



IBM Software Group

IBM® Rational® Application Developer V6.0

Application Analysis

Code Review



@business on demand.

© 2004 IBM Corporation
Updated January 24, 2005

This presentation will focus on code review features within IBM Rational Application Developer.

Goals

- Understand Code Review features
- Learn how to configure a Code Review
- Learn how to execute Code Review



This presentation covers the Code Review function in the IBM Rational product suite. In this presentation you will learn the features that are provided in Code Review, and you will also learn how to configure and run a Code Review.

Agenda

- Overview
- Code Review Features
- Code Review Process
- Configuration



This presentation will provide an overview of Code Review, then cover the features of Code Review, the typical process to use when doing a code review. It will then go over the configuration of Code Review.

There are two Show Me demonstrations, one showing you how to run Code Review, and one showing you how to configure Code Review.

Overview

- Code Review is a feature in IBM Rational Application Developer and IBM Rational Software Architect
- Code Review goals
 - ▶ Automation of various manual inspections of code
 - ▶ Unification of quality acceptance tests for all developers working on the same project
- Code Review Benefits
 - ▶ Improved efficiency and productivity
 - ▶ Higher quality code
 - ▶ Less time spent debugging
 - ▶ Improved maintainability and readability of code
- Developers use this feature to review Java™ classes
 - ▶ Application, applet, servlet, bean, EJB, web service



Code Review is included in the IBM Rational Application Developer (RAD) product but it is not included in the Rational Web Developer (RWD) product. The Rational Software Architect (RSA) product includes Code Review, and it contains rules and rule categories for Architectural Control that are not contained in RAD. RSA also contains a feature called Architectural Discovery which is not in the other products. Architectural Discovery is launched from the Diagram Navigator in the Modeling Perspective and it allows you to scan Java code for specific design patterns and display them in topic diagrams for easy viewing and updating.

Code Review is an automated means of enforcing coding rules, and it is an easy way to ensure consistent coding guidelines on a team. This can have many benefits including improved productivity and higher quality code.

You can run Code Review against any of your Java classes, including applications, applets, servlets, beans, EJBs and web services.

Code Review Features

- Rules based
 - ▶ Predefined rules
 - ▶ User defined rules
- Integration with Java editor
- Quick Fix
- Advanced filtering
 - ▶ Status and severity
 - ▶ Specific files, categories, rules



Code Review is rules based, and the rules are defined in categories. You can run specific rules, or specific categories of rules against your code. You can also create your own rules that are based on predefined templates.

Code Review is integrated with the Java editor, so that you can easily identify the location of findings in the source files. The source statements that relate to a code review finding are underlined for easy identification, and also the lines have markers that indicate the severity of the finding. Also, a quick fix is available for findings, so you can automatically apply a fix to the source code.

The Code Review findings view can be filtered so that you display findings based on status (unresolved, fixed, ignored) or severity (problem, warning, recommendation). You can also specify that only certain rules or categories of rules be applied to the code review, and you can run code review against specific files or projects or the entire workspace.

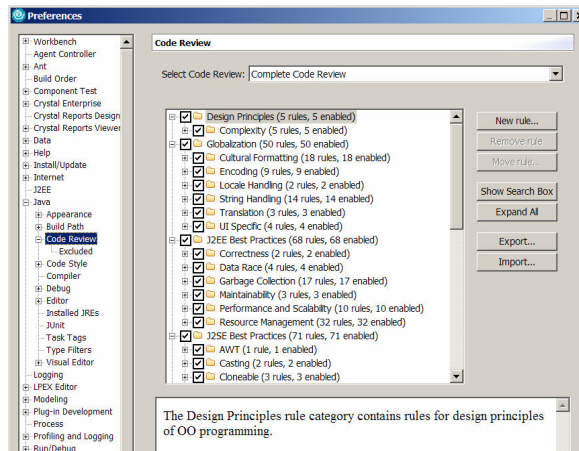
Code Review Example

The screenshot illustrates the IBM Rational Software Development Platform interface during a code review. The main editor shows the source code for `FindAccounts.java`. A yellow callout points to the line `float balance = float.parseFloat(args[1]);`, indicating the location of the finding in the code. Another yellow callout points to the `Code Review` button in the toolbar, indicating where to run the code review. A third yellow callout points to the `Code Review Details` view, which displays a list of findings. A fourth yellow callout points to the `Code Review Details` view, which shows a description and solution for the selected finding.

This example shows the results after a code review has been run. The green arrow on the Code Review view is used to run the code review. After it runs, the Code Review view shows the list of findings for the run. You can double click on a finding, and the Java file containing the finding will be opened in the editor view. You will see the findings underlined and markers for each of the findings, including markers for Quick Fixes. The Code Review Details view shows more detail about the selected finding, including example problems and example solutions.

Code Analysis

- Static analysis engine is rules based
- Rules and categories are setup in Preferences
- Rule Categories
 - ▶ Design Principles
 - ▶ Globalization
 - ▶ J2EE Best Practices
 - ▶ J2SE Best Practices
 - ▶ Naming Conventions
 - ▶ Performance
 - ▶ Private API Usage



The Code Review engine is rules based, and these rules and rule categories are predefined in the Preferences for Code Review. There are various predefined categories of rules, and you can select which of these you want enabled or disabled for your usage.

Examples:

Design Principles – avoid nesting if's, loops

Globalization – use resource bundles for strings rather than hardcoded strings

J2EE Best Practices – avoid calling certain methods and packages from a servlet (e.g. sound, Swing,...)

J2SE Best Practices – avoid negation in if/else conditions for readability

Naming Conventions – avoid incorrect main method signature

Performance – avoid method calls in the conditions of loops

Private API Usage – avoid using certain packages for Eclipse, Sun and WebSphere®

Rule Categories

- J2SE and J2EE Best Practice
 - ▶ Improves scalability, performance and maintainability of code
 - ▶ Enriched with vast IBM experience of building software applications within IBM and externally for IBM customers – through IBM Global Services organization
- Globalization
 - ▶ Automatic detection and Quick Fixes for a number of typical globalization hurdles
 - ▶ Developed by G11N group within IBM
 - ▶ Supplies Quick Fix solutions for projects that use *com.ibm.icu* library and also separate solutions for those projects that don't use it



The rule categories for J2SE and J2EE Best Practice help you design better code. The rules are not based on a standard, but they were created by IBM developers based on experience in the development of applications within IBM and for customers. There are rule categories in J2EE such as Correctness, Garbage Collection, Maintainability, Performance, Scalability, and others. There are rule categories in J2SE such as AWT, Casting, Comparison, Constructors, Declaration, Exceptions, Portability, Serialization, Threads, and others.

There are rule categories in Globalization such as Locale Handling, String Handling, Translation, and others.

Quick Fixes for Globalization rules are provided regardless of your usage of the ICU library. The International Components for Unicode (ICU) library is available at <http://oss.software.ibm.com/icu/>. ICU is an open source development project for Unicode support, and software globalization. It is available for Java and C/C++ environments. So, any findings for problems with Globalization will be listed with several options for Quick Fixes. Some of them are based on using the ICU library and others do not, so you can pick the ones that are most appropriate for your usage.

Code Review Process

- Configure the Code Review engine
- Add the Code Review view to your perspective
- Run Code Review against your file, project or workspace
- Review the findings in the Code Review view
- Update your code using Quick Fixes if available
- Repeat as required

To run Code Review, you should follow these steps.

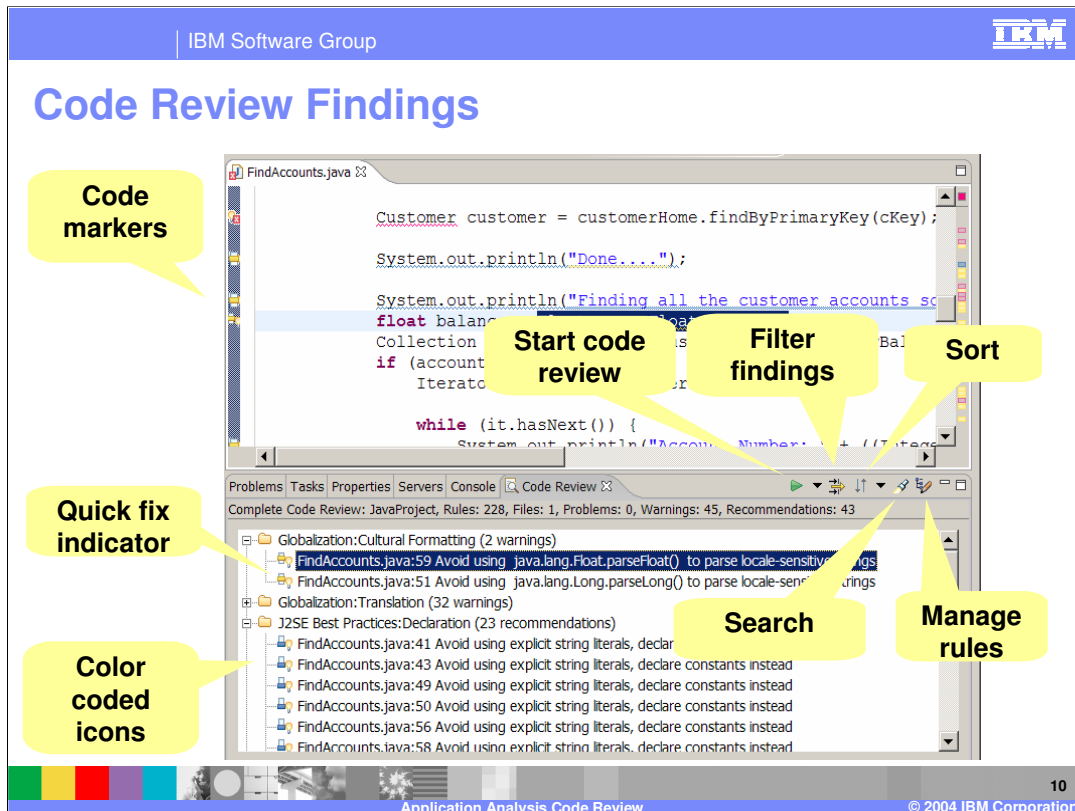
Use the Code Review preferences to tailor the run to your needs. You can enable rules and rule categories for the run. If you take the defaults, the you will run a Quick Code Review, which will run only rules associated with J2EE Best Practices, J2SE Best Practices, and Performance, so that other categories of rules such as Globalization will not be run. To run all the rules, you will have to select a code review type 'Complete Code Review' in the preferences.

Most of your work will be done using the Code Review view, so add it to your perspective.

Determine the scope of the Code Review by picking a file, or a project or the entire workspace. You can initiate a code review in the Code Review view or by using the context menu in the Project Explorer.

The findings are displayed in the Code Review view, so you can double click on any of them to bring up the editor on the affected Java file.

You can use Quick Fixes to fix the findings. After you apply the Quick Fix, the finding is marked as fixed in the Code Review view.



This example shows the findings that are displayed after a Code Review is performed. The findings have color coded icons to show the severity of the finding. Red means the finding is a problem. Yellow means the finding is a warning. Blue means the finding is a recommendation. A light bulb indicates that a Quick Fix is available. After the fix has been applied, the icon changes to a blue check mark.

This example also shows the various icons that control the Code Review view. You can start a code review for a file, project or workspace. You can filter the findings by status or severity, sort the findings by category or file name, search the findings, and manage rules.

The Editor view shows the Java code associated with the selected finding. The code associated with the findings are underlined and markers are also displayed for them.

Code Review View

- Review
 - ▶ Review a project or the entire workspace
 - ▶ If a Java file is open, you can also just review the opened file
 - ▶ From the Project Explorer context menu, you may also review individual projects, folders, packages or files
- Filters
 - ▶ By status – unresolved, fixed, ignored
 - ▶ By severity – problem, warning, recommendation
- Sort
 - ▶ By rule category or file
- Show Search Box
 - ▶ Allows filtering based on text searches in the findings
 - ▶ After searching, to reset the findings list, clear the search field and hit enter
- Manage Rules
 - ▶ Opens the Preferences for Code Review

This slide describes the various functions available to you on the Code Review view.


The Review option is used to execute a code review. You can select a specific project or the entire workspace. If you have a Java file opened in the editor, then that file is also listed, and you can review only it and nothing else. You can also select the target of your code review by using the Project Explorer and the context menu option called 'Code Review'. Using this method, you will be able to select projects, folders, packages or individual files.

The Filters option is used to filter the findings in the list. You can filter by status (unresolved, fixed, ignored) or by severity (problem, warning, recommendation).

The Sort option is used to sort the findings in the list. You can sort the findings by rule category name or by Java file name.

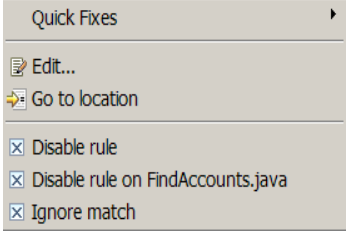
The Search option is used to search for specific strings contained in the findings. Only those findings contained the string will be displayed. To reset the list, you must clear the string in the search box and hit the enter key.

The Manage Rules option is used to bring up the Preferences for Code Review.

IBM Software Group 

Code Review View: Context Menus

- Quick Fixes
- Edit
- Go to location
- Disable rule
- Disable rule on <file name>
- Ignore Match



Application Analysis Code Review 12
© 2004 IBM Corporation

This slide describes the context menu available to you in the Code Review view.

The Quick Fixes option allows you to select from a list of available fixes for the selected finding, and apply them. If Quick Fixes are not available, then the option will not be listed on the menu.

The Edit option allows you to edit the properties of the rule for the selected finding. For canned rules, you can update the severity of the rule. For user defined rules, you can also edit the specific properties provided by the template for the rule.

The 'Go to location' option allows you to open the Java source file and then the editor is positioned at the location of the selected finding in the code.

The 'Disable rule' option allows you to disable the rule for the selected finding. Any disable rules will not be used in subsequent code reviews. This information is stored in the Code Review Preferences main page in the check box for the rule.

The 'Disable rule on <file name>' option allows you to disable a rule for the selected finding just for this specific Java file. This information is stored in the Code Review Preferences main page in the Resource Filters tab for the specified rule.

The 'Ignore Match' option is used to ignore this rule for this Java file at this specific location. If you edit the file and add or remove lines of code, the line number is updated automatically. This information is stored in the Code Review Preferences, in the Matches tab on the Excluded page.

IBM Software Group

Code Review View: Context Menus (Continued)

- Disable category Disable category
- Exclude File from Code Review Exclude file from Code Review

Application Analysis Code Review 13 © 2004 IBM Corporation

This slide describes the context menus available to you in the Code Review view.

‘Disable category’ is available when you right click on a category in the list of findings. If you disable a category, then the next time you run a code review, the entire category is not used. This information is stored in Code Review Preferences, in the check box for the specific category.

‘Exclude File from code Review’ is available when you right click on a file in the view, which is only applicable if you are sorted based on file names. Note that this information is stored in the Resources tab in the Excluded page of Code Review Preferences, and you can edit it there, but you cannot use wildcards in the filenames. You must specify individual file names.

Automatic Problem Resolution And Prevention

- Automatic detection of potential problems in Java code
 - Detection of problems prior to component testing, or regression testing
- Example
 - Avoid returning null instead of empty array

Quick Fix

Problem	Solution	Quick Fix
<pre> public NullEmptyArray_Example() { super(); } public void addValue(Integer value) { if (values == null) { values = new ArrayList(); } values.add(value); } public Integer[] getValues() { if (values == null) { return null; } return (Integer[])values.toArray(new Integer[0]); } private List values; public static void main(String[] args) { NullEmptyArray_Example example = new NullEmptyArray_Example(); Integer[] values = example.getValues(); for (int i = 0; i < values.length; i++) { System.out.println(values[i]); } } </pre>	<pre> public NullEmptyArray_Solution() { super(); } public void addValue(Integer value) { if (values == null) { values = new ArrayList(); } values.add(value); } public Integer[] getValues() { return (values == null) ? new Integer[0] : (Integer[])values.toArray(new Integer[0]); } private List values; public static void main(String[] args) { NullEmptyArray_Solution example = new NullEmptyArray_Solution(); Integer[] values = example.getValues(); for (int i = 0; i < values.length; i++) { System.out.println(values[i]); } } </pre>	

Use the Quick Fixes to automatically modify your code. The Code Review Detail View shows you Java code for typical problems with one or more solutions to the problem. The Quick Fix dialog, shows your original Java source and the proposed changes, so you can review the changes before you make them permanent.

Code Review Details View

Description of the finding with examples and solutions

Also found in Preferences


This slide shows you that the problem description, examples and solutions can be found in the Code Review Details view for the selected finding in the findings list. The examples and solutions can also be found in the Code Review Preferences for the selected rule in the rules list box.

Run Code Review: Demonstration

- Open the Code Review view
- Run Code Review against a project
- Look at details for a finding
- Perform a Quick Fix
- Filter, sort, search findings
- View Preferences




This is a tutorial that shows you how to run Code Review, apply a Quick Fix, and manage your findings in the Code Review View.

IBM Software Group 

Code Review Configuration: Preferences

- Code review views
- Enable, disable rules and categories
- Add, remove, edit rules
- Move rules to other categories
- Search rules
- Export, import configurations
- Exclude files and matches



17
Application Analysis Code Review © 2004 IBM Corporation

The Code Review preferences are used to pick the views that you wish to run. Views are predefined and they allow you to pick a subset of the rules categories to be used in the next run.

You can enable or disable specific rules or entire categories of rules. Each rule belongs to exactly one category.

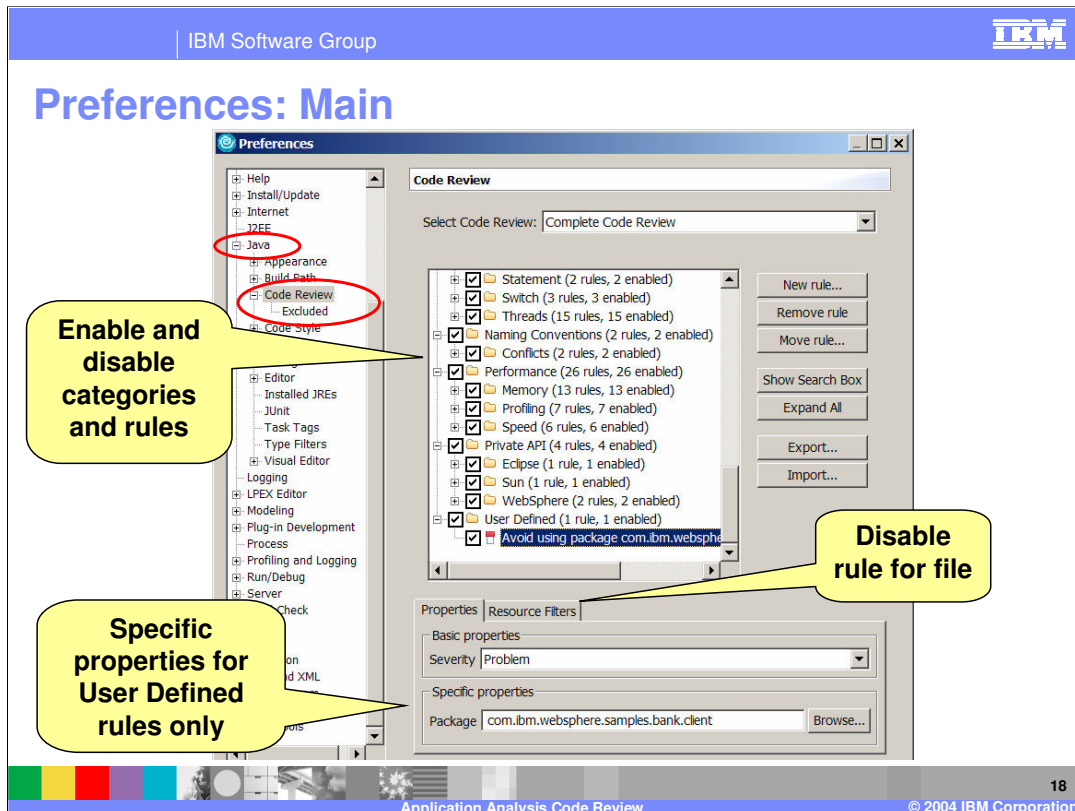
You can add user defined rules, or edit properties of existing rules or delete rules.

You can move rules from one category to another.

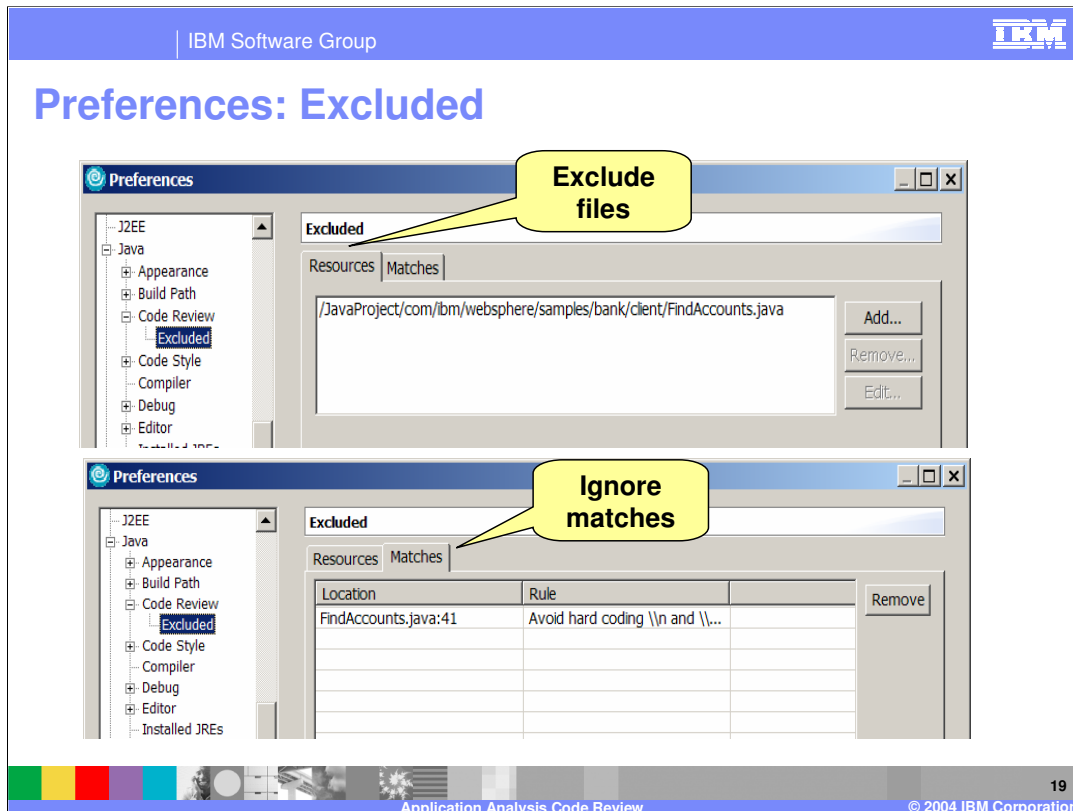
You can bring up a search box to find rules based on a string.

For team development, you can export or import a rule configuration.

You can also specify files and specific matches within files that you want to exclude.



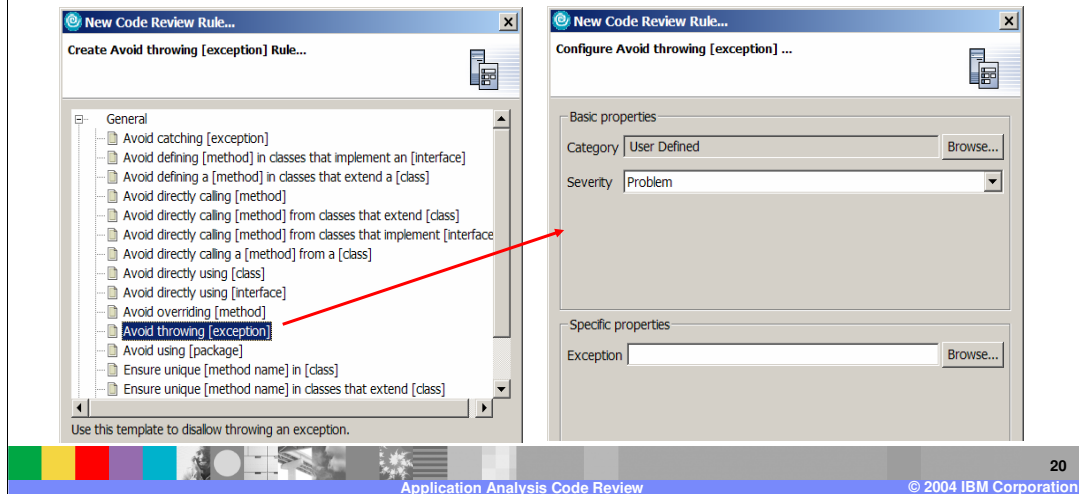
This is the preferences window for Code Review. Notice that it is located under the Java Preferences. You can enable and disable rules and rule categories here. You can create rules, remove rules and move rules between categories. You can import and export the rule configuration and you can also update properties or resource filters here.



This is the Excluded section of the Code Review preferences. On the Resources tab, you identify Java files that you want to exclude from Code Review processing. These are typically added via the context menu of the Code Review view, but you may add, remove or edit them here. On the Matches tab, you find specific file/rule combinations that will not included in the findings list for after Code Review processing. These are added via the context menu of the Code Review view, but you may remove them here.

Creating Rules

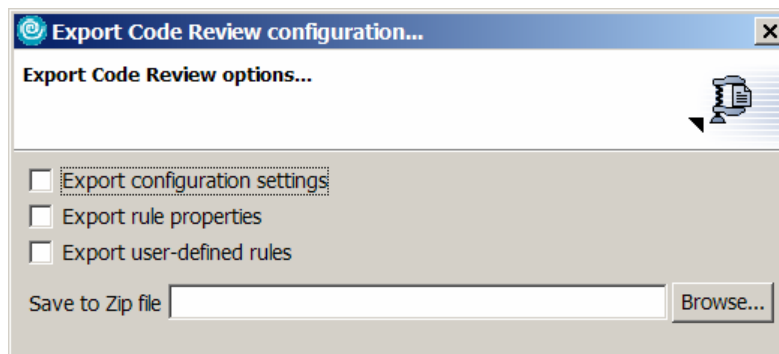
- Based on pre-existing templates
- Specific properties vary based on the template selected for the rule



To add a user defined rule, you select from preexisting templates. Then you can pick the severity of the rule and also you fill in properties specific to the template that you selected. In this example, template called 'Avoid throwing [exception]' has been selected, so you can see that you can choose the severity of this new rule and also you can pick the specific exception that will be assigned to this rule.

Team Development

- Code Review enables implementation and enforcement of coding guidelines for the whole team and not just one individual developer



Team development is enabled using the Import and Export features of the Code Review preferences. When you export, you pick the type of export. You can choose from configuration, rule properties or user defined rules, and you can choose any or all of them. The output of the export is a zipped file which can then be imported into the workspace of another team member.

If you pick 'Export configuration settings' then the zipped file includes a list of the standard rules and categories that are provided in the current configuration and also which ones are enabled or disabled. It also includes the information on the Resources tab and Matches tab on the Excluded page of the Code Review Preferences. If you pick this option, the user defined rules are not included in the zipped file.

If you pick 'Export rules properties' then the zipped file includes the severity of the standard rules which are supplied with Code Review, not including the User Defined rules.

If you pick 'Export user-defined rules' then the zipped file includes the User Defined rules and properties that are associated with them.

Usage model

- 1) The Lead Developer or the Architect configures Code Review
- 2) The customized configuration is shared through Import / Export capability
- 3) Every developer enforces the same coding guidelines

Reference

- Help > Help Contents > Detecting and analyzing runtime problems > Reviewing code automatically

Check the Help Contents in the tool to find more information about Code Review.

Trademarks, Copyrights, and Disclaimers

The following terms are trademarks or registered trademarks of International Business Machines Corporation in the United States, other countries, or both:

IBM	CICS	IMS	MQSeries	Tivoli
IBM (logo)	Cloudscape	Informix	OS/390	WebSphere
e(logo)/business	DB2	iSeries	OS/400	xSeries
AIX	DB2 Universal Database	Lotus	pSeries	zSeries

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

Microsoft, Windows, Windows NT, and the Windows logo are registered trademarks of Microsoft Corporation in the United States, other countries, or both.

Intel, ActionMedia, LANDesk, MMX, Pentium and ProShare are trademarks of Intel Corporation in the United States, other countries, or both.

UNIX is a registered trademark of The Open Group in the United States and other countries.

Linux is a registered trademark of Linus Torvalds.

Other company, product and 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 and/or changes in the product(s) and/or program(s) 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.

Information is provided "AS IS" without warranty of any kind. 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 (e.g., 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 2004. 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.