ASU CST216 Team 22

Checkstyle

A static analysis tool for Java

Contents

[1. Originator 2](#_Toc448337268)

[2. Purpose 2](#_Toc448337269)

[3. Programming Language 2](#_Toc448337270)

[4. Effective Use - What Phase? 2](#_Toc448337271)

[5. Tool Usage & Installation 3](#_Toc448337272)

[Installation 3](#_Toc448337273)

[6. Configuration 4](#_Toc448337274)

[7. How to use 6](#_Toc448337275)

[8. Checkstyle Strengths 10](#_Toc448337276)

[9. Checkstyle Limitations 10](#_Toc448337277)

[Appendix A 11](#_Toc448337278)

**Team 22** CST216 Spring B 2016 Arizona State University **Project - I3**

Category: Test Tool, Unit Test, Checkstyle

# 1. Originator

The Checkstyle tool was developed by Oliver Burn and released to public domain for over two decades under GNU Public License (GPL.) The tool is downloadable through Eclipse marketplace (<https://marketplace.eclipse.org/content/checkstyle-plug>) free of charge.

# 2. Purpose

The Eclipse Checkstyle Plugin (eclipse-cs) integrates the well-known source code analyzer Checkstyle into the Eclipse Integrated Development Environment (IDE.)

Checkstyle is a development tool to help Java programmers ensure that their Java code adheres to a set of coding standards. Checkstyle does this by inspecting their Java source code and pointing out items that deviate from a defined set of coding rules.

Checkstyle is easy to install, configure and use. The tool is very visual and can be used during development process to make corrections on the go. With the Eclipse Checkstyle Plugin the programmers’ code is constantly inspected for problems.

Within the Eclipse workbench they are notified of problems via the Eclipse Problems View and source code annotations just as they would see with compiler errors or warnings.

A comfortable Checkstyle configuration editor helps Java programmers create and maintain their audit rule configurations by using filters and file sets which enable Java programmers to define which files get checked and which do not.

The primary use of this tool is to enforce coding rules defined by software development teams.

See more at: [https://marketplace.eclipse.org/content/checkstyle-plug#sthash.z57MUmcE.dpuf](https://marketplace.eclipse.org/content/checkstyle-plug%23sthash.z57MUmcE.dpuf).

# 3. Programming Language

Currently Checkstyle only supports the Java programming language.

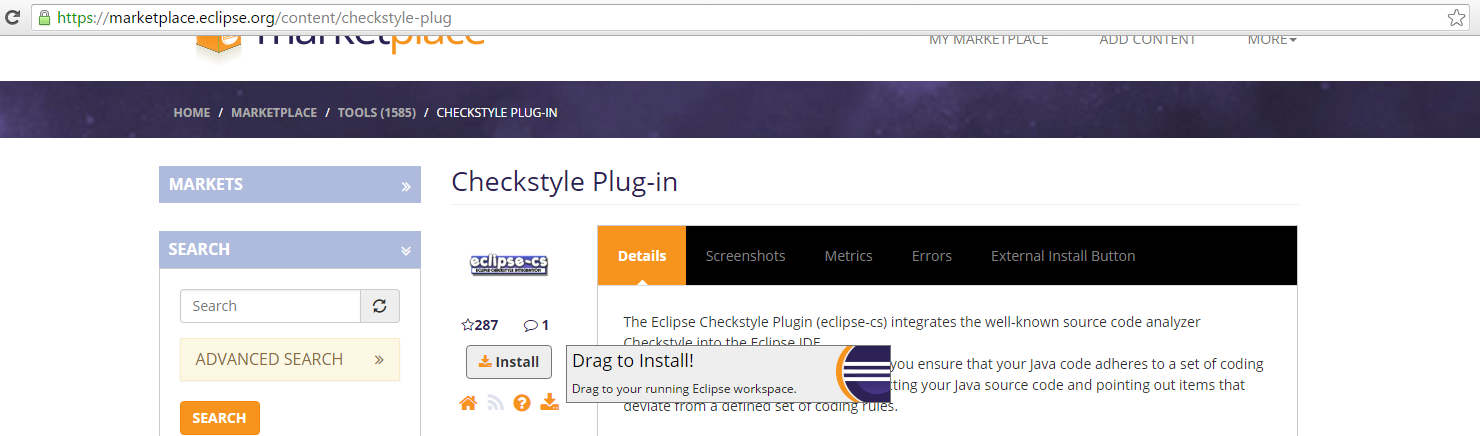
# 4. Effective Use - What Phase?

The Checkstyle tool is most effective during Unit Testing phase where it is a part of static analysis of the code base.

# 5. Tool Usage & Installation

## Installation

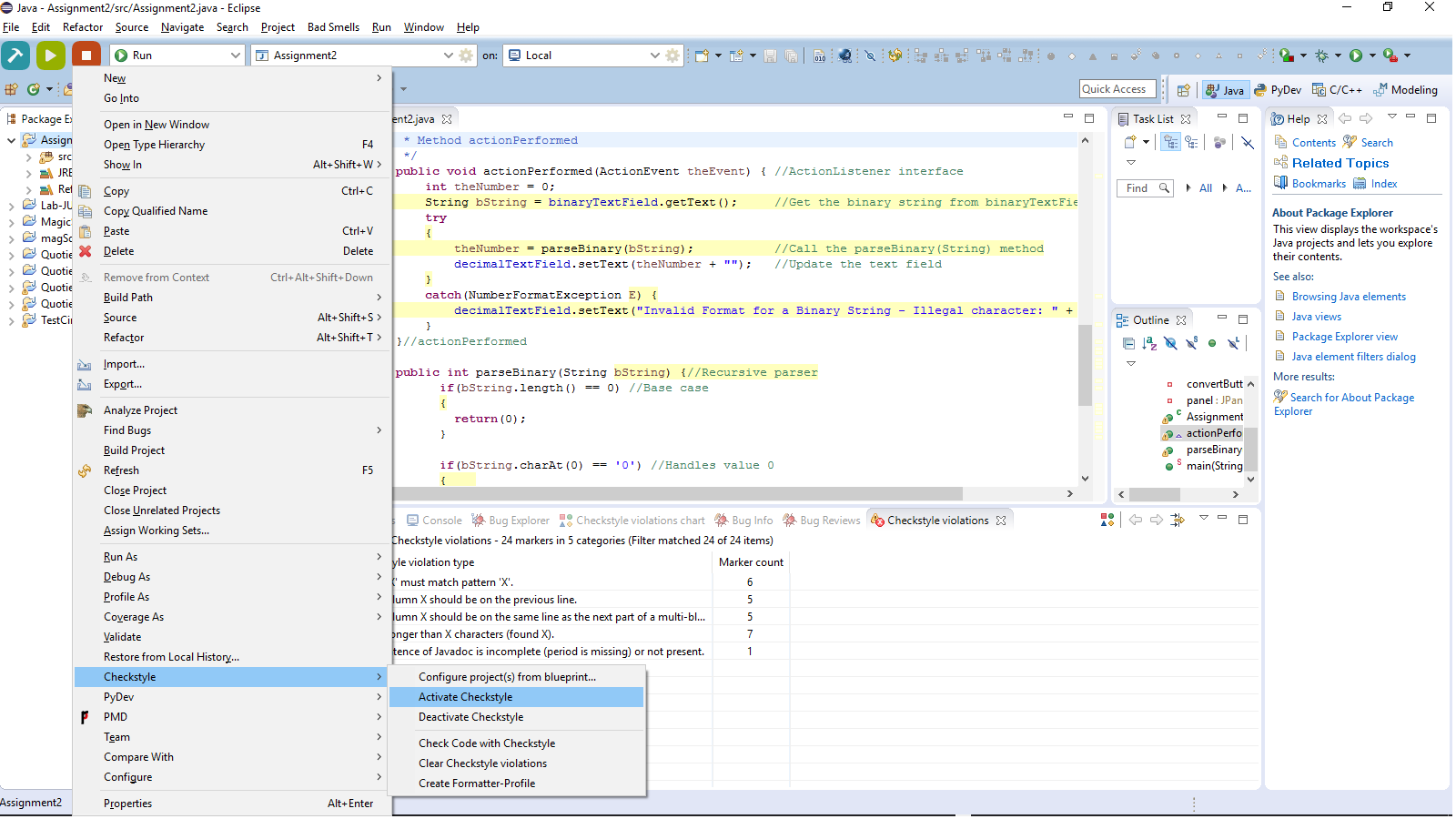
* Invoke the Eclipse Integrated Development and wait for it to completely open.
* Use any browser, navigate to <https://marketplace.eclipse.org/content/checkstyle-plug> site.
* Mouse over to the [Install] button on the web site, single left-click and hold down the left mouse button to drag the [Install] button onto the Eclipse IDE. See picture below:



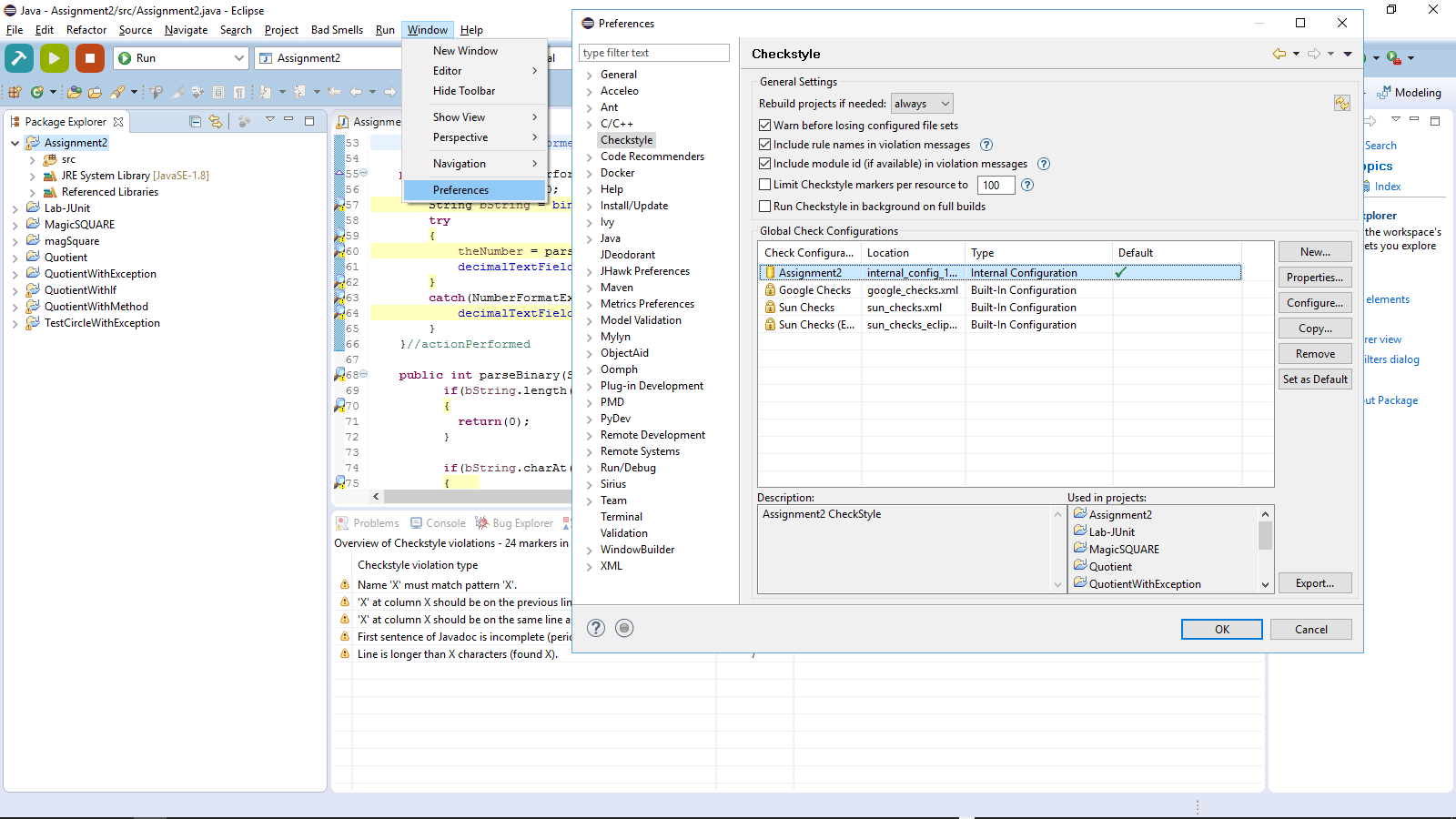
* Follow Eclipse IDE screen prompts to complete the installation. The Eclipse IDE will be restarted after the installation.

# 6. Configuration

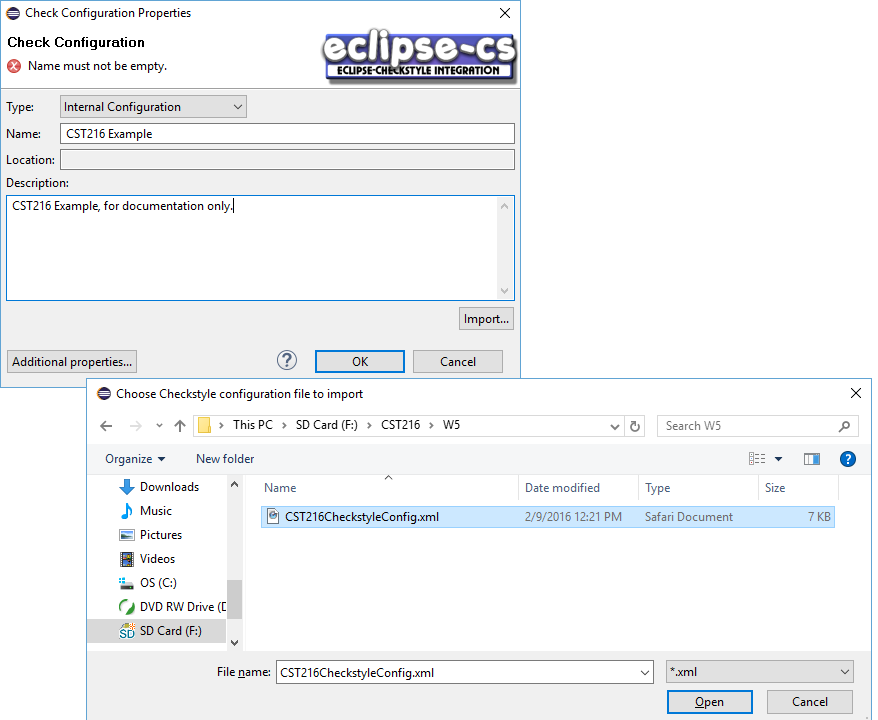
The Checkstyle tool works in concert with an external rule file written in Extensible Markup Language (XML) format which must be activated in Eclipse IDE. See picture below.



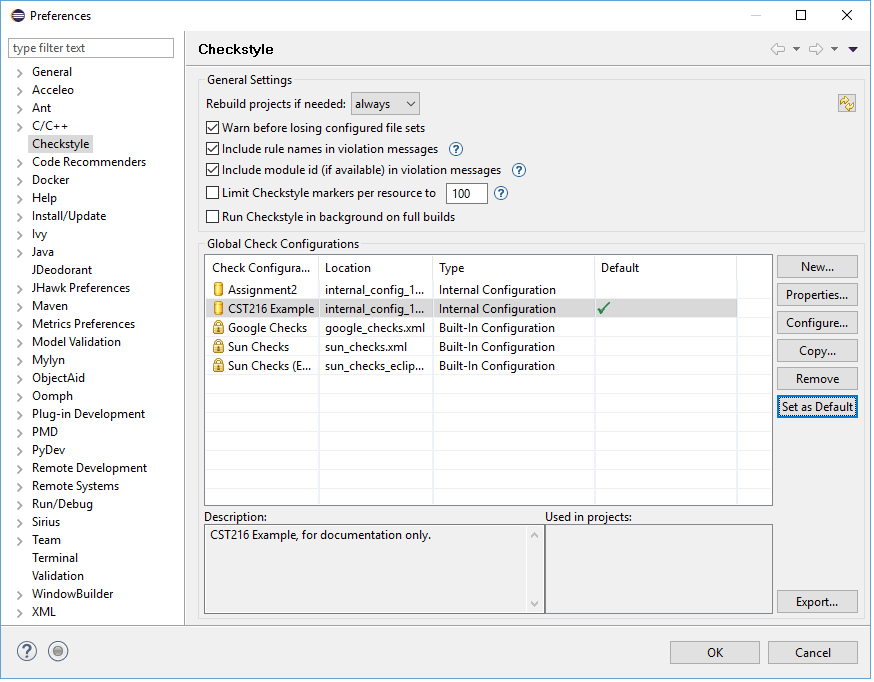
After activation, Checkstyle tool must be configured within Eclipse IDE. The picture below depicts Checkstyle already configured for use with quite a few projects.



To have a new configuration, click on the [New] button of this window, another window pops up. Leave the type as [Internal Configuration] at this point but enter a name in the [Name] box. Then click on [Import] button and navigate to where the rule (.xml) file resides and bring it in (assuming a natural GUI driven user without much difficulty with the Operating System.) See picture below.



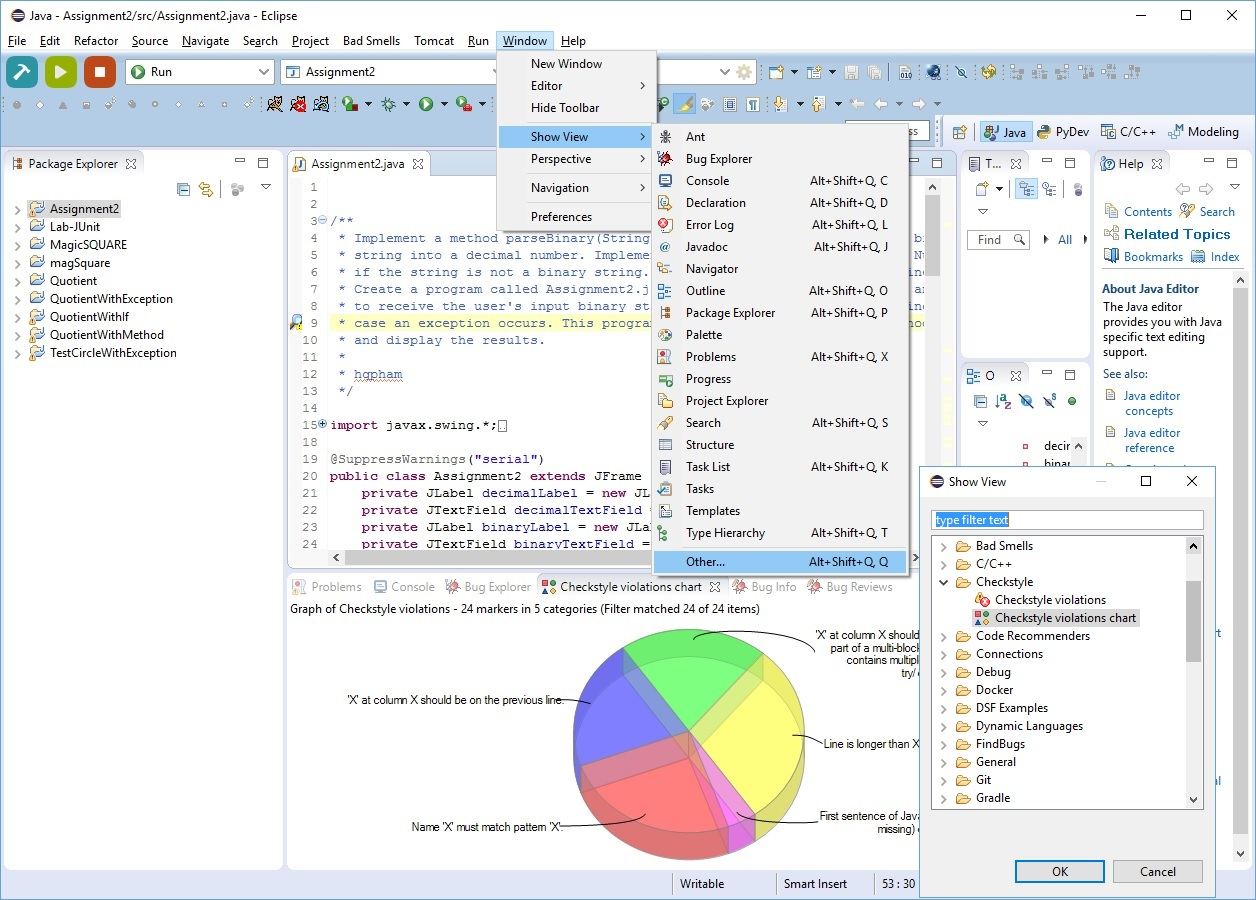
Now the new rule file has been brought in to Eclipse IDE. The user can make it the default rule file if so desired. See picture below.



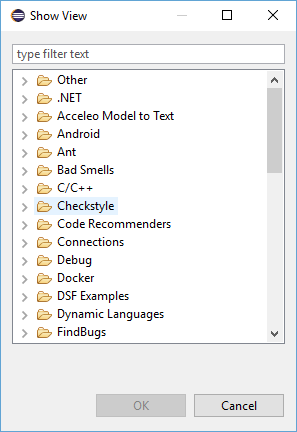
Click [OK] on this window to complete the configuration process.

# 7. How to use

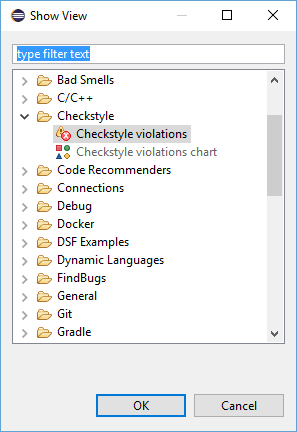
Once configured, Checkstyle takes action right after the project got compiled. To view the output, select [Window] and scroll down to [Show View] and then scroll down to [Other…]. See picture below.



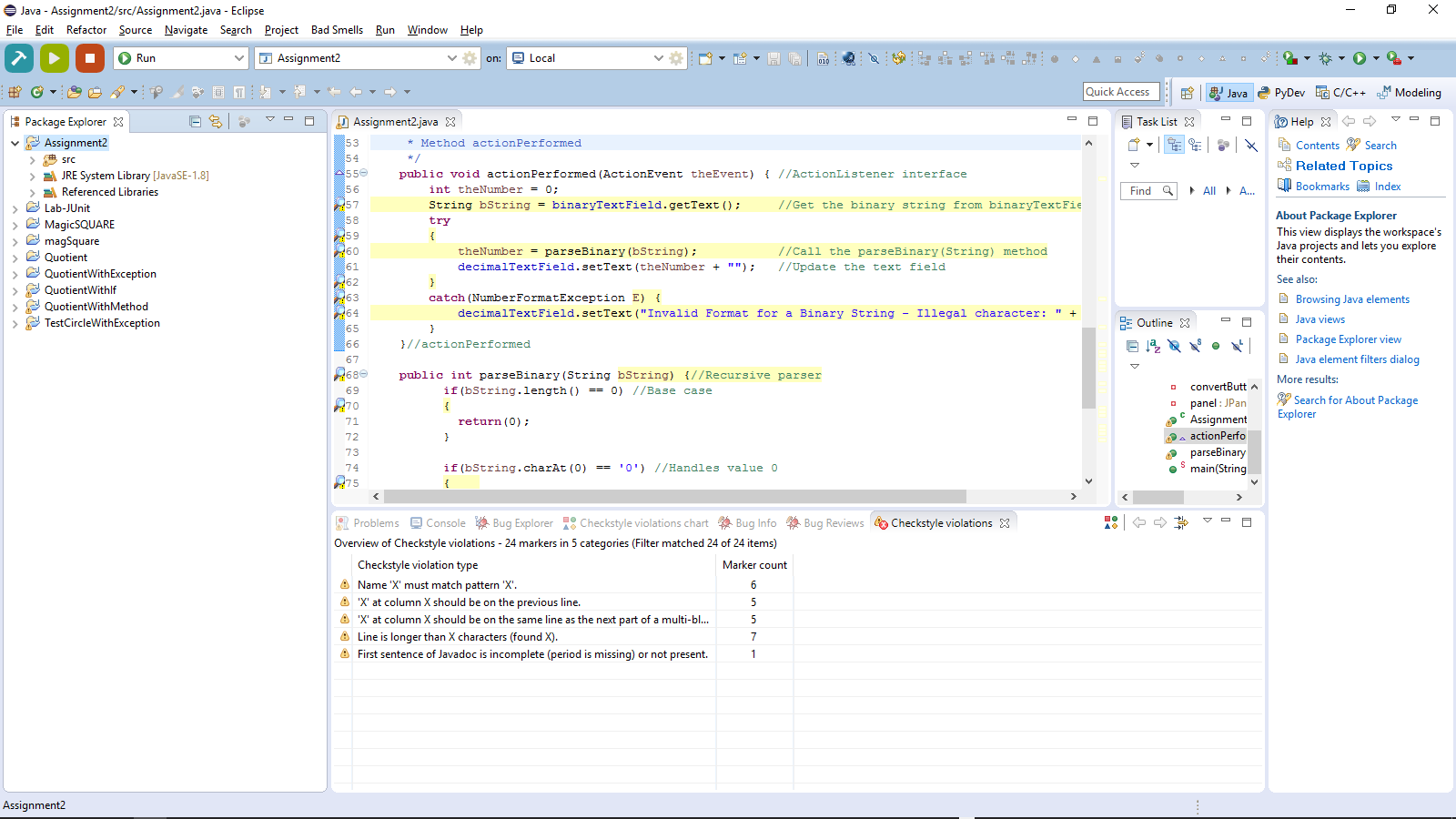
Another window pops up. As shown below.



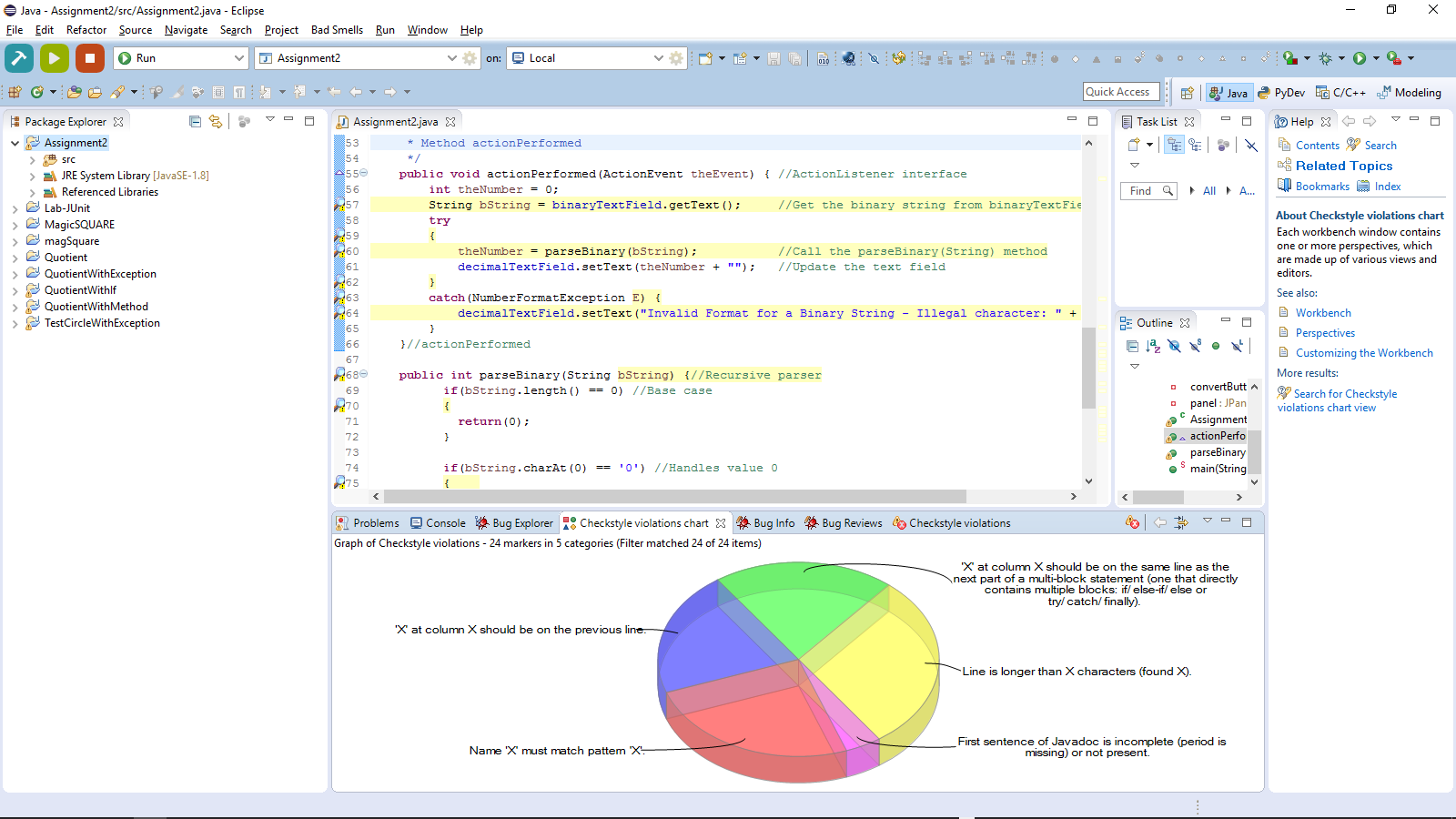
Click on [Checkstyle] and two forms of output are available there: Tablature form, and Pie-chart form. See picture below.



Select desired view and then click [OK] to continue. If the [Checkstyle violations] option was selected, then the output is displayed as shown below:



If the [Checkstyle violations chart] option was selected, then the output is displayed as shown below,



# 8. Checkstyle Strengths

The Checkstyle tool strengths are as follows:

* The programming style adopted by a software development project can help to comply with good programming practices which improve the code quality, readability, re-usability, and reduce the cost of development.

# 9. Checkstyle Limitations

The Checkstyle tool has the following limitations:

* The performed checks mainly limit themselves to the presentation and don't analyze content.
* The performed checks do not confirm the correctness or completeness of the program.
* In practice, it can be tedious to comply with all the style constraints, some of which could possibly harm the programming stages' dynamic; so, it may be useful to determine which level of check is needed for a certain type of program.

# Appendix A

