

SonarQube 配置与使用教程

Sonar 简介

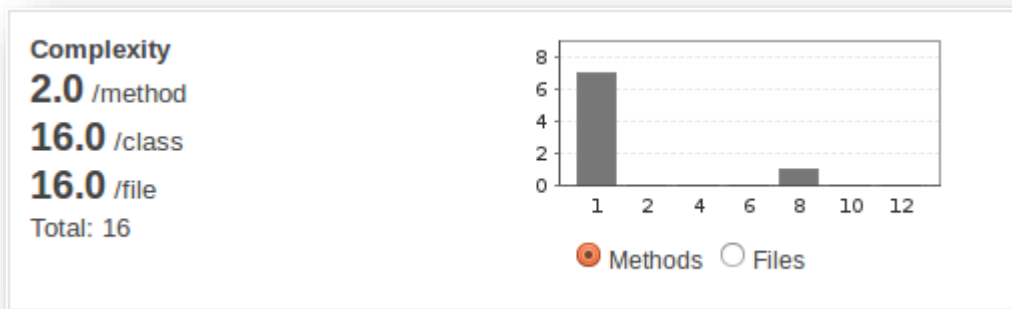
Sonar 是一个用于代码质量管理的开源平台，用于管理源代码的质量，可以从七个维度检测代码质量。

通过插件形式，可以支持包括 java,C#,C/C++,PL/SQL,Cobol,JavaScrip,Groovy 等等二十几种编程语言的代码质量管理与检测。

SonarQube 能带来什么？

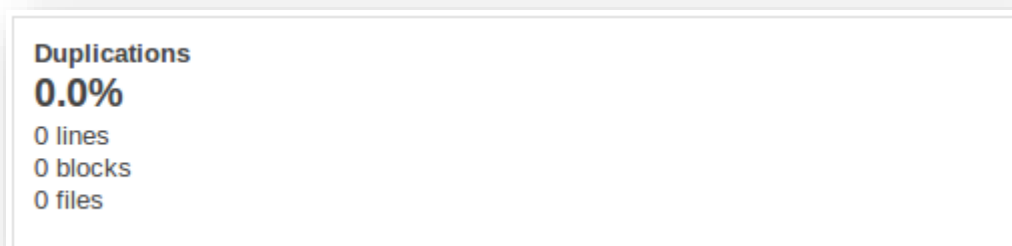
1. 糟糕的复杂度分布

文件、类、方法等，如果复杂度过高将难以改变，这会使得开发人员难以理解它们，且如果没有自动化的单元测试，对于程序中的任何组件的改变都将可能导致需要全面的回归测试。



2. 重复

显然程序中包含大量复制粘贴的代码是质量低下的，sonar 可以展示源码中重复严重的地方。



3. 缺乏单元测试

sonar 可以很方便地统计并展示单元测试覆盖率。



4. 没有代码标准

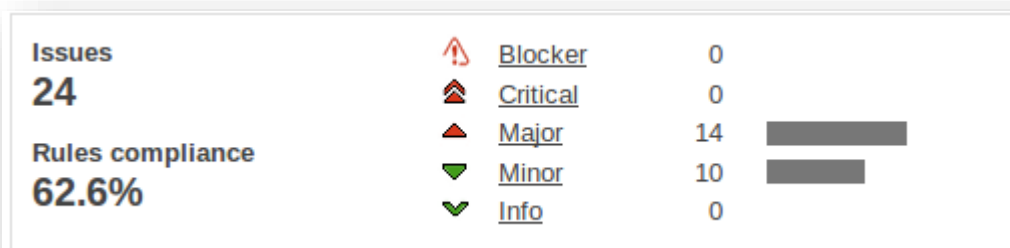
sonar 可以通过 PMD,CheckStyle,Findbugs 等等代码规则检测工具规范代码编写。

5. 没有足够的或者过多的注释

没有注释将使代码可读性变差，特别是当不可避免地出现人员变动时，程序的可读性将大幅下降。而过多的注释又会使得开发人员将精力过多地花费在阅读注释上，亦违背初衷。

6. 潜在的 bug

sonar 可以通过 PMD,CheckStyle,Findbugs 等等代码规则检测工具检测出潜在的 bug。



7. 糟糕的设计（原文 Spaghetti Design，意大利面式设计）

通过 sonar 可以找出循环，展示包与包、类与类之间的相互依赖关系；

可以检测自定义的架构规则；

通过 sonar 可以管理第三方的 jar 包；

可以利用 LCOM4 检测单个任务规则的应用情况；

检测耦合。

关于 Spaghetti Design: <http://docs.codehaus.org/display/SONAR/Spaghetti+Design>。

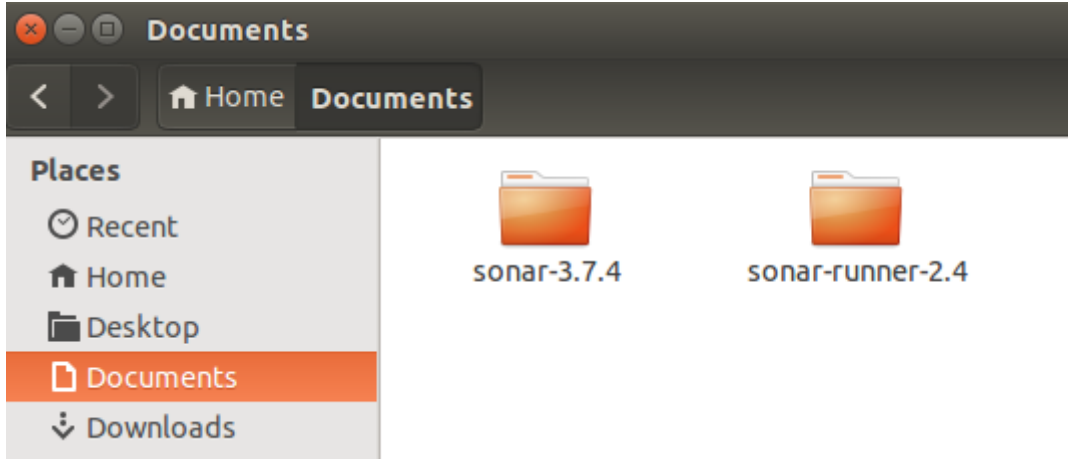
通过 sonar 可以有效检测以上在程序开发过程中的七大问题。

SonarQube 安装

预置条件

1. 已安装 Java 环境
2. 安装 sonar 与 sonar-runner
 - 2.1 将下载的 sonar-3.7.4.zip 包解压至 Linux 某路径如/usr/local,
 - 2.2 将下载的 sonar-runner-dist-2.4.zip 包解压某路径/usr/local;

小编直接解压缩到 Documents 文件夹里面，如下图：



2.3 添加 SONAR_HOME、SONAR_RUNNER_HOME 环境变量，并将 SONAR_RUNNER_HOME 加入 PATH（这步主要是方便大家在 shell 方便进入 sonar 文件夹和进行编译）：

2.3.1 在 shell 里面用 vi 或者 gedit 打开 /etc/profile

```
sudo vi /etc/profile
```

2.3.2 在文件末端添加你解压缩的 sonar 文件夹的路径，如下图：

```
export SONAR_HOME=/home/greenhill/Documents/sonar-3.7.4/bin/linux-x86-32
export SONAR_RUNNER_HOME=/home/greenhill/Documents/sonar-runner-2.4
export PATH=$SONAR_RUNNER_HOME/bin:$PATH
```

注意：小编选择的系统是 linux 32bit，具体的系统配置大家要看看自己机子。

2.3.3 最后保存退出，在 shell 里面键入 `source /etc/profile`，重启系统。

2.4 添加数据库

如果使用 Sonar 默认的数据库 H2，则无需配置，如果需要其他数据库，包括 mysql, Oracle 等，可以自行上网查询。由于我们的是小项目，所以用 Sonar 自带的数据库 H2 完全可以了。

2.5 启动服务

在 shell 里面键入 `cd $SONAR_HOME`，可以直接进入启动目录。在 shell 里面键入

`./sonar.sh start` 启动服务

`./sonar.sh stop` 停止服务

`./sonar.sh restart` 重启服务

```
sonar.sh start
Starting sonar...
Started sonar.
```

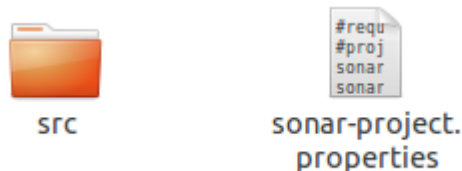
访问 `http://localhost:9000`，如果显示 SonarQube 的项目管理界面，表示安装成功。

使用 SonarQube Runner 分析源码

预置条件

已安装 SonarQube Runner 且环境变量已配置，即 sonar-runner 命令可在任意目录下执行

1.在项目源码的根目录下创建 sonar-project.properties 配置文件，内容如下：

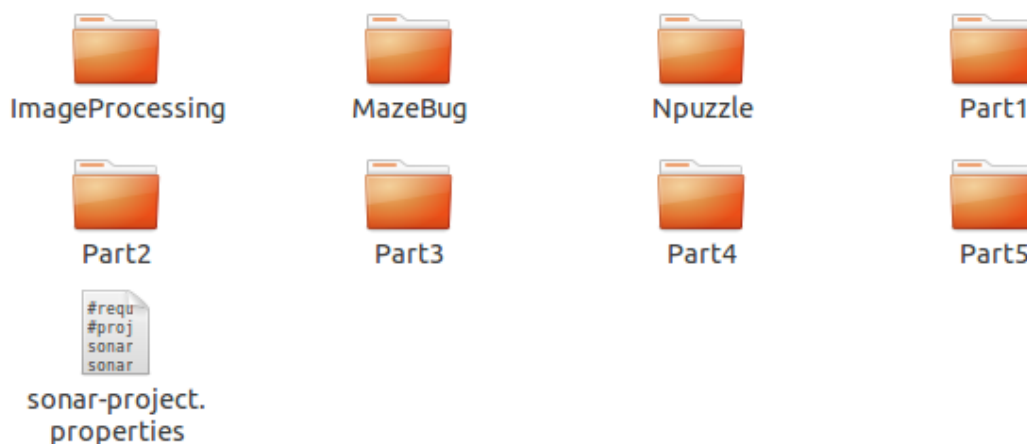


```
1 #required metadata
2 #projectKey项目的唯一标识，不能重复
3 sonar.projectKey=10389179
4 sonar.projectName=10389179
5 sonar.projectVersion=1.0
6 sonar.sourceEncoding=UTF-8
7 sonar.modules=java-module
8
9 # Java module
10 java-module.sonar.projectName=Java Module
11 java-module.sonar.language=java
12 # .表示projectBaseDir指定的目录
13 java-module.sonar.sources=.
14 java-module.sonar.projectBaseDir=src
```

其中 projectKey 是项目的唯一标识，不能重复；

大家要修改的内容包括 sonar.projectKey， sonar.projectName，
java-module.sonar.projectBaseDir 三项；

注意：每一 Part 都需要大家建立一个单独的项目进行测试，方便大家对每一 Part 的代码进行修改，以及 TA 检查。如下图：



到时候只需要修改 sonar.projectKey， sonar.projectName，
java-module.sonar.projectBaseDir 三项为对应文件夹名字即可。

```
1 #required metadata
2 #projectKey项目的唯一标识, 不能重复
3 sonar.projectKey=Part1
4 sonar.projectName=Part1
5 sonar.projectVersion=1.0
6 sonar.sourceEncoding=UTF-8
7 sonar.modules=java-module
8
9 # Java module
10 java-module.sonar.projectName=Java Module
11 java-module.sonar.language=java
12 # .表示projectBaseDir指定的目录
13 java-module.sonar.sources=.
14 java-module.sonar.projectBaseDir=Part1
```

2. 编写好 sonar-project.properties 文件后, 在 shell 里面进入含有 sonar-project.properties 文件的目录, 输入 sonar-runner, 运行测试;

```
INFO: -----
INFO: EXECUTION SUCCESS
INFO: -----
Total time: 1:07.802s
Final Memory: 11M/104M
INFO: -----
```

出现以上信息时, 证明运行成功, 打开 <http://localhost:9000> 查看对应项目的情况即可。

注意: 每次使用完 Sonar, 记得关闭, 进入启动目录, `./sonar.sh stop`

如果需要把 SonarQube 集成到 Eclipse, 可以参考官方文档进行配置, 这里不详述。

附:

不同参数的意思:

<http://docs.codehaus.org/display/SONAR/Analysis+Parameters>

不同项目的源码分析示例下载:

<https://github.com/SonarSource/sonar-examples/zipball/master>

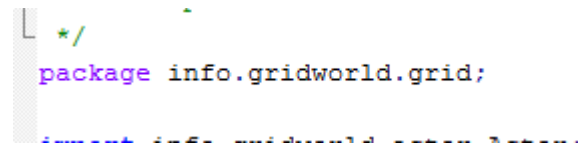
sonarQube 官网地址: <http://www.sonarqube.org/>

sonarQube 官方文档地址: <http://docs.codehaus.org/display/SONAR/Documentation>

Q&A:

Q1. 我用 Sonar 运行了项目，可是一下子就退出了，而且查看 localhost:9000 发现没有项目的结果生成。

A1: 这个问题要看看你的 java 文件里面是不是有 package 存在，如果有 package，则要建立相应的文件夹，然后把文件放入该文件夹，重新运行即可。如下图所示：



```
package info.gridworld.grid;
```

图 1 java 文件中有 package



```
Part5 ▶ info ▶ gridworld ▶ grid
```

图 2 在 Part5 文件夹下建立对应文件夹，并放入文件即可