

DAM

Contornos de desenvolvimento: CD

UD4 **SONARQUBE**

PROFESORA | CRISTINA PUGA FDEZ

UNIDADE | UD4

TRIMESTRE | 2

MÓDULO | CD

1.	SonarQube.....	3
1.1	Qué es SonarQube?	3
1.2	Descarga de SonarQube	3
1.3	Ejecución de SonarQube.....	4
2.	Análisis de un proyecto	5
2.1	Instalación de Maven.....	5
2.2	Creación de un proyecto Maven en Eclipse	6
2.3	Análisis en SonarQube	9
3.	Análisis con Sonar directamente en Eclipse.....	15
3.1	Instalar y configurar plugin SonarLint.....	15
3.2	Asociar proyecto Eclipse con Sonar	21
4.	Recursos	23

1. SonarQube

1.1 Qué es SonarQube?

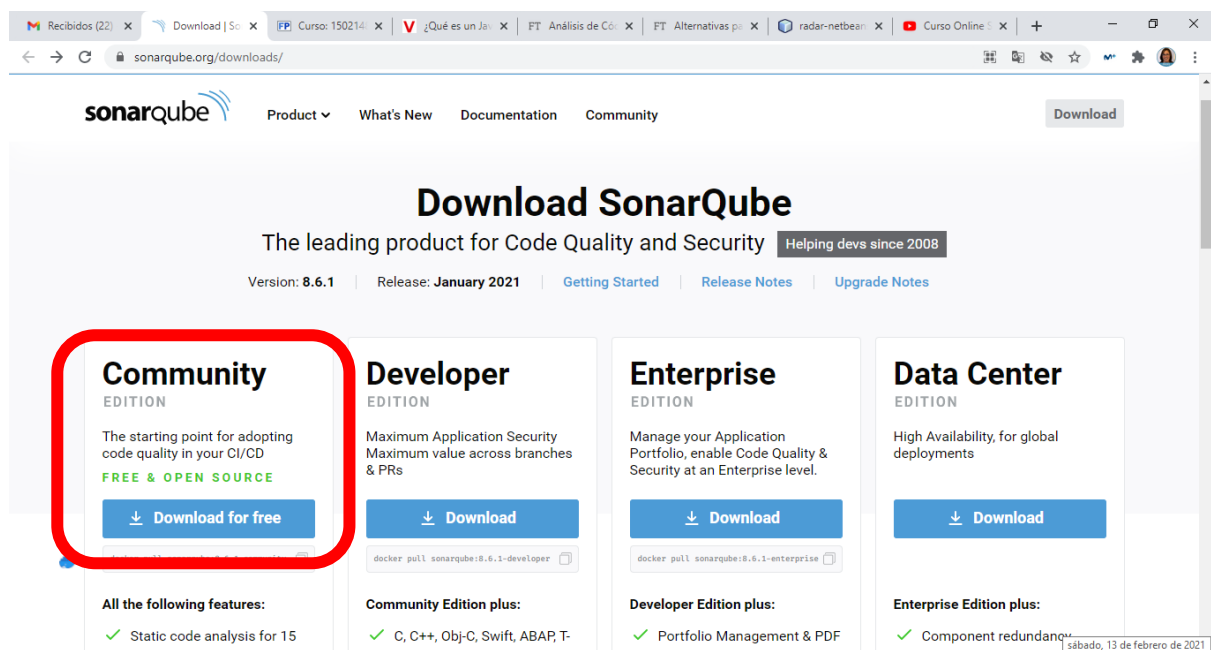
SonarQube es una plataforma de código abierto para el análisis de la calidad de código usando diversas herramientas de análisis estático de código fuente como Checkstyle, PMD o FindBugs para obtener métricas que pueden ayudar a mejorar la calidad del código de un programa. También pertenece al conjunto de herramientas de análisis de código estático, junto con Understand, semmle y otras.

Es una herramienta esencial para nuestra fase de testing y auditoria de código dentro de nuestro ciclo de desarrollo de nuestra aplicación.

1.2 Descarga de SonarQube

Podemos instalar SonarQube usando:

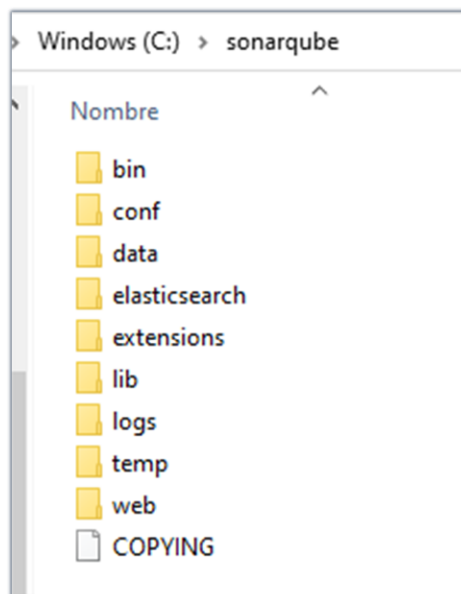
- una instalación tradicional con el archivo zip de la versión **Community**:
<https://www.sonarqube.org/downloads/>



- usando un contenedor Docker usando una de nuestras **imágenes de Docker**. En este documento no vamos a desarrollar esta opción.

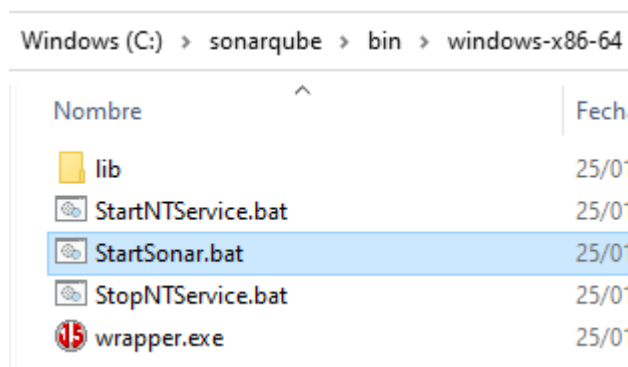
1.3 Ejecución de SonarQube

Descomprimos el fichero .zip descargado en una ubicación como puede ser: c:\sonarqube



Para iniciar SonarQube Server, debemos ejecutar el fichero StartSonar.bat que se encuentra en el directorio “\bin\windows-x86-64” de donde hayamos descomprimido el .zip.

Hacemos doble clic sobre el fichero **StartSonar.bat**:



Ahora veremos como se inicia SonarQube en una consola:

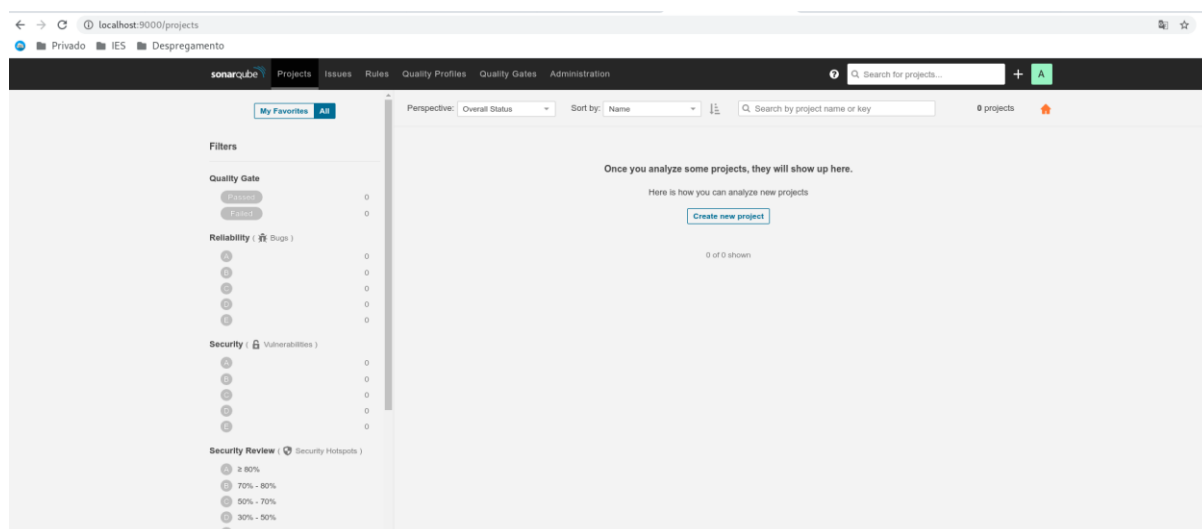
```

SonarQube
wrapper --> Wrapper Started as Console
wrapper Launching a JVM...
jvm 1 Wrapper (Version 3.2.3) http://wrapper.tanukisoftware.org
jvm 1 Copyright 1999-2006 Tanuki Software, Inc. All Rights Reserved.
jvm 1 2021.02.13 10:22:23 INFO app[[o.s.a.AppFileSystem] Cleaning or creating temp directory C:\sonarqube\temp
jvm 1 2021.02.13 10:22:23 INFO app[[o.s.a.es.EsSettings] Elasticsearch listening on [HTTP: 127.0.0.1:9001, TCP: 127.0.0.1:53096]
jvm 1 2021.02.13 10:22:23 INFO app[[o.s.a.ProcessLauncherImpl] Launch process[[key='es', ipcIndex=1, logFilenamePrefix=es]] from [C:\sonarqube\elasticsearch]: C:\Program Files\Java\jdk-14.0.1\bin\java -XX:+UseConcMarkSweepGC -XX:CMSInitiatingOccupancyFraction=75 -XX:+UseCMSInitiatingOccupancyOnly -Djava.io.tmpdir=C:\sonarqube\temp -XX:ErrorFile=../log/s/es_hs_err_pid%p.log -Des.networkaddress.cache.ttl=60 -Des.networkaddress.cache.negative.ttl=10 -XX:+AlwaysPreTouch -Xss1m -Djava.awt.headless=true -Dfile.encoding=UTF-8 -Djna.nosys=true -XX:-OmitStackTraceInFastThrow -Dio.netty.noUnsafe=true -Dio.netty.noKeySetOptimization=true -Dio.netty.recycler.maxCapacityPerThread=0 -Dio.netty allocator.numDirectArenas=0 -Dlog4j.shutdownHookEnabled=false -Dlog4j2.disable.jmx=true -Djava.locale.providers=COMPAT -Xmx512m -Xms512m -XX:MaxDirectMemorySize=256m -XX:+HeapDumpOnOutOfMemoryError -Delasticsearch -Des.path.home=C:\sonarqube\elasticsearch -Des.path.conf=C:\sonarqube\temp\conf\es -cp lib/* org.elasticsearch.bootstrap.Elasticsearch
jvm 1 2021.02.13 10:22:23 INFO app[[o.s.a.SchedulerImpl] Waiting for Elasticsearch to be up and running
jvm 1 Java HotSpot(TM) 64-Bit Server VM warning: Ignoring option UseConcMarkSweepGC; support was removed in 14.0
jvm 1 Java HotSpot(TM) 64-Bit Server VM warning: Ignoring option CMSInitiatingOccupancyFraction; support was removed in 14.0
jvm 1 Java HotSpot(TM) 64-Bit Server VM warning: Ignoring option UseCMSInitiatingOccupancyOnly; support was removed in 14.0
jvm 1 2021.02.13 10:22:32 INFO app[[o.s.a.SchedulerImpl] Process[es] is up
jvm 1 2021.02.13 10:22:32 INFO app[[o.s.a.ProcessLauncherImpl] Launch process[[key='web', ipcIndex=2, logFilenamePrefix=web]] from [C:\sonarqube]: C:\Program Files\Java\jdk-14.0.1\bin\java -Djava.awt.headless=true -Dfile.encoding=UTF-8 -Djava.io.tmpdir=C:\sonarqube\temp -XX:-OmitStackTraceInFastThrow --add-opens=java.base/java.util=ALL-UNNAMED --add-opens=java.base/java.lang=ALL-UNNAMED --add-opens=java.base/java.io=ALL-UNNAMED --add-opens=java.rmi/sun.rmi.transport=ALL-UNNAMED -Xmx512m -Xms128m -XX:+HeapDumpOnOutOfMemoryError -Dhttp.nonProxyHosts=localhost|127.*|:::1 -cp ./lib/common/*;C:\sonarqube\lib\jdbc\h2\h2-1.4.199.jar org.sonar.server.app.WebServer C:\sonarqube\temp\sq-process9972060858361881589

```

Una vez que nuestra instancia esté en funcionamiento, iniciamos sesión en <http://localhost:9000/> con las credenciales de administrador del sistema:

- iniciar sesión: admin
- contraseña: admin



2. Análisis de un proyecto

2.1 Instalación de Maven

Maven es una herramienta desarrollada en Java que simplifica las tareas de compilación y construcción de aplicaciones de software, principalmente usado con lenguajes de programación que tienen como destino ser ejecutadas en la Java Virtual Machine (JVM). Además incorpora una gestión de dependencias madura con acceso a los repositorios públicos de Maven con más de 16 Millones de librerías disponibles incluyendo todas las versiones de cada una de ellas.

Para su instalación, por ejemplo, sobre **Windows** únicamente debemos acceder a la página oficial del proyecto Maven y descargar la última versión estable.

<https://maven.apache.org/download.cgi>

La configuración de Maven para Windows es muy sencilla, lo único que debes hacer es descomprimir el archivo Zip descargado del paso anterior en un directorio. Por ejemplo, en el directorio donde se ha almacenado los archivos del JDK: c:\Archivos de Programa\Java\apache-maven-<version>.

Una vez descomprimidos los archivos se deberá crear la variable de entorno MAVEN_HOME apuntando a la ruta anterior, base de la instalación. El uso de esta variable permitirá cambiar de una distribución de Maven a otra versión más actualizada sin tener que cambiar todas las referencias a esta difundida por los proyectos.

Por último, añadiremos la ruta de los binarios de Java a la variable de entorno Path para que podamos invocar a maven con el comando mvn desde cualquier directorio:

```
Path=%MAVEN_HOME%\bin;%Path%
```

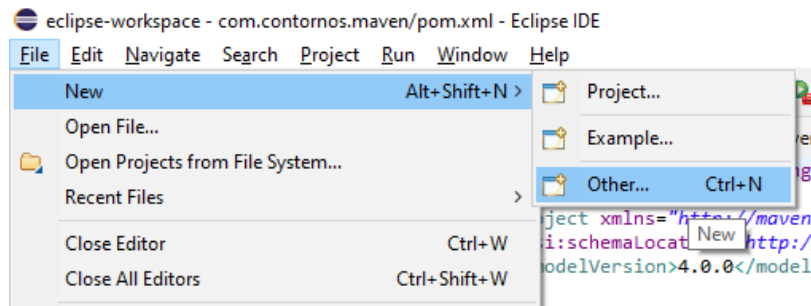
Si la configuración está bien hecha, desde un terminal de comandos podremos invocar a maven a modo de prueba con la siguiente instrucción:

```
mvn -version
```

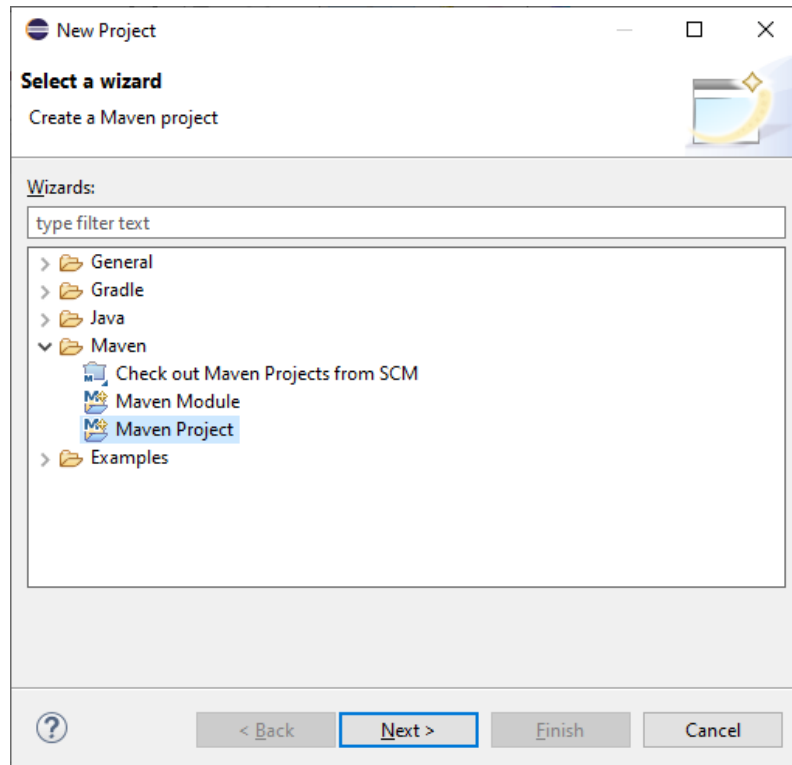
2.2 Creación de un proyecto Maven en Eclipse

Para llevar a cabo el análisis en SonarQube vamos a crear un proyecto Maven automáticamente desde Eclipse.

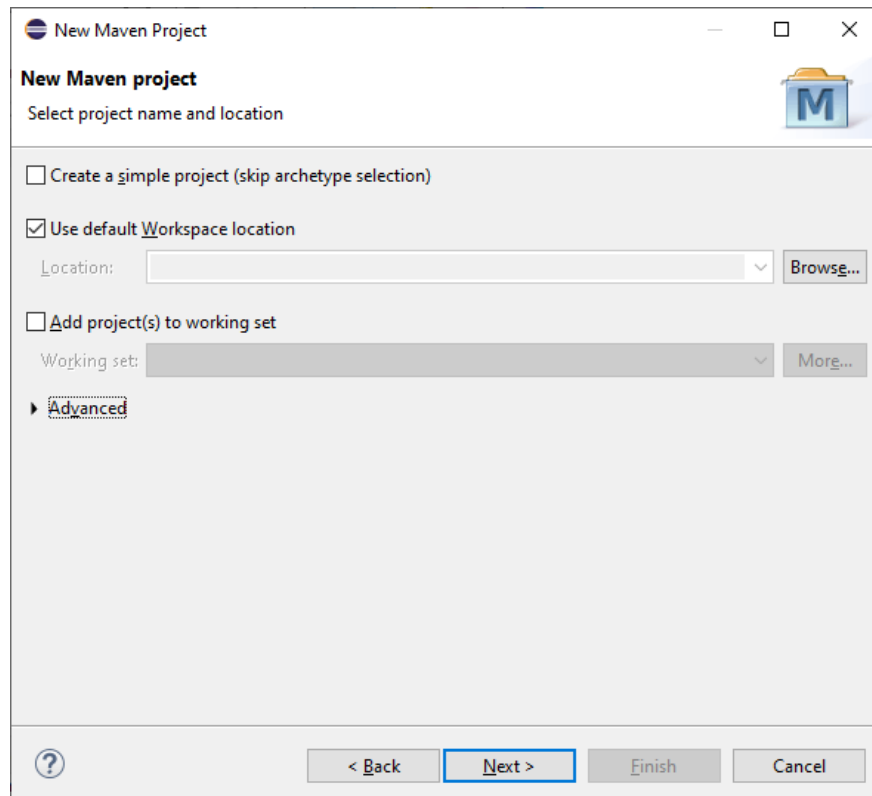
Para ello, iniciamos el IDE Eclipse y creamos un Nuevo proyecto: File -> New -> Other



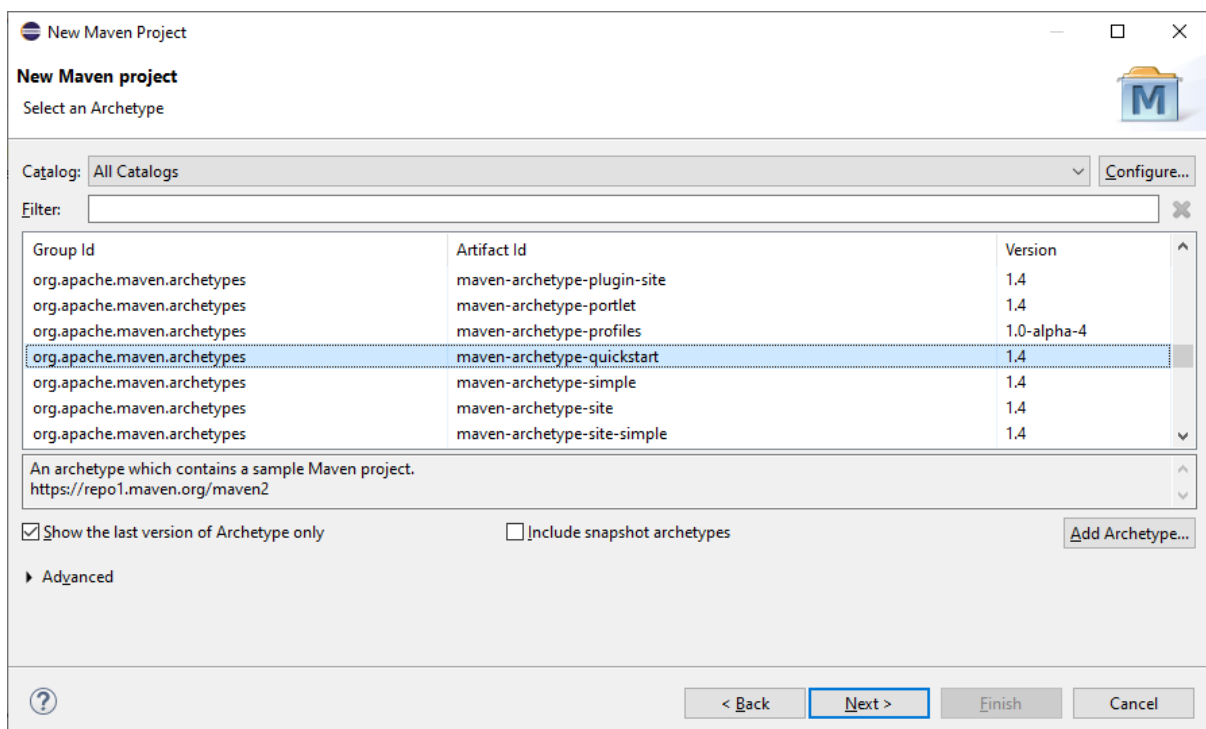
En el tipo de proyecto a crear seleccionamos Maven Project:



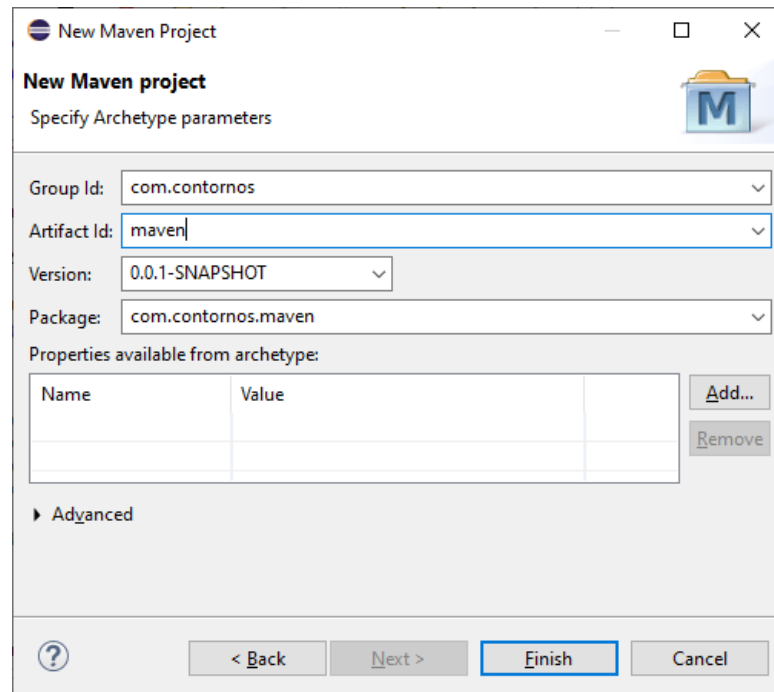
Indicamos el nombre y ubicación de nuestro proyecto:



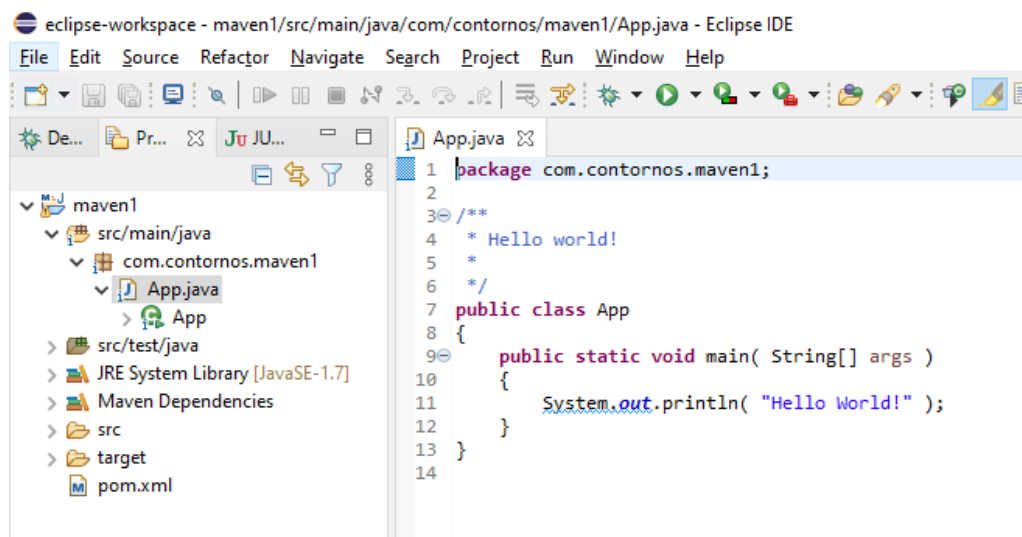
En Archetype seleccionamos quickstart en versión 1.4: **maven-archetype-quickstart**



Le daremos un ID de Grupo (Group ID) y un ID de Artefacto (Artifact ID) para identificar nuestro nuevo proyecto Maven. Por ejemplo, podríamos definir *com.contornos* como Group ID y *maven1* como Artifact ID:



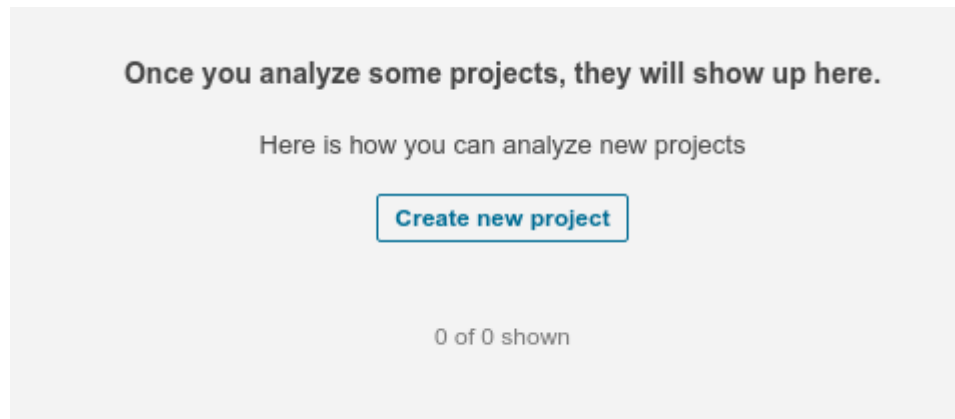
Una vez creado nuestro proyecto tendremos algo como lo siguiente:



2.3 Análisis en SonarQube

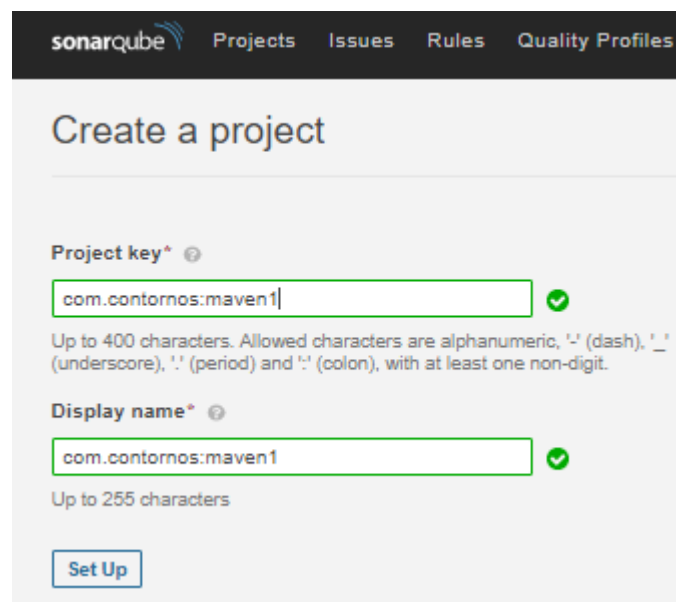
Ahora que hemos iniciado sesión en nuestra instancia local de SonarQube, analicemos un proyecto.

Haz clic en el botón **Crear nuevo proyecto** .



A continuación debemos indicar una clave de proyecto y un nombre a mostrar. En este caso vamos a usar un ejemplo de un proyecto Maven, por lo que debemos indicar:

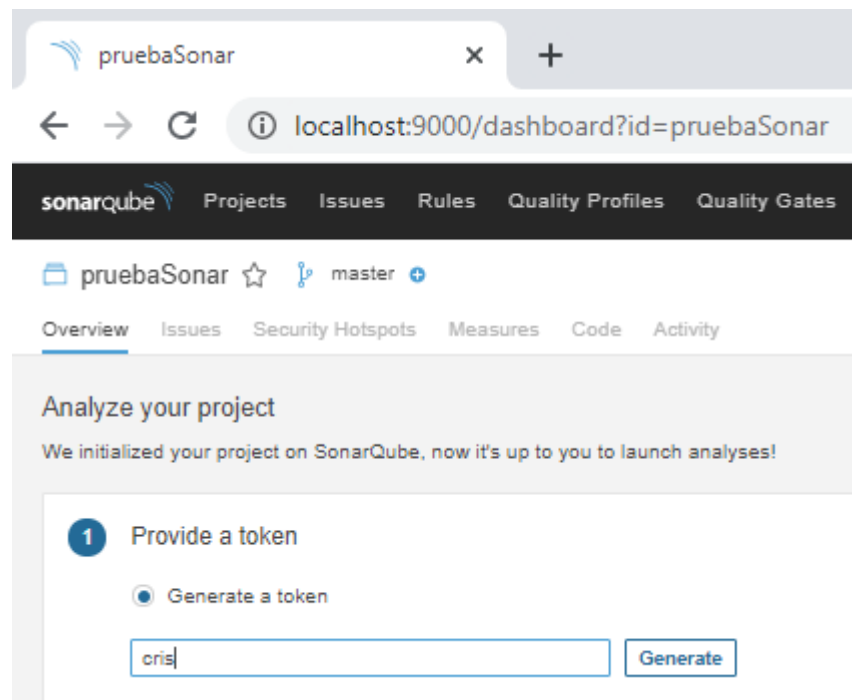
- **clave de proyecto** (esta clave es un identificador único para nuestro proyecto usado por maven): indicamos nuestro groupId y a continuación separado por : nuestro artifactId.



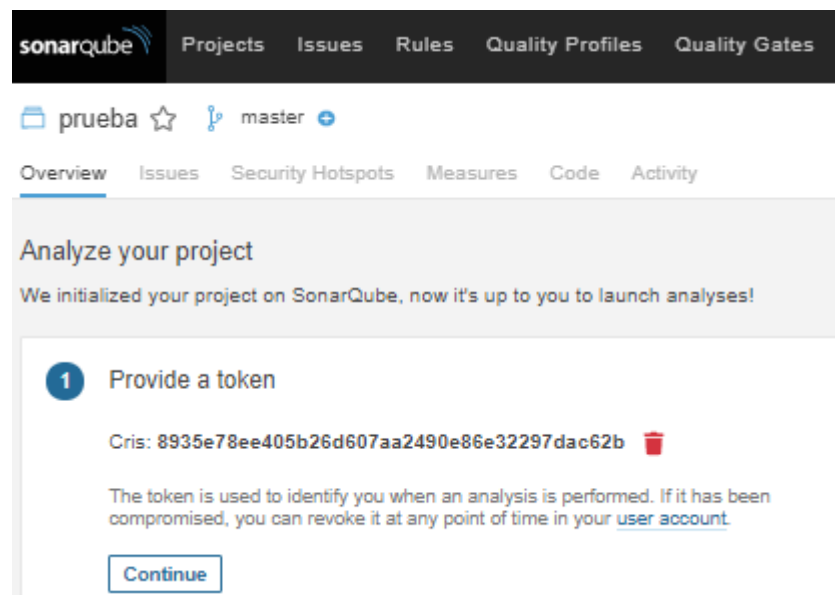
- **nombre para mostrar:** nos pone por defecto el mismo que indicamos en el project key. Podemos cambiarlo si quisiésemos.

Una vez configurado el nombre del proyecto, hacemos clic en el botón **Set Up**.

A continuación, nos pide **Proporcionar un token**, por ejemplo, indicamos nuestro nombre para identificarnos cuando se realice el análisis. Hacemos clic en el botón **Generar**.

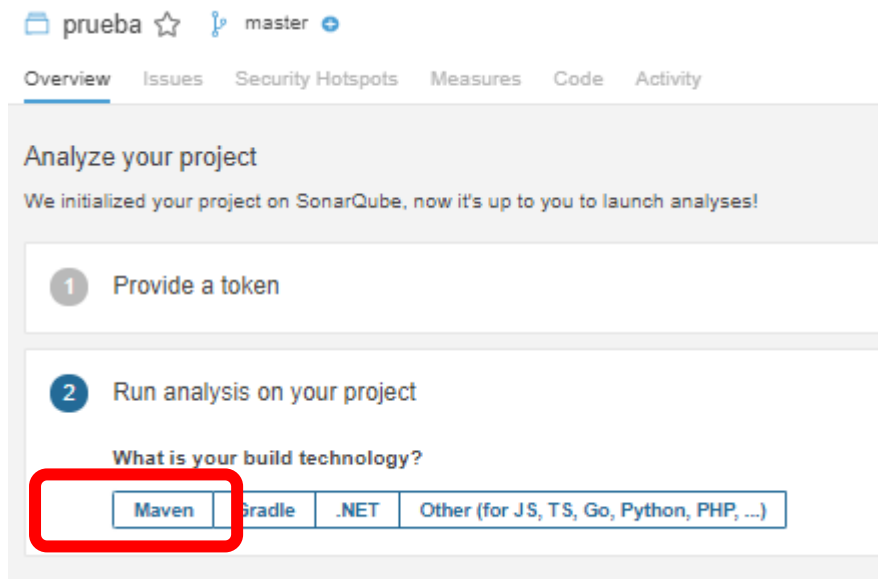


Nos hace una previsualización del token que se ha generado:

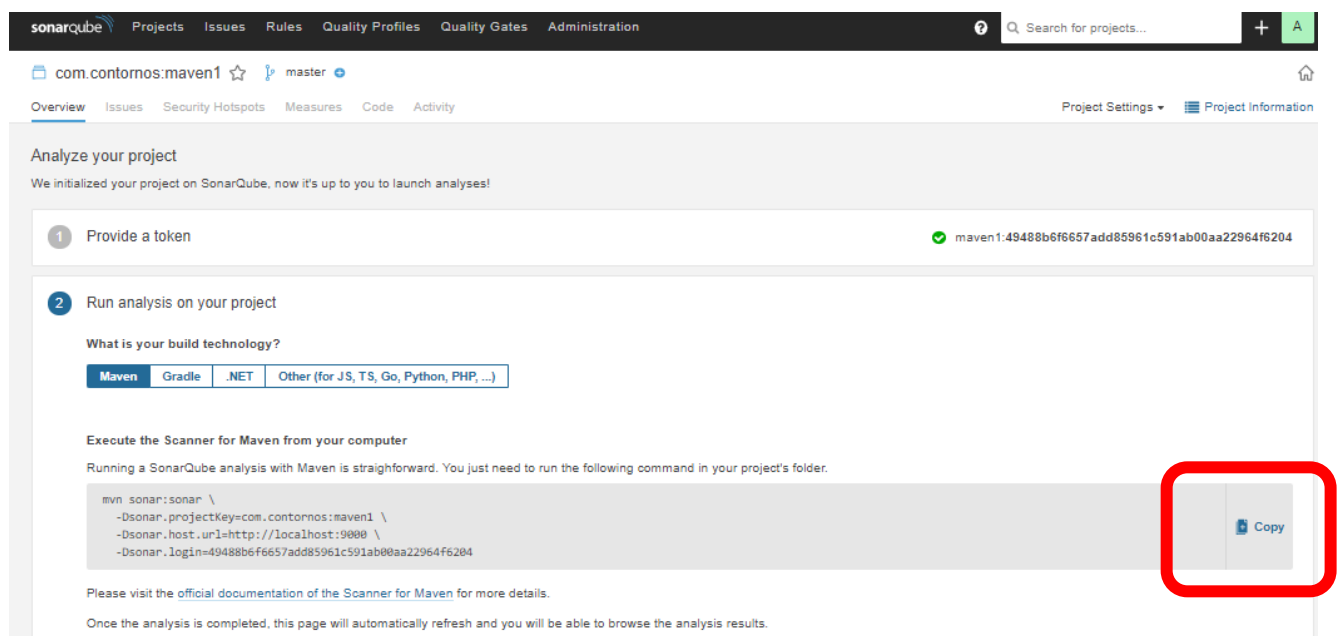


Y a continuación hacemos clic en **Continuar**.

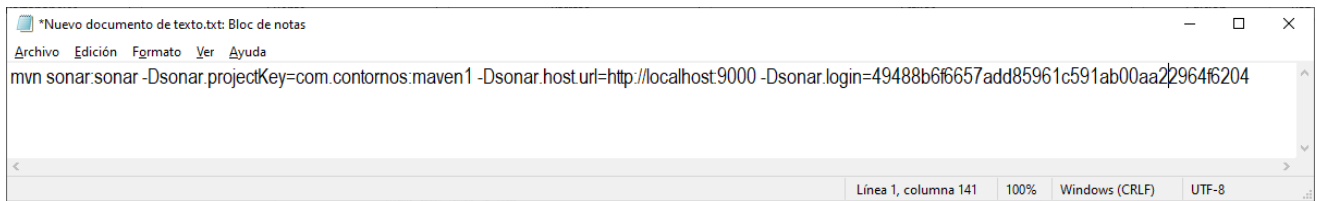
Como último paso para la creación de nuestro proyecto, debemos seleccionar la tecnología que tendrá nuestro proyecto, en nuestro caso Maven:



Y en ese momento nos mostrará unas líneas de código que debemos copiar para realizar el análisis de nuestro proyecto, haciendo uso de un botón que se muestra en el lateral derecho.

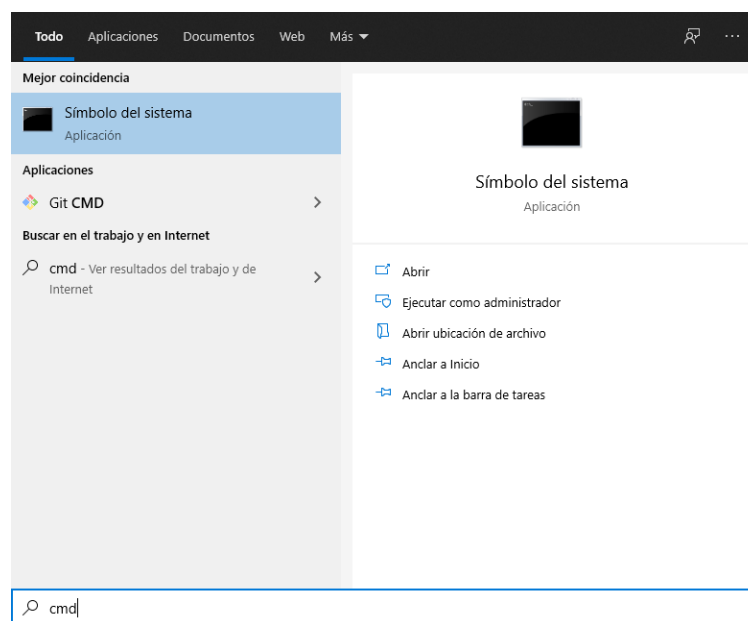


Una vez que hemos pulsado en el botón Copy que nos indica SonarQube, pegamos lo que tenemos en el portapapeles, por ejemplo en un documento de texto, ya que debemos eliminar todas las \ y dejar un único espacio en blanco entre cada bloque. En nuestro caso la línea final sería algo como:



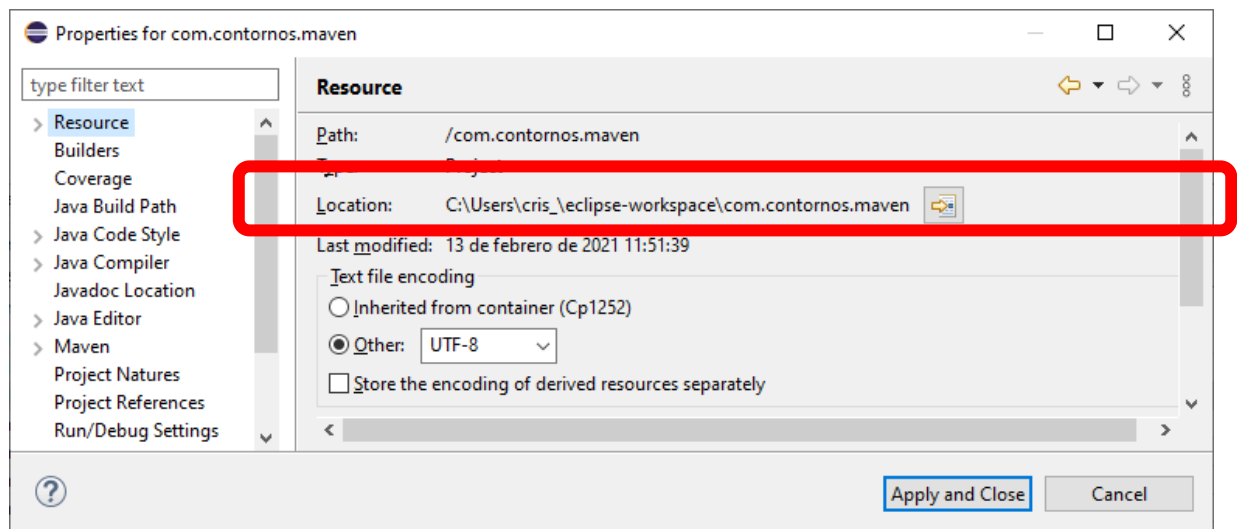
```
mvn sonar:sonar -Dsonar.projectKey=com.contornos:maven1 -Dsonar.host.url=http://localhost:9000 -Dsonar.login=49488b6f6657add85961c591ab00aa22964f6204
```

Una vez que hemos obtenido la línea de ejecución correspondiente, para analizar nuestro proyecto, abrimos una consola de windows:

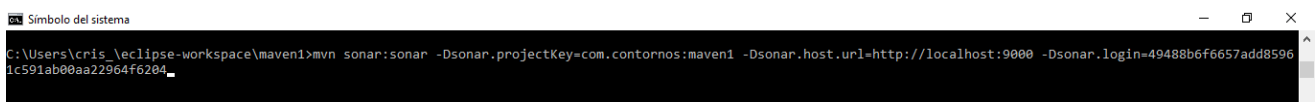


Y nos posicionamos en el workspace donde está guardado nuestro proyecto maven creado en Eclipse. En nuestro caso está en la ruta: `C:\Users\cris_\eclipse-workspace\maven1`

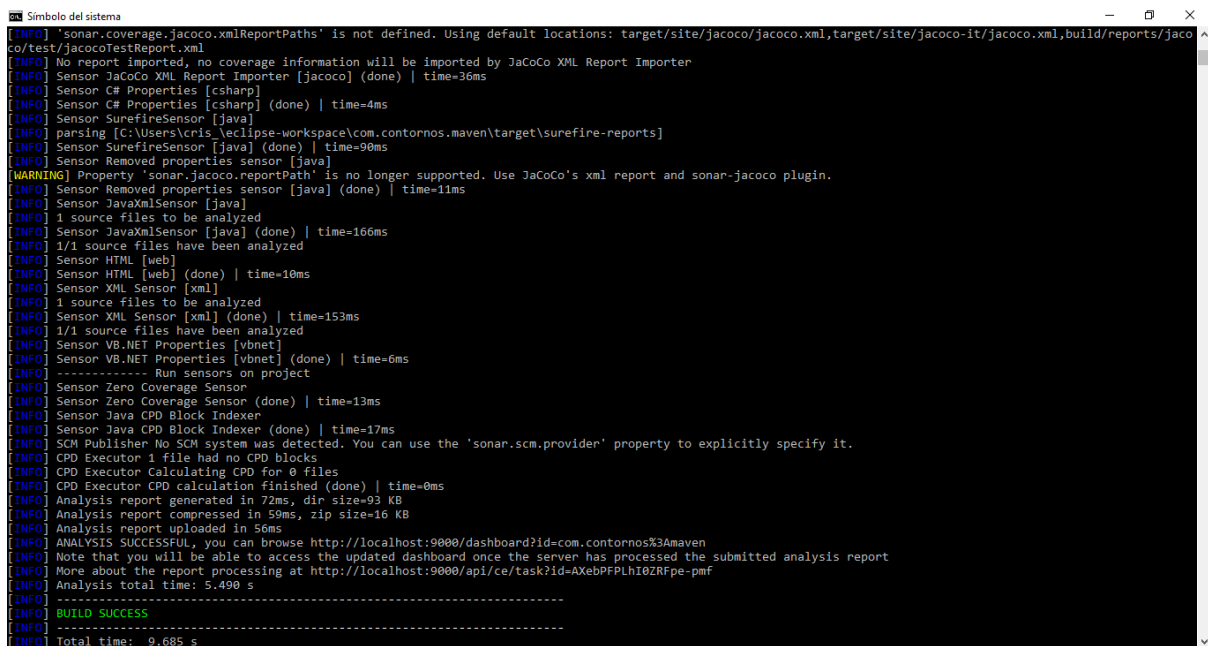
NOTA: Podemos ver la ruta pulsando en Eclipse con el botón derecho sobre nuestro proyecto -> Properties:



Una vez copiada la línea del editor de texto, pinchamos sobre la consola lo que hace que se pegue y pulsamos Intro para que se ejecute:

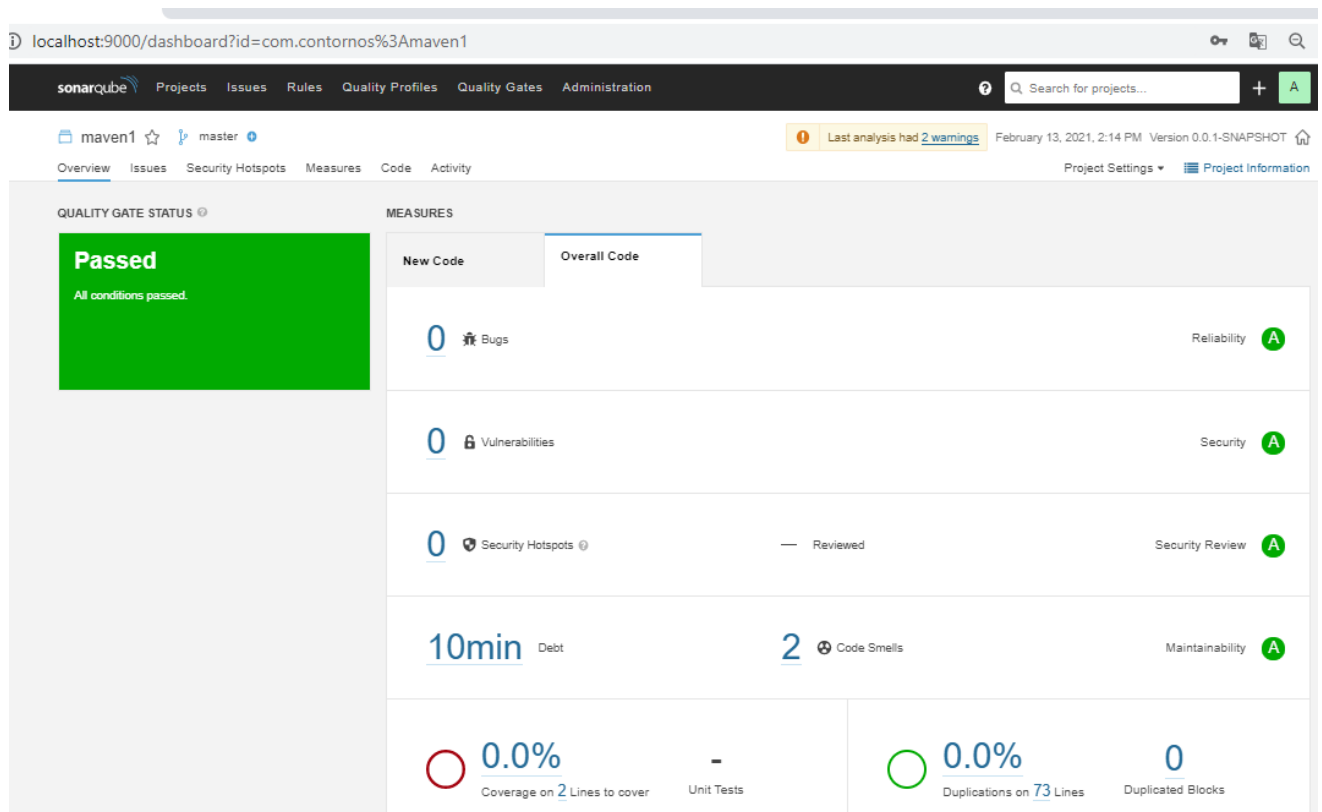


Ahora se empezará a analizar en SonarQube:



La primera vez que lo analizamos tardará un poco más tiempo que el resto de las ocasiones.

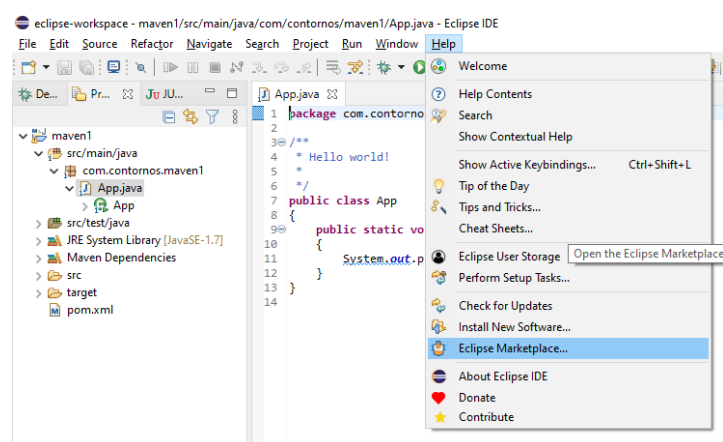
Si ahora vamos a la aplicación web, podremos ver los resultados del análisis ejecutado:



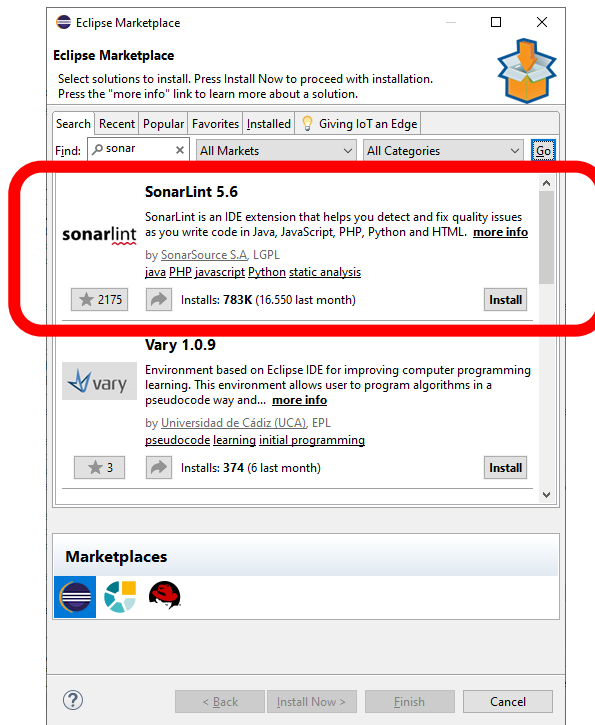
3. Análisis con Sonar directamente en Eclipse

3.1 Instalar y configurar plugin SonarLint

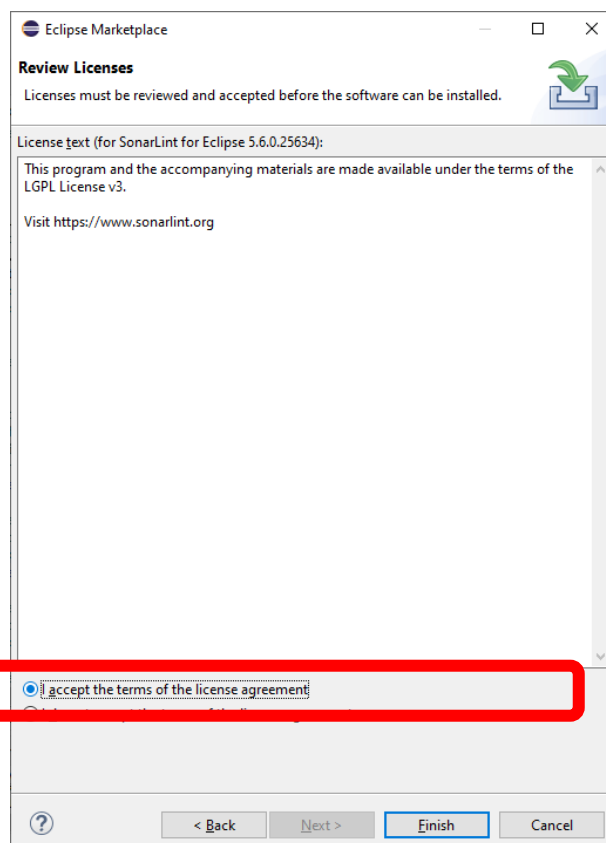
Para instalar el plugin de Sonar en Eclipse, debemos ir al MarketPlace a través de Help -> Eclipse MarketPlace:



Realizamos una búsqueda indicando Sonar e instalamos SonarLint:

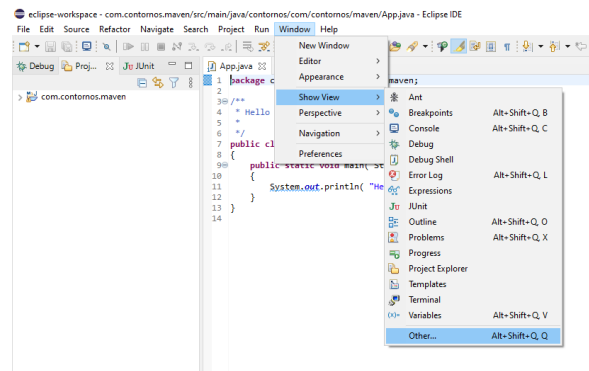


Aceptamos los términos de la licencia y pulsamos Finish:

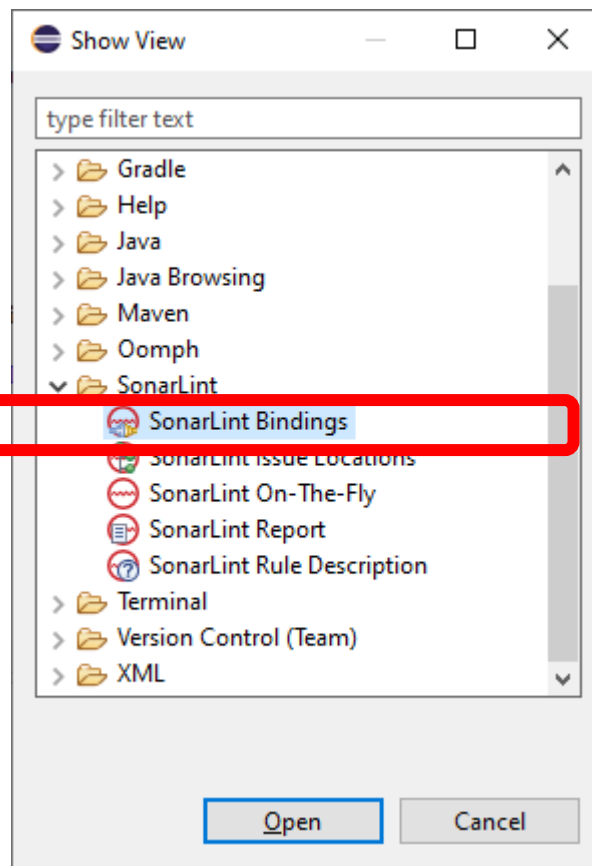


Una vez pulsado Finish, se nos reiniciará el Eclipse.

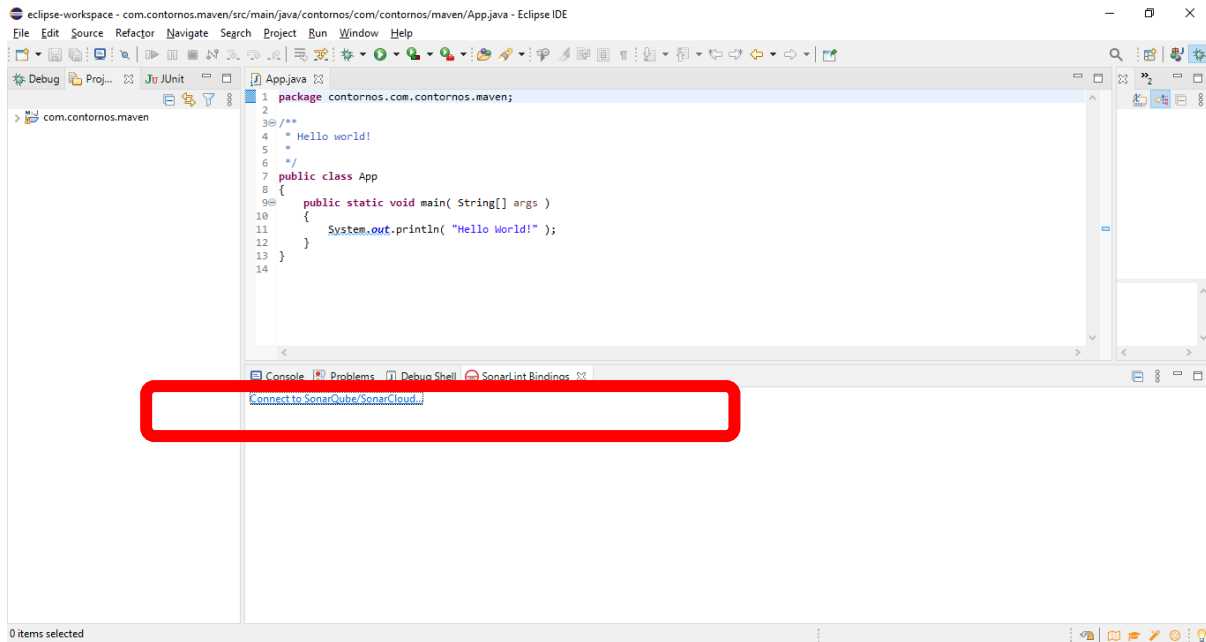
Una vez que se vuelva a iniciar, desde el menú Window -> Show View -> Other...



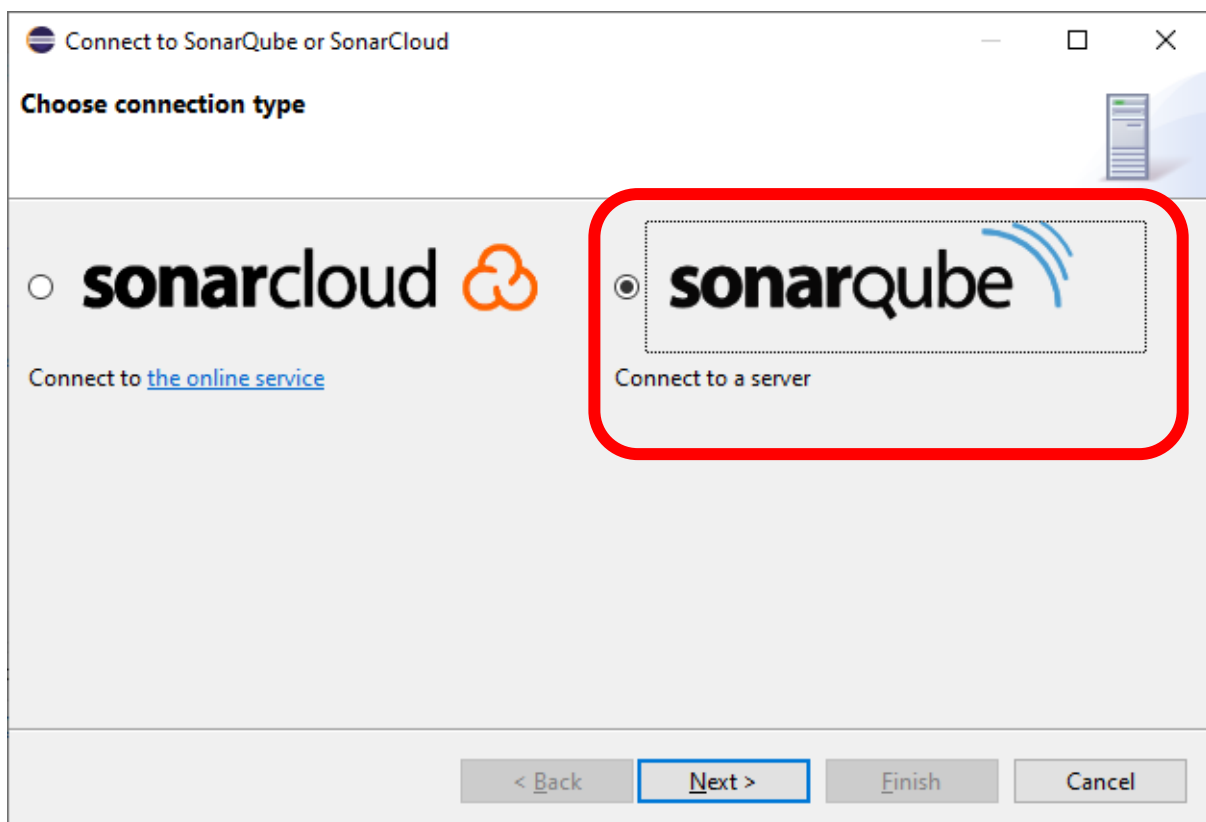
Configuramos SonarQube, seleccionando SonarLint Bindings:



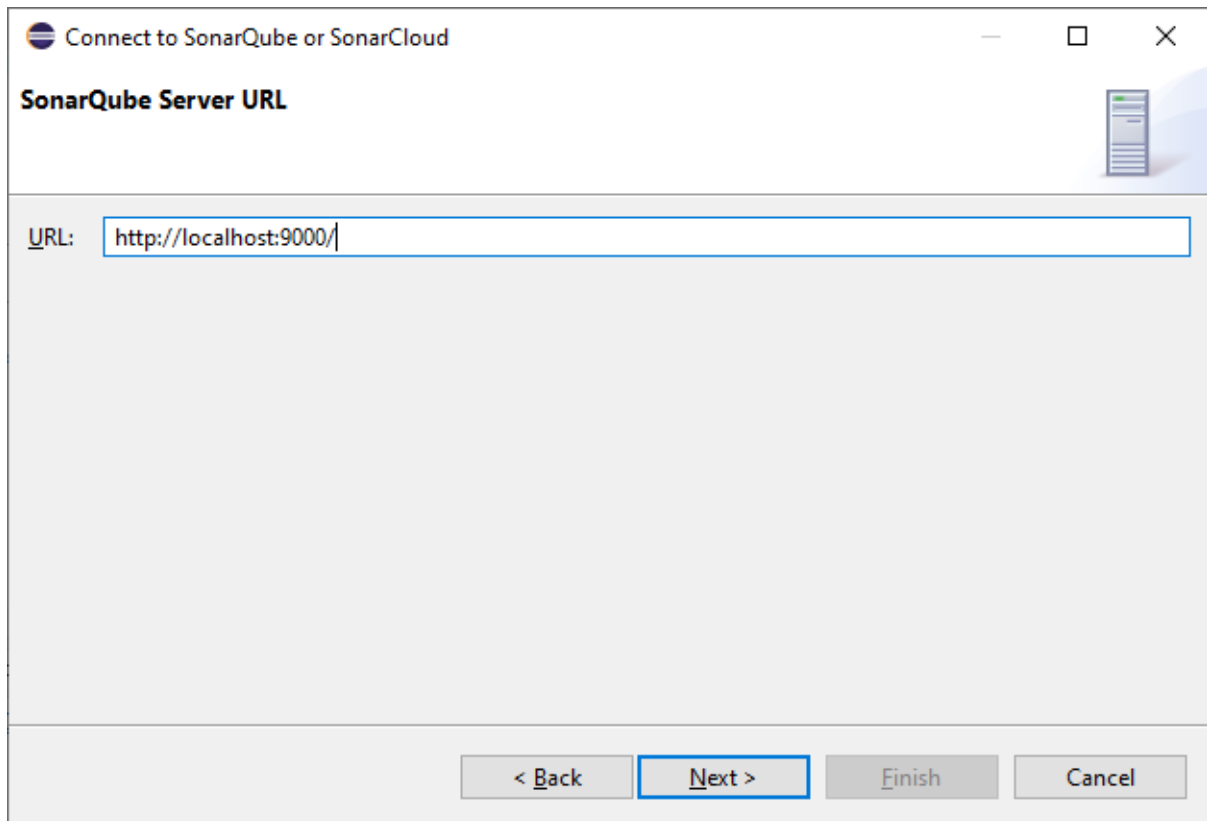
Ahora nos pide en la pestaña de Sonar Bindings, conectar con un SonarQube. Pinchamos en el enlace que se nos muestra:



Y se nos muestra la siguiente ventana donde seleccionamos sonarQube -> connect to a server:

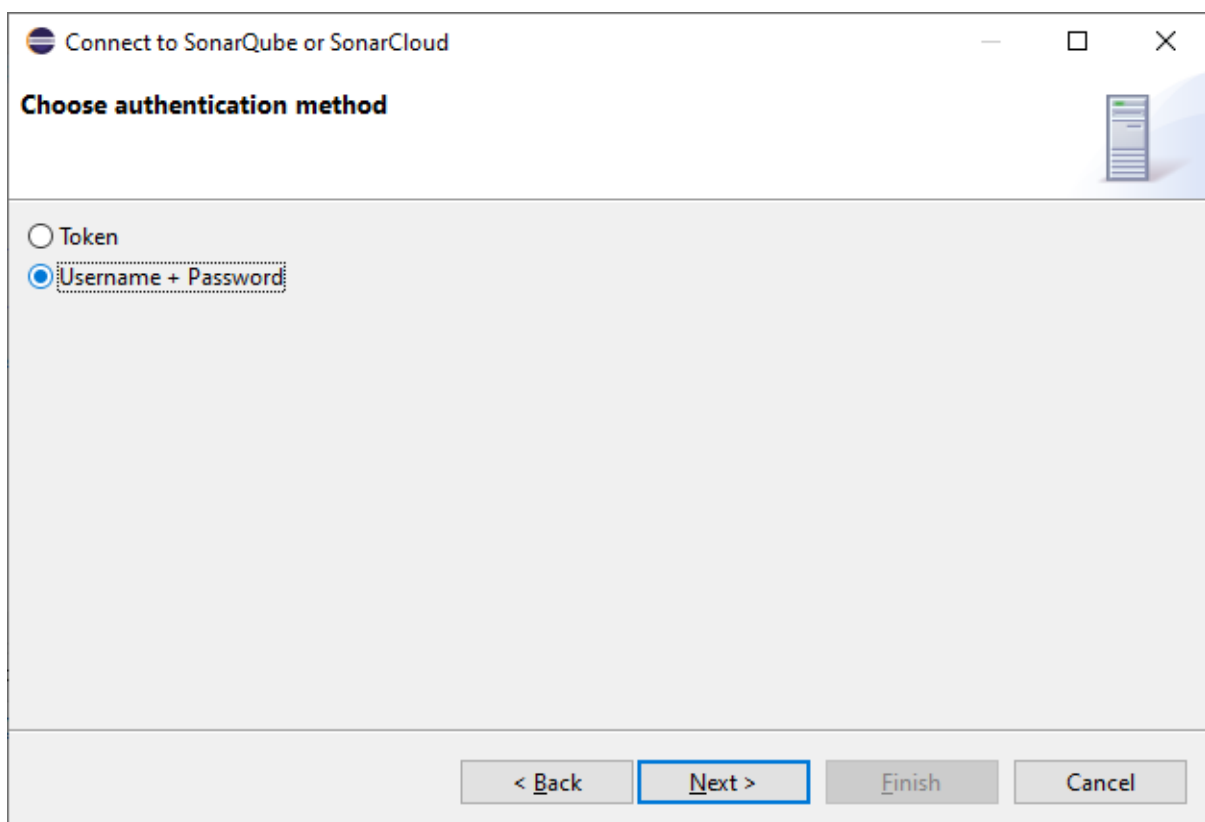


Pulsamos en Next y configuramos la URL de SonarQube, que en nuestro caso actual es: localhost y el puerto 9000.

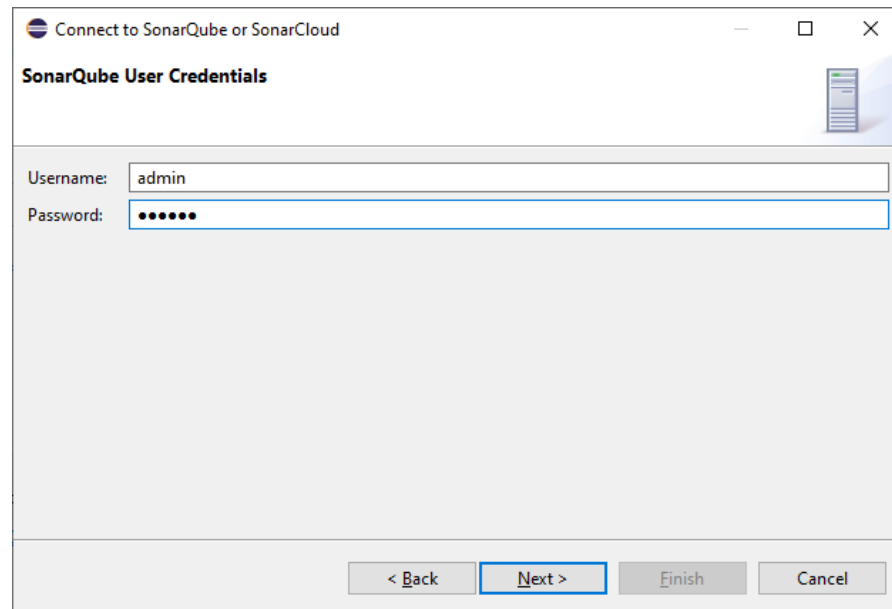


The screenshot shows a Windows-style dialog box titled "Connect to SonarQube or SonarCloud". The main heading is "SonarQube Server URL". Below this, there is a text input field labeled "URL:" containing the text "http://localhost:9000/". At the bottom of the dialog, there are four buttons: "< Back", "Next >", "Finish", and "Cancel". The "Next >" button is highlighted with a blue border.

Indicamos el Token o el usuario y password que hayamos configurado:



The screenshot shows the same dialog box, but now the heading is "Choose authentication method". There are two radio button options: "Token" and "Username + Password". The "Username + Password" option is selected, indicated by a blue dot. At the bottom, the same four buttons are present: "< Back", "Next >", "Finish", and "Cancel". The "Next >" button is highlighted with a blue border.



Connect to SonarQube or SonarCloud

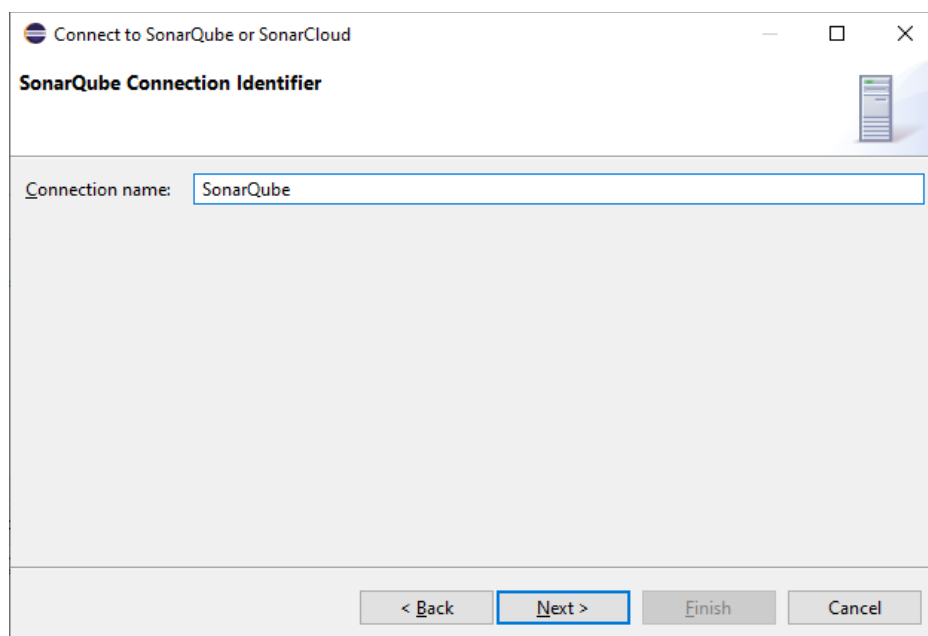
SonarQube User Credentials

Username:

Password:

< Back **Next >** Finish Cancel

Le damos un nombre a la conexión:

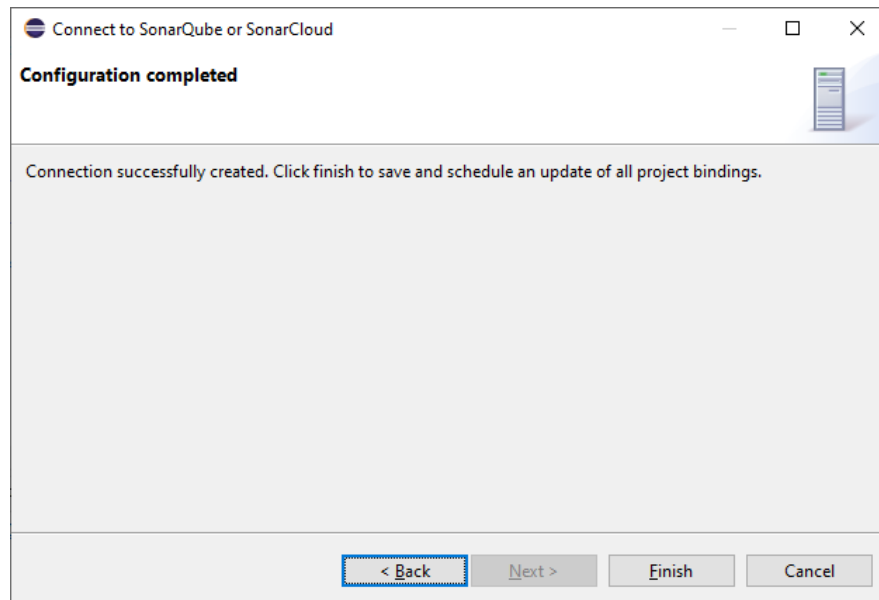


Connect to SonarQube or SonarCloud

SonarQube Connection Identifier

Connection name:

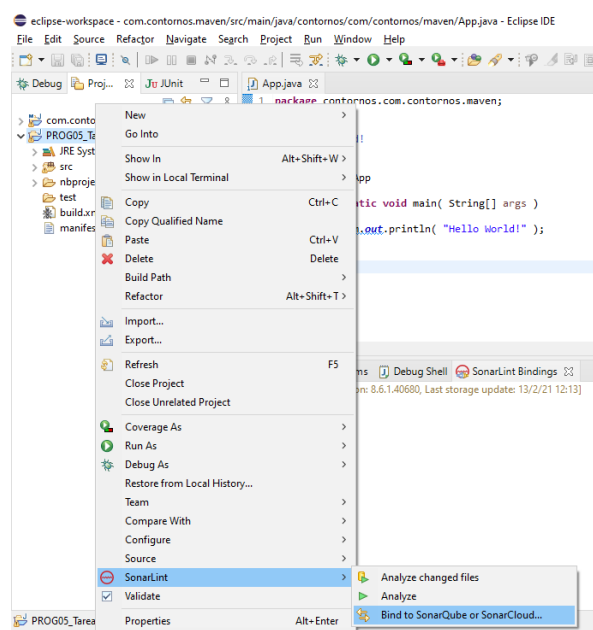
< Back **Next >** Finish Cancel

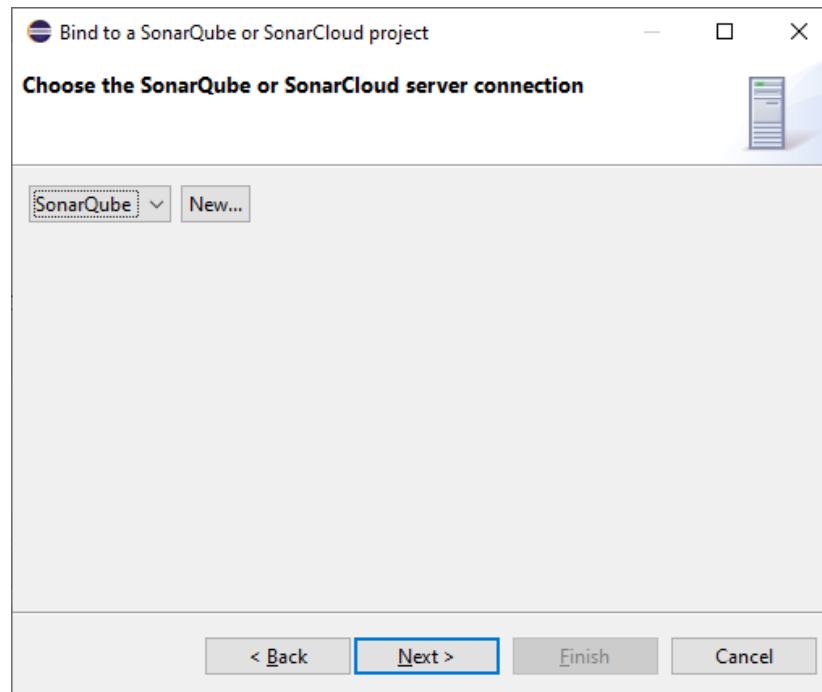


En este momento ya tenemos nuestro Eclipse enganchado con SonarQube.

3.2 Asociar proyecto Eclipse con Sonar

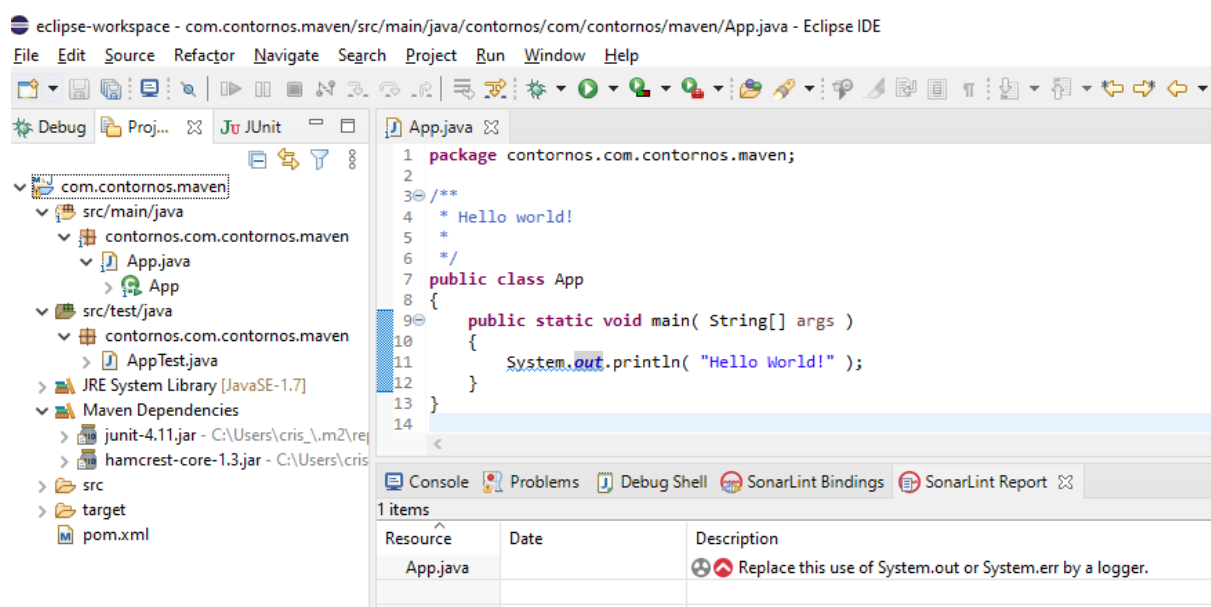
Ahora lo que tendríamos que hacer es coger nuestro proyecto y asociarlo a SonarLint. Para ello, con el botón derecho sobre nuestro proyecto, seleccionamos SonarLint -> Bind to SonarQube or SonarCloud..





Escribimos el nombre del proyecto que hemos creado en SonarQube y pulsamos en Finish. Una vez que tenemos asociado nuestro proyecto local con el proyecto en SonarQube, podemos analizar nuestro código con el botón derecho sobre nuestro proyecto en Eclipse -> SonarLint -> Analyze.

En la pestaña Problems, nos muestra lo que va encontrando a medida que nosotros vamos programando.



Description	Resource	Path	Location	Type
Warnings (2 items)				
Build path specifies execution environment JavaSE-1.7. The compiler compliance specified is 1.7 but a JRE 14 is used	com.contorn...		Build path	JRE System Library Problem
The compiler compliance specified is 1.7 but a JRE 14 is used	com.contorn...		Compiler Com...	JRE Compiler Compliance Problem
Infos (2 items)				
Replace this use of System.out or System.err by a logger.	App.java	com.contornos.maven/src/main/java/co...	line 11	SonarLint On-The-Fly Issue
Replace this use of System.out or System.err by a logger.	App.java	com.contornos.maven/src/main/java/co...	line 11	SonarLint Report Issue

Podemos configurar el nivel que queremos que nos muestre Sonar. Si vamos a Window -> Preferences -> SonarLint, dentro del campo Severity of SonarLint markers, podemos indicar si queremos que nos muestre sólo errores o warnings.

En el caso de que seleccionemos Error, si pulsamos en Apply, veremos que si en nuestro proyecto Sonar detecta algún error, nos avisará de en que fichero java se encuentra.

Description	Resource	Path	Location	Type
Errors (73 items)				
Add a default case to this switch.	Principal.java	PROG05_Tarea/src/prog05/ejerc1	line 57	SonarLint On-The-Fly Issue
Declare "anio_mat" on a separate line.	Principal.java	PROG05_Tarea/src/prog05/ejerc1	line 48	SonarLint On-The-Fly Issue
Declare "descripcion" on a separate line.	Principal.java	PROG05_Tarea/src/prog05/ejerc1	line 46	SonarLint On-The-Fly Issue
Declare "dni_propietario" on a separate line.	Principal.java	PROG05_Tarea/src/prog05/ejerc1	line 46	SonarLint On-The-Fly Issue
Declare "matricula" on a separate line.	Principal.java	PROG05_Tarea/src/prog05/ejerc1	line 46	SonarLint On-The-Fly Issue
Declare "mes_mat" on a separate line.	Principal.java	PROG05_Tarea/src/prog05/ejerc1	line 48	SonarLint On-The-Fly Issue
Declare "precio" on a separate line.	Principal.java	PROG05_Tarea/src/prog05/ejerc1	line 47	SonarLint On-The-Fly Issue
Declare "propietario" on a separate line.	Principal.java	PROG05_Tarea/src/prog05/ejerc1	line 46	SonarLint On-The-Fly Issue

De esta forma, a medida que vamos programando, podemos ver los errores, warnings, etc y de esta forma ir optimizando nuestro código en tiempo real a medida que avanzamos en el proyecto.

4. Recursos

- Documentación SonarQube: <https://docs.sonarqube.org/latest/>
- Página eclipse: <https://www.eclipse.org/>