SonarQube自定义规则开发

SonarQube 自定义规则开发

如何开发自己的代码规则
本文只是初步试了试自定义规则，可能有疏漏和错误，请大家仅做参考

以下内容需要了解 Java 和 Maven

开发自定义规则

开发步骤见，这是官方提供的 sonar-java 下面的指导文档，简述了怎么开发一个 Java 自定义规则。

文章的开始，给了一个模板地址，可以看到这个项目下还有别的语言模板。

我们下载这个 Java 自定义规则模板，为 Java 开发一种自定义规则。

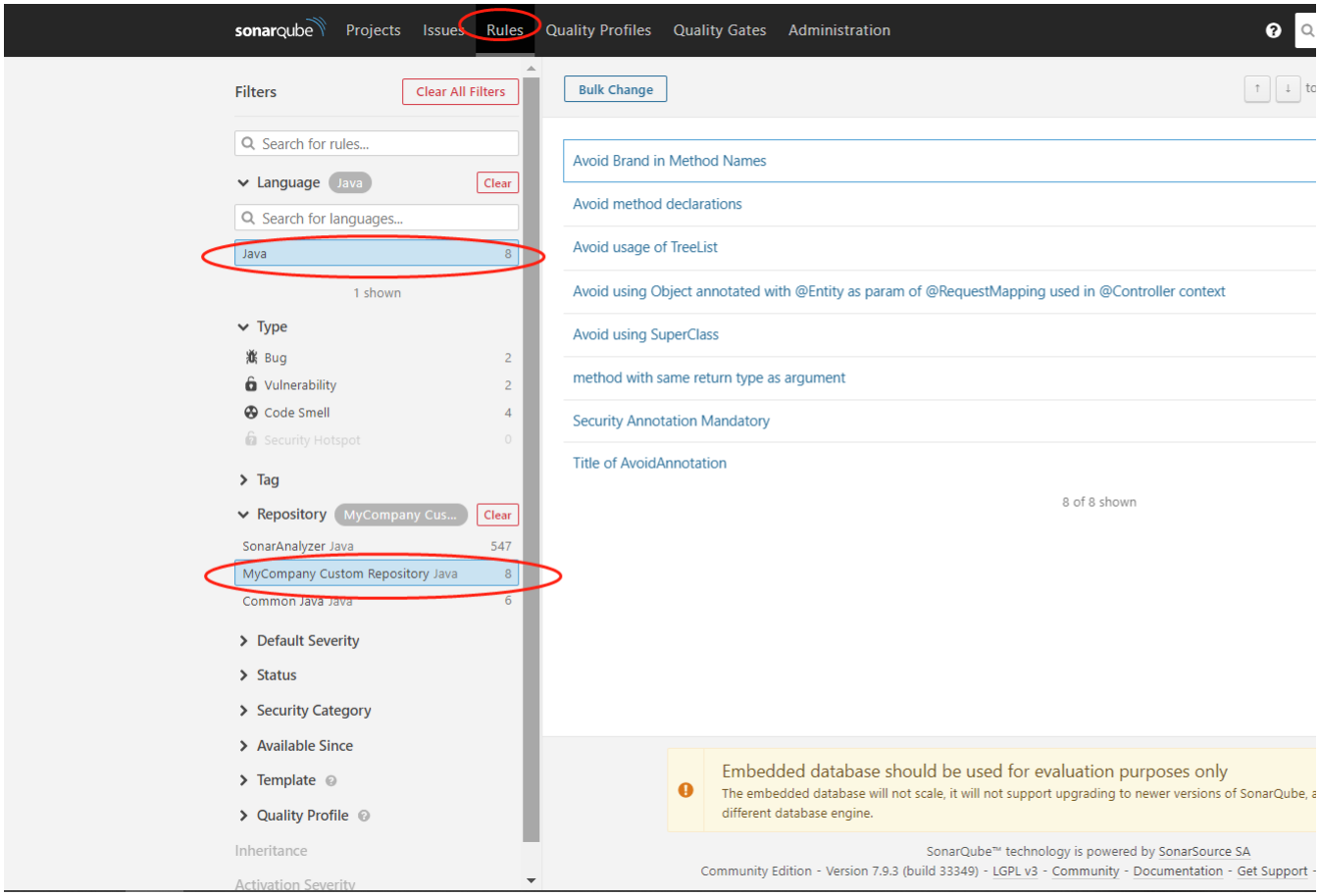
下载项目后编译

mvn clean package

把生成的文件放在 $SONAR_HOME/extensions/plugins 目录

重启 SonarQube

可以看到 Java 下面多了 MyCompany Custom Repository 仓库，下面有新增的规则，这是模板自带的例子。



接下来添加一个自定义规则，按照文档一步一步走，最终我们添加/修改了以下代码

创建 src/test/files/MyFirstCustomCheck.java 文件，这是被扫描的代码，用于单元测试

```
class MyClass {
  MyClass(MyClass mc) { }

  int     foo1() { return 0; }
  void    foo2(int value) { }
  int     foo3(int value) { return 0; } // Noncompliant
  Object  foo4(int value) { return null; }
  MyClass foo5(MyClass value) {return null; } // Noncompliant

  int     foo6(int value, String name) { return 0; }
  int     foo7(int ... values) { return 0;}
}
```

创建单元测试代码 src/test/java/org/sonar/samples/java/checks/MyFirstCustomCheckTest.java

```
package org.sonar.samples.java.checks;

import org.junit.Test;
import org.sonar.java.checks.verifier.JavaCheckVerifier;

public class MyFirstCustomCheckTest {

  @Test
  public void test() {
    JavaCheckVerifier.verify("src/test/files/MyFirstCustomCheck.java", new MyFirstCustomCheck());
  }

}
```

创建检查规则 src/main/java/org/sonar/samples/java/checks/MyFirstCustomCheck.java

```
package org.sonar.samples.java.checks;

import com.google.common.collect.ImmutableList;

import org.sonar.check.Priority;
import org.sonar.check.Rule;
import org.sonar.plugins.java.api.IssuableSubscriptionVisitor;
import org.sonar.plugins.java.api.semantic.Symbol.MethodSymbol;
import org.sonar.plugins.java.api.semantic.Type;
import org.sonar.plugins.java.api.tree.MethodTree;
import org.sonar.plugins.java.api.tree.Tree;
import org.sonar.plugins.java.api.tree.Tree.Kind;

import java.util.List;

@Rule(
        key = "MyFirstCustomCheck",
        name = "Return type and parameter of a method should not be the same",
        description = "For a method having a single parameter, the types of its return value and its parameter should never be the same.",
        priority = Priority.CRITICAL,
        tags = {"bug"})
public class MyFirstCustomCheck extends IssuableSubscriptionVisitor {


    @Override
    public void visitNode(Tree tree) {
        if (method.parameters().size() == 1) {
            MethodSymbol symbol = method.symbol();
            Type firstParameterType = symbol.parameterTypes().get(0);
            Type returnType = symbol.returnType().type();
            if (returnType.is(firstParameterType.fullyQualifiedName())) {
                reportIssue(method.simpleName(), "Never do that! 01!");
            }
        }
    }

    @Override
    public List<Kind> nodesToVisit() {
        return ImmutableList.of(Kind.METHOD);
    }
}
```

将我们自定义的规则添加到规则列表中，修改 src/main/java/org/sonar/samples/java/RulesList.java

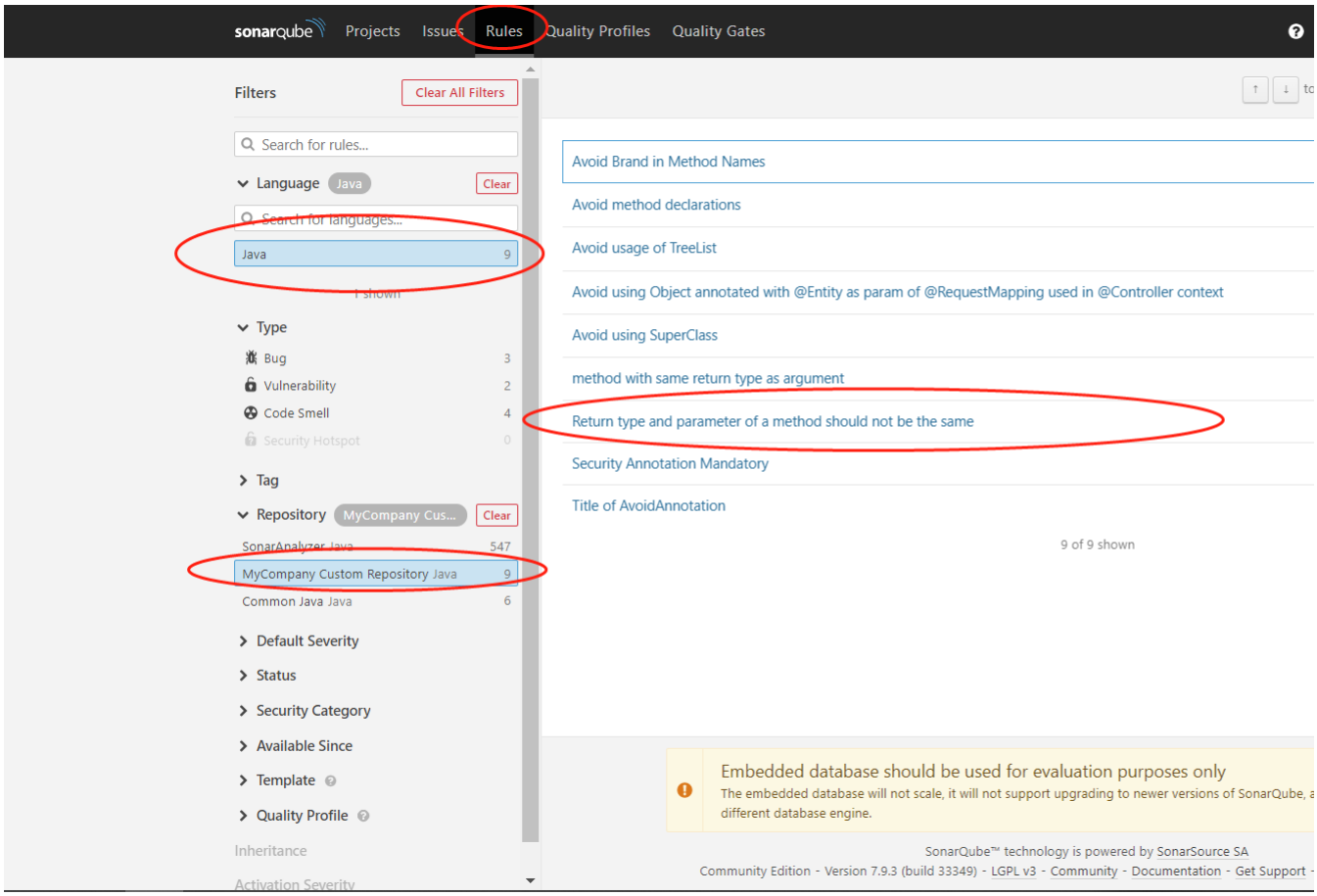
```
  public static List<Class<? extends JavaCheck>> getJavaChecks() {
    return Collections.unmodifiableList(Arrays.asList(
      SpringControllerRequestMappingEntityRule.class,
      AvoidAnnotationRule.class,
      AvoidBrandInMethodNamesRule.class,
      AvoidMethodDeclarationRule.class,
      AvoidSuperClassRule.class,
      AvoidTreeListRule.class,
      MyCustomSubscriptionRule.class,
      SecurityAnnotationMandatoryRule.class,
      MyFirstCustomCheck.class));  # 添加我们的规则
  }
```

再次编译

```
mvn clean package
```

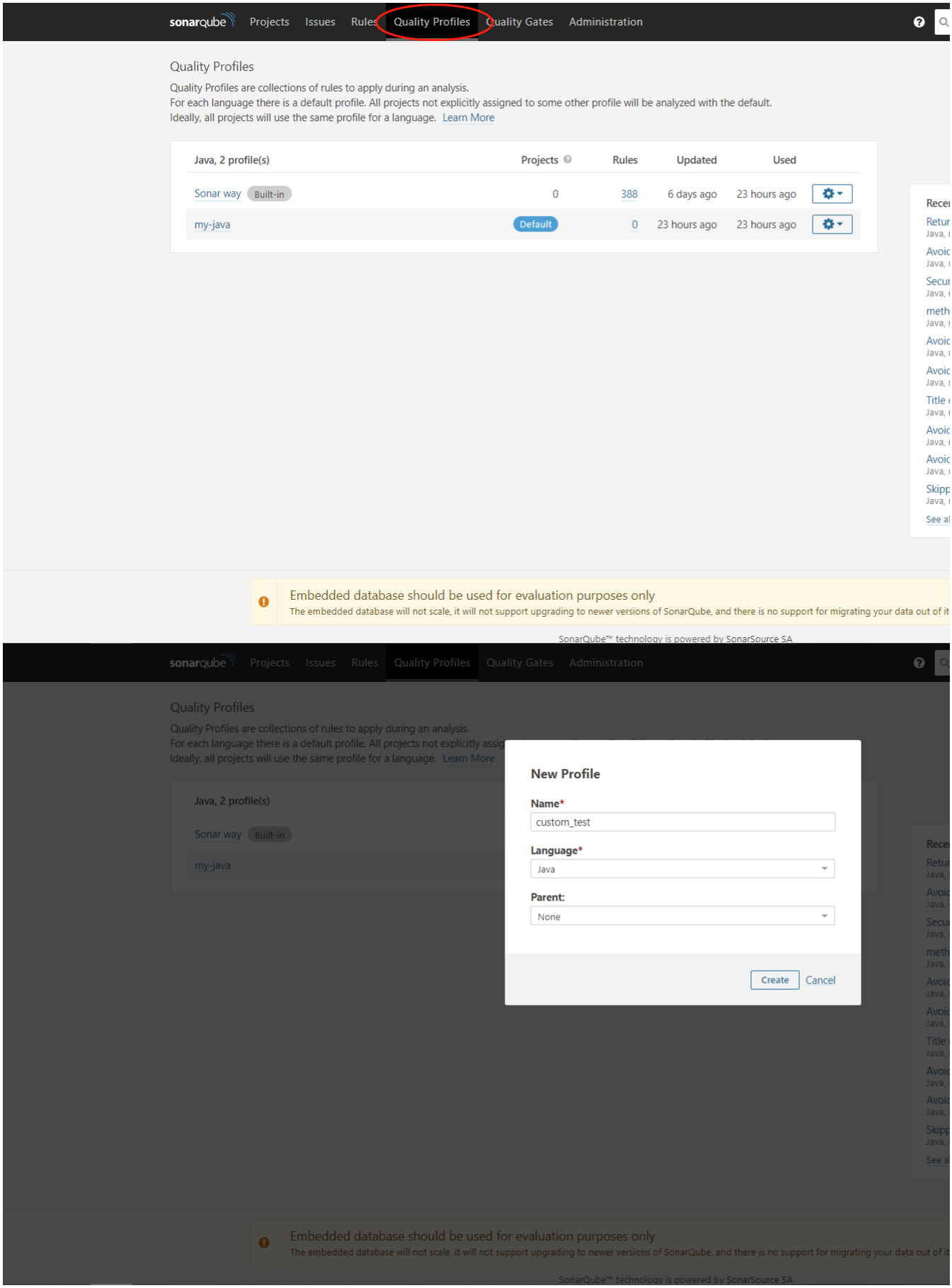
把生成的文件放在 $SONAR_HOME/extensions/plugins 目录

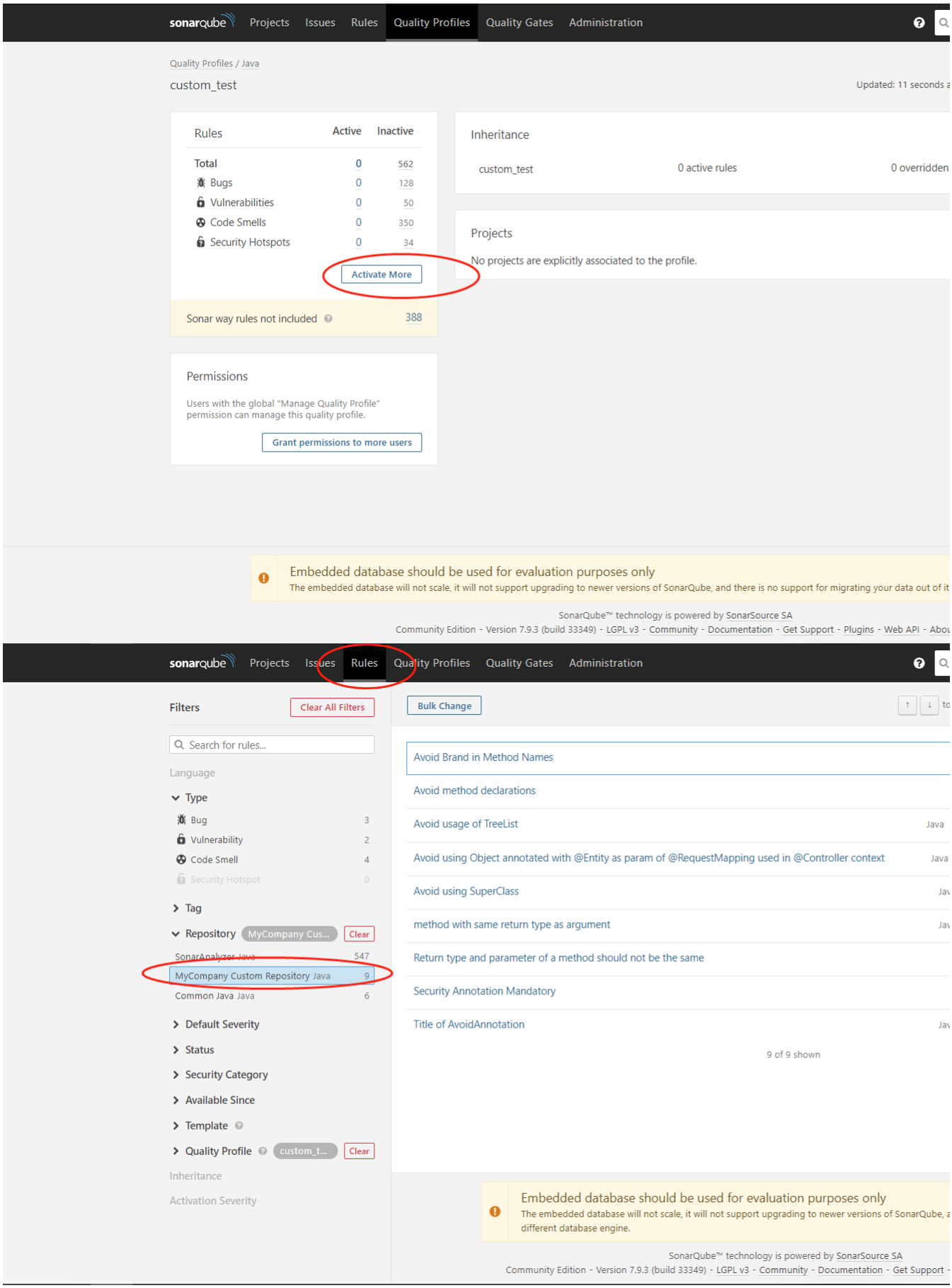
重启 SonarQube，可以看到我们添加的规则，自定义规则开发完毕

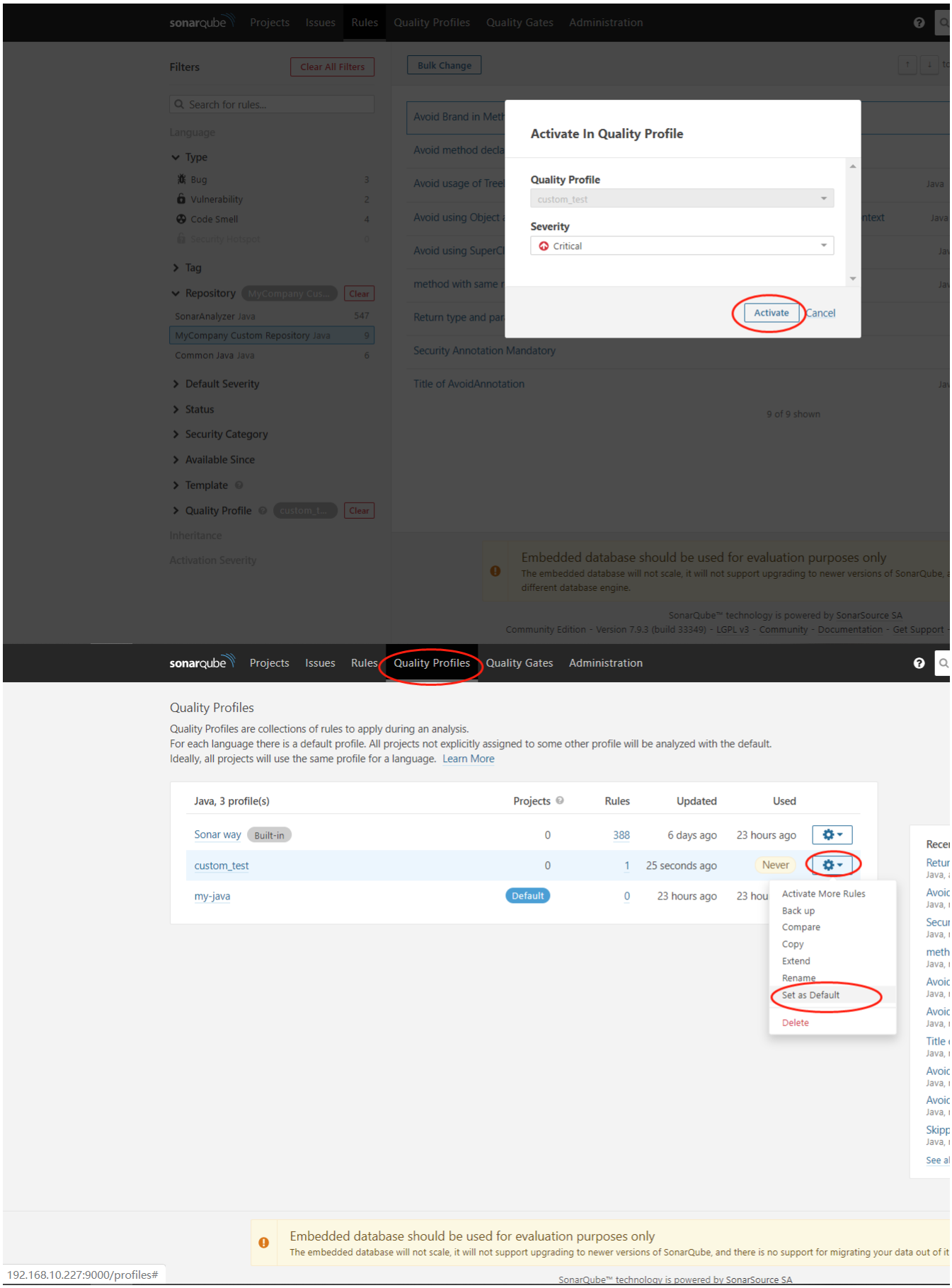


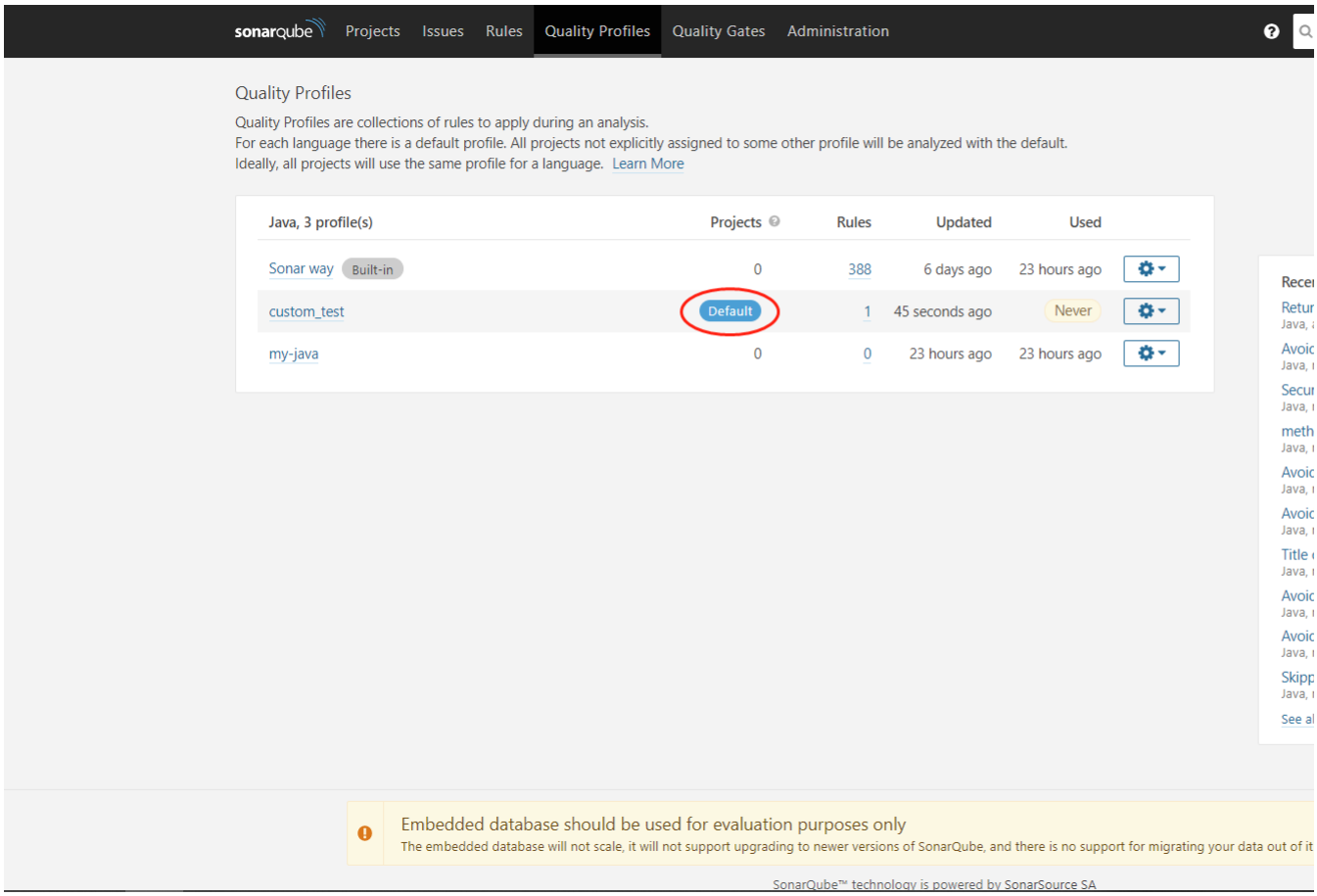
验证自定义规则

创建新的规则配置，只将我们添加自定义规则设置为检查项，步骤截图




sonarqube
Projects
Issues
Rules
Quality Profiles
Quality Gates
Administration
Quality Profiles / Java
custom_test
Updated: 11 seconds a
Rules
Active
Inactive
Total
0
562
Bugs
0
128
Vulnerabilities
0
50
Code Smells
0
350
Security Hotspots
0
34
Activate More
Sonar way rules not included
388
Inheritance
custom_test
0 active rules
0 overridden
Projects
No projects are explicitly associated to the profile.
Permissions
Users with the global "Manage Quality Profile" permission can manage this quality profile.
Grant permissions to more users
Embedded database should be used for evaluation purposes only
The embedded database will not scale, it will not support upgrading to newer versions of SonarQube, and there is no support for migrating your data out of it
SonarQube™ technology is powered by SonarSource SA
Community Edition - Version 7.9.3 (build 33349) - LGPL v3 - Community - Documentation - Get Support - Plugins - Web API - Abou
sonarqube
Projects
Issues
Rules
Quality Profiles
Quality Gates
Administration
Filters
Clear All Filters
Search for rules...
Language
Type
Bug
3
Vulnerability
2
Code Smell
4
Security Hotspot
0
Tag
Repository
MyCompany Cus...
Clear
SonarAnalyzer Java
547
MyCompany Custom Repository Java
9
Common Java Java
6
Default Severity
Status
Security Category
Available Since
Template
Quality Profile
custom_t...
Clear
Inheritance
Activation Severity
Bulk Change
Avoid Brand in Method Names
Avoid method declarations
Avoid usage of TreeList
Java
Avoid using Object annotated with @Entity as param of @RequestMapping used in @Controller context
Java
Avoid using SuperClass
Jav
method with same return type as argument
Jav
Return type and parameter of a method should not be the same
Security Annotation Mandatory
Title of AvoidAnnotation
Jav
9 of 9 shown
Embedded database should be used for evaluation purposes only
The embedded database will not scale, it will not support upgrading to newer versions of SonarQube, a
different database engine.
SonarQube™ technology is powered by SonarSource SA
Community Edition - Version 7.9.3 (build 33349) - LGPL v3 - Community - Documentation - Get Support -


sonarqube
Projects
Issues
Rules
Quality Profiles
Quality Gates
Administration
Filters
Clear All Filters
Bulk Change
Search for rules...
Language
Type
Bug 3
Vulnerability 2
Code Smell 4
Security Hotspot 0
Tag
Repository
MyCompany Cus...
Clear
SonarAnalyzer Java 547
MyCompany Custom Repository Java 9
Common Java Java 6
Default Severity
Status
Security Category
Available Since
Template
Quality Profile
custom_t...
Clear
Inheritance
Activation Severity
Activate In Quality Profile
Quality Profile
custom_test
Severity
Critical
Activate
Cancel
Avoid Brand in Meth
Avoid method decla
Avoid usage of Tree
Avoid using Object
Avoid using SuperCl
method with same r
Return type and par
Security Annotation Mandatory
Title of AvoidAnnotation
9 of 9 shown
Embedded database should be used for evaluation purposes only
The embedded database will not scale, it will not support upgrading to newer versions of SonarQube,
different database engine.
SonarQube™ technology is powered by SonarSource SA
Community Edition - Version 7.9.3 (build 33349) - LGPL v3 - Community - Documentation - Get Support
sonarqube
Projects
Issues
Rules
Quality Profiles
Quality Gates
Administration
Quality Profiles
Quality Profiles are collections of rules to apply during an analysis.
For each language there is a default profile. All projects not explicitly assigned to some other profile will be analyzed with the default.
Ideally, all projects will use the same profile for a language. Learn More
Java, 3 profile(s)
Projects
Rules
Updated
Used
Sonar way Built-in 0 388 6 days ago 23 hours ago
custom_test 0 1 25 seconds ago Never
my-java Default 0 23 hours ago 23 hou
Activate More Rules
Back up
Compare
Copy
Extend
Rename
Set as Default
Delete
Embedded database should be used for evaluation purposes only
The embedded database will not scale, it will not support upgrading to newer versions of SonarQube, and there is no support for migrating your data out of it
192.168.10.227:9000/profiles#
SonarQube™ technology is powered by SonarSource SA



创建项目验证自定义规则，和之前的扫描测试一样，在 Linux 服务器上执行

```
mkdir -p /usr/local/sonarqube/workspace/custom && cd /usr/local/sonarqube/workspace/custom
```

项目代码只有一个 MyFirstCustomCheck.java，就是前边用于单元测试的代码

```
class MyClass {
    MyClass(MyClass mc) { }

    int     foo1() { return 0; }
    void    foo2(int value) { }
    int     foo3(int value) { return 0; } // Noncompliant
    Object  foo4(int value) { return null; }
    MyClass foo5(MyClass value) {return null; } // Noncompliant

    int     foo6(int value, String name) { return 0; }
    int     foo7(int ... values) { return 0;}
}
```

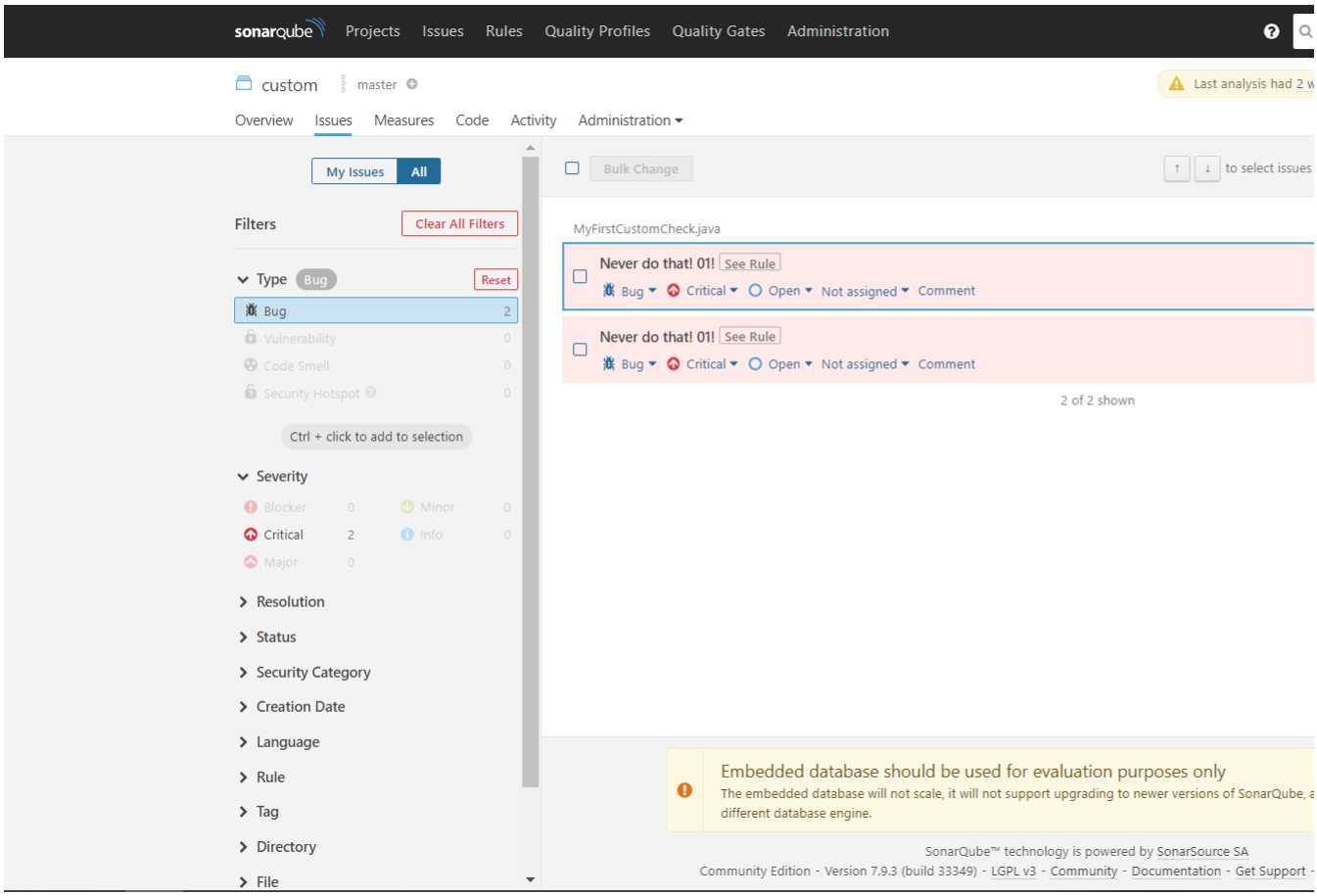
创建扫描配置文件 sonar-project.properties

```
sonar.projectKey=custom
sonar.sources=.
sonar.host.url=http://192.168.10.227:9000
sonar.login=c4765957e5ada82ebe21a7c2e1f56afbff4059d3
sonar.language=java
sonar.java.binaries=.
sonar.sourceEncoding=UTF-8
```

执行扫描

```
sonar-scanner
```

自定义规则生效了



备注

注意，以上自定义规则应该是对 SonarJava 的拓展，必须安装了 SonarJava 才会生效。