**结束语**

代码质量管理对提高项目质量意义重大。本文介绍了 SonarQube 的工作原理，并从项目实战的角度讲解了使用 SonarQube3.6 进行项目代码质量管理的流程和注意事项。希望能够给使用 SonarQube 的读者提供有价值的参考。