Removed Code Behavior Options

Learn about `removed_code_behavior` options.

Removed code behavior (RCB) helps keep your Continuous Integration (CI) pipeline from failing by adjusting how code removal impacts coverage.

Preface

Before configuring RCB, you should know the following:

- RCB only applies to project coverage and does not impact patch coverage.
- For RCB to be effective, you must set `Target` to `auto`.
- You can configure RCB with one of the following behaviors:
  - `adjust_base` (default behavior)
  - `fully_covered_patch`
  - `removals_only`
  - `off`

Why Use RCB?

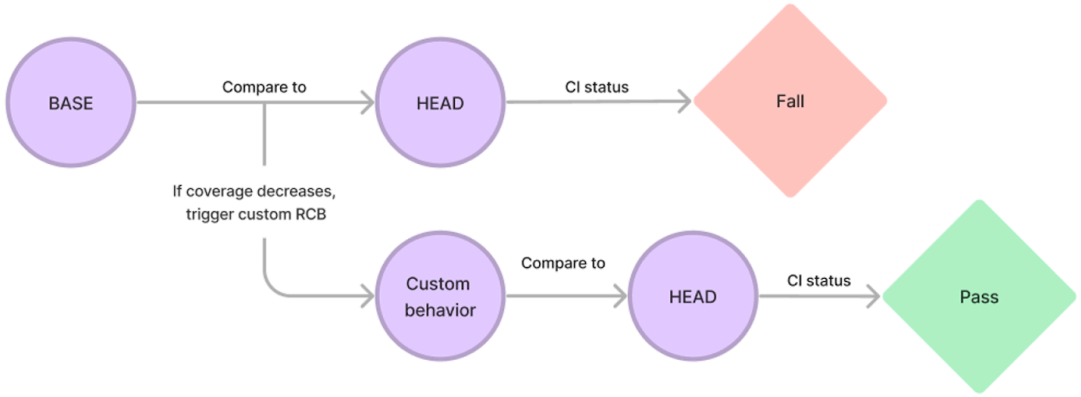
Sometimes, removing lines of code from a codebase can be a good thing, but doing so can reduce code coverage.

Code coverage measures how much of your code is run by automated tests in a CI pipeline. It is calculated by dividing the number of lines of code covered by tests by the total number of lines in the project, expressed as a percentage:

$$\text{Project Coverage } \% = \left(\frac{\text{Total covered lines}}{\text{Total lines}} \right) \times 100$$

If your CI pipeline is configured to fail if code coverage decreases or drops below a certain level, removing covered lines may cause a CI failure. RCB changes the way code removal impacts coverage, allowing developers to streamline their codebase without worrying about breaking the CI just because code coverage drops.

This diagram shows how RCB works:



On the top line, without RCB, a code-reducing patch can cause a CI pipeline to fail if it reduces coverage.

On the bottom line, with RCB, the coverage decrease from the same patch triggers RCB to adjust how the removed lines impact the coverage percentage, leading to a passing pipeline.

Options

You can configure RCB with one of the following behaviors:

`adjust_base`

This behavior recalculates the base to filter out any code removals. The recalculation removes misses, hits, and partials from lines removed in the diff of the base. This adjusted base coverage is then compared to the coverage of the patch with code removals ignored. If the adjusted base coverage is less than or equal to the patch coverage with code removals ignored, the project CI passes. This is the default behavior.

`fully_covered_patch`

The project CI passes if all lines added in the patch are fully covered, unless there are unexpected indirect changes. Indirect changes may result from modifications to both new and existing covered and uncovered tests, processing errors, CI errors, or other factors.

The table below describes the possible scenarios.

Patch Coverage	Indirect Changes	CI Status	Scenario
100%	False	Pass	Patch coverage 100%, no indirect changes

100%	True	Fail	Patch coverage 100%, indirect changes occurred
<100%	False	Fail	Patch coverage less than 100%, no indirect changes
<100%	True	Fail	Patch coverage less than 100%, indirect changes occurred

`removals_only`

The project CI passes if the patch only removes lines of code, with no additions or indirect changes.

If any lines are added, or any indirect changes exist, the project CI fails.

The table below describes the possible scenarios.

Lines Removed	Lines Added	Indirect Changes	CI Status	Scenario
True	False	False	Pass	Lines removed, no lines added, no indirect changes
True	False	True	Fail	Lines removed, no lines added, indirect changes occurred
True	True	False	Fail	Lines removed, lines added, no indirect changes
True	True	True	Fail	Lines removed, lines added, indirect changes occurred
False	False	False	Fail	No lines removed, no lines added, no indirect changes
False	False	True	Fail	No lines removed, no lines added, indirect changes occurred
False	True	False	Fail	No lines removed, lines added, no indirect changes
False	True	True	Fail	No lines removed, lines added, indirect changes

				occurred

`off`

With `off` or `False`, no custom behavior is applied to project coverage calculation.

Example Handling of a Code-Reducing Patch

Initial status:

- We have a project with 10 lines of code, with nine lines covered by automated tests. The initial project coverage is 90%, calculated as follows:

$$\text{Initial Project Coverage \%} = \left(\frac{9}{10}\right) \times 100 = 90\%$$

- Our CI pipeline is configured to fail if coverage drops below 90%.

- A developer has submitted a patch in a pull request that removes three covered lines and adds one fully-covered line, with no indirect changes. This would reduce the project to eight lines of code, with seven covered by tests. If the pull request was accepted, the updated project coverage would be 87.5%, calculated as follows:

$$\text{Updated Project Coverage \%} = \left(\frac{7}{8}\right) \times 100 = 87.5\%$$

Behavior Comparison

`off`

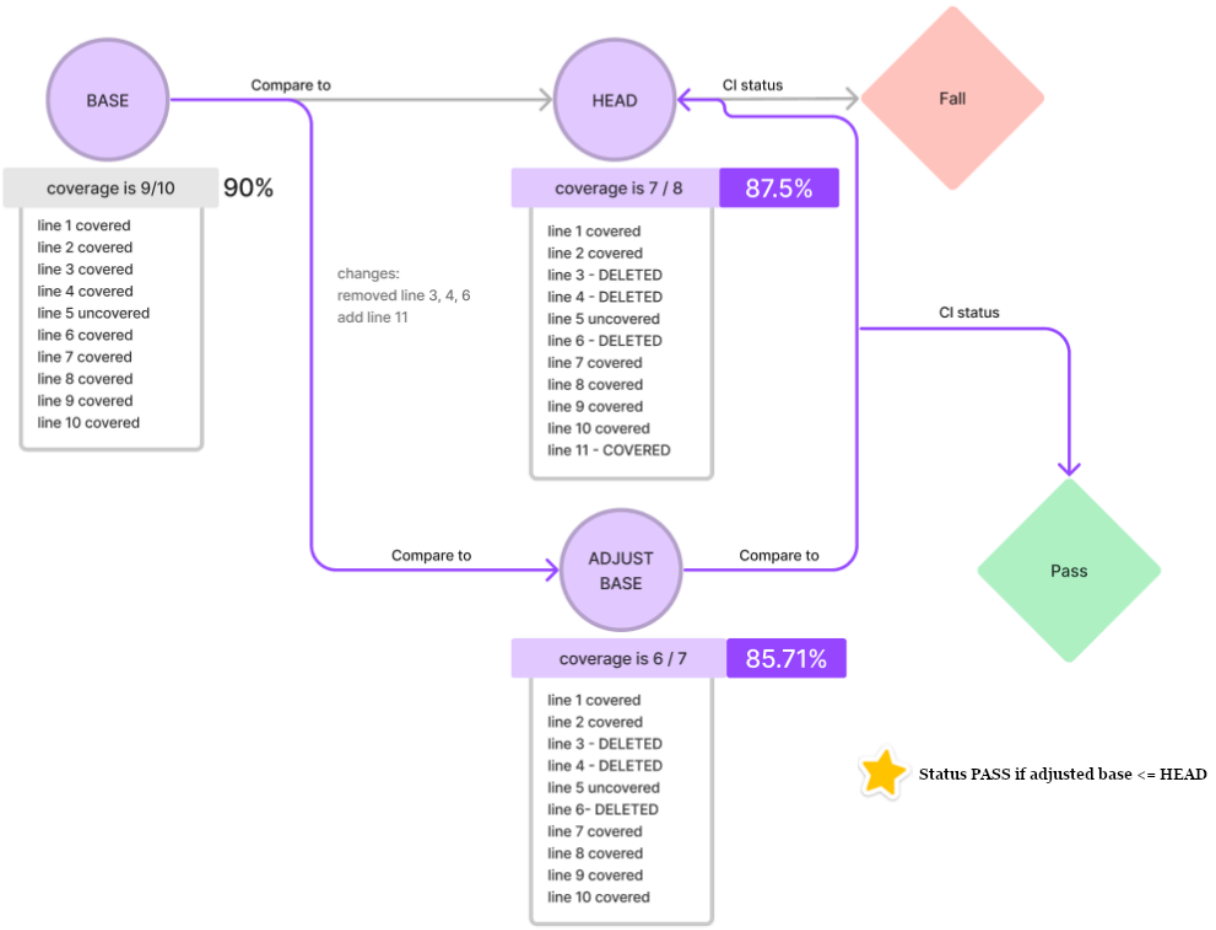
With `off` or `False`, or if no RCB is applied, no custom behavior is applied to the project coverage calculation. If the pull request is merged, the project will have eight total lines of code, with seven covered by tests. The updated project coverage then becomes 87.5%, calculated as follows:

$$\text{Updated Project Coverage \%} = \left(\frac{7}{8}\right) \times 100 = 87.5\%$$

Because our CI pipeline is set to fail if code coverage drops below 90%, the project status check will fail if the pull request is merged.

`adjust_base`

With `adjust_base`, removed lines are ignored in coverage calculations, and added lines are included only when evaluating the patch coverage. The diagram below shows the calculations made with the `adjust_base` behavior (if status `threshold` is default/not set or `automatic`) if the pull request was merged.



The adjusted base coverage is calculated by ignoring the three lines of code removed by the patch and is therefore 85.7%, calculated as follows:

$$\text{Adjusted Base Coverage \%} = \left(\frac{9 - 3 = 6}{10 - 3 = 7}\right) \times 100 = 85.7\%$$

The updated project coverage ignores the three lines of code removed by the patch, but includes the one line added, and is 87.5%:

$$\text{Updated Project Coverage \%} = \left(\frac{7}{8}\right) \times 100 = 87.5\%$$

Since the adjusted base coverage is less than or equal to the updated project coverage the project CI will pass if the pull request is merged.

`fully_covered_patch`

With `fully_covered_patch`, the project CI passes if all lines added in the patch are fully covered and there are no indirect changes.

The patch contains one new line of fully-covered code and no indirect changes, so if the pull request is merged, the project status check will pass.

`removals_only`

With `removals_only`, the project CI fails if any lines are added in a patch. The pull request adds one line of code, so the project status check will fail if the pull request is merged.