SonarQube (소나큐브)

CONTINUOUS INSPECTION TOOL

양경석

Military & Mobile Communications Lab
Graduate School of Ajou University

Contents

About SonarQube: intro

SonarQube

- 오픈소스 품질 관리 플랫폼
- 지속적으로 분석, 기술적 품질 분석
- reduced risk, lower cost 로 개발 및 장기적인 유지보수를 가능하게 함

Community Edition	Professional Edition
This Edition is best suited for small teams: • Located on the same site • Single technology • Effort driven by the development team	This Edition is best suited for medium-size teams (~50): • Located on the same site • Single technology • Effort driven by the development team • Regular management reporting
Enterprise Edition	**Ultimate Edition**
This Edition is best suited for large teams (100+): • Located on the same site • Single entity • Multiple technologies • Centralized management of quality • Quality gate in place and enforced • Effort driven by management and team	This Edition is best suited for large enterprises: • Located on multiple sites • Multiple lines of businesses • Numerous technologies • Centralized management of quality • Quality gate in place and enforced • Enterprise-wide initiative

free

12,500 euro
(약 1,600만원)

50,000 euro
(약 6,500만원)

Contact Us

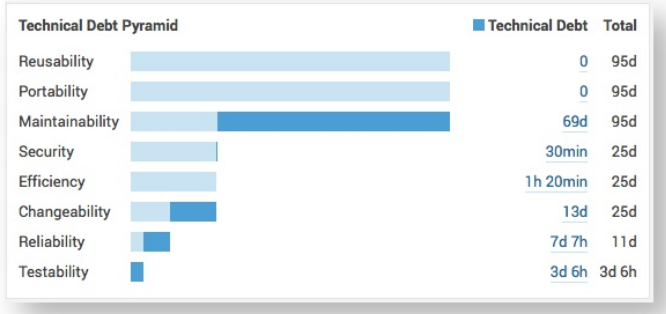
About SonarQube: Customers

About SonarQube: Developers' Seven Deadly Sins

- Bugs and Potential Bugs [버그와 잠재적 버그]
- Coding Standards Breach [코딩 표준 위반]
- Duplications [중복 코드]
- Lack of Unit Tests [단위테스트의 부족으로 인한 낮은 커버리지]
- Bad Distribution of Complexity [복잡한 코드 분포]
- Spaghetti Design [스파게티 설계]
- Not Enough or Too Many Comments [과다/과소 주석]

Technical Debt

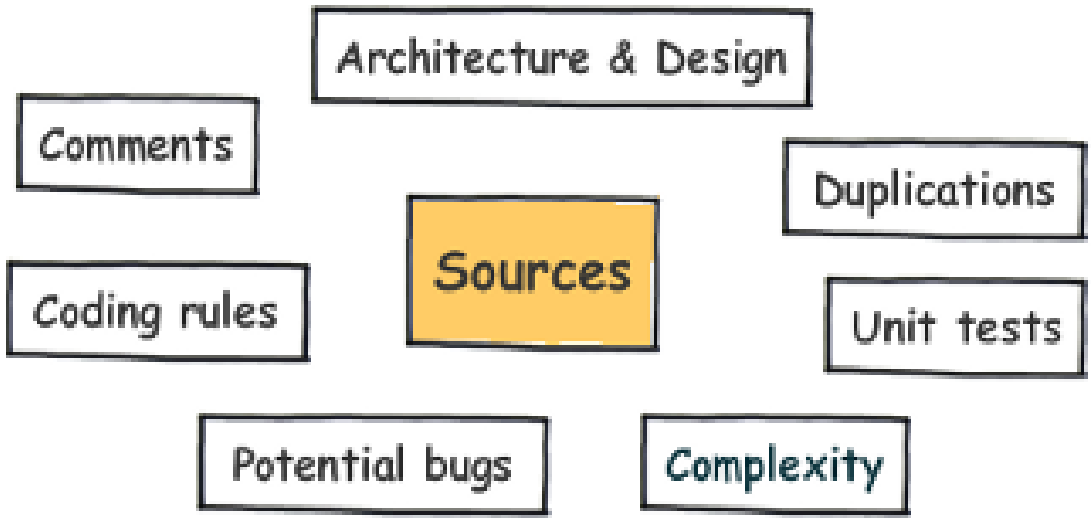
워드 커닝햄(Ward Cunningham):
코드를 **지저분**하게 만드는 것은 기술적인 빚을 지는 것.
빚을 줄여야 앞으로의 이자인 **생산성 향상**에 도움을 준다



Technical Debt Pyramid widget

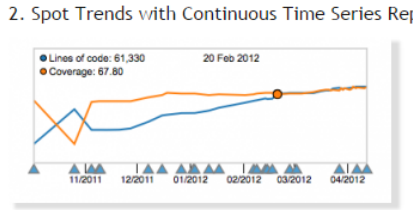
About SonarQube: 7 axes of code quality

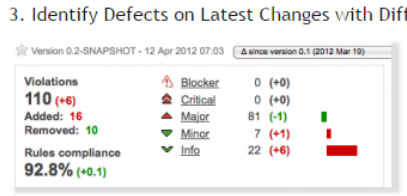
- Developers' Seven Deadly Sins 해결을 위한 SonarQube가 제공하는 7가지 기능 분류

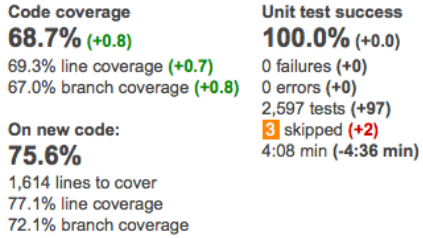


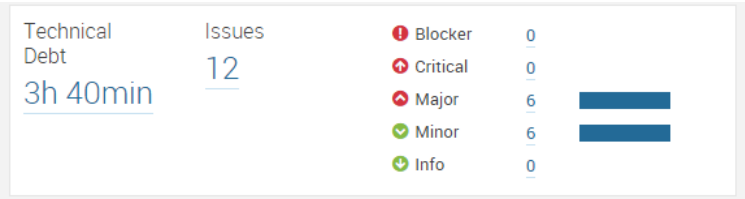
About SonarQube: Features

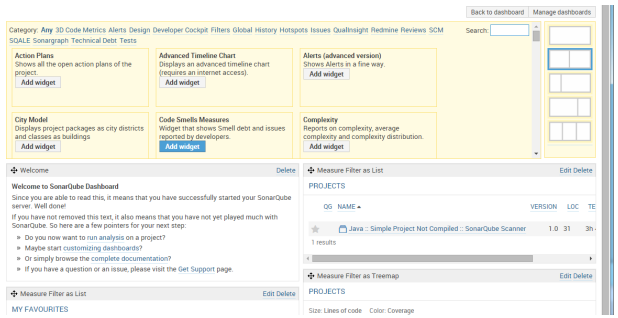
- Continuous Inspection [지속적인 점검]
- Multidimensional Analysis [다차원 분석]
- Customizable Dashboards [대쉬보드 변경 가능]
- Extensibility [플러그인을 통한 확장가능성]
- Technical Debt Evaluation [SW에 내재된 잠재 위험, 기술부채 평가]
- Actionable Reporting [의사결정에 사용가능]
- Centralized Portfolio Management [한군데서 관리]
- Developer Perspective [개발자의 관점 고려]
- Teamwork and Collaboration [svn, redmine, Jenkins 등과 연동 가능]
- Rule-Based Defect Identification [룰에 기반]
- Recent quality Issue Monitoring
- Application Lifecycle Management
- Multi-Technology Suppert
- Security
- Open Source
- ...





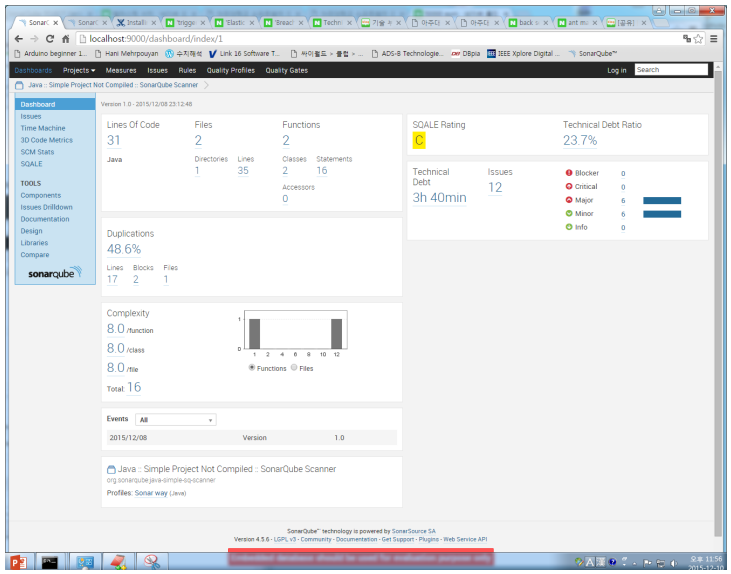


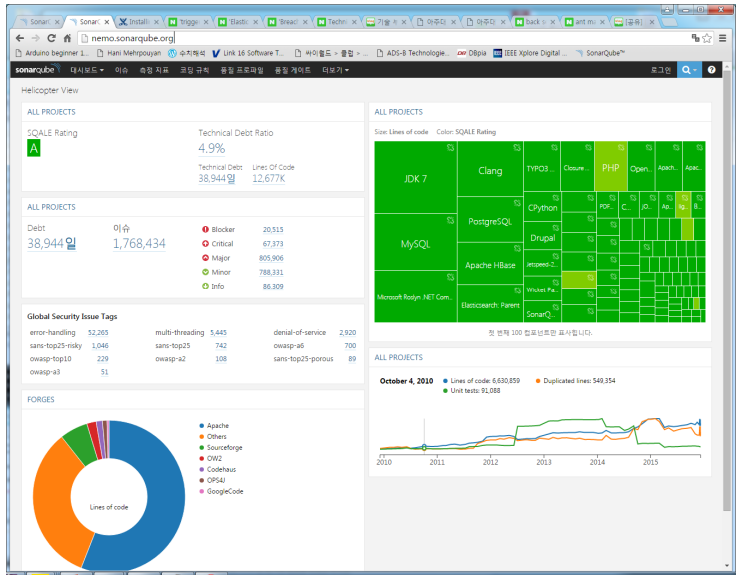




About SonarQube: Dashboard Examples

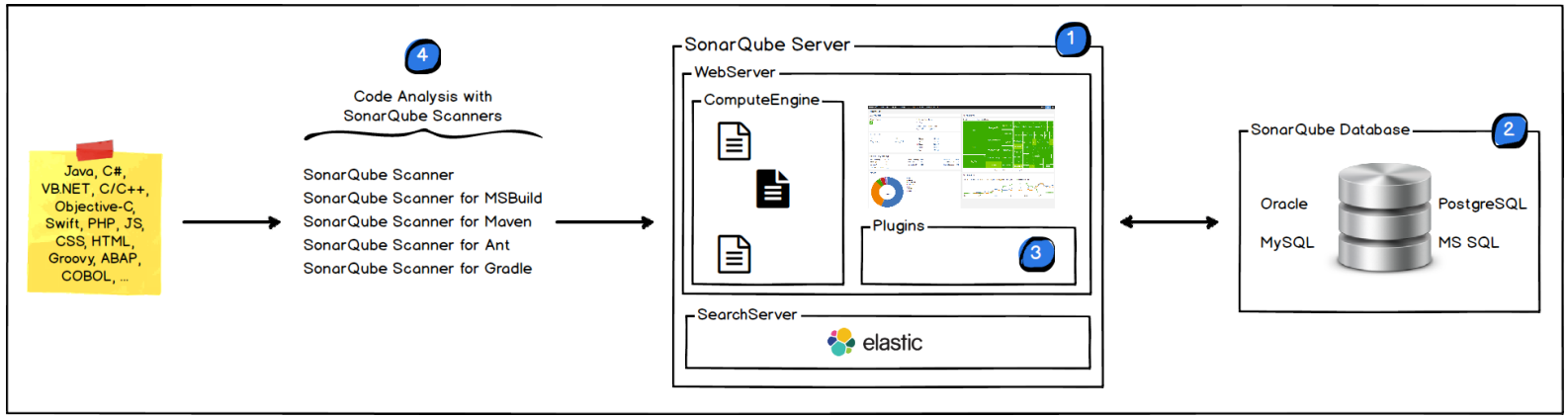
- Bugs and Potential Bugs [버그와 잠재적 버그]
- Coding Standards Breach [코딩 표준 위반]
- Duplications [중복 코드]
- Lack of Unit Tests [단위테스트의 부족으로 인한 낮은 커버리지]
- Bad Distribution of Complexity [복잡한 코드 분포]
- Spaghetti Design [스파게티 설계]
- Not Enough or Too Many Comments [과다/과소 주석]





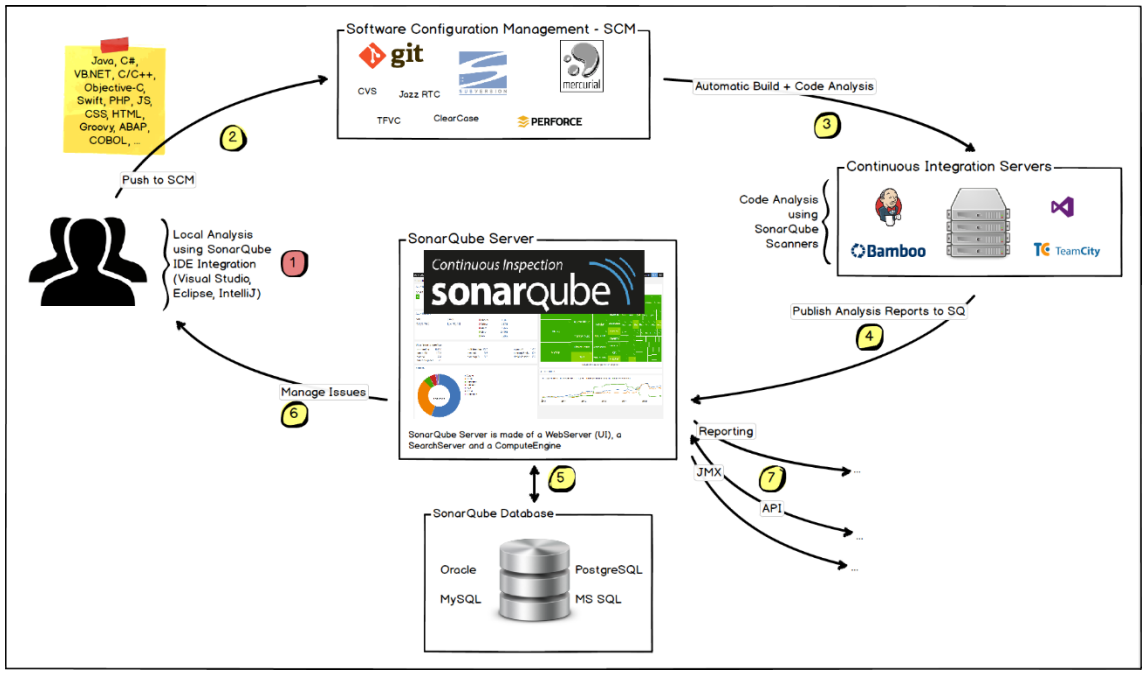
About SonarQube: Architecture

1. SonarQube Server
   - Web Server
   - Search Server
2. SonarQube Database
   - Configuration
   - Quality snapshots
3. SonarQube Plugins
4. SonarQube Scanners
   - Sonar Runner
   - SonarQube Scanner for MSBuild, Maven, Ant, Gradle

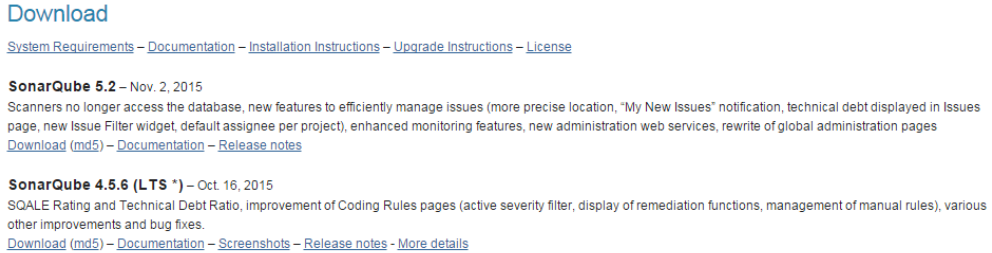


About SonarQube: Integration

1. Local IDE (Eclipse)에서 분석
2. 코드를 SCM으로 전송(SVN, git)
3. 자동 빌드, 코드 분석을 수행하는 CI서버로 전송
   - CI서버에서 SonarQube Scanner를 실행하여 분석 보고서 발행
4. CI서버에서 발행된 분석보고서가 SonarQube Server로 전송
5. 분석보고서가 SonarQube DB에 저장되고, UI로 보여짐
6. 개발자는 UI를 통해 확인



Setup: 설치방법



Download

System Requirements – Documentation – Installation Instructions – Upgrade Instructions – License

SonarQube 5.2 – Nov. 2, 2015
Scanners no longer access the database, new features to efficiently manage issues (more precise location, "My New Issues" notification, technical debt displayed in Issues page, new Issue Filter widget, default assignee per project), enhanced monitoring features, new administration web services, rewrite of global administration pages
Download (md5) – Documentation – Release notes

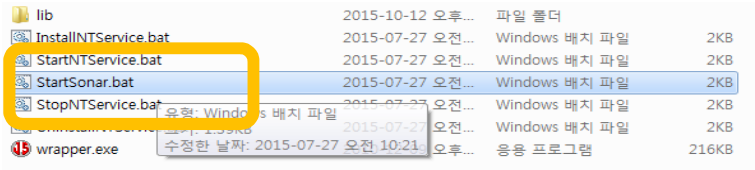
SonarQube 4.5.6 (LTS *) – Oct. 16, 2015
SQALE Rating and Technical Debt Ratio, improvement of Coding Rules pages (active severity filter, display of remediation functions, management of manual rules), various other improvements and bug fixes.
Download (md5) – Documentation – Screenshots – Release notes - More details

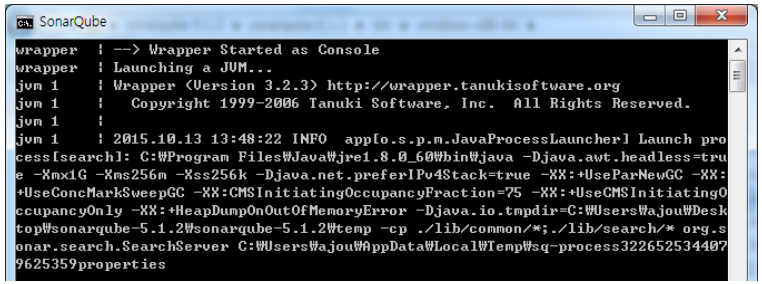
- 1. **Download and unzip** the **SonarQube** distribution
  - http://www.sonarsource.org/downloads/
  - C:₩SonarQube₩sonarqube-4.5.6
- 2. **Start** the SonarQube server
  - # On Windows, execute:
  - C:₩SonarQube₩sonarqube-5.1.2₩bin₩windows-x86-64₩StartSonar.bat
- 3. **Download and unzip** the SonarQube Scanner (**Sonar-runner**)
  - http://repo1.maven.org/maven2/org/codehaus/sonar/runner/sonar-runner-dist/2.4/sonar-runner-dist-2.4.zip
  - C:₩SonarQube₩sonar-runner-2.4
- 4. Download and unzip some project samples (Sonar-examples)
- 5. **Analyze a project**
  - # On Windows:
    - cd C:₩sonar-examples₩projects₩languages₩java₩sonar-runner₩java-sonar-runner-simple
    - C:₩sonar-runner₩bin₩sonar-runner.bat
- 6. Browse the results at http://localhost:9000

Setup: 설치 화면

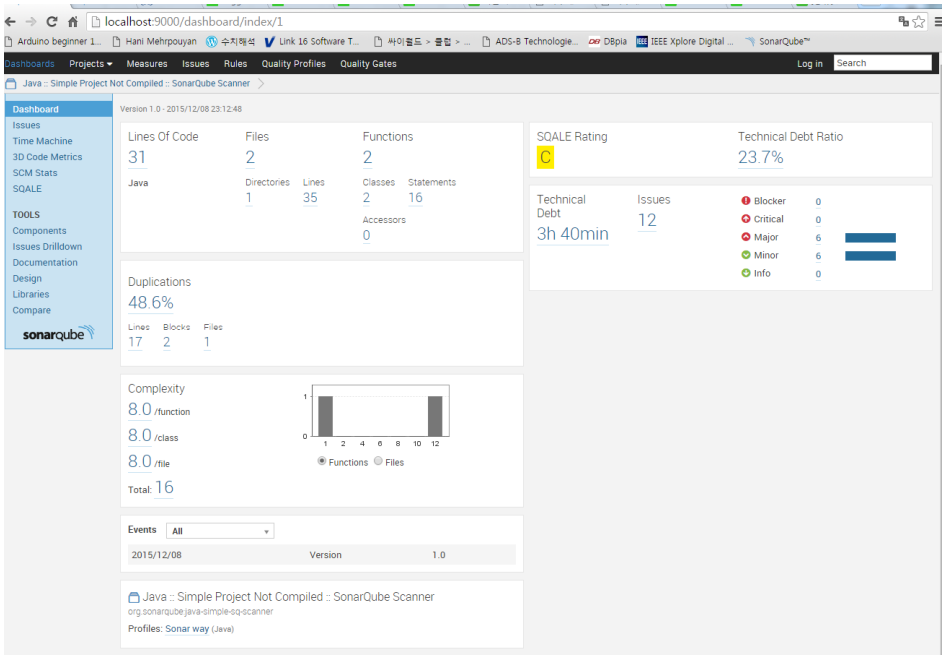
SonarQube Server설치

- /bin/StartSonar.bat 실행하여 서버 실행



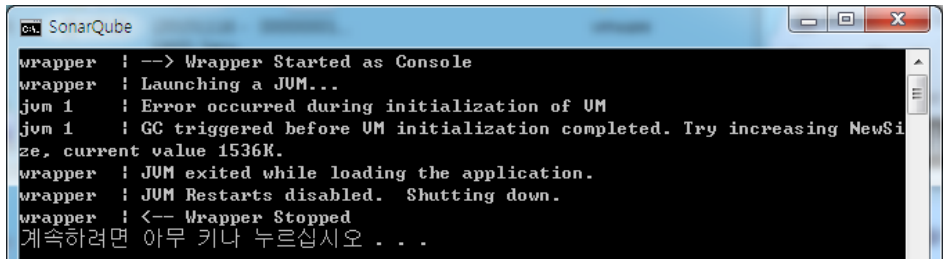


http://localhost:9000



Setup: 에러 발생시 해결 방법

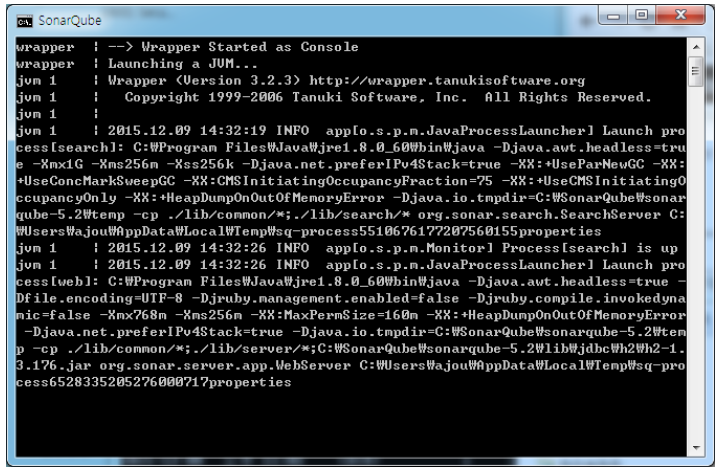
메모리 부족 메시지 뜨는 경우



- C:₩SonarQube₩sonarqube-4.5.6₩conf₩wrapper.conf 변경

```
12 #*****************************************************************
13 # Wrapper Java
14 #*****************************************************************
15 wrapper.java.additional.1=-Djava.awt.headless=true
16 wrapper.java.mainclass=org.tanukisoftware.wrapper.WrapperSimpleApp
17 wrapper.java.classpath.1=../../lib/jsw/*.jar
18 wrapper.java.classpath.2=../../lib/*.jar
19 wrapper.java.library.path.1=./lib
20 wrapper.app.parameter.1=org.sonar.application.App
21 wrapper.java.initmemory=3
22 wrapper.java.maxmemory=3
```

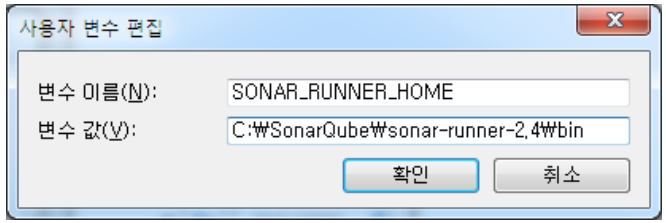
wrapper.java.initmemory=128
wrapper.java.maxmemory=256



StartSonar.bat 실행화면

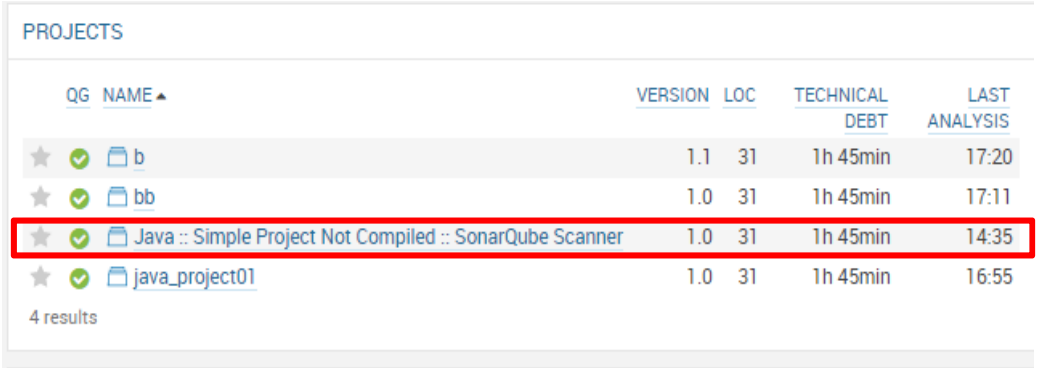
분석 방법1: Sonar Runner를 통한 실행 (manual)

- sonar-runner의 설치경로를 환경변수에 등록
- 소스코드가 있는 폴더로 이동
- **sonar-project.properties 파일 작성**




```
# Required metadata
sonar.projectKey=org.sonarqube:java-simple-sq-scanner
sonar.projectName=Java :: Simple Project Not Compiled :: SonarQube Scanner
sonar.projectVersion=1.0

# Comma-separated paths to directories with sources (required)
sonar.sources=src

# Language
sonar.language=java

# Encoding of the source files
sonar.sourceEncoding=UTF-8
```

- sonar-runner실행
- SonarQube 서버에서 확인



분석 방법1: Sonar Runner를 통한 실행 (autosonar)

자동화 도구(autosonar) 제작

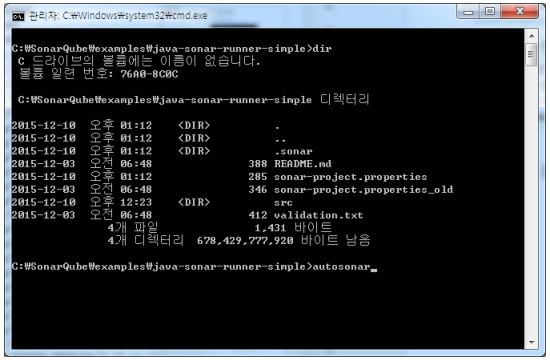
- sonar-project.properties 파일을 매번 제작해야하는 번거로움 해소

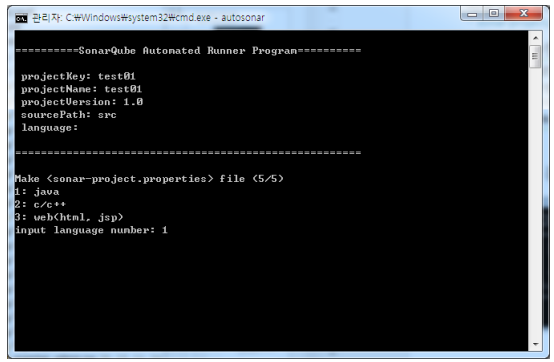
프로그램 수행내용

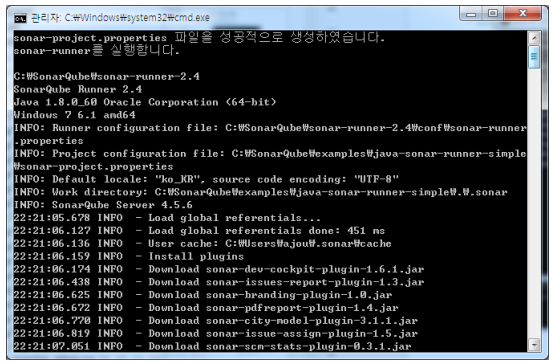
1. Project Key, Name, Version, Source dir, Language 입력받음
2. sonar-project.properties 파일 작성
3. sonar-runner실행

작성한 autosonar.exe는 sonar-runne와 동일한 경로에 이동(환경변수)

파일 위치한 경로에서 autosonar 실행







SonarQube 서버에서 확인

분석 방법1: Sonar Runner를 통한 실행 (autosonar)

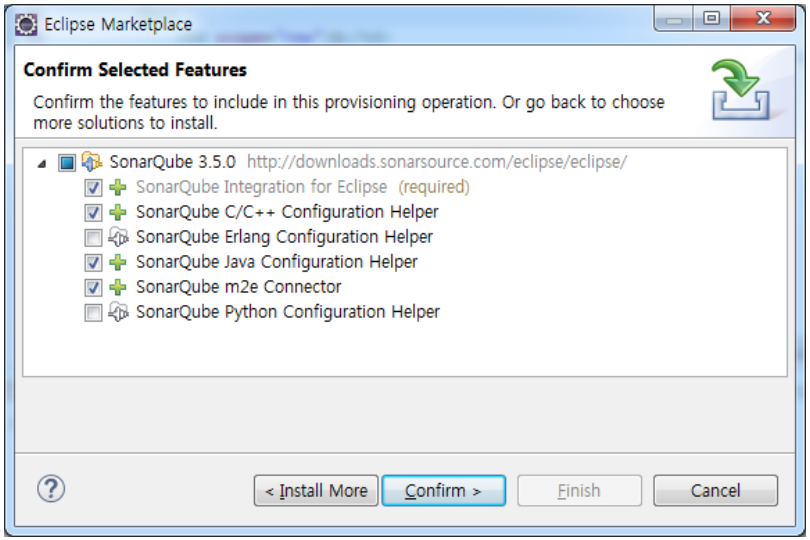
- autosonar 소스코드

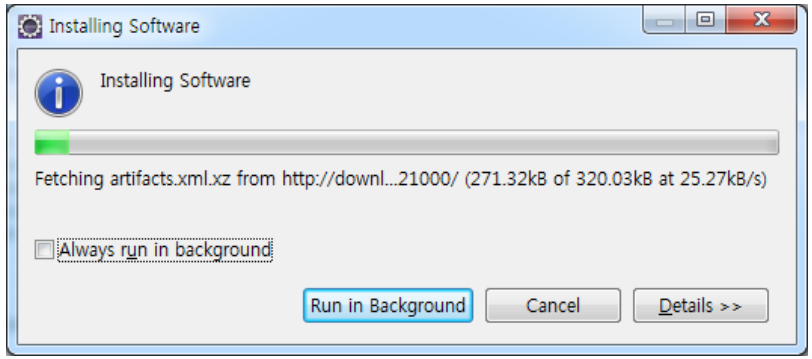
```
#include <stdio.h>
#include <string.h>

int main(){
FILE *fp;
char projectKey[30] = { 0, };
char projectName[30] = { 0, };
char projectVersion[30] = { 0, };
char sourcePath[30] = { 0, };
char langChoice[3][30] = { "java", "c", "web" };
char language[30] = { 0, };
char textBuffer[65535] = { 0, };

int langch = 0;
int cnt = 0;
int i = 0;

for (cnt = 1; cnt <= 5;cnt++){
system("cls");
printf("\n==========SonarQube Automated Runner Program==========\n\n");
printf(" projectKey: %s\n", projectKey);
printf(" projectName: %s\n", projectName);
printf(" projectVersion: %s\n", projectVersion);
printf(" sourcePath: %s\n", sourcePath);
printf(" language: %s\n", language);
printf("\n======================================================\n\n");

switch (cnt){
case 1:
printf("Make <sonar-project.properties> file (1/5)\n");
printf("input Project Key: ");
scanf("%s", projectKey);
break;

case 2:
printf("Make <sonar-project.properties> file (2/5)\n");
printf("input Project Key: ");
scanf("%s", projectName);
break;

case 3:
printf("Make <sonar-project.properties> file (3/5)\n");
printf("input Project Key: ");
scanf("%s", projectVersion);
break;

case 4:
system("dir");
printf("\n======================================================\n\n");
printf("Make <sonar-project.properties> file (4/5)\n");
printf("input sourcePath: ");
scanf("%s", sourcePath);
break;

case 5:
printf("Make <sonar-project.properties> file (5/5)\n");
printf("1: java\n2: c/c++\n3: web(html, jsp)\n");
printf("input language number: ");
scanf("%d", &langch);
sprintf(language, "%s", langChoice[langch-1]);
break;

default: break;

}//end switch
}//end for

sprintf(textBuffer, "# Required
metadata\nsonar.projectKey=%s\nsonar.projectName=%s\nsonar.projectVersion=%s\n\n# Comma -
separated paths to directories with sources(required)\nsonar.sources=%s\n\n#
Language\nsonar.language=%s\n\n# Encoding of the source files\nsonar.sourceEncoding=UTF-
8", projectKey, projectName, projectVersion, sourcePath, language);

//printf("%s\n", textBuffer);

fp = fopen("sonar-project.properties", "w");

for (i = 0; i < strlen(textBuffer); i++){
fprintf(fp, "%c", textBuffer[i]);
}
fclose(fp);

system("cls");
printf("sonar-project.properties 파일을 성공적으로 생성하였습니다.\n");
printf("sonar-runner를 실행합니다.\n\n");
system("sonar-runner.bat");

return 0;
}
```

분석 방법2: Eclipse Plugin을 통한 분석

Eclipse Plugin 설치

- eclipse – help – marketplace – SonarQube 검색 - install

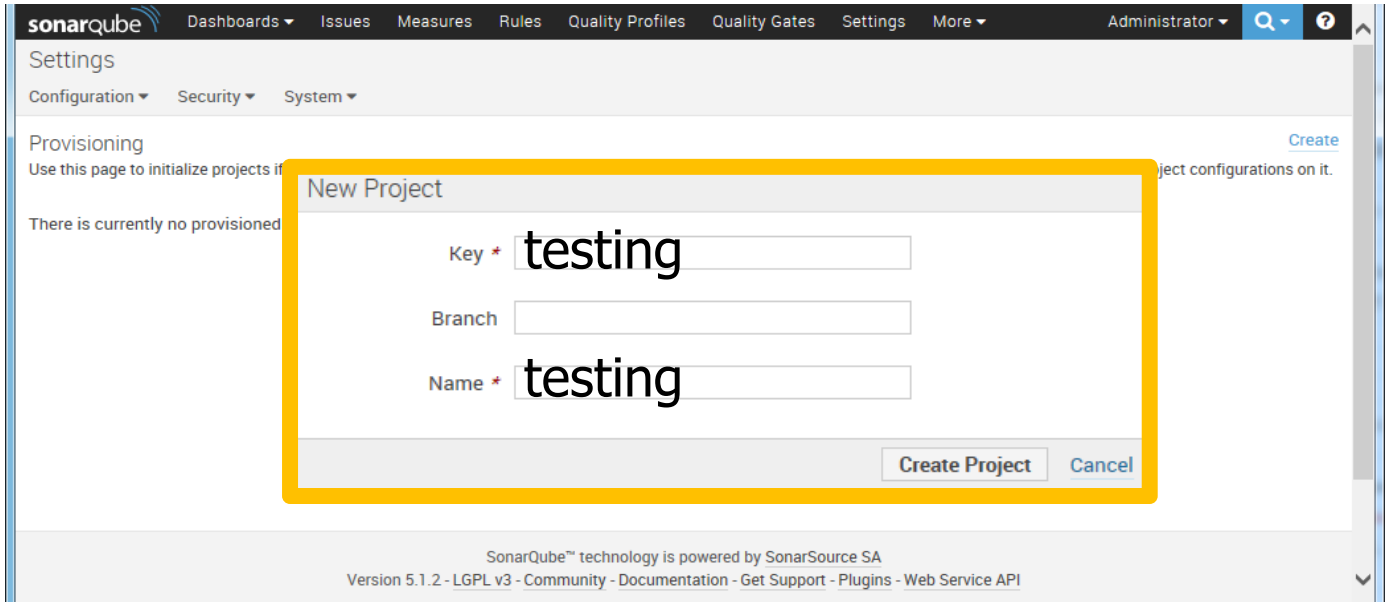




분석 방법2: Eclipse Plugin을 통한 분석

SonarQube server 설정

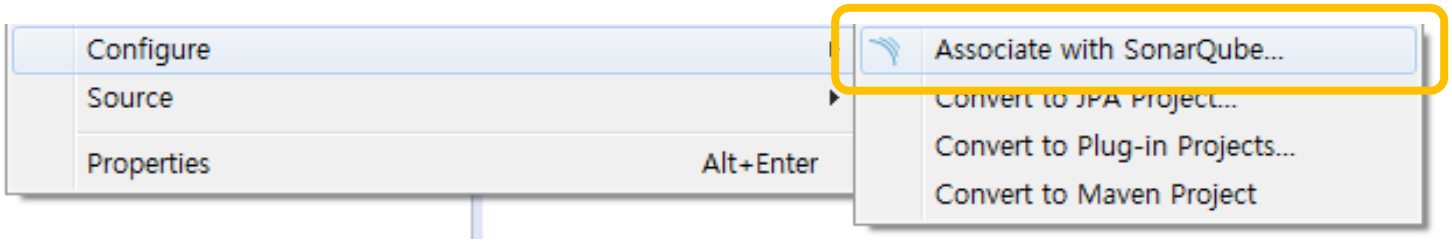
- New Project: 임의의 Key, Branch, Name 입력



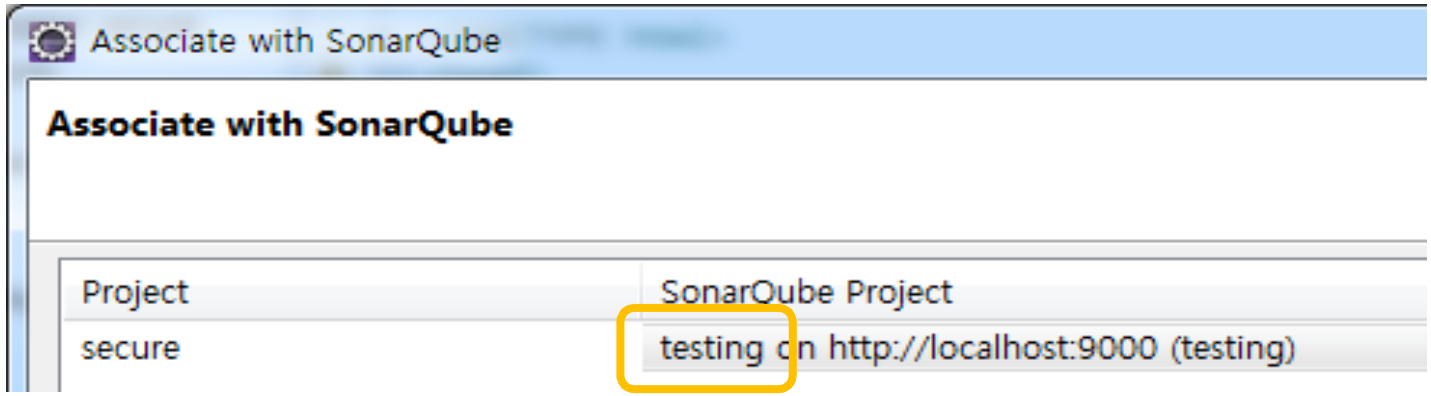
분석 방법2: Eclipse Plugin을 통한 분석

Eclipse와 SonarQubeServer 연동

- 프로젝트명-마우스 우클릭-Configure-Associate with SonarQube



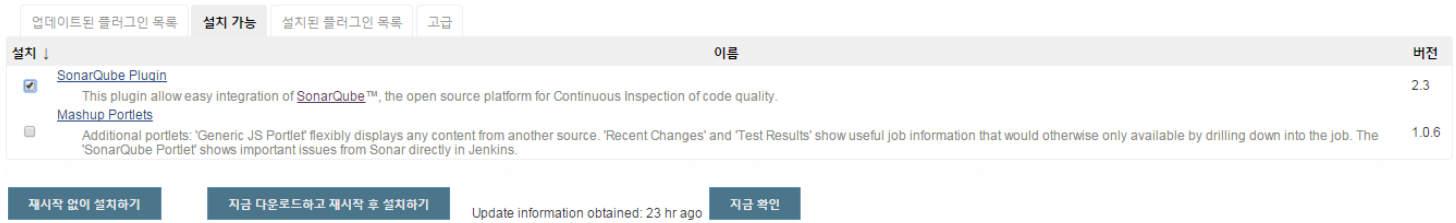
- SonarQube Server에서 입력했던 name 입력(testing)



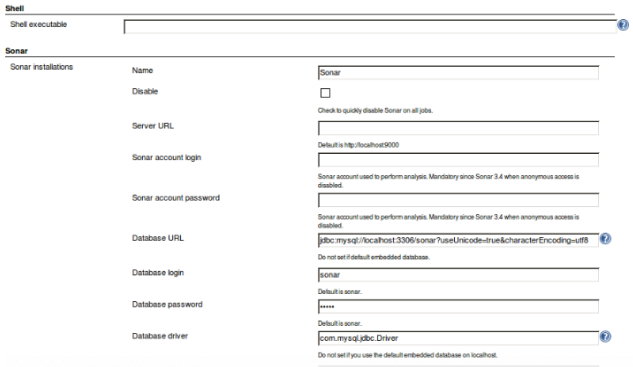
- SonarQube 서버에서 확인

분석 방법3: CI도구 젠킨스와 연동

- 젠킨스는 svn 등 SCM과 연결되어 있음
- 젠킨스의 SonarQube 플러그인 설치



- 설정- SonarQube 설정 추가



- ant/maven 빌드후 조치에 SonarQube 실행되도록 추가
- 빌드 실행 후 SonarQube Server에서 확인

분석 방법의 장단점 비교

	방법1 (Sonar-runner를 통한 실행)	방법2 (IDE와 연결)	방법3 (CI와 통합)
장점	간편함 (간단하게 콘솔창에 명령어 입력만으로 분석 결과 확인 가능)	익숙함 (개발환경과 직접 연동)	중앙집중형 관리
단점	번거로움 (매번 sonar-project.properties 파일 작성 autosonar로 파일작성 자동화되어도 역시 번거로움)	과거 정보가 DB에 저장되지 않음	초기 환경구축 어려움
용도	소규모 local 프로그램 분석시	협업에서 commit 전에 분석시 활용가능	대규모 장기 프로젝트

소프트웨어 품질관리를 위한 SW Visualization

효율적인 SW개발관리

- 계획: 지표 설정에 따른 명확한 목표 수립
- 수행: 시스템 기반의 효율적인 개발활동
- 검증: 시각화를 통한 지속적 모니터링 및 통제

with SonarQube

감사합니다