



This module will cover Setting up custom rules for IBM Rational® Software Analyzer versions 7.0 and higher.

## Module objectives

- Topics covered in this module:
  - ▶ What are rules?
  - ▶ How to write your own rules
  - ▶ Setting up a custom plug-in for the custom rule
- When you complete this module you are able to:
  - ▶ Set up custom rules in Rational Software Analyzer



This course covers three topics. It provides you with information about what a rule is, how to write your own rules and how to set up a custom plug-in for the custom rule. When you complete this module you are able to set up custom rules in Rational Software Analyzer.

## Custom rules in Rational Software Analyzer

- Rational Software Analyzer ships with a set of pre-defined rules, but you want to define your own rules.
- Rational Software Analyzer is a collection of plug-ins. These plug-ins define all the Rational Software Analyzer behavior including the rules and their definitions.
- To add new rules you need to expand the plug-ins by adding your own custom plug-ins to those already there.



When Rational Software Analyzer is installed it comes with its own set of pre-determined rules ready for analysis. However these rules are by no means exhaustive. In fact part of the purpose of Rational Software Analyzer is to allow you to create rules to bring their own form of governance to the code base. Rational Software Analyzer is a collection of plug-ins, so the rules are plug-ins as well. In order to create a new rule you need to create a new plug-in to extend the Rational Software Analyzer framework.

## How to write your own rules

- Eclipse offers a plug-in creation wizard that will help you get started.
  - ▶ In the Eclipse shell go to **File -> New -> Project -> Plug-In Project**.
- All the defaults will work except you will need to uncheck the box that states “**This plug-in makes contributions to the UI**”.



Creating a plug-in can appear to be a difficult task, but Eclipse has a wizard in place to do most of the work for you. The Eclipse plug-in wizard will create the project and all the preliminary configuration information. This will create the basis from which you can create your Rational Software Analyzer custom rule. Other various values can be changed to something meaningful to you.

## Setting up a custom plug-in

- Steps to set up the custom plug-in
  - ▶ Once the plug-in Project has been generated, find the project in the navigator.
  - ▶ Open the plug-in.xml file.
  - ▶ Go to the Dependencies tab. Plug-ins that are required for this plug-in. The custom plug-in will always require:
    - com.ibm.rsaz.analysis.core – This holds all the core functionality Software Analyzer stuff.
    - For Java™
      - com.ibm.analysis.codereview.java
      - com.eclipse.jdt.core
    - For C
      - com.ibm.analysis.codereview.cpp
      - com.eclipse.cdt.core

Now that you have run the wizard and you have your preliminary plug-in, you will need to tailor it to make it grab all the Rational Software Analyzer information. You will do this through the plug-in.xml file. This file holds all the configuration information for the plug-in. Right now it is a blank slate. You will add in some dependencies. These are the plug-ins that your plug-in will look for to extend. You will configure it to look for the Rational Software Analyzer plug-ins you need. For the purposes of this presentation, proceed with making this plug-in a Java rule.

## Setting up a custom plug-in: Extending

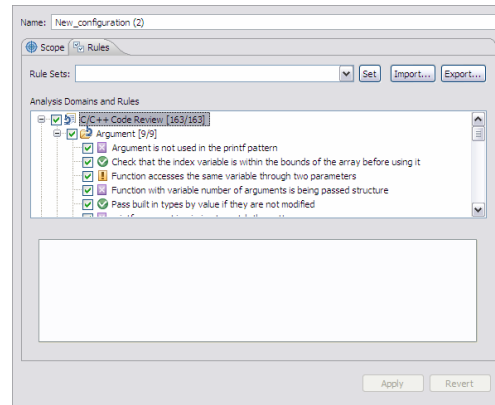
- From the same plugin.xml file, go to the Extensions tab and add a new Extension  
`com.ibm.rsaz.analysis.core.analysisCategory`. This will create the rule category for where this rule is placed in Rational Software Analyzer.
- Set these values in the extension:
  - ▶ **id** - Set to a unique value
  - ▶ **Label** – This is how the rule shows in the Rational Software Analyzer rule menu
  - ▶ **Class** should be set to  
`com.ibm.rsaz.analysis.core.category.DefaultAnalysisCategory`. That class has most of the functionality needed for a category



You now have your plug-in extending the Rational Software Analyzer plug-ins, but you need to configure your plug-in so Rational Software Analyzer knows what to do with it. First, you need to tell Rational Software Analyzer how to categorize your rule, so it knows how to display it in the Rules tab. You can do this by adding an Extension to the plug-in that will create an analysisCategory extension.

## Setting up a custom plug-in: Extending (2)

- The hierarchy of Rational Software Analyzer rule displays is:
  - ▶ Providers (one to many)
    - Categories (one to many)
      - Rules



To clarify why you were doing a category extension in the previous slide, here is an explanation of the hierarchy with Rational Software Analyzer rules. Every rule must belong somewhere in the hierarchy. There must be at least one provider, who must have at least one category, who can have many rules. It is up to you on how you want to extend the existing rule providers or create an entirely new provider to implement the plug-in extension.

## Setting up a custom plug-in: Extending (3)

- Define the rule
  - ▶ From the same Extensions tab, create a new Extension for `com.ibm.rsaz.analysis.core.analysisRule`.
  - ▶ In the extension, set this data:
    - **id** – This must be a unique value.
    - **label** – This is how the rule shows in Rational Software Analyzer.
    - **severity** - This sets how important the rule is from a warning to a critical defect.
    - **class** – This defines the Java class that will define the actual behavior of the rule. You will discuss what needs to be implemented in this class in the next set of slides.



Now you know what category you are going to put your rule in Rational Software Analyzer, but you need to define the rule itself. You can do this by creating a new extension to the Rational Software Analyzer `analysisRule`. This will tell Rational Software Analyzer what your rule name is and where it is located. However the behavior of the rule will need to be implemented in the Java class. This is where all of the true implementation will take place.



## Summary

- Custom rules in Rational Software Analyzer are the creation of custom plug-ins.
- Eclipse has a wizard to aid with the creation of the plug-in.
- Configuration of the plug-in to extend the rule is done in the plugin.xml file.
- The actual implementation of the rule is done in the Java class of the rule definition.



This module provided an overview of rules in Rational Software Analyzer. You should now be comfortable with understanding what a rule is and you should know how to write and set up a custom plug-in for a custom rule.

## Additional resources

- **Additional resources on ibm.com**

<http://www-01.ibm.com/software/awdtools/swanalyzer/enterprise/index.html>

<http://www-01.ibm.com/software/awdtools/swanalyzer/enterprise/support/>

[http://www.ibm.com/developerworks/rational/library/08/0429\\_gutz1/index.html?S\\_TACT=105AGX54&S\\_CMP=B0612](http://www.ibm.com/developerworks/rational/library/08/0429_gutz1/index.html?S_TACT=105AGX54&S_CMP=B0612)



Additional resources can be found on the ibm.com Web site and in the Whitepaper on “**Static analysis IBM Rational Software Analyzer: Getting started**”.

## Feedback

### Your feedback is valuable

You can help improve the quality of IBM Education Assistant content to better meet your needs by providing feedback.

- Did you find this module useful?
- Did it help you solve a problem or answer a question?
- Do you have suggestions for improvements?

Click to send e-mail feedback:

<mailto:iea@us.ibm.com?subject=Feedback about rsar setting up custom files.ppt>

This module is also available in PDF format at: [../rsar\\_setting\\_up\\_custom\\_files.pdf](http://rsar_setting_up_custom_files.pdf)



You can help improve the quality of IBM Education Assistant content by providing feedback.

## Trademarks, copyrights, and disclaimers

IBM, the IBM logo, ibm.com, and the following terms are trademarks or registered trademarks of International Business Machines Corporation in the United States, other countries, or both:

Rational

If these and other IBM trademarked terms are marked on their first occurrence in this information with a trademark symbol (® or ™), these symbols indicate U.S. registered or common law trademarks owned by IBM at the time this information was published. Such trademarks may also be registered or common law trademarks in other countries. A current list of other IBM trademarks is available on the Web at "Copyright and trademark information" at <http://www.ibm.com/legal/copytrade.shtml>

Java, and all Java-based trademarks and logos are trademarks of Sun Microsystems, Inc. in the United States, other countries, or both.

Other company, product, or service names may be trademarks or service marks of others.

Product data has been reviewed for accuracy as of the date of initial publication. Product data is subject to change without notice. This document could include technical inaccuracies or typographical errors. IBM may make improvements or changes in the products or programs described herein at any time without notice. Any statements regarding IBM's future direction and intent are subject to change or withdrawal without notice, and represent goals and objectives only. References in this document to IBM products, programs, or services does not imply that IBM intends to make such products, programs or services available in all countries in which IBM operates or does business. Any reference to an IBM Program Product in this document is not intended to state or imply that only that program product may be used. Any functionally equivalent program, that does not infringe IBM's intellectual property rights, may be used instead.

THE INFORMATION PROVIDED IN THIS DOCUMENT IS DISTRIBUTED "AS IS" WITHOUT ANY WARRANTY, EITHER EXPRESS OR IMPLIED. IBM EXPRESSLY DISCLAIMS ANY WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE OR NONINFRINGEMENT. IBM shall have no responsibility to update this information. IBM products are warranted, if at all, according to the terms and conditions of the agreements (for example, IBM Customer Agreement, Statement of Limited Warranty, International Program License Agreement, etc.) under which they are provided. Information concerning non-IBM products was obtained from the suppliers of those products, their published announcements or other publicly available sources. IBM has not tested those products in connection with this publication and cannot confirm the accuracy of performance, compatibility or any other claims related to non-IBM products.

IBM makes no representations or warranties, express or implied, regarding non-IBM products and services.

The provision of the information contained herein is not intended to, and does not, grant any right or license under any IBM patents or copyrights. Inquiries regarding patent or copyright licenses should be made, in writing, to:

IBM Director of Licensing  
IBM Corporation  
North Castle Drive  
Armonk, NY 10504-1785  
U.S.A.

Performance is based on measurements and projections using standard IBM benchmarks in a controlled environment. All customer examples described are presented as illustrations of how those customers have used IBM products and the results they may have achieved. The actual throughput or performance that any user will experience will vary depending upon considerations such as the amount of multiprogramming in the user's job stream, the I/O configuration, the storage configuration, and the workload processed. Therefore, no assurance can be given that an individual user will achieve throughput or performance improvements equivalent to the ratios stated here.

© Copyright International Business Machines Corporation 2009. All rights reserved.

Note to U.S. Government Users - Documentation related to restricted rights-Use, duplication or disclosure is subject to restrictions set forth in GSA ADP Schedule Contract and IBM Corp.

