Working with the Code Review UI

This topic introduces how the Code Review UI presents code review tasks, provides an overview of how you can take action on your assigned tasks, and explains how to customize the UI to meet your specific preferences.

Section include:

Introducing the Code Review UI

Parasoft Test provides the following provides the following views to facilitate code reviews:

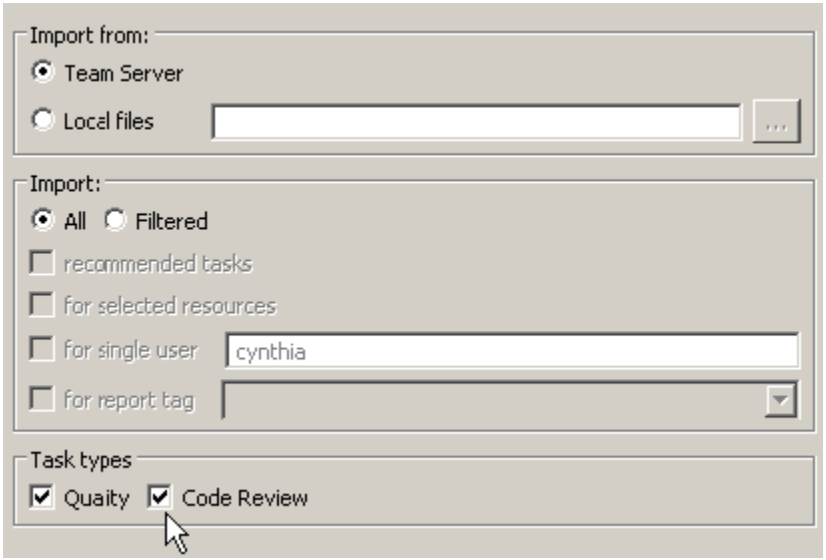
- **Quality Tasks view –** In Code Review mode, this view displays the code review task tree. It presents two types of tasks: reviewers see "to review" tasks for code revisions they need to review, and authors see "to fix" tasks for responding to reviewer comments. This view should be open by default; to put it in Code Review mode, click the **Code Review** or **User Code Review** button in the view's toolbar.

- **Code Review Issue–** Allows team members to add and review code comments about the submitted code modifications. This allows the author and reviewer to have a conversation about how to revise a submitted revision. To access this view, choose **Parasoft> Show View> Code Review Issue**.
- **Compare Editor –** Highlights the differences between the most recent version of the file and the previous version that was stored under source control. In post-commit mode, it shows the difference between two or more revisions reported to review. It also shows the difference between local changes on desktop and the previous version under source control. The Com-pare Editor will open when you double-click a specific revision in the code review task tree.

Importing Code Review Tasks into the UI

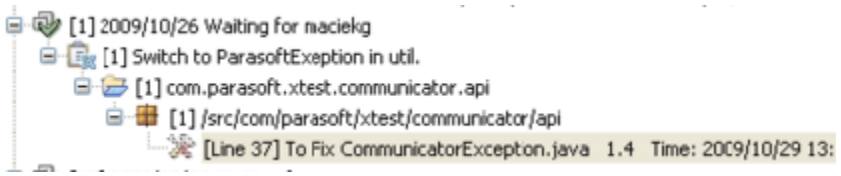
Depending on your import settings, your code review tasks may be imported automatically when you import your quality tasks (as described in Importing Results into the UI).

You can always import your assigned code review tasks into the **Quality Tasks view** by choosing **Parasoft> Import> [desired_import_option]** or clicking the Import My Recommended Tasks toolbar button. If you choose to import a custom set of tasks, be sure to enable the **Code Review** option.



Understanding Task Status Indicators

Depending on how you configure the code review tasks tree, each code review task may be marked with a status indicator, as well as the name of the file, the revision version number, and the time when the latest revision was committed into the source control.



The following table describes the various status indicators used:

Action	Description
To Review	Indicates that a review should be performed on the revision package.
To Fix	Indicates that some improvements should be made within the files included in the revision package.
Monitor	Indicates that the designated monitor should review the status of files included in the revision packages.
Waiting	Indicates that your revision package is waiting for someone's action.
Done	If you want such tasks to be shown, you need to set the **Show completed tasks by** option in the Preferences panel (see Configuring Code Review Preferences).

Taking Action on a Task or Set of Tasks

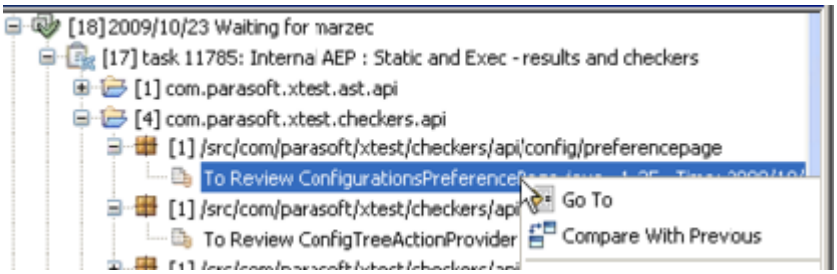
The main ways to take action on a task are to double-click a code review task tree node, or right-click it and choose the appropriate shortcut menu command. You can take action on a set of tasks (such as all code review tasks for a code review package), a single code review task (such as a single modification to review or reviewer comment to address), or anything in between.

Available Actions

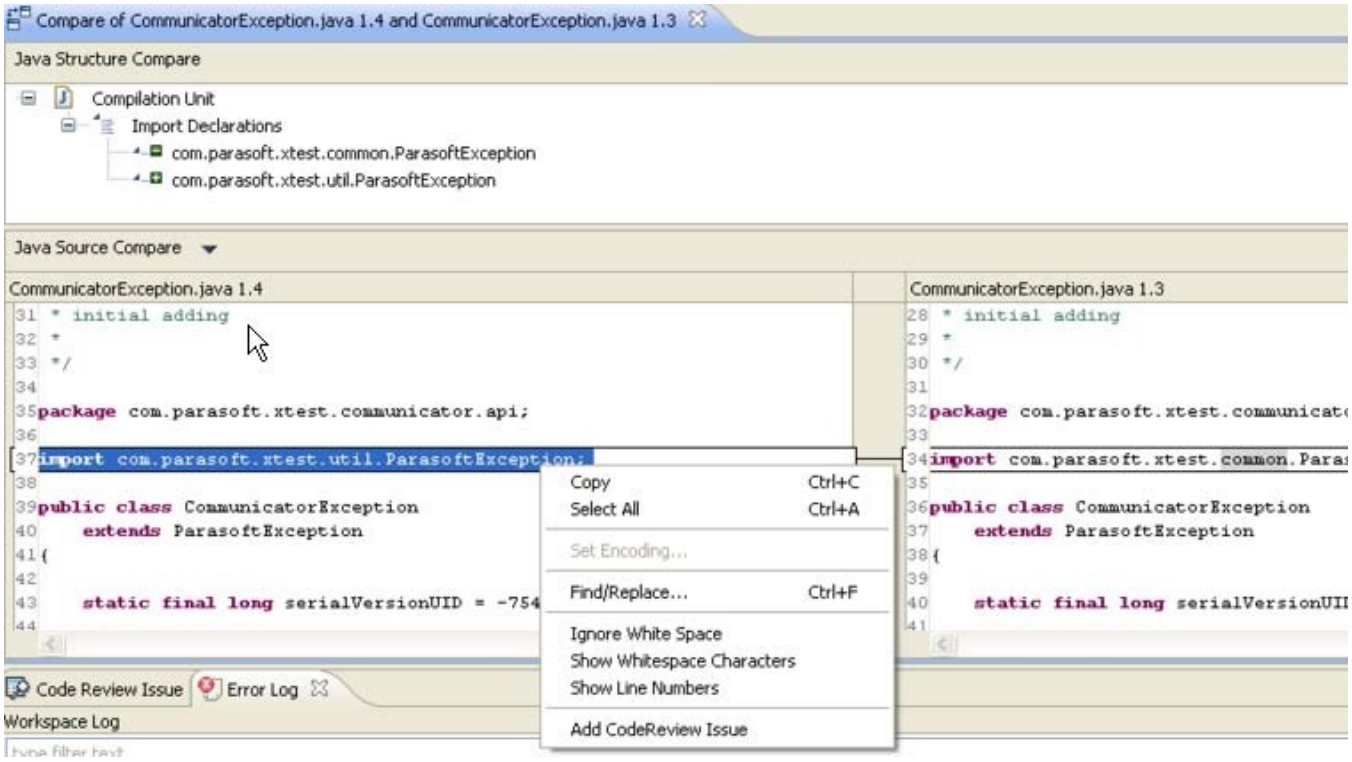
Different shortcut menu commands are available depending on what code review task tree item you right-click and your specific role. Role-relevant commands are discussed in the topics for authors and reviewer. Generally, shortcut menus can be used to perform actions such as:

- Reassign a task to another team member.
- Remove a task.
- Navigate from one task to the next (or previous) tasks.
- Expand or collapse the tree.
- Change the tree layout.
- Mark a task as done, cancel tasks, or reject tasks.
- Open a compare editor that highlights the differences between the most recent version of the file and the previous version that was stored under source control.
- Add a new reviewer.
- Open comments that a reviewer added.

For example, a reviewer with the following task might choose **Compare with Previous**

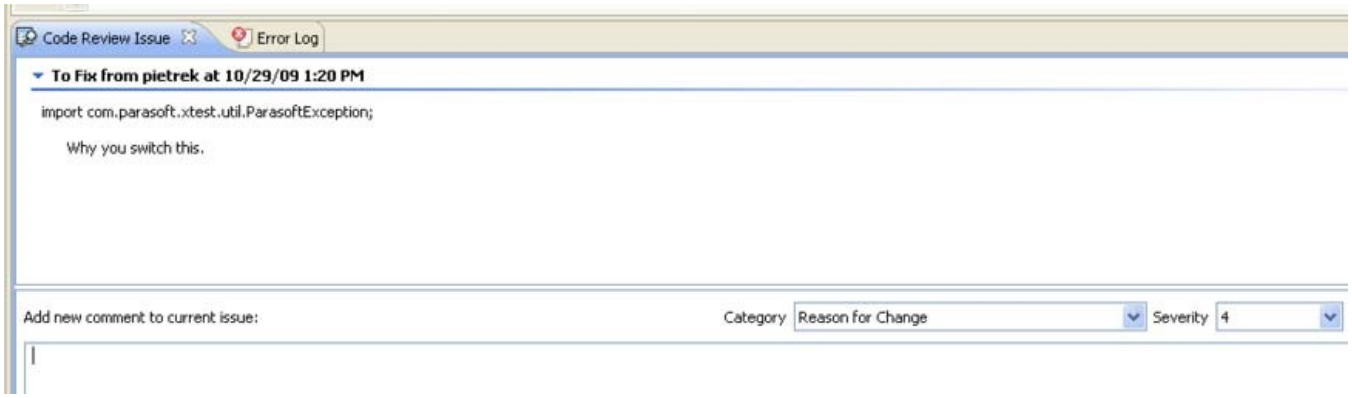


review the code change in the compare editor



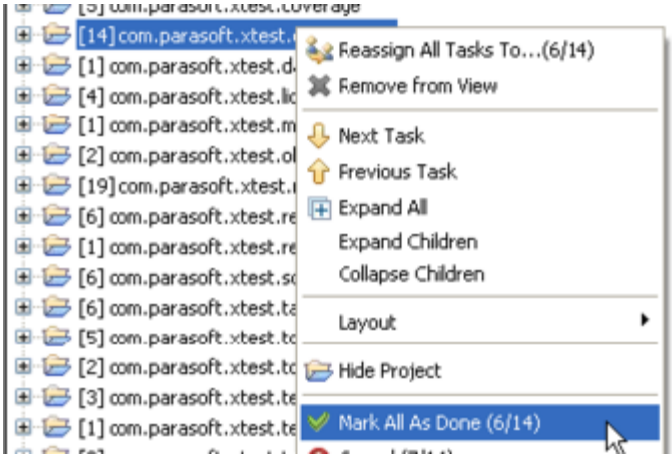
then add an issue in the Code Review Issue area.

The author might then open up that comment, then respond to it in the Code Review Issue area.



Applying an Action to Multiple Tasks

If you right-click a code review task tree node that represents a group of items (for instance, all reviews for a specific file), you can use a single command to perform the same action on all appropriate items in that group. For example, if you want to mark all active code review tasks in a package as "done," you could do as follows:



Customizing the Code Review Tasks Tree

There are numerous ways to configure the code review tasks tree to suit your needs and preferences.

Layout Template Modifications

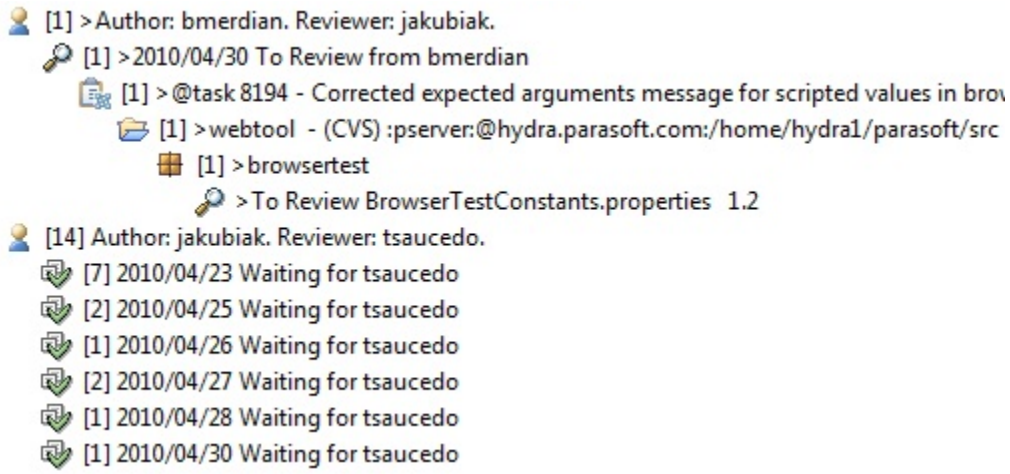
To customize which elements are shown or hidden, you can select, configure, and customize layout templates, which are described in Changing the Display Format and Contents.

Sample Code Review Layouts

You can organize the code review layout by file, by date, by committer, or by comment.

We encourage you to experiment with different layouts for each view until you find the one that works best for you. Here are some sample layouts that developers use for code review.

For instance, the following screenshot shows one possible configuration for Code Review:



This was configured by a developer trying to group items in a way that lets him to look at things by developer, then by task, and then by location in the code. That's how his brain breaks down the information, so that's how we wanted it presented. More specifically:

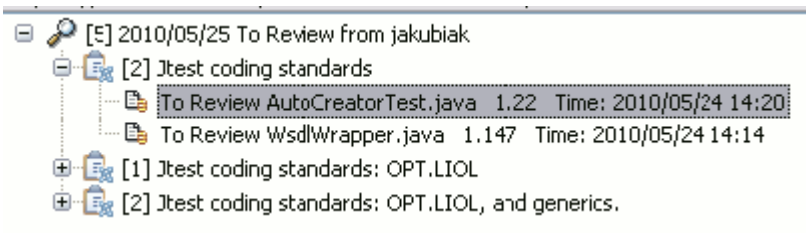
- The top-level node shows him reviews broken down by developer-reviewer. This helps him distinguish between reviews of code he wrote and reviews that he needs to perform.
- The 2nd-level node tells him the date of the review.

- The 3rd node shows him the commit or pre-commit comment that indicates what the code author was working on.
- The 4th and 5th level nodes are successively more granular breakdowns of where in the code that the file lives.
- The 6th node indicates the actual file, along with the current state of that file in the review.

A developer looking for a simpler peer code review layout might decide to have only 4 levels of information:

- The Perform Code Review Tasks node.
- The date of the review.
- The commit or pre-commit comment that indicates what the code author was working on.
- The actual file, along with the current state of that file in the review.

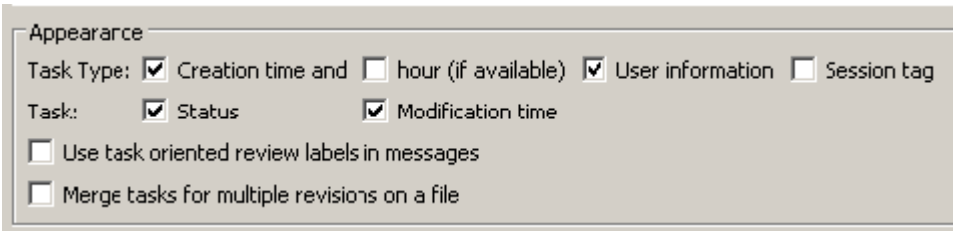
Another developer who wants to focus on 1) what code reviews to address first and 2) what each code modification was designed to achieve might have the following layout:



This layout shows the "Task Type" and the "Task or Comment." The "Task Type" sorts the check-ins by date, allowing the reviewer to see which code review to look at first. The "Task or Comment" shows him the intent of the check-in by including the author's comment.

Label Decorations

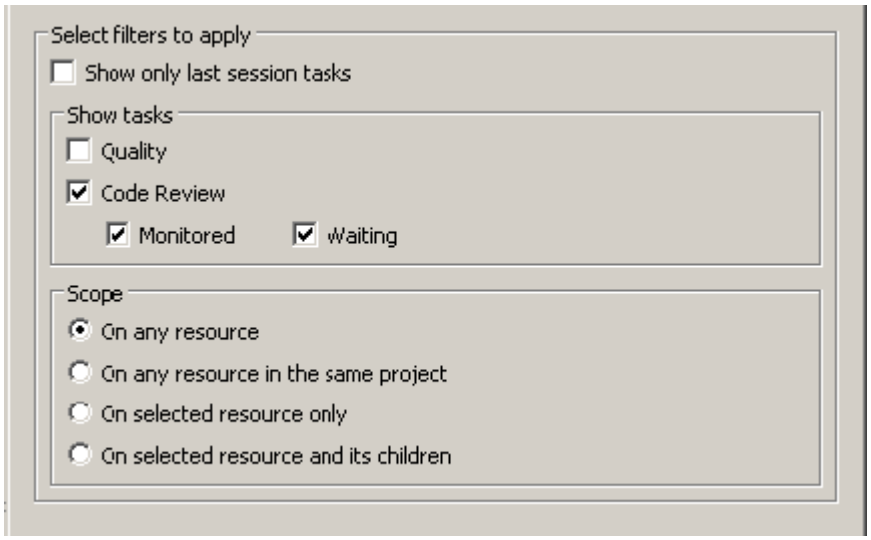
To fine-tune what data is displayed in the various tree nodes that you choose to display, you can use the Preferences panel's Code Review controls to configure which labels are displayed.



Filters

Additionally, you can filter the content shown by clicking the **Filter** button in the **Quality Tasks** view

then specifying the desired filter conditions.


Select filters to apply
Show only last session tasks
Show tasks
Quality
Code Review
Monitored
Waiting
Scope
On any resource
On any resource in the same project
On selected resource only
On selected resource and its children