

version v2.9

Table of Contents

- [Change Screen](#)
- [Commit Message Block](#)
- [Commit Info Block](#)
- [Change Info Block](#)
- [File List](#)
- [Patch Sets](#)
- [Download](#)
- [Included In](#)
- [Star Change](#)
- [Related Changes](#)
- [Reply](#)
- [History](#)
- [Update Notification](#)
- [Plugin Extensions](#)
- [Old Change Screen](#)
- [Side-by-Side Diff Screen](#)
- [Inline Comments](#)
- [File Level Comments](#)
- [Search](#)
- [Key Navigation](#)
- [Diff Preferences](#)
- [Keyboard Shortcuts](#)
- [New Review UI vs. Old Review UI](#)

Reviewing changes is an important task and the Gerrit Web UI provides many functionalities to make the review process comfortable and efficient. This is a guide through the review UI that explains the different functions and UI elements.

Change Screen

The change screen shows the details of a single change and provides various actions on it.

The screenshot shows the Gerrit Change Screen for Change 55060. At the top, there's a header with "Edit Message" and "Reply..." buttons, and tabs for "Patch Sets (2/2)" and "Download". Below the header, there's a summary section with fields for Owner (David Pursehouse), Reviewers (Dify Cuckoo (Automated Verifier) and Dave Borowitz), Project (gergit), Branch (master), Topic, Strategy (Merge if Necessary), and Updated (26 hours ago). There are buttons for "Cherry Pick" and "Abandon". A "Code-Review" section shows a green "Verified" status from Dify Cuckoo (Automated Verifier) and a yellow "David Pursehouse" status. The "Files" section lists "Commit Message" and "lib/BUCK" with their file paths, comments, and sizes. The "History" section shows a log of activity: David Pursehouse uploaded patch set 1, Patch Set 1 was Verified+, Dify Cuckoo (Aut...) merged Patch Set 2, and David Pursehouse uploaded patch set 2, which was also Verified+.

Commit Message Block

The focus of the change screen is on the commit message since this is the most important information about a change. The numeric change ID and the change status are displayed right above the commit message.

Change 55060 - Needs Code-Review

Update gwtorm to version 1.9

Updates the versions of protobuf, asm and guava libraries to synchronize with the versions used in GWT 2.6.0

Change-ID: Id1badda0d0f24f6022404be479d4681560ed0eda

Reply...

Owner: David Pursehouse
Reviewers: Difly Cuckoo (Automated Verifier) [x] [Add...]
Dave Borowitz [x] [Add...]
Project: gerrit
Branch: master
Topic:
Strategy: Merge if Necessary
Updated: 26 hours ago

Related Changes (2) Conflicts With (1)

- Hide 'refs/heads/' prefix in cherry-pick branch suggestion list
- Update gwtorm to version 1.9

Files

File Path

Commit Message

lib/BUCK

History

David Pursehouse Uploaded patch set 1. 03/11/14 02:25
David Pursehouse Patch Set 1: Verified+1 03/11/14 02:34
David Pursehouse Uploaded patch set 2. 03/11/14 03:24
Difly Cuckoo (Aut...) Patch Set 2: Verified-1 Patchset merges but requires downloading new artifacts. The build cannot be completed from this point. A human will need to p... 03/11/14 04:35
David Pursehouse Patch Set 2: Verified+1 03/11/14 07:14

The commit message can be edited directly in the Web UI by clicking on the `Edit Message` button in the change header. This opens a drop-down editor box in which the commit message can be edited. Saving modifications of the commit message automatically creates a new patch set for the change. The commit message may only be edited on the current patch set.

Change 55060 - Needs Code-Review

Update gwtorm to version 1.9

Updates the versions of protobuf, asm and guava libraries to synchronize with the versions used in GWT 2.6.0

Change-ID: Id1badda0d0f24f6022404be479d4681560ed0eda

Reply...

Update gwtorm to version 1.9

Updates the versions of protobuf, asm and guava libraries to synchronize with the versions used in GWT 2.6.0

Change-ID: Id1badda0d0f24f6022404be479d4681560ed0eda

Patch Sets (2/2) Download ▾

Conflicts With (1)

- Cherry-pick branch suggestion list

Files

File Path

Commit Message

lib/BUCK

History

David Pursehouse Uploaded patch set 1. 03/11/14 02:24
David Pursehouse Patch Set 1: Verified+1 03/11/14 03:24
David Pursehouse Uploaded patch set 2. 03/11/14 03:24
Difly Cuckoo (Aut...) Patch Set 2: Verified-1 Patchset merges but requires downloading new artifacts. The build cannot be completed from this point. A human will need to p... 03/11/14 04:35
David Pursehouse Patch Set 2: Verified+1 03/11/14 07:14

Edit Commit Message

Save **Cancel**

The numeric change ID is a link to the change and clicking on it refreshes the change screen. By copying the link location you can get the permalink of the change.

Change 55060 - Needs Code-Review

Update gwtorm to version 1.9

Updates the versions of protobuf, asm and guava libraries to synchronize with the versions used in GWT 2.6.0

Change-ID: Id1badda0d0f24f6022404be479d4681560ed0eda

Reply...

Owner: David Pursehouse
Reviewers: Difly Cuckoo (Automated Verifier) [x] [Add...]
Dave Borowitz [x] [Add...]
Edwin Kempin [x] [Add...]
Project: gerrit
Branch: master
Topic:
Strategy: Merge if Necessary
Updated: 6 minutes ago

Related Changes (2) Conflicts With (1)

- Hide 'refs/heads/' prefix in cherry-pick branch suggestion list
- Update gwtorm to version 1.9

Code-Review

Library-Compliance

Verified: -1 Difly Cuckoo (Automated Verifier)
+1 David Pursehouse

Files

File Path

Commit Message

lib/BUCK

History

David Pursehouse Uploaded patch set 1. 03/11/14 02:24
David Pursehouse Patch Set 1: Verified+1 03/11/14 03:24
David Pursehouse Uploaded patch set 2. 03/11/14 03:24
Difly Cuckoo (Aut...) Patch Set 2: Verified-1 Patchset merges but requires downloading new artifacts. The build cannot be completed from this point. A human will need to p... 03/11/14 04:35
David Pursehouse Patch Set 2: Verified+1 03/11/14 07:14

Permalink

Copy Link Location

Inspect Element

The change status shows the state of the change:

- Needs <label>:

The change is in review and an approval on the shown label is still required to make the change submittable.

- Not <label>:

The change is in review and a veto vote on the shown label is preventing the submit.

- Not Current:

The currently viewed patch set is outdated.

Please note that some operations, like voting, are not available on outdated patch sets, but only on the current patch set.

- Ready to Submit:

The change has all necessary approvals and may be submitted.

- Submitted, Merge Pending:

The change was submitted and was added to the merge queue.

The change stays in the merge queue if it depends on a change that is still in review. In this case it will get automatically merged when all predecessor changes have been merged.

This status can also mean that the change depends on an abandoned change or on an outdated patch set of another change. In this case you may want to rebase the change.

- Merged:

The change was successfully merged into the destination branch.

- Abandoned:

The change was abandoned.

- Draft:

The change is a draft that is only visible to the change owner, the reviewers that were explicitly added to the change, and users who have the [View Drafts](#) global capability assigned.

Commit Info Block

The commit info block shows information about the commit of the currently viewed patch set.

It displays the author and the committer as links to a list of this person's changes that have the same status as the currently viewed change.

The commit ID and the [Change-ID](#) are both displayed with a copy-to-clipboard icon that allows the ID to be copied into the clipboard.

If a Git web browser, such as GitWeb or Gitlets, is configured, there is also a link to the commit in the Git web browser.

The screenshot shows a detailed view of a commit in a Git interface. At the top, it displays the commit message: "Update gwtorm to version 1.9". Below this, it lists the author (David Pursehouse), committer (David Pursehouse), commit ID (7dbc394cc3d80d30c1ff9da7013d59f6009e23f8), and Change-ID (ld1badda0d0f24f602240be479d4681560ed0eda). A red circle highlights the "Commit Info" section, which includes the Author/Committer, Commit ID, and Change-ID. The interface also shows a list of files (lib/BUCK) and a history of previous commits by David Pursehouse and Dify Cuckoo.

If a merge commit is viewed this is highlighted by an icon. In this case the parent commits are also shown.

Author	Edwin Kempin <edwin.kempin@sap.com>	14.05.2014 10:05 AM
Committer	Edwin Kempin <edwin.kempin@sap.com>	14.05.2014 10:05 AM
Commit	64d986f48a2505cde0ef7aff8069a7761854cb6f	(gitlets)
Parents	976ced8f4fc0909d7e1584d18455299545881d60	(gitlets)
	3c10ee65396d681dc91689a9a8baa5590f603b58	(gitlets)
Change-ID	l64d986f48a2505cde0ef7aff8069a7761854cb6f	(gitlets)

Change Info Block

The change info block contains detailed information about the change and offers actions on the change.

A screenshot of the Gerrit interface showing a specific change. A red oval highlights the 'Change Info' section at the top left. Another red box highlights the 'Owner' and 'Reviewers' fields. A red arrow points from the 'Change Owner' label below to the 'Owner' field. The 'Owner' is listed as 'David Pursehouse'. The 'Reviewers' section shows 'Dify Cuckoo (Automated Verifier)' and 'Dave Borowitz' with a remove icon. Below this, project, branch, topic, strategy, and update information are listed. A 'Code-Review' section shows a -1 from Dify Cuckoo and a +1 from David Pursehouse. A 'Files' section shows a diff against base. A 'History' section shows patch set uploads and reviews. Buttons for 'Cherry Pick' and 'Abandon' are at the bottom.

- Change Owner:

The owner of the change is displayed as a link to a list of the owner's changes that have the same status as the currently viewed change.

A screenshot of the Gerrit interface showing the 'Change Owner' link. A red box highlights the 'Owner' field, which is 'David Pursehouse'. A red arrow points from the 'Change Owner' label below to this field. The 'Reviewers' section shows 'Dify Cuckoo (Automated Verifier)' and 'Dave Borowitz'. Below this, project, branch, topic, strategy, and update information are listed. A 'Code-Review' section shows a -1 from Dify Cuckoo and a +1 from David Pursehouse. A 'History' section shows patch set uploads and reviews. Buttons for 'Cherry Pick' and 'Abandon' are at the bottom.

- Reviewers:

The reviewers of the change are displayed as chip tokens.

For each reviewer there is a tooltip that shows on which labels the reviewer is allowed to vote.

New reviewers can be added by clicking on the `Add...` button. Typing into the pop-up text field activates auto completion of user and group names.

Reviewers can be removed from the change by clicking on the `x` icon in the reviewer's chip token. Removing a reviewer also removes the current votes of the reviewer. The removal of votes is recorded as a message on the change.

Removing reviewers is protected by permissions:

- Users can always remove themselves.
- The change owner may remove any zero or positive score.
- Users with the [Remove Reviewer](#) access right, the branch owner, the project owner and Gerrit administrators may remove anyone.

Reply...

Owner David Pursehouse

Reviewers Diffy Cuckoo (Automated Verifier) Add...

Dave Borowitz

Project gerrit

Branch master

Topic

Strategy Merge if Necessary

Updated 27 hours ago

Cherry Pick **Abandon**

Code-Review

Library-Compliance

Verified -1 Diffy Cuckoo (Automated Verifier)
+1 David Pursehouse

- Project / Branch / Topic:

The name of the project for which the change was done is displayed as a link to the [default dashboard](#) of the project. If no default dashboard is defined, the link opens a list of open changes on the project.

Clicking on the settings icon on the right side navigates to the project administration screen.

The name of the destination branch is displayed as a link to a list with all changes on this branch that have the same status as the currently viewed change.

If a topic was assigned to the change it is displayed below the branch. By clicking on the edit icon the topic can be set. This requires the [Edit Topic Name](#) access right. To be able to set a topic on a closed change, the `Edit Topic Name` must be assigned with the `force` flag.

Reply...

Owner David Pursehouse

Reviewers Diffy Cuckoo (Automated Verifier) Add...

Dave Borowitz

Project gerrit

Branch master

Topic

Strategy Merge if Necessary

Updated 27 hours ago

Cherry Pick **Abandon**

Code-Review

Library-Compliance

Verified -1 Diffy Cuckoo (Automated Verifier)
+1 David Pursehouse

- Submit Strategy:

The [submit strategy](#) that will be used to submit the change. The submit strategy is only displayed for open changes.

[Reply...](#)

Owner [David Pursehouse](#)
Reviewers [Diffy Cuckoo \(Automated Verifier\)](#) [Dave Borowitz](#) [Add...](#)

Project [gerrit](#)
Branch [master](#)
Topic
Strategy **Merge if Necessary**
Updated 27 hours ago

[Cherry Pick](#) [Abandon](#)

Code-Review
Library-Compliance
Verified -1 [Diffy Cuckoo \(Automated Verifier\)](#)
+1 [David Pursehouse](#)



If a change cannot be merged due to path conflicts this is highlighted by a bold red `Cannot Merge` label.

[Reply...](#)

Owner [David Pursehouse](#)
Reviewers [Diffy Cuckoo \(Automated Verifier\)](#) [Dave Borowitz](#) [Add...](#)

Project [gerrit](#)
Branch [master](#)
Topic
Strategy **Merge if Necessary**
Updated 27 hours ago

[Cherry Pick](#) [Abandon](#)

Code-Review
Library-Compliance
Verified -1 [Diffy Cuckoo \(Automated Verifier\)](#)
+1 [David Pursehouse](#)



- Time of Last Update:

Reply...

Owner	David Pursehouse
Reviewers	Diffy Cuckoo (Automated Verifier) <input checked="" type="checkbox"/> Add... Dave Borowitz <input checked="" type="checkbox"/>
Project	gerrit
Branch	master
Topic	
Strategy	Merge if Necessary
Updated	27 hours ago

Cherry Pick **Abandon**

Code-Review	
Library-Compliance	
Verified	-1 Diffy Cuckoo (Automated Verifier) +1 David Pursehouse

Time of
Last Update

- Actions:

Depending on the change state and the permissions of the user, different actions are available on the change:

- Submit:
Submits the change and adds it to the merge queue. If possible the change is merged into the destination branch.
The [Submit](#) button is available if the change is submittable and the [Submit](#) access right is assigned.
It is also possible to submit changes that have merge conflicts. This allows to do the conflict resolution for a change series in a single merge commit and submit the changes in reverse order.
- Abandon:
Abandons the change.
The [Abandon](#) button is only available if the change is open and the [Abandon](#) access right is assigned.
When a change is abandoned, a panel appears that allows one to type a comment message to explain why the change is being abandoned.
- Restore:
Restores the change.
The [Restore](#) button is only available if the change is abandoned and the [Abandon](#) and the [Push](#) access right is assigned.
When a change is restored, a panel appears that allows one to type a comment message to explain why the change is being restored.
- Rebase:
Rebases the change. The rebase is always done with content merge enabled. If the rebase is successful a new patch set with the rebased commit is created. If the rebase fails, there are conflicts that have to be resolved manually.
If the change does not depend on another open change, it is rebased onto the tip of the destination branch.
If the change depends on another open change, it is rebased onto the current patch set of that other change.
The [Rebase](#) button is only available if the change can be rebased and the [Rebase](#) access right is assigned. Rebasing merge commits is not supported.
- Cherry-Pick:
Allows to cherry-pick the change to another branch. The destination branch can be selected from a dialog. Cherry-picking a change creates a new open change on the selected destination branch.
It is also possible to cherry-pick a change to the same branch. This is effectively the same as rebasing it to the current tip of the destination branch. This can be used to remove dependencies on other open changes.
Users can only cherry-pick changes to branches for which they are allowed to upload changes for review.
- Publish:
Publishes the currently viewed draft patch set. If this is the first patch set of a change that is published, the change will be published as well.
The [Publish](#) button is only available if a draft patch set is viewed and the user is the change owner or has the [Publish Drafts](#) access right assigned.
- Delete Change / Delete Revision:

Deletes the draft change / the currently viewed draft patch set.

The Delete Change / Delete Revision buttons are only available if a draft patch set is viewed and the user is the change owner or has the [Delete Drafts](#) access right assigned.

- Further actions may be available if plugins are installed.

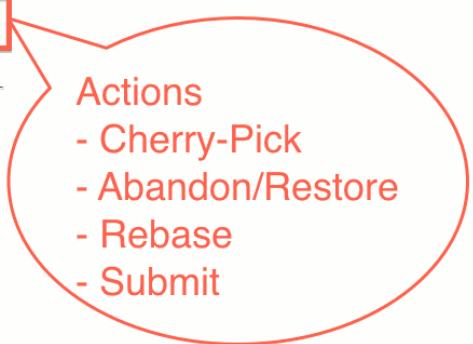
Reply...

Owner David Pursehouse
Reviewers Dify Cuckoo (Automated Verifier)
 Dave Borowitz

Project gerrit
Branch master
Topic
Strategy Merge if Necessary
Updated 27 hours ago

Cherry Pick **Abandon**

Code-Review
Library-Compliance
Verified -1 Dify Cuckoo (Automated Verifier)
+1 David Pursehouse



- Labels & Votes:

Approving votes are colored green; veto votes are colored red.

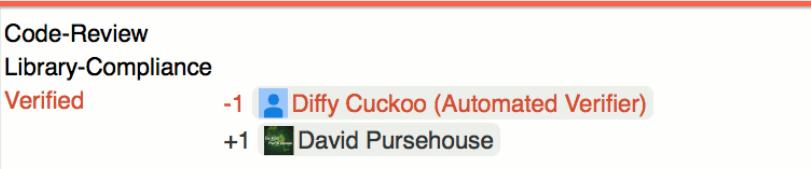
Reply...

Owner David Pursehouse
Reviewers Dify Cuckoo (Automated Verifier)
 Dave Borowitz

Project gerrit
Branch master
Topic
Strategy Merge if Necessary
Updated 27 hours ago

Cherry Pick **Abandon**

Code-Review
Library-Compliance
Verified -1 Dify Cuckoo (Automated Verifier)
+1 David Pursehouse



File List

The file list shows the files that are modified in the currently viewed patch set.

Change 56711 - Needs Code-Review

Extract the Web-static resources into a dedicated component

Preparation work in order to define a new type of server-side plugin capable of:
- servicing web-resources and their server-side expansions
- provide static Plugin for the HttpPluginService

Change-Id: Ib38455eed107d920389953e51dae8046360396dd

Owner: Luca Milanesio
Reviewers: David Pursehouse
Project: plugins/scripting/scala-provider
Branch: master
Topic: scripting-reloaded
Strategy: Merge if Necessary
Updated: 24 hours ago

Related Changes (5) Same Topic (4)

- Extract the Web-static resources into a dedicated component
- Serve static resources from directory-based Scala scripts
- Allows multiple Scala scripts to be included in a single plugin
- Display language features warnings during compile
- Support for Scala classes organised in packages

Code-Review Verified

Author: Luca Milanesio <lucamilanesio@gmail.com> 05/01/14 09:50
Committer: Luca Milanesio <lucamilanesio@gmail.com> 05/01/14 09:50
Commit: 637fbed9e2525099576e02bd44c1cb5fafe661
Change-Id: Ib38455eed107d920389953e51dae8046360396dd

Files

File Path	Comments	Size
Commit Message		
src/main/java/com/googlesource/gerrit/plugins/scripting/scala/ScalaPluginScanner.java	75	red
A src/main/java/com/googlesource/gerrit/plugins/web/LookAheadFileInputStream.java	106	green
A src/main/java/com/googlesource/gerrit/plugins/web/SSIPageInputStream.java	145	green
A src/main/java/com/googlesource/gerrit/plugins/web/WebPluginScanner.java	131	green
	+390, -67	green

History

Author	Message	Date
Luca Milanesio	Uploaded patch set 1.	05/01/14 09:51
Luca Milanesio	Topic set to scripting-reloaded	05/01/14 09:51
David Pursehouse	Patch Set 1: (3 comments)	05/01/14 10:23

The checkboxes in front of the file names allow files to be marked as reviewed.

Files

File Path	Comments	Size
gerrit/plugins/scripting/scala/ScalaPluginScanner.java	comments: 3	75
A src/main/java/com/googlesource/gerrit/plugins/web/LookAheadFileInputStream.java		106
A src/main/java/com/googlesource/gerrit/plugins/web/SSIPageInputStream.java		145
A src/main/java/com/googlesource/gerrit/plugins/web/WebPluginScanner.java		131
	+390, -67	green

The type of a file modification is indicated by the character in front of the file name:

- *no character* (Modified):
The file existed before this change and is modified.
- A (Added):
The file is newly added.
- D (Deleted):
The file is deleted.
- R (Renamed):
The file is renamed.
- C (Copied):
The file is new and is copied from an existing file.

Files

File Path	Comments	Size
Commit Message	comments: 3	
src/main/java/com/googlesource/gerrit/plugins/scripting/scala/ScalaPluginScanner.java		75
A src/main/java/com/googlesource/gerrit/plugins/web/LookAheadFileInputStream.java		106
A src/main/java/com/googlesource/gerrit/plugins/web/SSIPageInputStream.java		145
A src/main/java/com/googlesource/gerrit/plugins/web/WebPluginScanner.java		131
	+390, -67	green

If a file is renamed or copied, the name of the original file is displayed in gray below the file name.

Files

File Path	Comments	Size
Commit Message		
Documentation/config-gerrit.txt		
R gerrit-httppd/src/main/java/com/google/gerrit/httpd/GetUserFilter.java		
gerrit-pgm/src/main/java/com/google/gerrit/pgm/http/jetty/GetUserFilter.java		
gerrit-pgm/src/main/java/com/google/gerrit/pgm/Daemon.java		
gerrit-pgm/src/main/java/com/google/gerrit/pgm/http/jetty/HttpLog.java		

Repeating path segments are grayed out.

Repeating Path Segments are grayed out

File Path	Comments	Size
gerrit-pgm/src/main/java/com/google/gerrit/pgm/Daemon.java	drafts: 1	33
gerrit-pgm/src/main/java/com/google/gerrit/pgm/http/jetty/HttpLog.java	comments: 3	7
	new: 3	5

Inline comments on a file are shown in the `Comments` column.

Draft comments, i.e. comments that have been written by the current user but not yet published, are highlighted in red.

New comments from other users, that were published after the current user last reviewed this change, are highlighted in bold.

Drafts and new comments are highlighted

File Path	Comments	Size
Commit Message		
Documentation/dev-plugins.txt		
Documentation/rest-api-changes.txt		
gerrit-extension-api/src/main/java/com/google/gerrit/extensions/common/ListChangesOption.java		

The size of the modifications in the files can be seen in the `Size` column. The footer row shows the total size of the change.

For files, the `Size` column shows the sum of inserted and deleted lines as one number. For the total size, inserted and deleted lines are shown separately. In addition, the number of insertions and deletions is shown as a bar. The size of the bar indicates the amount of changed lines, and its coloring in green and red shows the proportion of insertions to deletions.

The size information is useful to easily spot the files that contain the most modifications; these files are likely to be the most relevant files for this change. The total change size gives an estimate of how long a review of this change may take.

Size of the File Modifications, Total Size of the Change

File Path	Comments	Size
src/main/java/com/googlesource/gerrit/plugins/web/WebPluginScanner.java		75
A src/main/java/com/googlesource/gerrit/plugins/web/WebPluginScanner.java		106
A src/main/java/com/googlesource/gerrit/plugins/web/WebPluginScanner.java		145
A src/main/java/com/googlesource/gerrit/plugins/web/WebPluginScanner.java		131
		+390, -67

In the header of the file list, the `Diff Against` selection can be changed. This selection allows one to choose if the currently viewed patch set should be compared against its base or against another patch set of this change. The file list is updated accordingly.

The file list header also provides an `Open All` button that opens the diff views for all files in the file list.

Open All File Diffs, Diff Against Selection

File Path	Comments	Size
src/main/java/com/googlesource/gerrit/plugins/scripting/sc		75
A src/main/java/com/googlesource/gerrit/plugins/web/LookAt		106
A src/main/java/com/googlesource/gerrit/plugins/web/SSIPageInputS		145
A src/main/java/com/googlesource/gerrit/plugins/web/WebPluginScanner.java		131
		+390, -67

Patch Sets

The change screen only presents one patch set at a time. Which patch set is currently viewed can be seen from the `Patch Sets` drop-down panel in the change header. It shows the number of the currently viewed patch set and the total number of patch sets, in the form: "current patch set/number of patch sets".

If a non-current patch set is viewed this is indicated by the `Not Current` change state. Please note that some operations are only available on the current patch set.

The screenshot shows the Gerrit interface for a code review. At the top right, there is a dropdown menu labeled "Patch Sets (2/2)". A red callout bubble points to this menu with the text "Patch Sets" and "- currently viewed patch set". Another part of the callout points to the number "2/2" with the text "number of patch sets".

Patch Sets
- currently viewed patch set
- number of patch sets

Below the header, there is a table for "Patch Sets" with two rows:

Patch Set	Commit	Date	Author / Committer
2	7dbc394cc3d80d30c1f9da7013d59f6009e23f8	03/11 03:24	David Pursehouse
1	2c400d5ee	03/11 02:24	David Pursehouse

The "Patch Set" column is highlighted with a red border.

The patch set drop-down list shows the list of patch sets and allows to switch between them. The patch sets are sorted in descending order so that the current patch set is always on top.

Patch sets that have unpublished draft comments are marked by a comment icon.

Draft patch sets are marked with DRAFT.

A red callout bubble points to the "Patch Sets (2/2)" dropdown with the text "List of Patch Sets" and "- newest on top".

Keys for Navigation between Patch Sets:
- 'n' = Next Patch Set
- 'p' = Previous Patch Set
- 'R' = Reload Change / Current Patch Set

Below the header, there is a table for "Patch Sets" with two rows:

Patch Set	Commit	Date	Author / Committer
2	7dbc394cc3d80d30c1f9da7013d59f6009e23f8	03/11 03:24	David Pursehouse
1	2c400d5ee	03/11 02:24	David Pursehouse

The "Patch Set" column is highlighted with a red border.

Download

The Download drop-down panel in the change header offers commands and links for downloading the currently viewed patch set.

Change 55243 - Needs Code-Review

Owner: Phil Lello

Reviewers: David Pursehouse, Dify Cuckoo, Kyle Laker, Nasser Grainawi, Rob Ward

Project: gerriet

Branch: master

Topic:

Strategy: Merge if Necessary

Updated: 6 days ago

Code-Review: Verified

Files:

File Path	Comments	Size
Commit Message	3	green
gerrit-gwtui/src/main/java/com/google/gerrit/client/change/RelatedChanges.java	29	green
gerrit-gwtui/src/main/java/com/google/gerrit/client/changes/ChangeScreen.java	54	green
gerrit-server/src/main/java/com/google/gerrit/server/change/GetRelated.java	77	green
gerrit-server/src/main/java/com/google/gerrit/server/git/validators/CommitValidators.java	+153, -10	green

History:

- Phil Lello Uploaded patch set 1. 03/18/14 21:51
- Dify Cuckoo (Aut... Patch Set 1: Verified+1 Patchset merges and builds. Congratulations! 03/18 22:47
- Phil Lello Patch Set 2: Commit message was updated 03/18 23:13

Patch Sets (15/15) ▾ Download ▾

The available download commands depend on the installed Gerrit plugins. The most popular plugin for download commands, the [download-commands](#) plugin, provides commands to checkout, pull and cherry-pick a patch set.

Each command has a copy-to-clipboard icon that allows the command to be copied into the clipboard. This makes it easy to paste and execute the command on a Git command line.

If several download schemes are configured on the server (e.g. SSH and HTTP) there is a drop-down list to switch between the download schemes. Gerrit automatically remembers the download scheme that was last chosen and selects this download scheme the next time the download commands drop-down panel is opened.

The **Patch-File** links provide the Git patch file for the currently viewed patch set for download. The patch file can be base64 encoded or zipped.

The **Archive** links allow one to download an archive with the contents of the currently viewed patch set. The archive is offered in several formats (e.g. tar and tbz2); which formats are available depends on the configuration of the server.

Change 55243 - Needs Code-Review

Cross-project dependencies

This is a patch to allow manual addition of dependencies in other projects, by adding a Depends-On: footer.

e.g. Depends-On: <projectname>~<branch>~<Change-Id>

Note that this is a work-in-progress requiring further testing, but is hopefully complete.

Additional considerations:

- Need to check what happens if a dependency is added
- It might be nice to allow a numeric range

Feature: Issue 377

Change-Id: Idc47ce55677c87fb528a1d1c5225ba70fc3abc8

Owner: Phil Lello <philip@catalyst-eu.net>

Reviewer: Rob Ward <robert.ward114@gmail.com>

Commit: d556cea4601a4758e567b308221657d8bd95a0e

Change-Id: Idc47ce55677c87fb528a1d1c5225ba70fc3abc8

Files:

File Path	Comments	Size
Commit Message	3	green
gerrit-gwtui/src/main/java/com/google/gerrit/client/change/RelatedChanges.java	29	green
gerrit-gwtui/src/main/java/com/google/gerrit/client/changes/ChangeScreen.java	54	green
gerrit-server/src/main/java/com/google/gerrit/server/change/GetRelated.java	77	green
gerrit-server/src/main/java/com/google/gerrit/server/git/validators/CommitValidators.java	+153, -10	green

History:

- Phil Lello Uploaded patch set 1. 03/18/14 21:51
- Dify Cuckoo (Aut... Patch Set 1: Verified+1 Patchset merges and builds. Congratulations! 03/18 22:47
- Phil Lello Patch Set 2: Commit message was updated 03/18 23:13
- David Pursehouse Patch Set 3: Commit message was updated 03/19 01:44

Patch Sets (15/15) ▾ Download ▾

Download Commands

- available Commands depend on Installed Plugins
- download Patch File & Git Archive

Included In

For merged changes the **Included In** drop-down panel is available in the change header.

Included In

The screenshot shows a Gerrit change detail page for Change 51560 - Merged. The top right corner features a dropdown menu labeled "Included in" with a red oval around it. Below the menu, the "Patch Sets (1/1)" and "Download" buttons are visible. The main content area displays the change's details, including the owner (Shawn Pearce), reviewers (Colby Ranger), project (gergit), branch (stable-2.8), topic, and update time (4 months ago). It also shows code reviews (+2), cherry picks, and verified status. A "Files" section lists a single file path (gerrit-server/src/main/java/com/google/gerrit/server/change/Files.java) with a diff against the base. The "History" section shows a merge commit from Shawn Pearce at 11/09/13 01:02, indicating the change has been successfully merged into the git repository.

The **Included In** drop-down panel shows the branches and tags in which the change is included. E.g. if a change fixes a bug, this allows to quickly see in which released versions the bug-fix is contained (assuming that every release is tagged).

Branches/Tags in which this Change is included

This screenshot is similar to the first one but focuses on the "Cherry-Picks" dropdown menu, which is highlighted with a red oval. The menu lists "Cherry-Picks (1)" pointing to the master branch, along with "Branches" (master, stable-2.8) and "Tags" (v2.8, v2.8-rc3, v2.8.1, v2.8.2). The rest of the page content is identical to the first screenshot, showing the change details, file history, and merge commit.

Star Change

The star icon in the change header allows to mark the change as a favorite. Clicking on the star icon again, unstars the change.

Mark Change as Favorite

This screenshot shows a Gerrit change detail page for Change 55060 - Needs Code-Review. The top right corner features a star icon with a red oval around it, indicating the change is marked as a favorite. The main content area displays the change's details, including the owner (David Pursehouse), reviewers (Dify Cuckoo (Automated Verifier) and Dave Borowitz), project (gergit), branch (master), strategy (Merge if Necessary), and update time (26 hours ago). It also shows code reviews (-1), cherry picks, and abandoned status. A "Files" section lists a single file path (libBUCK) with a diff against the base. The "History" section shows a patch set upload from David Pursehouse at 03/11/14 02:24 and a verification from Dify Cuckoo at 03/11/14 03:24.

Starring a change turns on email notifications for this change.

Starred changes are listed under [My > Starred Changes](#), and can be queried by the [is:starred](#) search operator.

Related Changes

If there are changes that are related to the currently viewed change they are displayed in the third column of the change screen.

There are several lists of related changes and a tab control is used to display each list of related changes in its own tab.

The following tabs may be displayed:

- Related Changes:

This tab page shows changes on which the current change depends (ancestors) and open changes that depend on the current change (descendants). For merge commits it also shows the closed changes that will be merged into the destination branch by submitting the merge commit.

The changes are sorted in the same way as the output of `git log`. This means the relationship between the changes can be inferred from the position of the changes in the list. Changes listed above the current change are descendants; changes below the current change are ancestors.

For merged changes this tab is only shown if there are open descendants.

Dependencies

- sorted same way as 'git log' output
- orange dot = not current
- green tilde = indirect descendant
- black dot = merged commit

Change 56097 - Needs Code-Review

Owner Luca Milanesio
Reviewers Dariusz Łukasz, David Ostrovsky, Difly Cuckoo (Automated Verifier), Shawn Pearce, gerit

Project gerit

Branch master

Patch Sets (12/15) ▾ Download ▾

Related Changes (5) Same Topic (8) Conflicts With (2)

- Serve static resources for non-jar Server plugins
- Introduce support for externally loaded plugins
- Decouple plugins from their "jar" external form
- Renaming JarPlugin to ServerPlugin for future extensions
- Removed unused methods and fields in JarScanner

Reply... Add...

Change-Id: 1769595a030545a5f272f453c3cf435b747719e167

Author Luca Milanesio <luka.milanesio@gmail.com>
Committer Luca Milanesio <luka.milanesio@gmail.com>
Commit ab9dcbca2b25974c6979a85f2bb19dbba27e44
Change-Id: 1769595a030545a5f272f453c3cf435b747719e167

Files

- Commit Message
- gerit-server/src/main/java/com/google/gerit/server/plugins/AutoRegister.java
- gerit-server/src/main/java/com/google/gerit/server/plugins/JarScanner.java
- gerit-server/src/main/java/com/google/gerit/server/plugins/JsPlugin.java
- gerit-server/src/main/java/com/google/gerit/server/plugins/ListPlugins.java
- gerit-server/src/main/java/com/google/gerit/server/plugins/Plugin.java
- A gerit-server/src/main/java/com/google/gerit/server/plugins/PluginContentScanner.java
- A gerit-server/src/main/java/com/google/gerit/server/plugins/PluginEntry.java
- gerit-server/src/main/java/com/google/gerit/server/plugins/PluginLoader.java
- gerit-server/src/main/java/com/google/gerit/server/plugins/ServerPlugin.java

History

Luca Milanesio	Uploaded patch set 1.	04/18 00:46
Luca Milanesio	Topic set to scripting-reloaded	04/18 00:47
Difly Cuckoo (Aut...	Patch Set 1: Verified+1 Patchset merges and builds. Congratulations!	04/18 05:41

Open All Diff All Expand All

2
130
64
4
24
+307, -35

Related changes may be decorated with an icon to signify dependencies on outdated patch sets, or commits that are not associated to changes under review:

- Orange Dot:

The selected patch set of the change is outdated; it is not the current patch set of the change.

If an ancestor change is marked with an orange dot it means that the currently viewed patch set depends on a outdated patch set of the ancestor change. This is because a new patch set for the ancestor change was uploaded in the meantime and as result the currently viewed patch set now needs to be rebased.

If a descendant change is marked with an orange dot it means that an old patch set of the descendant change depends on the currently viewed patch set. It may be that the descendant was rebased in the meantime and with the new patch set this dependency was removed.

- Green Tilde:

The selected patch set of the change is an indirect descendant of the currently viewed patch set; it has a dependency to another patch set of this change. E.g. this could mean that a new patch set was uploaded for this change and the descendant change now needs to be rebased. Please note that following the link to an indirect descendant change may result in a completely different related changes listing.

- Black Dot:

Indicates a merged ancestor, e.g. the commit was directly pushed into the repository bypassing code review, or the ancestor change was reviewed and submitted on another branch. The latter may indicate that the user has accidentally pushed the commit to the wrong branch, e.g. the commit was done on `branch-a`, but was then pushed to `refs/for/branch-b`.

Related Changes (5)

Same Topic (8)

Conflicts With (2)

Serve static resources for non-jar Server plugins

Introduce support for externally loaded plugins

► Decouple plugins from their "jar" external form

Renaming JarPlugin to ServerPlugin for future extensions

Removed unused methods and fields in JarScanner

• Conflicts With:

This tab page shows changes that conflict with the current change. Non-mergeable changes are filtered out; only conflicting changes that are mergeable are shown.

If this change is merged, its conflicting changes will have merge conflicts and must be rebased. The rebase of the other changes with the conflict resolution must then be done manually.

Change 55060 - Needs Code-Review

Update gwtorm to version 1.9

Updates the versions of protobuf, asm and guava libraries to synchronize with the versions used in GWT 2.6.0

Change-Id: Id1badda0d0f24f6022404be479d4681560ed0eda

Author: David Pursehouse <david.pursehouse@sonymobile.com> 03/11/14 02:24
Commiter: David Pursehouse <david.pursehouse@sonymobile.com> 03/11/14 03:24
Commit: 76bc394cc3d80d30c1ff9da7013d59f6009e23f8
Change-Id: Id1badda0d0f24f6022404be479d4681560ed0eda

Files

File Path	Comments	Size
Commit Message	6	-
libBUCK	+3, -3	-

History

Author	Message	Date
David Pursehouse	Uploaded patch set 1.	03/11/14 02:25
David Pursehouse	Patch Set 1: Verified+1	03/11/14 02:34
David Pursehouse	Uploaded patch set 2.	03/11/14 03:24
Diffy Cuckoo (Aut...)	Patch Set 1: Verified-1 Patchset merges but requires downloading new artifacts. The build cannot be completed from this point. A human will need to p...	03/11/14 04:35
David Pursehouse	Patch Set 2: Verified+1	03/11/14 07:14

• Same Topic:

This tab page shows changes that have the same topic as the current change. Only open changes are included in the list.

Change 54508 - Needs Code-Review

Support for Scripting Plugins self-registration

As scripting plugins do not have a MANIFEST.MF associated we need to allow the self-registration of classes defined and exported by the script, using the same annotation defined for regular plugins (@Export, @Listen).

Change-Id: I4f1e30b074c0bce818cdaec21d1760061f5e56f1

Author: Luca Milanesio <lucamilanesio@gmail.com> 02/11/14 22:50
Commiter: David Ostrovsky <david.ostrovsky.org> 03/07/14 01:13
Commit: c2c5e3ae361b4c8c2883c38f154a78a4fd772ef3
Change-Id: I4f1e30b074c0bce818cdaec21d1760061f5e56f1

Files

File Path	Comments	Size
Commit Message	comments: 2	210
gerrit-server/src/main/java/com/google/gerrit/server/plugins/AutoRegisterScript.java	+210, -0	-

History

Author	Message	Date
Luca Milanesio	Uploaded patch set 1.	02/11/14 22:52
Luca Milanesio	Patch Set 2: Commit message was updated	02/11/14 22:53
Luca Milanesio	Uploaded patch set 3.	02/11/14 23:34
Luca Milanesio	Uploaded patch set 4.	02/12/14 00:48
Luca Milanesio	Topic set to scripting-plugins	02/12/14 00:55

• Cherry-Picks:

This tab page shows changes with the same [Change-Id](#) for the current project.

Abandoned changes are filtered out.

For each change in this list the destination branch is shown as a prefix in front of the change subject.

Change 54942 - Needs Code-Review

Fix false rework detection on rebase and commit message update
When a change was both rebased and its commit message was modified, it is detected as REWORK instead of TRIVIAL_REBASE.

Change-Id: [Ic54ef16f4a48fcc5c8ddc48d715cb5c2fc263ea2](#)

Owner: Orgad Shanh
Reviewers: David Ostrovsky, Dify Cuckoo (Automated Verifier), Dave Borowitz
Project: gerit
Branch: master
Topic: [\(gfiles\)](#)
Strategy: Merge if Necessary
Updated: 2 hours ago

Code-Review: -1 (David Ostrovsky)
Verified: +1 (Dify Cuckoo (Automated Verifier))

Cherry-Picks (1)
stable-2.8: Fix false rework detection on rebase and comm...

Author: Orgad Shanh <orgads@gmail.com>
Committer: Orgad Shanh <orgads@gmail.com>
Commit: 4e991489321bc422a61d5c09ae0fe4932cccd499
Change-Id: Ic54ef16f4a48fcc5c8ddc48d715cb5c2fc263ea2

Files: Open All Diff against: Base

File Path	Comments	Size
Commit Message	comments: 2 13	+2, -11
gerit-server/src/main/java/com/google/gerrit/server/change/ChangeKindCache.java		

History: Expand All

Orgad Shanh	Uploaded patch set 1.	03/03/14 13:25
Dify Cuckoo (Aut...)	Patch Set 1: Verified+1 Patchset merges and builds. Congratulations!	03/03/14 09:27
Orgad Shanh	Uploaded patch set 2.	09:28
Dify Cuckoo (Aut...)	Patch Set 2: Verified+1 Patchset merges and builds. Congratulations!	11:41

If there are no related changes for a tab, the tab is not displayed.

Reply

The Reply... button in the change header allows to reply to the currently viewed patch set; one can add a summary comment, publish inline draft comments, and vote on the labels.

Change 55060 - Needs Code-Review

Update gwterm to version 1.9
Updates the versions synchronize with

Change-Id: [re](#)

Owner: David Pursehouse
Reviewers: David Pursehouse, Dify Cuckoo (Automated Verifier), Dave Borowitz
Project: gerit
Branch: master
Topic: [\(gfiles\)](#)
Strategy: Merge if Necessary
Updated: 26 hours ago

Code-Review: Library-Compliance
Verified: -1 (Dify Cuckoo (Automated Verifier))
+1 (David Pursehouse)

Related Changes (2) Conflicts With (1)
Hide 'refs/heads/' prefix in cherry-pick branch suggestion list
Update gwterm to version 1.9

Author: David Pursehouse <david.pursehouse@sonymobile.com>
Committer: David Pursehouse <david.pursehouse@sonymobile.com>
Commit: 7dbc394cc3d80d30c1ff9da7013d59f6009e23fb
Change-Id: Id1badda0d0124f6022404be479d4681560ed0eda

Files: Open All Diff against: Base

File Path	Comments	Size
Commit Message	6	+3, -3
lib/BUCK		

History: Expand All

David Pursehouse	Uploaded patch set 1.	03/11/14 02:24
David Pursehouse	Patch Set 1: Verified+1	03/11/14 03:24
David Pursehouse	Uploaded patch set 2.	03/11/14 03:24
Dify Cuckoo (Aut...)	Patch Set 2: Verified-1 Patchset merges but requires downloading new artifacts. The build cannot be completed from this point. A human will need to p...	03/11/14 04:35
David Pursehouse	Patch Set 2: Verified+1	03/11/14 07:14

Clicking on the Reply... button opens a popup panel.

A text box allows to type a summary comment for the currently viewed patch set.

If the current patch set is viewed, radio buttons are displayed for each label on which the user is allowed to vote. Voting on non-current patch sets is not possible.

Typing "LGTM" (acronym for *Looks Good To Me*) in the summary comment text box automatically selects the highest possible score for the *Code-Review* label.

The inline draft comments that will be published are displayed in a separate section so that they can be reviewed before publishing. There are links to navigate to the inline comments which can be used if a comment needs to be edited.

The Post button publishes the comments and the votes.

The send email checkbox controls whether the reply should trigger email notifications for other users. Deselecting the checkbox means that there will be no email notification about the change update to the change author, the reviewers or any other user.

The screenshot shows a Gerrit change detail page for Change 55060. A red circle highlights the 'Reply' button in the top right corner of the main content area. Below it, a red box encloses the reply dialog window. The dialog contains a summary comment: 'Seems to work pretty well, but I would like that another committer also approves this.' It includes a 'Code-Review' rating scale from -2 to +2, with a blue dot at +1. The 'Verified' score is +1. A note says 'Looks good to me, but someone else must approve'. There's an 'inline comment' for 'lib/BUCK' with the message 'This is an inline comment for demo purposes.' Buttons for 'Post' and 'Cancel' are at the bottom. The 'History' section below shows a log of changes made by various users, including David Purhouse and Dify Cuckoo.

If a user can approve a label that is still required, a quick approve button appears in the change header that allows to add this missing approval by a single click. The quick approve button only appears if there is a single label that is still required and can be approved by the user.

E.g. if a change requires approvals on the *Code-Review* and the *Verified* labels, and there is already a *+1 Verified* vote, then if the user is allowed to vote the max score on *Code-Review*, a *Code-Review+2* quick approve button appears that approves the *Code-Review* label if clicked.

Using the quick approve button also publishes all inline draft comments; a summary comment is only added if the reply popup panel is open when the quick approve button is clicked.

This screenshot shows a Gerrit change detail page for Change 54942. A red circle highlights the 'Quick Approve' button in the top right corner of the main content area. Below it, a red box encloses the 'Code-Review' section of the change header, which shows a 'Code-Review+2' button. The 'History' section shows a log of changes made by Orgad Shaneh, David Ostrovsky, and Dify Cuckoo.

History

The history of the change can be seen in the lower part of the screen.

The history contains messages for all kinds of change updates, e.g. a message is added when a new patch set is uploaded or when a review was done.

Messages with new comments from other users, that were published after the current user last reviewed this change, are automatically expanded.

Change 55243 - Needs Code-Review

[Edit Message](#) [Reply...](#)

Owner: Phil Lello
Reviewers: [David Pursehouse](#) [Dify Cuckoo \(Automated Verifier\)](#) [Kyle Laker](#) [Nasser Grainawi](#) [Rob Ward](#) [Add...](#)

Project: gerriet
Branch: master
Topic:
Strategy: Merge if Necessary
Updated: 7 days ago

[Cherry Pick](#) [Rebase](#) [Abandon](#)

Code-Review: Verified

Author: Phil Lello <philip@catalyst-eu.net> 03/18/14 21:51
Committer: Rob Ward <robert.ward114@googlemail.com> 04/29/14 21:33
Commit: d556cea46014a758e567308221657d8bd95a0e
Change-Id: ldc47ce55677c87fb528a1d1c5225ba70fc3abc8

[\(gitiles\)](#)

Files: [Open All](#) Diff against: Base

File Path	Comments	Size
gerriet-gwtui/src/main/java/com/google/gerrit/client/change/RelatedChanges.java	3	
gerriet-gwtui/src/main/java/com/google/gerrit/client/changes/ChangeScreen.java	29	
gerriet-server/src/main/java/com/google/gerrit/server/change/GetRelated.java	54	
gerriet-server/src/main/java/com/google/gerrit/server/glitValidators/CommitValidators.java	77	
	+153, -10	

History: [Expand All](#)

- Phil Lello Uploaded patch set 1.
- Dify Cuckoo (Aut... Patch Set 1: Verified+1 Patchset merges and builds. Congratulations!
- Phil Lello Patch Set 2: Commit message was updated
- David Pursehouse Patch Set 3: Commit message was updated
- Nasser Grainawi Patch Set 3: (1 comment)
- Phil Lello Patch Set 3: (1 comment)
- Phil Lello Uploaded patch set 4.
- Dify Cuckoo (Aut... Patch Set 4: Verified+1 Patchset merges and builds. Congratulations!
- David Pursehouse Patch Set 4: (1 comment)
- David Pursehouse Patch Set 4: Note that something like this was attempted before: <https://gerrit-review.googlesource.com/#/c/44052/>
- Kyle Laker Patch Set 4: Code-Review-1 (1 comment)
- Phil Lello Uploaded patch set 5.
- Phil Lello Uploaded patch set 6.
- Phil Lello Uploaded patch set 7.
- Dify Cuckoo (Aut... Patch Set 7: Verified+1 Patchset merges but does not build. The following snippet of output may point you in the right direction: [-] PARSENG BUILD FIL...
- Phil Lello Patch Set 8: Commit message was updated
- Phil Lello Patch Set 9: Patch Set 8 was rebased
- Dify Cuckoo (Aut... Patch Set 9: Verified+1 Patchset merges but does not build. The following snippet of output may point you in the right direction: [-] PARSENG BUILD FIL...
- Phil Lello Uploaded patch set 10.

Patch Set 10:

| Note that something like this was attempted before:<https://gerrit-review.googlesource.com/#/c/44052/>

It looks like activity stalled on that about 9 months ago, and that it had a wider scope. I'd like to stick with this feature, then progressively enhance it once it's in the wild.

Change History
- New Comments are automatically expanded

It is possible to directly reply to a change message by clicking on the reply icon in the right upper corner of a change message. This opens the reply popup panel and prefills the text box with the quoted comment. Then the reply can be written below the quoted comment or inserted inline. Lines starting with ">" will be rendered as a block quote. Please note that for a correct rendering it is important to leave a blank line between a quoted block and the reply to it.

The screenshot shows a Gerrit Code Review change page. At the top right, there's a red box highlighting a comment from 'Difly Cuckoo' which has been quoted in a reply message. A red arrow points from this box to a callout bubble labeled 'Quote Old Comment when Replying'. Below the comment, a red box highlights the original comment text again, with a red arrow pointing to a callout bubble labeled 'Block Quote for Old Comment on Reply'. Another red box highlights the original comment text at the bottom of the page, with a red arrow pointing to a callout bubble labeled 'Reply to Comment'.

Inline comments are directly displayed in the change history and there are links to navigate to the inline comments.

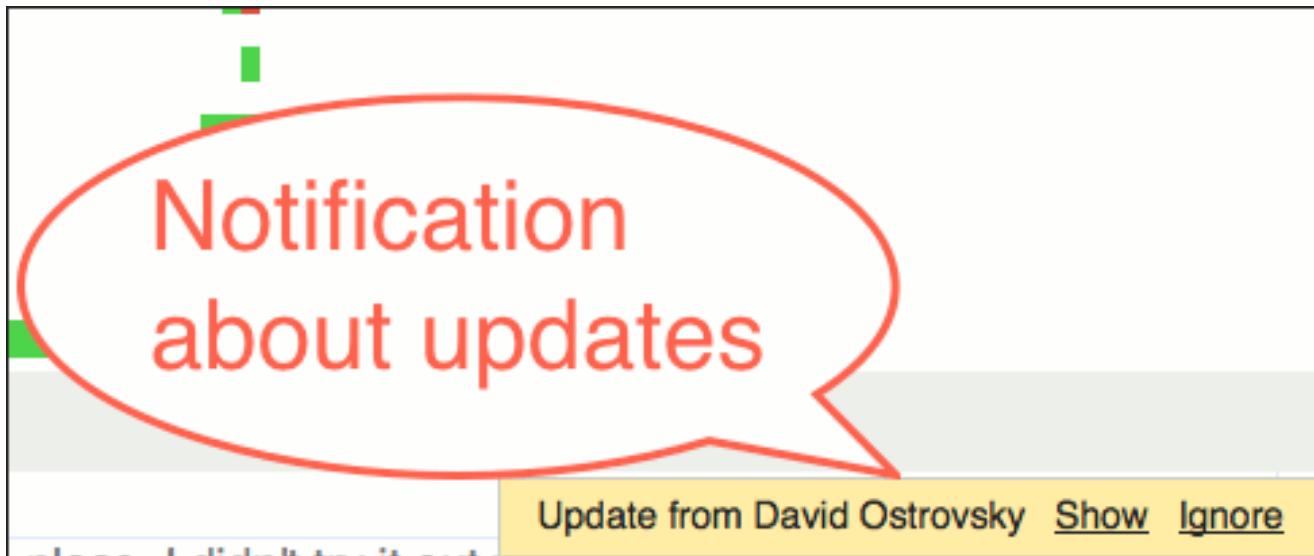
This screenshot shows another Gerrit Code Review change page. A red box highlights a comment from 'Sasa Zivkov' which contains a link to another comment. A red arrow points from this box to a callout bubble labeled 'Inline Comments - with links'. The original comment text is also highlighted with a red arrow pointing to a callout bubble labeled 'Reply to Comment'.

The **Expand All** button expands all messages; the **Collapse All** button collapses all messages.

Update Notification

The change screen automatically polls for updates to the currently viewed change. If there is an update the user is informed by a popup panel in the bottom right corner.

The polling frequency depends on the server configuration; by default it is 30 seconds. Polling may also be completely disabled by the administrator.



Plugin Extensions

Gerrit plugins may extend the change screen; they can add buttons for additional actions to the change info block and display arbitrary UI controls below the change info block.

A screenshot of the Gerrit interface showing a change detail page. The top navigation bar includes "Edit Message", "Reply...", "Code-Review+2", "Patch Sets (1/1)", "Download", and other icons. Below the header, there's a "Modify access rules" section. The main content area shows author information (Administrator <edwin.kempin@sap.com>, 16.01.2014 2:54 PM), committer information (Gerrit Code Review <d044411@wdflm00349532a.dhcp.wdf.sap.corp>, 16.01.2014 2:54 PM), commit details (Commit 6659a20aae949a344784c365c6bea8ef2d0b30fa, Change-Id l6659a20aae949a344784c365c6bea8ef2d0b30fa), and a "Files" section with a "Commit Message" file. A "Code-Review" section shows "Verified" status with three buttons: "Bonjour France", "Hallo Germany", and "Hello USA". A red speech bubble labeled "Plugin Extensions" points to the "Hello USA" button. Another red speech bubble points to the "Say hello" button in the "Code-Review" section. The bottom of the page has a "History" section and a "Expand All" button.

Old Change Screen

In addition to the normal change screen, this Gerrit version still includes the old change screen that was used in earlier Gerrit versions. Users that want to continue using the old change screen can configure it in their preferences under `Settings > Preferences > Change View`:

Settings

Profile
Preferences
Watched Projects
Contact Information
HTTP Password
Identities
Groups
Agreements

Maximum Page Size: 100 rows per page

Date/Time Format: 04/30 ; 04/30/14 : 09:50

Comment Visibility (deprecated: Old Change Screen): Collapse All

Change View: Server Default (New Screen)

Diff View (New Change Screen): Side by Side

The 'Change View' dropdown menu contains four options: 'Server Default (New Screen)', 'Server Default (New Screen)', 'Old Screen', and 'New Screen'. The 'Old Screen' option is highlighted with a blue background.

Warning The old change screen will be removed in a later version of Gerrit.

Side-by-Side Diff Screen

The side-by-side diff screen shows a single patch; the old file version is displayed on the left side of the screen; the new file version is displayed on the right side of the screen.

This screen allows to review a patch and to comment on it.

```
gitiles / gitiles-servlet/src/main/java/com/google/gitiles/LogServlet.java
Patch Set Base 1 2 3 4 5 6 7
...
11 // WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
12 // See the License for the specific language governing permissions and
13 // limitations under the License.
14
15 package com.google.gitiles;
16
17 import static com.google.common.base.Preconditions.checkNotNull;
18 import static javax.servlet.http.HttpServletResponse.SC_INTERNAL_SERVER_ERROR;
19 import static javax.servlet.http.HttpServletResponse.SC_NOT_FOUND;
20
21 import com.google.common.base.Optional;
22 import com.google.common.collect.Iterables;
23 import com.google.common.collect.ListMultimap;
24 import com.google.common.collect.Lists;
25 import com.google.common.collect.Maps;
26 import com.google.common.primitives.Longs;
27 import com.google.gson.reflect.TypeToken;
28
29 import org.eclipse.jgit.errors.IncorrectObjectTypeException;
30 import org.eclipse.jgit.errors.ObjectTypeException;
31 import org.eclipse.jgit.errors.UnsupportedObjectException;
32
33 import java.util.Map;
34
35 import javax.servlet.http.HttpServletRequest;
36 import javax.servlet.http.HttpServletResponse;
37
38 /**
39 * Serves an HTML page with a shortlog for commits and paths. */
40 public class LogServlet extends BaseServlet {
41     private static final long serialVersionUID = 1L;
42     private static final Logger log = LoggerFactory.getLogger(LogServlet.class);
43
44     ...
45
46     static final String LIMIT_PARAM = "n";
47     static final String START_PARAM = "s";
48     private static final int DEFAULT_LIMIT = 100;
49     private static final int MAX_LIMIT = 10000;
50
51     private final Linkifier linkifier;
52
53     public LogServlet(Renderer renderer, Linkifier linkifier) {
54         super(renderer);
55
56         this.linkifier = checkNotNull(linkifier, "linkifier");
57     }
58
59     @Override
60     protected void doGetHTML(HttpServletRequest req, HttpServletResponse res) throws IOException {
61         Repository repo = ServletUtils.getRepository(req);
62         GitilesView view = getView(req, repo);
63         Paginator paginator = newPaginator(repo, view);
64         if (paginator == null) {
65             ...
66
67             ...
68
69             ...
70
71             ...
72             static final String LIMIT_PARAM = "n";
73             static final String START_PARAM = "s";
74             private static final int DEFAULT_LIMIT = 100;
75             private static final int MAX_LIMIT = 10000;
76
77             private final GitilesAccess.Factory accessFactory;
78             private final Linkifier linkifier;
79
80             public LogServlet(GitilesAccess.Factory accessFactory, Renderer renderer, Linkifier linkifier) {
81                 super(renderer);
82                 this.accessFactory = checkNotNull(accessFactory, "accessFactory");
83                 this.linkifier = checkNotNull(linkifier, "linkifier");
84             }
85
86             ...
87             ...
88             ...
89             ...
90             ...
91             ...
92         }
93     }
94 }
```

Shawn Pearce We should get logDetail worked into this somehow. logDetail for the ... 03/10 17:34
Stefan Zager I defer to your judgement. Changing the subsection to "logDetail" me... 03/11 21:32

In the screen header the project name and the name of the viewed patch file are shown.

If a Git web browser is configured on the server, the project name and the file path are displayed as links to the project and the folder in the Git web

browser.

The screenshot shows a Git commit diff for a Java file named LogServlet.java. A red oval highlights the project name 'gitiles' and the file name 'LogServlet.java' at the top left. Another red oval highlights the word 'Patch Set Base' just below the file path. The code itself is mostly standard Java, with some annotations and imports. A large green bar covers the right side of the code area, containing several inline comments from users like Shawn Pearce and Stefan Zager. These comments include timestamps and brief descriptions, such as 'We should get logDetail worked into this somehow. logDetail for the ...'. The scrollbar on the right indicates that there are many more lines of code and comments visible.

The checkbox in front of the project name and the file name allows the patch to be marked as reviewed. The [Mark Reviewed](#) diff preference allows to control whether the files should be automatically marked as reviewed when they are viewed.

This screenshot is similar to the previous one, showing the same Java file LogServlet.java. A prominent red oval on the left side of the code area contains the text 'Mark Patch as Reviewed'. The rest of the interface is identical to the first screenshot, showing the code, annotations, and inline comments from users. The scrollbar on the right is also present.

The scrollbar shows patch diffs and inline comments as annotations. This provides a good overview of the lines in the patch that are relevant for reviewing. By clicking on an annotation one can quickly navigate to the corresponding line in the patch.

This screenshot shows a side-by-side comparison of two versions of a Java file, LogServlet.java, from the gitiles project. The left pane shows the base version, and the right pane shows the current version. A red oval highlights the right pane's header which displays a list of patch sets (Base, 1, 2, 3, 4, 5, 6, 7) and a note about skipped common lines. Another red oval highlights the right pane's sidebar, which contains three sections: 'Scrollbar with Annotations', '- Overview of Diffs / Comments', and '- Quick Navigation'. Below the sidebar, a green box contains a message from Shawn Pearce and Stefan Zager. A vertical red bar is visible between the two panes, indicating a gap between lines.

A gap between lines in the file content that is caused by aligning the left and right side or by displaying inline comments is shown as a vertical red bar in the line number column. This prevents a gap from being mistaken for blank lines in the file

This screenshot shows the same code diff interface as the previous one, but with a red bar drawn across the line number column between line 61 and line 62. A red oval surrounds this bar and the text 'Red Bar to visualize Gap between Lines'. The rest of the interface is identical to the first screenshot, showing the patch set selection at the top and the detailed sidebar below.

In the header, on each side, the list of patch sets is shown. Clicking on a patch set changes the selection for the patch set comparison and the screen is refreshed to show the diff between the selected patch sets. The currently selected patch set is highlighted by a light blue background.

On the left side Base can be selected to compare a patch set against its base. For merge commits Auto Merge is available instead which allows to compare the patch against the result of the auto merge. The auto merge version may contain Git conflict markers and is useful for reviewing how conflicts are resolved by a patch.

Reviewers that are reviewing a patch for the first time look at its diff against its base; reviewers that have reviewed an old patch version before, may see what has changed since that version by comparing the old patch against the current patch.

Choose Patch Sets that are diffed

```

Patch Set Base 1 2 3 4 5 6 7 ...
... Skipped 10 common lines ...
11 // WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
12 // See the License for the specific language governing permissions and
13 // limitations under the License.
14
15 package com.google.gitiles;
16
17 import static com.google.common.base.Preconditions.checkNotNull;
18 import static javax.servlet.http.HttpServletResponse.SC_INTERNAL_SERVER_ERROR;
19 import static javax.servlet.http.HttpServletResponse.SC_NOT_FOUND;
20
21 import com.google.common.base.Optional;
22 import com.google.common.base.Strings;
23 import com.google.common.collect.Iterables;
24 import com.google.common.collect.ListMultimap;
25 import com.google.common.collect.Lists;
26 import com.google.common.collect.Maps;
27 import com.google.common.primitives.Longs;
28 import com.google.gson.reflect.TypeToken;
29
30 import org.eclipse.jgit.errors.IncorrectObjectTypeException;
31 import org.eclipse.jgit.errors.ObjectNotFoundException;
32 import org.eclipse.jgit.errors.RepositoryFormatException;
33 import org.eclipse.jgit.errors.RevWalkException;
34 import org.eclipse.jgit.http.server.ServletUtils;
35 import org.eclipse.jgit.lib.AbbreviatedObjectId;
36 import org.eclipse.jgit.lib.Constants;
37 import org.eclipse.jgit.lib.ObjectReader;
38 import org.eclipse.jgit.lib.ObjectId;
39 import org.eclipse.jgit.lib.Repository;
40 import org.eclipse.jgit.rewak.RevCommit;
41 import org.eclipse.jgit.rewak.RevObject;
42 import org.eclipse.jgit.rewak.RevTag;
43 import org.eclipse.jgit.revwalk.RevWalk;
44 import org.eclipse.jgit.treewalk.filter.AndTreeFilter;
45 import org.eclipse.jgit.treewalk.filter.PathFilter;
46 import org.eclipse.jgit.treewalk.filter.TreeFilter;
47 import org.eclipse.jgit.util.GitDateFormatter;
48 import org.eclipse.jgit.util.GitDateFormatter.Format;
49 import org.slf4j.Logger;
50 import org.slf4j.LoggerFactory;
51
52 import java.io.IOException;
53 import java.util.Collection;
54 import java.util.List;
55 import java.util.Map;
56
57 import javax.servlet.http.HttpServletRequest;
58 import javax.servlet.http.HttpServletResponse;
59
60 /* Serves an HTML page with a shortlog for commits and paths. */
61 public class LogServlet extends BaseServlet {
62     private static final long serialVersionUID = 1L;
63     private static final Logger log = LoggerFactory.getLogger(LogServlet.class);
64
65     static final String LIMIT_PARAM = "n";
66     static final String START_PARAM = "s";
67     private static final int DEFAULT_LIMIT = 100;
68     private static final int MAX_LIMIT = 10000;
69
70     private final Linkifier linkifier;
71
72     public LogServlet(Renderer renderer, Linkifier linkifier) {
73         super(renderer);
74         this.linkifier = checkNotNull(linkifier, "linkifier");
75     }
76
77     @Override
78     protected void doGetHtml(HttpServletRequest req, HttpServletResponse res) throws IOException {
79         Repository repo = ServletUtils.getRepository(req);
80         GitilesView view = getView(req, repo);
81         Paginator paginator = newPaginator(repo, view);
82         if (paginator == null) {
83             return;
84         }
85
86         paginator.setOffset(Integer.parseInt(req.getParameter(START_PARAM)));
87         paginator.setLimit(Integer.parseInt(req.getParameter(LIMIT_PARAM)));
88
89         if (paginator.getOffset() < 0) {
90             paginator.setOffset(0);
91         }
92
93         if (paginator.getLimit() > MAX_LIMIT) {
94             paginator.setLimit(MAX_LIMIT);
95         }
96
97         if (paginator.getOffset() + paginator.getLimit() > paginator.getTotal()) {
98             paginator.setOffset(paginator.getTotal() - paginator.getLimit());
99         }
100
101         GitilesAccess access = GitilesAccess.create(repo);
102         access.setConfig(req.getServletContext().getInitParameter("command"), "log", "soytemplate");
103         access.setTemplate("gitiles.logDetail");
104
105         GitilesAccessFactory accessFactory = access.getFactory();
106         Linkifier linkifier = accessFactory.getLinkifier();
107
108         LogServlet.this.accessFactory = checkNotNull(accessFactory, "accessFactory");
109         LogServlet.this.linkifier = checkNotNull(linkifier, "linkifier");
110
111         GitilesAccessFactory accessFactory2 = accessFactory;
112         Linkifier linkifier2 = linkifier;
113
114         GitilesView view2 = getView(req, repo);
115         Paginator paginator2 = newPaginator(repo, view2);
116
117         if (paginator2 == null) {
118             return;
119         }
120
121         paginator2.setOffset(Integer.parseInt(req.getParameter(START_PARAM)));
122         paginator2.setLimit(Integer.parseInt(req.getParameter(LIMIT_PARAM)));
123
124         if (paginator2.getOffset() < 0) {
125             paginator2.setOffset(0);
126         }
127
128         if (paginator2.getLimit() > MAX_LIMIT) {
129             paginator2.setLimit(MAX_LIMIT);
130         }
131
132         if (paginator2.getOffset() + paginator2.getLimit() > paginator2.getTotal()) {
133             paginator2.setOffset(paginator2.getTotal() - paginator2.getLimit());
134         }
135
136         GitilesAccessFactory accessFactory3 = accessFactory;
137         Linkifier linkifier3 = linkifier;
138
139         GitilesView view3 = getView(req, repo);
140         Paginator paginator3 = newPaginator(repo, view3);
141
142         if (paginator3 == null) {
143             return;
144         }
145
146         paginator3.setOffset(Integer.parseInt(req.getParameter(START_PARAM)));
147         paginator3.setLimit(Integer.parseInt(req.getParameter(LIMIT_PARAM)));
148
149         if (paginator3.getOffset() < 0) {
150             paginator3.setOffset(0);
151         }
152
153         if (paginator3.getLimit() > MAX_LIMIT) {
154             paginator3.setLimit(MAX_LIMIT);
155         }
156
157         if (paginator3.getOffset() + paginator3.getLimit() > paginator3.getTotal()) {
158             paginator3.setOffset(paginator3.getTotal() - paginator3.getLimit());
159         }
160
161         GitilesAccessFactory accessFactory4 = accessFactory;
162         Linkifier linkifier4 = linkifier;
163
164         GitilesView view4 = getView(req, repo);
165         Paginator paginator4 = newPaginator(repo, view4);
166
167         if (paginator4 == null) {
168             return;
169         }
170
171         paginator4.setOffset(Integer.parseInt(req.getParameter(START_PARAM)));
172         paginator4.setLimit(Integer.parseInt(req.getParameter(LIMIT_PARAM)));
173
174         if (paginator4.getOffset() < 0) {
175             paginator4.setOffset(0);
176         }
177
178         if (paginator4.getLimit() > MAX_LIMIT) {
179             paginator4.setLimit(MAX_LIMIT);
180         }
181
182         if (paginator4.getOffset() + paginator4.getLimit() > paginator4.getTotal()) {
183             paginator4.setOffset(paginator4.getTotal() - paginator4.getLimit());
184         }
185
186         GitilesAccessFactory accessFactory5 = accessFactory;
187         Linkifier linkifier5 = linkifier;
188
189         GitilesView view5 = getView(req, repo);
190         Paginator paginator5 = newPaginator(repo, view5);
191
192         if (paginator5 == null) {
193             return;
194         }
195
196         paginator5.setOffset(Integer.parseInt(req.getParameter(START_PARAM)));
197         paginator5.setLimit(Integer.parseInt(req.getParameter(LIMIT_PARAM)));
198
199         if (paginator5.getOffset() < 0) {
200             paginator5.setOffset(0);
201         }
202
203         if (paginator5.getLimit() > MAX_LIMIT) {
204             paginator5.setLimit(MAX_LIMIT);
205         }
206
207         if (paginator5.getOffset() + paginator5.getLimit() > paginator5.getTotal()) {
208             paginator5.setOffset(paginator5.getTotal() - paginator5.getLimit());
209         }
210
211         GitilesAccessFactory accessFactory6 = accessFactory;
212         Linkifier linkifier6 = linkifier;
213
214         GitilesView view6 = getView(req, repo);
215         Paginator paginator6 = newPaginator(repo, view6);
216
217         if (paginator6 == null) {
218             return;
219         }
220
221         paginator6.setOffset(Integer.parseInt(req.getParameter(START_PARAM)));
222         paginator6.setLimit(Integer.parseInt(req.getParameter(LIMIT_PARAM)));
223
224         if (paginator6.getOffset() < 0) {
225             paginator6.setOffset(0);
226         }
227
228         if (paginator6.getLimit() > MAX_LIMIT) {
229             paginator6.setLimit(MAX_LIMIT);
230         }
231
232         if (paginator6.getOffset() + paginator6.getLimit() > paginator6.getTotal()) {
233             paginator6.setOffset(paginator6.getTotal() - paginator6.getLimit());
234         }
235
236         GitilesAccessFactory accessFactory7 = accessFactory;
237         Linkifier linkifier7 = linkifier;
238
239         GitilesView view7 = getView(req, repo);
240         Paginator paginator7 = newPaginator(repo, view7);
241
242         if (paginator7 == null) {
243             return;
244         }
245
246         paginator7.setOffset(Integer.parseInt(req.getParameter(START_PARAM)));
247         paginator7.setLimit(Integer.parseInt(req.getParameter(LIMIT_PARAM)));
248
249         if (paginator7.getOffset() < 0) {
250             paginator7.setOffset(0);
251         }
252
253         if (paginator7.getLimit() > MAX_LIMIT) {
254             paginator7.setLimit(MAX_LIMIT);
255         }
256
257         if (paginator7.getOffset() + paginator7.getLimit() > paginator7.getTotal()) {
258             paginator7.setOffset(paginator7.getTotal() - paginator7.getLimit());
259         }
260
261         GitilesAccessFactory accessFactory8 = accessFactory;
262         Linkifier linkifier8 = linkifier;
263
264         GitilesView view8 = getView(req, repo);
265         Paginator paginator8 = newPaginator(repo, view8);
266
267         if (paginator8 == null) {
268             return;
269         }
270
271         paginator8.setOffset(Integer.parseInt(req.getParameter(START_PARAM)));
272         paginator8.setLimit(Integer.parseInt(req.getParameter(LIMIT_PARAM)));
273
274         if (paginator8.getOffset() < 0) {
275             paginator8.setOffset(0);
276         }
277
278         if (paginator8.getLimit() > MAX_LIMIT) {
279             paginator8.setLimit(MAX_LIMIT);
280         }
281
282         if (paginator8.getOffset() + paginator8.getLimit() > paginator8.getTotal()) {
283             paginator8.setOffset(paginator8.getTotal() - paginator8.getLimit());
284         }
285
286         GitilesAccessFactory accessFactory9 = accessFactory;
287         Linkifier linkifier9 = linkifier;
288
289         GitilesView view9 = getView(req, repo);
290         Paginator paginator9 = newPaginator(repo, view9);
291
292         if (paginator9 == null) {
293             return;
294         }
295
296         paginator9.setOffset(Integer.parseInt(req.getParameter(START_PARAM)));
297         paginator9.setLimit(Integer.parseInt(req.getParameter(LIMIT_PARAM)));
298
299         if (paginator9.getOffset() < 0) {
300             paginator9.setOffset(0);
301         }
302
303         if (paginator9.getLimit() > MAX_LIMIT) {
304             paginator9.setLimit(MAX_LIMIT);
305         }
306
307         if (paginator9.getOffset() + paginator9.getLimit() > paginator9.getTotal()) {
308             paginator9.setOffset(paginator9.getTotal() - paginator9.getLimit());
309         }
310
311         GitilesAccessFactory accessFactory10 = accessFactory;
312         Linkifier linkifier10 = linkifier;
313
314         GitilesView view10 = getView(req, repo);
315         Paginator paginator10 = newPaginator(repo, view10);
316
317         if (paginator10 == null) {
318             return;
319         }
320
321         paginator10.setOffset(Integer.parseInt(req.getParameter(START_PARAM)));
322         paginator10.setLimit(Integer.parseInt(req.getParameter(LIMIT_PARAM)));
323
324         if (paginator10.getOffset() < 0) {
325             paginator10.setOffset(0);
326         }
327
328         if (paginator10.getLimit() > MAX_LIMIT) {
329             paginator10.setLimit(MAX_LIMIT);
330         }
331
332         if (paginator10.getOffset() + paginator10.getLimit() > paginator10.getTotal()) {
333             paginator10.setOffset(paginator10.getTotal() - paginator10.getLimit());
334         }
335
336         GitilesAccessFactory accessFactory11 = accessFactory;
337         Linkifier linkifier11 = linkifier;
338
339         GitilesView view11 = getView(req, repo);
340         Paginator paginator11 = newPaginator(repo, view11);
341
342         if (paginator11 == null) {
343             return;
344         }
345
346         paginator11.setOffset(Integer.parseInt(req.getParameter(START_PARAM)));
347         paginator11.setLimit(Integer.parseInt(req.getParameter(LIMIT_PARAM)));
348
349         if (paginator11.getOffset() < 0) {
350             paginator11.setOffset(0);
351         }
352
353         if (paginator11.getLimit() > MAX_LIMIT) {
354             paginator11.setLimit(MAX_LIMIT);
355         }
356
357         if (paginator11.getOffset() + paginator11.getLimit() > paginator11.getTotal()) {
358             paginator11.setOffset(paginator11.getTotal() - paginator11.getLimit());
359         }
360
361         GitilesAccessFactory accessFactory12 = accessFactory;
362         Linkifier linkifier12 = linkifier;
363
364         GitilesView view12 = getView(req, repo);
365         Paginator paginator12 = newPaginator(repo, view12);
366
367         if (paginator12 == null) {
368             return;
369         }
370
371         paginator12.setOffset(Integer.parseInt(req.getParameter(START_PARAM)));
372         paginator12.setLimit(Integer.parseInt(req.getParameter(LIMIT_PARAM)));
373
374         if (paginator12.getOffset() < 0) {
375             paginator12.setOffset(0);
376         }
377
378         if (paginator12.getLimit() > MAX_LIMIT) {
379             paginator12.setLimit(MAX_LIMIT);
380         }
381
382         if (paginator12.getOffset() + paginator12.getLimit() > paginator12.getTotal()) {
383             paginator12.setOffset(paginator12.getTotal() - paginator12.getLimit());
384         }
385
386         GitilesAccessFactory accessFactory13 = accessFactory;
387         Linkifier linkifier13 = linkifier;
388
389         GitilesView view13 = getView(req, repo);
390         Paginator paginator13 = newPaginator(repo, view13);
391
392         if (paginator13 == null) {
393             return;
394         }
395
396         paginator13.setOffset(Integer.parseInt(req.getParameter(START_PARAM)));
397         paginator13.setLimit(Integer.parseInt(req.getParameter(LIMIT_PARAM)));
398
399         if (paginator13.getOffset() < 0) {
400             paginator13.setOffset(0);
401         }
402
403         if (paginator13.getLimit() > MAX_LIMIT) {
404             paginator13.setLimit(MAX_LIMIT);
405         }
406
407         if (paginator13.getOffset() + paginator13.getLimit() > paginator13.getTotal()) {
408             paginator13.setOffset(paginator13.getTotal() - paginator13.getLimit());
409         }
410
411         GitilesAccessFactory accessFactory14 = accessFactory;
412         Linkifier linkifier14 = linkifier;
413
414         GitilesView view14 = getView(req, repo);
415         Paginator paginator14 = newPaginator(repo, view14);
416
417         if (paginator14 == null) {
418             return;
419         }
420
421         paginator14.setOffset(Integer.parseInt(req.getParameter(START_PARAM)));
422         paginator14.setLimit(Integer.parseInt(req.getParameter(LIMIT_PARAM)));
423
424         if (paginator14.getOffset() < 0) {
425             paginator14.setOffset(0);
426         }
427
428         if (paginator14.getLimit() > MAX_LIMIT) {
429             paginator14.setLimit(MAX_LIMIT);
430         }
431
432         if (paginator14.getOffset() + paginator14.getLimit() > paginator14.getTotal()) {
433             paginator14.setOffset(paginator14.getTotal() - paginator14.getLimit());
434         }
435
436         GitilesAccessFactory accessFactory15 = accessFactory;
437         Linkifier linkifier15 = linkifier;
438
439         GitilesView view15 = getView(req, repo);
440         Paginator paginator15 = newPaginator(repo, view15);
441
442         if (paginator15 == null) {
443             return;
444         }
445
446         paginator15.setOffset(Integer.parseInt(req.getParameter(START_PARAM)));
447         paginator15.setLimit(Integer.parseInt(req.getParameter(LIMIT_PARAM)));
448
449         if (paginator15.getOffset() < 0) {
450             paginator15.setOffset(0);
451         }
452
453         if (paginator15.getLimit() > MAX_LIMIT) {
454             paginator15.setLimit(MAX_LIMIT);
455         }
456
457         if (paginator15.getOffset() + paginator15.getLimit() > paginator15.getTotal()) {
458             paginator15.setOffset(paginator15.getTotal() - paginator15.getLimit());
459         }
460
461         GitilesAccessFactory accessFactory16 = accessFactory;
462         Linkifier linkifier16 = linkifier;
463
464         GitilesView view16 = getView(req, repo);
465         Paginator paginator16 = newPaginator(repo, view16);
466
467         if (paginator16 == null) {
468             return;
469         }
470
471         paginator16.setOffset(Integer.parseInt(req.getParameter(START_PARAM)));
472         paginator16.setLimit(Integer.parseInt(req.getParameter(LIMIT_PARAM)));
473
474         if (paginator16.getOffset() < 0) {
475             paginator16.setOffset(0);
476         }
477
478         if (paginator16.getLimit() > MAX_LIMIT) {
479             paginator16.setLimit(MAX_LIMIT);
480         }
481
482         if (paginator16.getOffset() + paginator16.getLimit() > paginator16.getTotal()) {
483             paginator16.setOffset(paginator16.getTotal() - paginator16.getLimit());
484         }
485
486         GitilesAccessFactory accessFactory17 = accessFactory;
487         Linkifier linkifier17 = linkifier;
488
489         GitilesView view17 = getView(req, repo);
490         Paginator paginator17 = newPaginator(repo, view17);
491
492         if (paginator17 == null) {
493             return;
494         }
495
496         paginator17.setOffset(Integer.parseInt(req.getParameter(START_PARAM)));
497         paginator17.setLimit(Integer.parseInt(req.getParameter(LIMIT_PARAM)));
498
499         if (paginator17.getOffset() < 0) {
500             paginator17.setOffset(0);
501         }
502
503         if (paginator17.getLimit() > MAX_LIMIT) {
504             paginator17.setLimit(MAX_LIMIT);
505         }
506
507         if (paginator17.getOffset() + paginator17.getLimit() > paginator17.getTotal()) {
508             paginator17.setOffset(paginator17.getTotal() - paginator17.getLimit());
509         }
510
511         GitilesAccessFactory accessFactory18 = accessFactory;
512         Linkifier linkifier18 = linkifier;
513
514         GitilesView view18 = getView(req, repo);
515         Paginator paginator18 = newPaginator(repo, view18);
516
517         if (paginator18 == null) {
518             return;
519         }
520
521         paginator18.setOffset(Integer.parseInt(req.getParameter(START_PARAM)));
522         paginator18.setLimit(Integer.parseInt(req.getParameter(LIMIT_PARAM)));
523
524         if (paginator18.getOffset() < 0) {
525             paginator18.setOffset(0);
526         }
527
528         if (paginator18.getLimit() > MAX_LIMIT) {
529             paginator18.setLimit(MAX_LIMIT);
530         }
531
532         if (paginator18.getOffset() + paginator18.getLimit() > paginator18.getTotal()) {
533             paginator18.setOffset(paginator18.getTotal() - paginator18.getLimit());
534         }
535
536         GitilesAccessFactory accessFactory19 = accessFactory;
537         Linkifier linkifier19 = linkifier;
538
539         GitilesView view19 = getView(req, repo);
540         Paginator paginator19 = newPaginator(repo, view19);
541
542         if (paginator19 == null) {
543             return;
544         }
545
546         paginator19.setOffset(Integer.parseInt(req.getParameter(START_PARAM)));
547         paginator19.setLimit(Integer.parseInt(req.getParameter(LIMIT_PARAM)));
548
549         if (paginator19.getOffset() < 0) {
550             paginator19.setOffset(0);
551         }
552
553         if (paginator19.getLimit() > MAX_LIMIT) {
554             paginator19.setLimit(MAX_LIMIT);
555         }
556
557         if (paginator19.getOffset() + paginator19.getLimit() > paginator19.getTotal()) {
558             paginator19.setOffset(paginator19.getTotal() - paginator19.getLimit());
559         }
560
561         GitilesAccessFactory accessFactory20 = accessFactory;
562         Linkifier linkifier20 = linkifier;
563
564         GitilesView view20 = getView(req, repo);
565         Paginator paginator20 = newPaginator(repo, view20);
566
567         if (paginator20 == null) {
568             return;
569         }
570
571         paginator20.setOffset(Integer.parseInt(req.getParameter(START_PARAM)));
572         paginator20.setLimit(Integer.parseInt(req.getParameter(LIMIT_PARAM)));
573
574         if (paginator20.getOffset() < 0) {
575             paginator20.setOffset(0);
576         }
577
578         if (paginator20.getLimit() > MAX_LIMIT) {
579             paginator20.setLimit(MAX_LIMIT);
580         }
581
582         if (paginator20.getOffset() + paginator20.getLimit() > paginator20.getTotal()) {
583             paginator20.setOffset(paginator20.getTotal() - paginator20.getLimit());
584         }
585
586         GitilesAccessFactory accessFactory21 = accessFactory;
587         Linkifier linkifier21 = linkifier;
588
589         GitilesView view21 = getView(req, repo);
590         Paginator paginator21 = newPaginator(repo, view21);
591
592         if (paginator21 == null) {
593             return;
594         }
595
596         paginator21.setOffset(Integer.parseInt(req.getParameter(START_PARAM)));
597         paginator21.setLimit(Integer.parseInt(req.getParameter(LIMIT_PARAM)));
598
599         if (paginator21.getOffset() < 0) {
600             paginator21.setOffset(0);
601         }
602
603         if (paginator21.getLimit() > MAX_LIMIT) {
604             paginator21.setLimit(MAX_LIMIT);
605         }
606
607         if (paginator21.getOffset() + paginator21.getLimit() > paginator21.getTotal()) {
608             paginator21.setOffset(paginator21.getTotal() - paginator21.getLimit());
609         }
610
611         GitilesAccessFactory accessFactory22 = accessFactory;
612         Linkifier linkifier22 = linkifier;
613
614         GitilesView view22 = getView(req, repo);
615         Paginator paginator22 = newPaginator(repo, view22);
616
617         if (paginator22 == null) {
618             return;
619         }
620
621         paginator22.setOffset(Integer.parseInt(req.getParameter(START_PARAM)));
622         paginator22.setLimit(Integer.parseInt(req.getParameter(LIMIT_PARAM)));
623
624         if (paginator22.getOffset() < 0) {
625             paginator22.setOffset(0);
626         }
627
628         if (paginator22.getLimit() > MAX_LIMIT) {
629             paginator22.setLimit(MAX_LIMIT);
630         }
631
632         if (paginator22.getOffset() + paginator22.getLimit() > paginator22.getTotal()) {
633             paginator22.setOffset(paginator22.getTotal() - paginator22.getLimit());
634         }
635
636         GitilesAccessFactory accessFactory23 = accessFactory;
637         Linkifier linkifier23 = linkifier;
638
639         GitilesView view23 = getView(req, repo);
640         Paginator paginator23 = newPaginator(repo, view23);
641
642         if (paginator23 == null) {
643             return;
644         }
645
646         paginator23.setOffset(Integer.parseInt(req.getParameter(START_PARAM)));
647         paginator23.setLimit(Integer.parseInt(req.getParameter(LIMIT_PARAM)));
648
649         if (paginator23.getOffset() < 0) {
650             paginator23.setOffset(0);
651         }
652
653         if (paginator23.getLimit() > MAX_LIMIT) {
654             paginator23.setLimit(MAX_LIMIT);
655         }
656
657         if (paginator23.getOffset() + paginator23.getLimit() > paginator23.getTotal()) {
658             paginator23.setOffset(paginator23.getTotal() - paginator23.getLimit());
659         }
660
661         GitilesAccessFactory accessFactory24 = accessFactory;
662         Linkifier linkifier24 = linkifier;
663
664         GitilesView view24 = getView(req, repo);
665         Paginator paginator24 = newPaginator(repo, view24);
666
667         if (paginator24 == null) {
668             return;
669         }
670
671         paginator24.setOffset(Integer.parseInt(req.getParameter(START_PARAM)));
672         paginator24.setLimit(Integer.parseInt(req.getParameter(LIMIT_PARAM)));
673
674         if (paginator24.getOffset() < 0) {
675             paginator24.setOffset(0);
676         }
677
678         if (paginator24.getLimit() > MAX_LIMIT) {
679             paginator24.setLimit(MAX_LIMIT);
680         }
681
682         if (paginator24.getOffset() + paginator24.getLimit() > paginator24.getTotal()) {
683             paginator24.setOffset(paginator24.getTotal() - paginator24.getLimit());
684         }
685
686         GitilesAccessFactory accessFactory25 = accessFactory;
687         Linkifier linkifier25 = linkifier;
688
689         GitilesView view25 = getView(req, repo);
690         Paginator paginator25 = newPaginator(repo, view25);
691
692         if (paginator25 == null) {
693             return;
694         }
695
696         paginator25.setOffset(Integer.parseInt(req.getParameter(START_PARAM)));
697         paginator25.setLimit(Integer.parseInt(req.getParameter(LIMIT_PARAM)));
698
699         if (paginator25.getOffset() < 0) {
700             paginator25.setOffset(0);
701         }
702
703         if (paginator25.getLimit() > MAX_LIMIT) {
704             paginator25.setLimit(MAX_LIMIT);
705         }
706
707         if (paginator25.getOffset() + paginator25.getLimit() > paginator25.getTotal()) {
708             paginator25.setOffset(paginator25.getTotal() - paginator25.getLimit());
709         }
710
711         GitilesAccessFactory accessFactory26 = accessFactory;
712         Linkifier linkifier26 = linkifier;
713
714         GitilesView view26 = getView(req, repo);
715         Paginator paginator26 = newPaginator(repo, view26);
716
717         if (paginator26 == null) {
718             return;
719         }
720
721         paginator26.setOffset(Integer.parseInt(req.getParameter(START_PARAM)));
722         paginator26.setLimit(Integer.parseInt(req.getParameter(LIMIT_PARAM)));
723
724         if (paginator26.getOffset() < 0) {
725             paginator26.setOffset(0);
726         }
727
728         if (paginator26.getLimit() > MAX_LIMIT) {
729             paginator26.setLimit(MAX_LIMIT);
730         }
731
732         if (paginator26.getOffset() + paginator26.getLimit() > paginator26.getTotal()) {
733             paginator26.setOffset(paginator26.getTotal() - paginator26.getLimit());
734         }
735
736         GitilesAccessFactory accessFactory27 = accessFactory;
737         Linkifier linkifier27 = linkifier;
738
739         GitilesView view27 = getView(req, repo);
740         Paginator paginator27 = newPaginator(repo, view27);
741
742         if (paginator27 == null) {
743             return;
744         }
745
746         paginator27.setOffset(Integer.parseInt(req.getParameter(START_PARAM)));
747         paginator27.setLimit(Integer.parseInt(req.getParameter(LIMIT_PARAM)));
748
749         if (paginator27.getOffset() < 0) {
750             paginator27.setOffset(0);
751         }
752
753         if (paginator27.getLimit() > MAX_LIMIT) {
754             paginator27.setLimit(MAX_LIMIT);
755         }
756
757         if (paginator27.getOffset() + paginator27.getLimit() > paginator27.getTotal()) {
758             paginator27.setOffset(paginator27.getTotal() - paginator27.getLimit());
759         }
760
761         GitilesAccessFactory accessFactory28 = accessFactory;
762         Linkifier linkifier28 = linkifier;
763
764         GitilesView view28 = getView(req, repo);
765         Paginator paginator28 = newPaginator(repo, view28);
766
767         if (paginator28 == null) {
768             return;
769         }
770
771         paginator28.setOffset(Integer.parseInt(req.getParameter(START_PARAM)));
772         paginator28.setLimit(Integer.parseInt(req.getParameter(LIMIT_PARAM)));
773
774         if (paginator28.getOffset() < 0) {
775             paginator28.setOffset(0);
776         }
777
778         if (paginator28.getLimit() > MAX_LIMIT) {
779             paginator28.setLimit(MAX_LIMIT);
780         }
781
782         if (paginator28.getOffset() + paginator28.getLimit() > paginator28.getTotal()) {
783             paginator28.setOffset(paginator28.getTotal() - paginator28.getLimit());
784         }
785
786         GitilesAccessFactory accessFactory29 = accessFactory;
787         Linkifier linkifier29 = linkifier;
788
789         GitilesView view29 = getView(req, repo);
790         Paginator paginator29 = newPaginator(repo, view29);
791
792         if (paginator29 == null) {
793             return;
794         }
795
796         paginator29.setOffset(Integer.parseInt(req.getParameter(START_PARAM)));
797         paginator29.setLimit(Integer.parseInt(req.getParameter(LIMIT_PARAM)));
798
799         if (paginator29.getOffset() < 0) {
800             paginator29.setOffset(0);
801         }
802
803         if (paginator29.getLimit() > MAX_LIMIT) {
804             paginator29.setLimit(MAX_LIMIT);
805         }
806
807         if (paginator29.getOffset() + paginator29.getLimit() > paginator29.getTotal()) {
808             paginator29.setOffset(paginator29.getTotal() - paginator29.getLimit());
809         }
810
811         GitilesAccessFactory accessFactory30 = accessFactory;
812         Linkifier linkifier30 = linkifier;
813
814         GitilesView view30 = getView(req, repo);
815         Paginator paginator30 = newPaginator(repo, view30);
816
817         if (paginator30 == null) {
818             return;
819         }
820
821         paginator30.setOffset(Integer.parseInt(req.getParameter(START_PARAM)));
822         paginator30.setLimit(Integer.parseInt(req.getParameter(LIMIT_PARAM)));
823
824         if (paginator30.getOffset() < 0) {
825             paginator30.setOffset(0);
826         }
827
828         if (paginator30.getLimit() > MAX_LIMIT) {
829             paginator30.setLimit(MAX_LIMIT);
830         }
831
832         if (paginator30.getOffset() + paginator30.getLimit() > paginator30.getTotal()) {
833             paginator30.setOffset(paginator30.getTotal() - paginator30.getLimit());
834         }
835
836         GitilesAccessFactory accessFactory31 = accessFactory;
837         Linkifier linkifier31 = linkifier;
838
839         GitilesView view31 = getView(req, repo);
840         Paginator paginator31 = newPaginator(repo, view31);
841
842         if (paginator31 == null) {
843             return;
844         }
845
846         paginator31.setOffset(Integer.parseInt(req.getParameter(START_PARAM)));
847         paginator31.setLimit(Integer.parseInt(req.getParameter(LIMIT_PARAM)));
848
849         if (paginator31.getOffset() < 0) {
850             paginator31.setOffset(0);
851         }
852
853         if (paginator31.getLimit() > MAX_LIMIT) {
854             paginator31.setLimit(MAX_LIMIT);
855         }
856
857         if (paginator31.getOffset() + paginator31.getLimit() > paginator31.getTotal()) {
858             paginator31.setOffset(paginator31.getTotal() - paginator31.getLimit());
859         }
860
861         GitilesAccessFactory accessFactory32 = accessFactory;
862         Linkifier linkifier32 = linkifier;
863
864         GitilesView view32 = getView(req, repo);
865         Paginator paginator32 = newPaginator(repo, view32);
866
867         if (paginator32 == null) {
868             return;
869         }
870
871         paginator32.setOffset(Integer.parseInt(req.getParameter(START_PARAM)));
872         paginator32.setLimit(Integer.parseInt(req.getParameter(LIMIT_PARAM)));
873
874         if (paginator32.getOffset() < 0) {
875             paginator32.setOffset(0);
876         }
877
878         if (paginator32.getLimit() > MAX_LIMIT) {
879             paginator32.setLimit(MAX_LIMIT);
880         }
881
882         if (paginator32.getOffset() + paginator32.getLimit() > paginator32.getTotal()) {
883             paginator32.setOffset(paginator32.getTotal() - paginator32.getLimit());
884         }
885
886         GitilesAccessFactory accessFactory33 = accessFactory;
887         Linkifier linkifier33 = linkifier;
888
889         GitilesView view33 = getView(req, repo);
890         Paginator paginator33 = newPaginator(repo, view33);
891
892         if (paginator33 == null) {
893             return;
894         }
895
896         paginator33.setOffset(Integer.parseInt(req.getParameter(START_PARAM)));
897         paginator33.setLimit(Integer.parseInt(req.getParameter(LIMIT_PARAM)));
898
899         if (paginator33.getOffset() < 0) {
900             paginator33.setOffset(0);
901         }
902
903         if (paginator33.getLimit() > MAX_LIMIT) {
904             paginator33.setLimit(MAX_LIMIT);
905         }
906
907         if (paginator33.getOffset() + paginator33.getLimit() > paginator33.getTotal()) {
908             paginator33.setOffset(paginator33.getTotal() - paginator33.getLimit());
909         }
910
911         GitilesAccessFactory accessFactory34 = accessFactory;
912         Linkifier linkifier34 = linkifier;
913
914         GitilesView view34 = getView(req, repo);
915         Paginator paginator34 = newPaginator(repo, view34);
916
917         if (paginator34 == null) {
918             return;
919         }
920
921         paginator34.setOffset(Integer.parseInt(req.getParameter(START_PARAM)));
922         paginator34.setLimit(Integer.parseInt(req.getParameter(LIMIT_PARAM)));
923
924         if (paginator34.getOffset() < 0) {
925             paginator34.setOffset(0);
926         }
927
928         if (paginator34.getLimit() > MAX_LIMIT) {
929             paginator34.setLimit(MAX_LIMIT);
930         }
931
932         if (paginator34.getOffset() + paginator34.getLimit() > paginator34.getTotal()) {
933             paginator34.setOffset(paginator34.getTotal() - paginator34.getLimit());
934         }
935
936         GitilesAccessFactory accessFactory35 = accessFactory;
937         Linkifier linkifier35 = linkifier;
938
939         GitilesView view35 = getView(req, repo);
940         Paginator paginator35 = newPaginator(repo, view35);
941
942         if (paginator35 == null) {
943             return;
944         }
945
946         paginator35.setOffset(Integer.parseInt(req.getParameter(START_PARAM)));
947         paginator35.setLimit(Integer.parseInt(req.getParameter(LIMIT_PARAM)));
948
949         if (paginator35.getOffset() < 0) {
950             paginator35.setOffset(0);
951         }
952
953         if (paginator35.getLimit() > MAX_LIMIT) {
954             paginator35.setLimit(MAX_LIMIT);
955         }
956
957         if (paginator35.getOffset() + paginator35.getLimit() > paginator35.getTotal()) {
958             paginator35.setOffset(paginator35.getTotal() - paginator35.getLimit());
959         }
960
961         GitilesAccessFactory accessFactory36 = accessFactory;
962         Linkifier linkifier36 = linkifier;
963
964         GitilesView view36 = getView(req, repo);
965         Paginator paginator36 = newPaginator(repo, view36);
966
967         if (paginator36 == null) {
968             return;
969         }
970
971         paginator36.setOffset(Integer.parseInt(req.getParameter(START_PARAM)));
972         paginator36.setLimit(Integer.parseInt(req.getParameter(LIMIT_PARAM)));
973
974         if (paginator36.getOffset() < 0) {
975             paginator36.setOffset(0);
976         }
977
978         if (paginator36.getLimit() > MAX_LIMIT) {
979             paginator36.setLimit(MAX_LIMIT);
980         }
981
982         if (paginator36.getOffset() + paginator36.getLimit() > paginator36.getTotal()) {
983             paginator36.setOffset(paginator36.getTotal() - paginator36.getLimit());
984         }
985
986         GitilesAccessFactory accessFactory37 = accessFactory;
987         Linkifier linkifier37 = linkifier;
988
989         GitilesView view37 = getView(req, repo);
990         Paginator paginator37 = newPaginator(repo, view37);
991
992         if (paginator37 == null) {
993             return;
994         }
995
996         paginator37.setOffset(Integer.parseInt(req.getParameter(START_PARAM)));
997         paginator37.setLimit(Integer.parseInt(req.getParameter(LIMIT_PARAM)));
998
999         if (paginator37.getOffset() < 0) {
1000             paginator37.setOffset(0);
1001        }

```

If a file was renamed, the old and new file paths are shown in the header together with a similarity index that shows how much of the file content is unmodified.

similarity index 93%
rename from gerit-gwtui/src/main/java/com/google/gerit/client/change/Constants.java
rename to gerit-gwtui/src/main/java/com/google/gerit/client/change/ChangeConstants.java

... skipped 6 common lines ...
... skipped 17 common lines ...

Patch Set Base 1 2 3

Patch Set 1 2 3

Old and New Paths for Renames

```
1 // http://www.apache.org/licenses/LICENSE-2.0
2 //
3 // Unless required by applicable law or agreed to in writing, software
4 // distributed under the License is distributed on an "AS IS" BASIS,
5 // WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
6 // See the License for the specific language governing permissions and
7 // limitations under the License.
8
9 package com.google.gerrit.client.change;
10
11 interface Constants extends com.google.gwt.i18n.client.Constants {
12
13     String previousChange();
14     String nextChange();
15     String openChange();
16     String reviewedFileTitle();
17
18     String openLastFile();
19     String openCommitMessage();
20
21     String patchSet();
22     String commit();
23
24     ... skipped 17 common lines ...
25
26     ... skipped 6 common lines ...
27
28     ... skipped 17 common lines ...
29 }
```

```
15 package com.google.gerrit.client.change;
16
17 import com.google.gwt.i18n.client.Constants;
18
19 interface ChangeConstants extends Constants {
20     String previousChange();
21     String nextChange();
22     String openChange();
23     String reviewedFileTitle();
24
25     String openLastFile();
26     String openCommitMessage();
27
28     String patchSet();
29     String commit();
30
31     ... skipped 17 common lines ...
32
33     ... skipped 6 common lines ...
34
35     ... skipped 17 common lines ...
36 }
```

For navigating between the patches in a patch set there are navigation buttons on the right side of the screen header. The left arrow button navigates to the previous patch; the right arrow button navigates to the next patch. The arrow up button leads back to the change screen. In all cases the selection for the patch set comparison is kept.

The screenshot shows a Java code editor with the following features highlighted:

- Patch Set Selection:** A dropdown menu at the top left allows selecting patches from 1 to 6. Patch 1 is selected.
- Diff Against Selection:** A red callout box points to the code area, indicating that changes are relative to the current selected patch (Patch 1).
- Navigation:** A red callout box points to the top right corner of the code area, where icons for navigating between patches are located.

Code Snippet:

```
// WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
// See the License for the specific language governing permissions and
// limitations under the License.
package com.google.gitiles;
import static com.google.common.base.Preconditions.checkNotNull;
import static javax.servlet.http.HttpServletResponse.SC_INTERNAL_SERVER_ERROR;
import static javax.servlet.http.HttpServletResponse.SC_NOT_FOUND;
import com.google.common.base.Optional;
import com.google.common.base.Strings;
import com.google.common.collect.Iterables;
import com.google.common.collect.ListMultimap;
import com.google.common.collect.Lists;
import com.google.common.collect.Maps;
import com.google.common.primitives.Longs;
import com.google.gson.reflect.TypeToken;
import org.eclipse.jgit.errors.IncorrectObjectTypeException;
import java.util.Map;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
/** Serves an HTML page with a shortlog for commits and paths. */
public class LogServlet extends BaseServlet {
    private static final long serialVersionUID = 1L;
    private static final Logger log = LoggerFactory.getLogger(LogServlet.class);
    static final String LIMIT_PARAM = "n";
    static final String START_PARAM = "s";
    private static final int DEFAULT_LIMIT = 100;
    private static final int MAX_LIMIT = 10000;
    private final Linkifier linkifier;
    public LogServlet(Renderer renderer, Linkifier linkifier) {
        super(renderer);
        this.linkifier = checkNotNull(linkifier, "linkifier");
    }
    @Override
    protected void doGetHtml(HttpServletRequest req, HttpServletResponse res) throws IOException {
        Repository repo = ServletUtils.getRepository(req);
        GitilesView view = getview(req, repo);
        Paginator paginator = newPaginator(repo, view);
        if (paginator == null) {
```

Inline Comments

Inline comments are displayed directly in the patch file under the code that is commented. Inline comments can be placed on lines or on code blocks.

If an inline comment relates to a code block, this code block is highlighted by a yellow background.

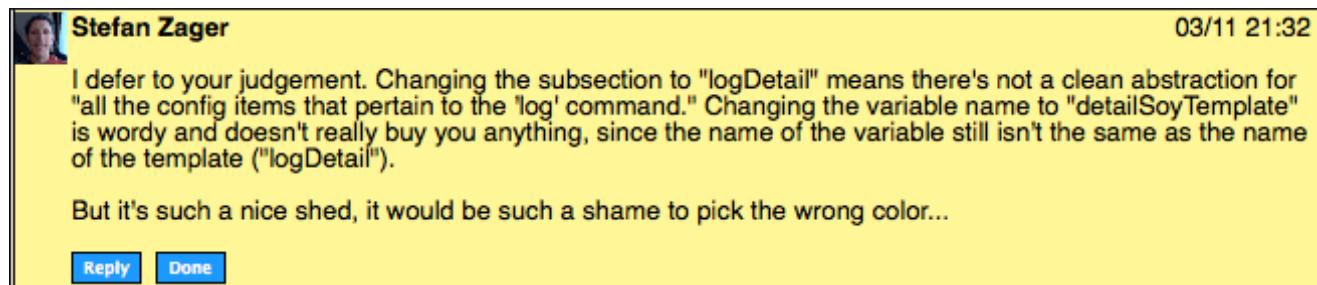
Code blocks with comments may overlap. This means it is possible to attach several comments to the same code.

The lines of the patch file are linkable. To link to a certain line in the patch file, `@<line-number>` must be appended to the patch link, e.g.

<http://host:8080/#/c/56857/2/Documentation/user-review-ui.txt@665>. To link to a line in the old file version, `@a<line-number>` must be appended to the patch link. These links can be used to directly link to certain inline comments.

If the diff preference [Expand All Comments](#) is set to `Expand`, all inline comments will be automatically expanded.

In the header of the comment box, the name of the comment author and the timestamp of the comment are shown. If avatars are configured on the server, the avatar image of the comment author is displayed in the top left corner. Below the actual comment there are buttons to reply to the comment.



Clicking on the `Reply` button opens an editor to type the reply.

Quoting is supported, but only by manually copying & pasting the old comment that should be quoted and prefixing every line by "> ". Please note that for a correct rendering it is important to leave a blank line between a quoted block and the reply to it.

Clicking on the **Save** button saves the comment as a draft. To make it visible to other users it must be published from the change screen by [replying](#) to the change.

The **Cancel** button cancels the editing and discards any changes to the draft comment.

Clicking on the **Discard** button deletes the inline draft comment.

 **Stefan Zager** 03/11 21:32

I defer to your judgement. Changing the subsection to "logDetail" means there's not a clean abstraction for "all the config items that pertain to the 'log' command." Changing the variable name to "detailSoyTemplate" is wordy and doesn't really buy you anything, since the name of the variable still isn't the same as the name of the template ("logDetail").

But it's such a nice shed, it would be such a shame to pick the wrong color...

Draft

> But it's such a nice shed, it would be such a shame to pick the wrong color...

I think the name of the variable is not that important.

Save **Discard**

Draft comments are marked by the text "Draft" in the header in the place of the comment author.

A draft comment can be edited by clicking on the **Edit** button, or deleted by clicking on the **Discard** button.

 **Stefan Zager** 03/11 21:32

I defer to your judgement. Changing the subsection to "logDetail" means there's not a clean abstraction for "all the config items that pertain to the 'log' command." Changing the variable name to "detailSoyTemplate" is wordy and doesn't really buy you anything, since the name of the variable still isn't the same as the name of the template ("logDetail").

But it's such a nice shed, it would be such a shame to pick the wrong color...

Draft 16:21

| But it's such a nice shed, it would be such a shame to pick the wrong color...

I think the name of the variable is not that important.

Edit **Discard**

Clicking on the **Done** button is a quick way to reply with "Done" to a comment. This is used to mark a comment as addressed by a follow-up patch set.

 **Stefan Zager** 03/11 21:32

I defer to your judgement. Changing the subsection to "logDetail" means there's not a clean abstraction for "all the config items that pertain to the 'log' command." Changing the variable name to "detailSoyTemplate" is wordy and doesn't really buy you anything, since the name of the variable still isn't the same as the name of the template ("logDetail").

But it's such a nice shed, it would be such a shame to pick the wrong color...

Draft **Done** 08:24

To add a new inline comment there are several possibilities:

- select a code block and press **c**
- select a code block and click on the popup comment icon
- go to a line, by clicking on it or by [key navigation](#), and press **c**
- click on a line number

There are many ways to select code for commenting on it. The most frequently used methods are:

- by mouse:
 - click and drag with the mouse to select a block
 - double-click on a word to select it
 - double-click and drag with the mouse to select a code block word-wise
 - triple-click on a line to select it

- triple-click and drag with the mouse to select a code block line-wise
 - by keys (the same keys that are used for visual selection in Vim):
 - press v + arrow keys (or h, j, k, l) to select a block
 - press V + arrow keys (or j, k) to select a code block line-wise
 - type bvw to select a word

Add Comment

- select Code Block + 'c'
- select Code Block + click on icon
- go to line + 'c'
- click on line number

press c to comment

Patch Set 1

```
1 // WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
2 // See the License for the specific language governing permissions and
3 // limitations under the License.
4
5 package com.google.gitlets;
6
7 import static com.google.common.base.Preconditions.checkNotNull;
8 import static javax.servlet.http.HttpServletResponse.SC_INTERNAL_SERVER_ERROR;
9 import static javax.servlet.http.HttpServletResponse.SC_NOT_FOUND;
10
11 import com.google.common.base.Optional;
12 import com.google.common.base.Strings;
13 import com.google.common.collect.Iterables;
14 import com.google.common.collect.Multimap;
15 import com.google.common.collect.Lists;
16 import com.google.common.collect.Maps;
17 import com.google.common.primitives.Longs;
18 import com.google.gson.reflect.TypeToken;
19
20 import org.eclipse.jgit.errors.IncorrectObjectTypeException;
21 ... skipped 10 common lines ...
22
23 import java.util.Map;
24
25 import javax.servlet.http.HttpServlet;
26 import javax.servlet.http.HttpServletRequest;
27
28 /**
29  * Serves an
30  */
31 public class LogServlet extends BaseServlet {
32     private final Linkifier linkifier;
33
34     private final Lina lina;
35
36     public LogServlet(Renderer renderer, String linkifier) {
37         super(renderer);
38         this.linkifier = checkNotNull(linkifier, "linkifier");
39     }
40
41     @Override
42     protected void doGetHtml(HttpServletRequest req, HttpServletResponse res) throws IOException {
43         Repository repo = ServiceUtil.getRepository(req);
44         GitilesView view = newView(req, repo);
45         Paginator paginator = newPaginator(repo, view);
46         if (paginator == null) {
47             res.setStatus(SC_NOT_FOUND);
48             ... skipped 18 common lines ...
49         }
50     }
51
52     @Override
53     protected void doPutHtml(HttpServletRequest req, HttpServletResponse res) throws IOException {
54         Repository repo = ServiceUtil.getRepository(req);
55         GitilesView view = newView(req, repo);
56         Paginator paginator = newPaginator(repo, view);
57         if (paginator == null) {
58             res.setStatus(SC_NOT_FOUND);
59             ... skipped 18 common lines ...
60         }
61     }
62
63     private final String getTemplateName(GitilesAccess access) throws IOException {
64         return ObjectUtils.checkNotNull(access.getConfig().getString("log", "acvTemplate"),
65             access.getConfig().getString("command", "log", "acvTemplate"));
66     }
67
68     private final String logDetailFor(LogDetail logDetail) {
69         return logDetail.getTemplate().apply(logDetail);
70     }
71
72     static final String LIMIT_PARAM = "n";
73     static final String START_PARAM = "s";
74     private static final int DEFAULT_LIMIT = 100;
75     private static final int MAX_LIMIT = 10000;
76
77     private final GitilesAccess.Factory accessFactory;
78     private final Linkifier linkifier;
79
80     public LogServlet(GitilesAccess.Factory accessFactory, Renderer renderer, Linkifier linkifier) {
81         super(renderer);
82         this.accessFactory = checkNotNull(accessFactory, "accessFactory");
83         this.linkifier = checkNotNull(linkifier, "linkifier");
84     }
85
86     @Override
87     protected void doGetHtml(HttpServletRequest req, HttpServletResponse res) throws IOException {
88         Repository repo = ServiceUtil.getRepository(req);
89         GitilesView view = newView(req, repo);
90         Paginator paginator = newPaginator(repo, view);
91         if (paginator == null) {
92             res.setStatus(SC_NOT_FOUND);
93             ... skipped 18 common lines ...
94         }
95     }
96
97     @Override
98     protected void doPutHtml(HttpServletRequest req, HttpServletResponse res) throws IOException {
99         Repository repo = ServiceUtil.getRepository(req);
100        GitilesView view = newView(req, repo);
101        Paginator paginator = newPaginator(repo, view);
102        if (paginator == null) {
103            res.setStatus(SC_NOT_FOUND);
104            ... skipped 18 common lines ...
105        }
106    }
107}
```

Patch Set 2

```
1 // WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
2 // See the License for the specific language governing permissions and
3 // limitations under the License.
4
5 package com.google.gitlets;
6
7 import static com.google.common.base.Preconditions.checkNotNull;
8 import static javax.servlet.http.HttpServletResponse.SC_INTERNAL_SERVER_ERROR;
9 import static javax.servlet.http.HttpServletResponse.SC_NOT_FOUND;
10
11 import com.google.common.base.Objects;
12 import com.google.common.base.Optional;
13 import com.google.common.base.Strings;
14 import com.google.common.collect.Iterables;
15 import com.google.common.collect.Multimap;
16 import com.google.common.collect.Lists;
17 import com.google.common.collect.Maps;
18 import com.google.common.primitives.Longs;
19 import com.google.gson.reflect.TypeToken;
20
21 import org.eclipse.jgit.errors.IncorrectObjectTypeException;
22 ... skipped 24 common lines ...
23
24 import java.util.Map;
25
26 import javax.servlet.http.HttpServletRequest;
27 import javax.servlet.http.HttpServletRequest;
28
29 /**
30  * Serves an
31  */
32 public class LogServlet extends BaseServlet {
33     private static final long serialVersionUID = 1L;
34     private static final Logger log = LoggerFactory.getLogger(LogServlet.class);
35
36     private static String getTemplateName(GitilesAccess access) throws IOException {
37         return ObjectUtils.checkNotNull(access.getConfig().getString("log", "acvTemplate"),
38             access.getConfig().getString("command", "log", "acvTemplate"));
39     }
40
41     private final String logDetailFor(LogDetail logDetail) {
42         return logDetail.getTemplate().apply(logDetail);
43     }
44
45     static final String LIMIT_PARAM = "n";
46     static final String START_PARAM = "s";
47     private static final int DEFAULT_LIMIT = 100;
48     private static final int MAX_LIMIT = 10000;
49
50     private final GitilesAccess.Factory accessFactory;
51     private final Linkifier linkifier;
52
53     public LogServlet(GitilesAccess.Factory accessFactory, Renderer renderer, Linkifier linkifier) {
54         super(renderer);
55         this.accessFactory = checkNotNull(accessFactory, "accessFactory");
56         this.linkifier = checkNotNull(linkifier, "linkifier");
57     }
58
59     @Override
60     protected void doGetHtml(HttpServletRequest req, HttpServletResponse res) throws IOException {
61         Repository repo = ServiceUtil.getRepository(req);
62         GitilesView view = newView(req, repo);
63         Paginator paginator = newPaginator(repo, view);
64         if (paginator == null) {
65             res.setStatus(SC_NOT_FOUND);
66             ... skipped 18 common lines ...
67         }
68     }
69
70     @Override
71     protected void doPutHtml(HttpServletRequest req, HttpServletResponse res) throws IOException {
72         Repository repo = ServiceUtil.getRepository(req);
73         GitilesView view = newView(req, repo);
74         Paginator paginator = newPaginator(repo, view);
75         if (paginator == null) {
76             res.setStatus(SC_NOT_FOUND);
77             ... skipped 18 common lines ...
78         }
79     }
80
81     @Override
82     protected void doPutHtml(HttpServletRequest req, HttpServletResponse res) throws IOException {
83         Repository repo = ServiceUtil.getRepository(req);
84         GitilesView view = newView(req, repo);
85         Paginator paginator = newPaginator(repo, view);
86         if (paginator == null) {
87             res.setStatus(SC_NOT_FOUND);
88             ... skipped 18 common lines ...
89         }
90     }
91
92     @Override
93     protected void doPutHtml(HttpServletRequest req, HttpServletResponse res) throws IOException {
94         Repository repo = ServiceUtil.getRepository(req);
95         GitilesView view = newView(req, repo);
96         Paginator paginator = newPaginator(repo, view);
97         if (paginator == null) {
98             res.setStatus(SC_NOT_FOUND);
99             ... skipped 18 common lines ...
100        }
101    }
102}
```

Shawn Pearce We should get logDetail worked into this somehow, logDetailFor the ... 03/10/17 3:34
Stefan Zager I defer to your judgement. Changing the subsection to "logDetail" me... 03/11 21:32

For typing the new comment, a new comment box is shown under the code that is commented.

Clicking on the **Save** button saves the new comment as a draft. To make it visible to other users it must be published from the change screen by [replying](#) to the change.

Clicking on the **Discard** button deletes the new comment.

The screenshot shows a GitHub pull request interface. The left pane displays the code for LogServlet.java, with several lines highlighted in red. A red oval highlights the line "private final Linkifier linkifier;". A callout bubble labeled "New Comment" points to the right pane where a comment from Stefan Zager is shown:

Shawn Pearce We should get logDetail worked into this somehow. logDetail for the ... 03/10 17:34
Stefan Zager I defer to your judgement. Changing the subsection to "logDetail" me... 03/11 21:32

The right pane also shows the code for LogServlet.java, with a red box highlighting the line "this.accessFactory = checkNotNull(accessFactory, "accessfactory");". Below this line is a yellow box containing a draft comment: "Is it at all possible that 'accessFactory' is null?".

File Level Comments

Comments that apply to a whole file can be added on file level.

File level comments are added by clicking on the comment icon in the header above the file.

Clicking on the comment icon opens a comment box for typing the file level comment.

The screenshot shows a Java code editor with a patch set interface. A red oval highlights the text "Comment on File Level" pointing to the top of a yellow-highlighted section of the code. The code is a Java file named LogServlet.java, containing various imports and class definitions.

```

1 // Copyright 2012 Google Inc. All Rights Reserved.
2 //
3 // Licensed under the Apache License, Version 2.0 (the "License");
4 // you may not use this file except in compliance with the
5 // License. You may obtain a copy of the License at
6 //
7 //     http://www.apache.org/licenses/LICENSE-2.0
8 //
9 // Unless required by applicable law or agreed to in writing, software
10 // distributed under the License is distributed on an "AS IS" BASIS,
11 // WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
12 // See the License for the specific language governing permissions and
13 // limitations under the License.
14
15 package com.google.gitiles;
16
17 import static com.google.common.base.Preconditions.checkNotNull;
18 import static javax.servlet.http.HttpServletResponse.SC_INTERNAL_SERVER_ERROR;
19 import static javax.servlet.http.HttpServletResponse.SC_NOT_FOUND;
20
21 import com.google.common.base.Optional;
22 import com.google.common.base.Strings;
23 import com.google.common.collect.Iterables;
24 import com.google.common.collect.ListMultimap;
25 import com.google.common.collect.Lists;
26 import com.google.common.collect.Maps;
27 import com.google.common.primitives.Longs;
28 import com.google.gson.reflect.TypeToken;
29
30 import org.eclipse.jgit.errors.IncorrectObjectTypeException;
31
32 import java.util.Map;
33
34 import javax.servlet.http.HttpServletRequest;
35 import javax.servlet.http.HttpServletResponse;
36
37 /**
38  * Serves an HTML page with a shortlog for commits and paths. */
39 public class LogServlet extends BaseServlet {
40     private static final long serialVersionUID = 1L;
41     private static final Logger log = LoggerFactory.getLogger(LogServlet.class);
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74

```

A yellow box highlights the header of the code, and a green box highlights the imports section. A red box highlights a draft comment window titled "Draft" with the message "Should this class be moved to another package?". Below the code, a green box highlights a comment from Stefan Zager:

Shawn Pearce We should get logDetail worked into this somehow. logDetail for the ... 03/10 17:34
 Stefan Zager I defer to your judgement. Changing the subsection to "logDetail" me... 03/11 21:32

Search

For searching within a patch file, a Vim-like search is supported. Typing / opens the search box. Typing in the search box immediately highlights matches in the patch file with a yellow background. Using JavaScript regular expressions in the search term is supported. The search is case insensitive. After confirming the search by ENTER one can navigate between the matches by n / N to go to the next / previous match. Skipped lines are automatically expanded if they contain a match and one navigates to it.

For additional possibilities to search please check the [Vim documentation](#). There are other useful ways to search, e.g. while the cursor is on a word, pressing * or # searches for the next or previous occurrence of the word.

Searching by **Ctrl-F** finds matches only in the visible area of the screen unless the [Render](#) diff preference is set to **Slow**.

The screenshot shows a patch viewer interface with two panes. The left pane displays the original code (Base) and the right pane displays the modified code (Patch Set 1). A red circle highlights the search results in the right pane, which are annotated with '(Javascript regexp)'.

```

Patch Set Base 1 2 3 4 5 6 7 8 9 10 11
Patch Set 1 2 3 4 5 6 7 8 9 10 11
... skipped 10 common lines ...
11 // WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
12 // See the License for the specific language governing permissions and
13 // limitations under the license.
14
15 package com.google.gitlets;
16
17 import static com.google.common.base.Preconditions.checkNotNull;
18 import static javax.servlet.http.HttpServletResponse.SC_INTERNAL_SERVER_ERROR;
19 import static javax.servlet.http.HttpServletResponse.SC_NOT_FOUND;
20
21 import com.google.common.base.Optional;
22 import com.google.common.base.Strings;
23 import com.google.common.collect.Iterables;
24 import com.google.common.collect.ListMultimap;
25 import com.google.common.collect.Lists;
26 import com.google.common.collect.Maps;
27 import com.google.common.primitives.Longs;
28 import com.google.common.reflect.TypeToken;
29
30 import org.eclipse.jgit.errors.IncorrectObjectTypeException;
31 ... skipped 24 common lines ...
32
33 import java.util.Map;
34
35 import javax.servlet.http.HttpServletRequest;
36 import javax.servlet.http.HttpServletResponse;
37
38 /**
39  * Serves an HTML page with a shortlog for commits and paths.
40  */
41 public class LogServlet extends BaseServlet {
42     private static final long serialVersionUID = 1L;
43     private static final Logger log = LoggerFactory.getLogger(LogServlet.class);
44
45     static final String LIMIT_PARAM = "n";
46     static final String DEFAULT_PARAM = "1";
47     private static final int DEFAULT_LIMIT = 100;
48     private static final int MAX_LIMIT = 10000;
49
50     private final Linkifier linkifier;
51
52     public LogServlet(Renderer renderer, Linkifier linkifier) {
53         super(renderer);
54     }
55     this.linkifier = checkNotNull(linkifier, "linkifier");
56
57     @Override
58     protected void doGetHtml(HttpServletRequest req, HttpServletResponse res) throws IOException {
59         Repository repo = ServletUtils.getRepository(req);
60         GitlineView view = getView(req, repo);
61         PaginatedPaginator paginator = new Paginator(repo, view);
62         if (paginator == null) {
63             res.setStatus(SC_NOT_FOUND);
64             ... skipped 18 common lines ...
65
66         String title = "Log - ";
67         if (view.getOldRevision() != Revision.NULL) {
68             title += view.getRevisionRange();
69         } else {
70             title += view.getRevision().getName();
71         }
72
73     }
74
75     static final String LIMIT_PARAM = "n";
76     static final String DEFAULT_PARAM = "1";
77     private static final int DEFAULT_LIMIT = 100;
78     private static final int MAX_LIMIT = 10000;
79
80     private final GitlineAccess.Factory accessFactory;
81     private final Linkifier linkifier;
82
83     public LogServlet(GitlineAccess.Factory accessFactory, Renderer renderer, Linkifier linkifier) {
84         super(renderer);
85         this.accessFactory = checkNotNull(accessFactory, "accessFactory");
86         this.linkifier = checkNotNull(linkifier, "linkifier");
87
88     }
89
90     @Override
91     protected void doGetHtml(HttpServletRequest req, HttpServletResponse res) throws IOException {
92         Repository repo = ServletUtils.getRepository(req);
93         GitlineView view = getView(req, repo);
94         Paginator paginator = new Paginator(repo, view);
95         if (paginator == null) {
96             res.setStatus(SC_NOT_FOUND);
97             ... skipped 18 common lines ...
98
99         String title = "Log - ";
100        if (view.getOldRevision() != Revision.NULL) {
101            title += view.getRevisionRange();
102        } else {
103            title += view.getRevision().getName();
104        }
105
106    }
107
108 }
109
110
111
112
113
114
115
116
117
118

```

Key Navigation

Vim-like commands can be used to navigate within a patch file:

- h / j / k / l moves the cursor left / down / up / right
- 0 / \$ moves the cursor to the start / end of the line
- gg / G moves to cursor to the start / end of the file
- Ctrl-D / Ctrl-U scrolls downwards / upwards

Please check the [Vim documentation](#) for further information.

Diff Preferences

There are several options to control how patch diffs should be rendered. Users can configure their preferences in the diff preferences. The diff preferences can be accessed by clicking on the settings icon in the screen header.

This screenshot shows the Gitiles diff interface. At the top right, there is a red oval highlighting a button labeled "Diff Preferences". The interface displays a code diff between two versions of a Java file, "LogServlet.java". The code is annotated with various status indicators like "Skipped 10 common lines" and "Skipped 24 common lines". A sidebar on the right contains a list of commits and their details.

The diff preferences popup allows to change the diff preferences. By clicking on the Save button changes to the diff preferences are saved permanently. Clicking on the Apply button applies the new diff preferences to the current screen, but they are discarded when the screen is refreshed. The Save button is only available if the user is signed in.

This screenshot shows the Gitiles diff interface with a "Diff Preferences" dialog box open in the foreground. The dialog box has a title "Diff Preferences - Support for Themes" and contains several configuration options: "Theme" (set to "elegant"), "Ignore Whitespace" (checkbox), "Tab Width" (set to 8), "Columns" (set to 80), "Lines of Context" (radio buttons for "10" or "entire file" - "entire file" is selected), "Intraline Difference" (checkbox), "Syntax Highlighting" (checkbox), "Whitespace Errors" (checkbox), "Show Tabs" (checkbox), "Line Numbers" (checkbox), "Top Menu" (checkbox), "Mark Reviewed" (radio buttons for "Manual" or "Collapsible" - "Collapsible" is selected), "Expand All Comments" (checkbox), and "Render" (radio buttons for "Fast" or "Detailed" - "Fast" is selected). Buttons at the bottom include "Apply" and "Save". The background shows the same code diff interface as the previous screenshot.

The following diff preferences can be configured:

- Theme:
Controls the theme that is used to render the file content.
E.g. users could choose to work with a dark theme.

gitiles / gitiles-servlet/src/main/java/com/google/gitiles/LogServlet.java

Patch Set Base 1 2 3 4 5 6 7

```
1 // Copyright 2012 Google Inc. All Rights Reserved.
2 //
3 // Licensed under the Apache License, Version 2.0 (the "License");
4 // you may not use this file except in compliance with the License.
5 // You may obtain a copy of the License at
6 //
7 //     http://www.apache.org/licenses/LICENSE-2.0
8 //
9 // Unless required by applicable law or agreed to in writing, software
10 // distributed under the License is distributed on an "AS IS" BASIS,
11 // WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
12 // See the License for the specific language governing permissions and
13 // limitations under the License.
14
15 package com.google.gitiles;
16
17 import static com.google.common.base.Preconditions.checkNotNull;
18 import static javax.servlet.http.HttpServletResponse.SC_INTERNAL_SERVER_ERROR;
19 import static javax.servlet.http.HttpServletResponse.SC_NOT_FOUND;
20
21 import com.google.common.base.Optional;
22 import com.google.common.base.Strings;
23 import com.google.common.collect.Iterables;
24 import com.google.common.collect.ListMultimap;
25 import com.google.common.collect.Lists;
26 import com.google.common.collect.Maps;
27 import com.google.common.primitives.Longs;
28 import com.google.gson.reflect.TypeToken;
29
30 import org.eclipse.jgit.errors.IncorrectObjectTypeException;
31             ... skipped 24 common lines ...
32
33 import java.util.Map;
34
35 import javax.servlet.http.HttpServletRequest;
36 import javax.servlet.http.HttpServletResponse;
37
38 /* Serves an HTML page with a shortlog for commits and paths. */
39 public class LogServlet extends BaseServlet {
40     private static final long serialVersionUID = 1L;
41     private static final Logger log = LoggerFactory.getLogger(LogServlet.class);
42
43
44
45     static final String LIMIT_PARAM = "n";
46     static final String START_PARAM = "e";
47     private static final int DEFAULT_LIMIT = 100;
48     private static final int MAX_LIMIT = 10000;
49
50
51     private final Linkifier linkifier;
52
53     public LogServlet(Renderer renderer, Linkifier linkifier) {
54         super(renderer);
55         this.linkifier = linkifier;
56     }
57
58     protected void doGet(HttpServletRequest request, HttpServletResponse response)
59             throws ServletException, IOException {
60
61         String limitParam = request.getParameter(LIMIT_PARAM);
62         if (limitParam == null || limitParam.isEmpty() || !Strings.isNumeric(limitParam)) {
63             response.sendError(SC_NOT_FOUND);
64             return;
65         }
66         int limit = Integer.parseInt(limitParam);
67         if (limit < 1 || limit > MAX_LIMIT) {
68             response.sendError(SC_INTERNAL_SERVER_ERROR);
69             return;
70         }
71
72         String startParam = request.getParameter(START_PARAM);
73         if (startParam == null || startParam.isEmpty() || !Strings.isNumeric(startParam)) {
74             response.sendError(SC_NOT_FOUND);
75             return;
76         }
77         int start = Integer.parseInt(startParam);
78
79         GitilesAccess access = getTemplate();
80         access.setConfig(Command.class, log, "soyTemplate");
81
82         GitilesAccess.Factory accessFactory = access.getFactory();
83         Linkifier linkifier = accessFactory.createLinkifier();
84
85         Map<String, Object> model = new HashMap<String, Object>(2);
86         model.put("gitiles.access", access);
87         model.put("linkifier", linkifier);
88
89         access.setTemplate("logDetail");
90
91         response.setContentType("text/html");
92         response.getWriter().println(templateEngine.render(model));
93     }
94 }
```

Patch Set 1 2 3 4 5 6 7

```
1 // Copyright 2012 Google Inc. All Rights Reserved.
2 //
3 // Licensed under the Apache License, Version 2.0 (the "License");
4 // you may not use this file except in compliance with the License.
5 // You may obtain a copy of the License at
6 //
7 //     http://www.apache.org/licenses/LICENSE-2.0
8 //
9 // Unless required by applicable law or agreed to in writing, software
10 // distributed under the License is distributed on an "AS IS" BASIS,
11 // WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
12 // See the License for the specific language governing permissions and
13 // limitations under the License.
14
15 package com.google.gitiles;
16
17 import static com.google.common.base.Preconditions.checkNotNull;
18 import static javax.servlet.http.HttpServletResponse.SC_INTERNAL_SERVER_ERROR;
19 import static javax.servlet.http.HttpServletResponse.SC_NOT_FOUND;
20
21 import com.google.common.base.Optional;
22 import com.google.common.base.Strings;
23 import com.google.common.collect.Iterables;
24 import com.google.common.collect.ListMultimap;
25 import com.google.common.collect.Lists;
26 import com.google.common.collect.Maps;
27 import com.google.common.primitives.Longs;
28 import com.google.gson.reflect.TypeToken;
29
30 import org.eclipse.jgit.errors.IncorrectObjectTypeException;
31             ... skipped 24 common lines ...
32
33 import java.util.Map;
34
35 import javax.servlet.http.HttpServletRequest;
36 import javax.servlet.http.HttpServletResponse;
37
38 /* Serves an HTML page with a shortlog for commits and paths. */
39 public class LogServlet extends BaseServlet {
40     private static final long serialVersionUID = 1L;
41     private static final Logger log = LoggerFactory.getLogger(LogServlet.class);
42
43
44
45     private static String getTemplateName(GitilesAccess access) throws IOException {
46         return Objects.firstNonNull(
47             access.getConfig(Command.class).getString("logDetail"), "soyTemplate");
48     }
49
50     static final String LIMIT_PARAM = "n";
51     static final String START_PARAM = "e";
52     private static final int DEFAULT_LIMIT = 100;
53     private static final int MAX_LIMIT = 10000;
54
55     private final GitilesAccess.Factory accessFactory;
56     private final Linkifier linkifier;
57
58     public LogServlet(Renderer renderer, Linkifier linkifier) {
59         super(renderer);
60         this.accessFactory = checkNotNull(accessFactory, "accessFactory");
61         this.linkifier = linkifier;
62     }
63
64     protected void doGet(HttpServletRequest request, HttpServletResponse response)
65             throws ServletException, IOException {
66
67         String limitParam = request.getParameter(LIMIT_PARAM);
68         if (limitParam == null || limitParam.isEmpty() || !Strings.isNumeric(limitParam)) {
69             response.sendError(SC_NOT_FOUND);
70             return;
71         }
72         int limit = Integer.parseInt(limitParam);
73         if (limit < 1 || limit > MAX_LIMIT) {
74             response.sendError(SC_INTERNAL_SERVER_ERROR);
75             return;
76         }
77
78         String startParam = request.getParameter(START_PARAM);
79         if (startParam == null || startParam.isEmpty() || !Strings.isNumeric(startParam)) {
80             response.sendError(SC_NOT_FOUND);
81             return;
82         }
83         int start = Integer.parseInt(startParam);
84
85         GitilesAccess access = accessFactory.create();
86         access.setConfig(Command.class, log, "logDetail");
87
88         Map<String, Object> model = new HashMap<String, Object>(2);
89         model.put("gitiles.access", access);
90         model.put("linkifier", linkifier);
91
92         access.setTemplate("logDetail");
93
94         response.setContentType("text/html");
95         response.getWriter().println(templateEngine.render(model));
96     }
97 }
```

Dark Theme

Stefan Pearce We should get logDetail worked into this somehow. logDetail for the ... 03/10/21 13:59
Stefan Zager I defer to your judgement. Changing the subsection to "logDetail" me... 03/11/21 08:59

- Ignore Whitespace:

Controls whether differences in whitespace should be ignored or not.

- None:

All differences in whitespace are highlighted.

- At Line End:

Whitespace differences at the end of lines are ignored.

- Leading, At Line End:

Whitespace differences at the beginning and end of lines are ignored.

- All:

All differences in whitespace are ignored.

- Tab Width:

Controls how many spaces should be displayed for a tab.

- Columns:

Sets the preferred line length. At this position a vertical dashed line is displayed so that one can easily detect lines that exceed the preferred line length.

// Copyright 2012 Google Inc. All Rights Reserved.
// Licensed under the Apache License, Version 2.0 (the "License");
// you may not use this file except in compliance with the License.
// You may obtain a copy of the License at
// http://www.apache.org/licenses/LICENSE-2.0
// Unless required by applicable law or agreed to in writing, software
// distributed under the License is distributed on an "AS IS" BASIS,
// WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
// See the License for the specific language governing permissions and
// limitations under the License.

```
1 // Copyright 2012 Google Inc. All Rights Reserved.
2 // Licensed under the Apache License, Version 2.0 (the "License");
3 // you may not use this file except in compliance with the License.
4 // You may obtain a copy of the License at
5 // http://www.apache.org/licenses/LICENSE-2.0
6 //
7 // Unless required by applicable law or agreed to in writing, software
8 // distributed under the License is distributed on an "AS IS" BASIS,
9 // WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
10 // See the License for the specific language governing permissions and
11 // limitations under the License.
12
13 package com.google.gitiles;
14
15 import java.util.List;
16 import static com.google.common.base.Preconditions.checkNotNull;
17 import static javax.servlet.http.HttpServletResponse.SC_INTERNAL_SERVER_ERROR;
18 import static javax.servlet.http.HttpServletResponse.SC_NOT_FOUND;
19
20
21 import com.google.common.base.Optional;
22 import com.google.common.base.Strings;
23 import com.google.common.collect.Iterables;
24 import com.google.common.collect.ListMultimap;
25 import com.google.common.collect.Lists;
26 import com.google.common.collect.Maps;
27 import com.google.common.primitives.Longs;
28 import com.google.gson.reflect.TypeToken;
29
30 import org.eclipse.jgit.errors.IncorrectObjectTypeException;
31 import org.eclipse.jgit.errors.IncorrectObjectTypeException;
32 ... skipped 24 common lines ...
33
34 import java.util.Map;
35
36 import javax.servlet.http.HttpServletRequest;
37 import javax.servlet.http.HttpServletResponse;
38
39 /**
40 * Serves an HTML page with a shortlog for commits and paths. */
41 public class LogServlet extends BaseServlet {
42     private static final long serialVersionUID = 1L;
43     private static final Logger log = LoggerFactory.getLogger(LogServlet.class);
44
45
46     static final String LIMIT_PARAM = "n";
47     static final String START_PARAM = "s";
48     private static final int DEFAULT_LIMIT = 100;
49     private static final int MAX_LIMIT = 10000;
50
51
52     private final Linkifier linkifier;
53     private final PasswordManager passwordManager;
54
55     public LogServlet(Renderer renderer, Linkifier linkifier) {
56         super(renderer);
57     }
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
```

Dashed Column Line - shows preferred line length

Shawn Pearce We should get logDetail worked into this somehow. logDetail for the ... 03/10 17:44
Stefan Zager I defer to your judgement. Changing the subsection to "logDetail" me... 03/10 21:12

- Lines Of Context:

The number of context lines that should be displayed before and after any diff. If the `entire file` checkbox is selected, the full file is rendered.

Skipped common lines can be expanded by clicking on the placeholder for the skipped lines.

Clicking on "... skipped <n> common lines ..." expands the complete block of skipped lines.

If many lines are skipped there are additional links to expand the context by ten lines before and after the skipped block.

gerit / gerit-server/src/main/java/com/google/gerrit/server/git/ReceiveCommits.java

Pattern Set Base 1 2 3 4 5 6 7 8 9 10

```
98 import com.google.gerrit.server.project.ChangeControl;
99 import com.google.gerrit.server.project.ProjectCache;
100 import com.google.gerrit.server.project.ProjectControl;
101 import com.google.gerrit.server.project.ProjectState;
102 import com.google.gerrit.server.project.RefControl;
103 import com.google.gerrit.server.query.change.ChangeData;
104 import com.google.gerrit.server.ssh.SshInfo;
105 import com.google.gerrit.server.util.MagicBranch;
106 import com.google.gerrit.server.util.RequestScopePropagator;
107 import com.google.gerrit.server.util.TimeUtil;
108 import com.google.gerrit.util.cli.CmdLineParser;
109 import com.google.gwtorm.server.AtomicUpdate;
110 import com.google.gwtorm.serverOrmException;
111 import com.google.gwtorm.server.ResultSet;
112 import com.google.gwtorm.server.SchemaFactory;
113 import com.google.inject.Inject;
114 import com.google.inject.Provider;
115
116     *+100 ... skipped 948 common lines ... +100*
117
118 private static class MagicBranchInput {
119     private static final Splitter COMMAS = Splitter.on(',').omitEmptyStrings();
120
121     final ReceiveCommand cmd;
122     BranchNameKey dest;
123     RefControl ctl;
124     Set<Account.Id> reviewer = Sets.newLinkedHashSet();
125     Set<Account.Id> cc = Sets.newLinkedHashSet();
126
127     List<RevCommit> baseCommit;
128
129
130     #Option(name = "--base", metaVar = "BASE", usage = "merge base of changes")
131     List<ObjectId> base;
132
133     #Option(name = "--topic", metaVar = "NAME", usage = "attach topic to changes")
134     String topic;
135
136     #Option(name = "--draft", usage = "mark new/updated changes as draft")
137     boolean draft;
138
139     #Option(name = "--submit", usage = "immediately submit the change")
140     boolean submit;
141
142
143     #Option(name = "-r", metaVar = "EMAIL", usage = "add reviewer to changes")
144     void reviewer(Account.Id id) {
145         reviewer.add(id);
146     }
147
148     #Option(name = "--cc", metaVar = "EMAIL", usage = "notify user by CC")
149     void cc(Account.Id id) {
150         cc.add(id);
151     }
152
153     #Option(name = "--publish", usage = "publish new/updated changes")
154     void publish(boolean publish) {
155         draft = !publish;
156     }
157
158     MagicBranchInput(ReceiveCommand cmd) {
159
160
161     }
162
163     *+100 ... skipped 948 common lines ... +100*
164
165     private static class MagicBranchInput {
166         private static final Splitter COMMAS = Splitter.on(',').omitEmptyStrings();
167
168         final ReceiveCommand cmd;
169         BranchNameKey dest;
170         RefControl ctl;
171         Set<Account.Id> reviewer = Sets.newLinkedHashSet();
172         Set<Account.Id> cc = Sets.newLinkedHashSet();
173         Set<LabelType> labelTypes;
174         CmdLineParser cli;
175
176         #Option(name = "--base", metaVar = "BASE", usage = "merge base of changes")
177         List<ObjectId> base;
178
179         #Option(name = "--topic", metaVar = "NAME", usage = "attach topic to changes")
180         String topic;
181
182         #Option(name = "--draft", usage = "mark new/updated changes as draft")
183         boolean draft;
184
185         #Option(name = "--submit", usage = "immediately submit the change")
186         boolean submit;
187
188
189         #Option(name = "-r", metaVar = "EMAIL", usage = "add reviewer to changes")
190         void reviewer(Account.Id id) {
191             reviewer.add(id);
192         }
193
194         #Option(name = "--cc", metaVar = "EMAIL", usage = "notify user by CC")
195         void cc(Account.Id id) {
196             cc.add(id);
197         }
198
199         #Option(name = "--publish", usage = "publish new/updated changes")
200         void publish(boolean publish) {
201             draft = !publish;
202         }
203
204         #Option(name = "-l", metaVar = "LABEL/VALUE"
205             usage = "label(s) to assign (defaults to +1 if no value provided")
206         void addLabel(final String token) throws CmdLineException {
207             if (token.startsWith("+")) {
208                 labelTypes.add(LabelType.valueOf(token.substring(1)));
209             } else {
210                 LabelValue lv = new LabelValue(token);
211                 labelTypes.add(lv);
212                 labels.put(lv.getLabel(), lv.getValue());
213             }
214         }
215
216     }
217
218     *+100 ... skipped 948 common lines ... +100*
```

- Intraline Difference:

Controls whether intraline differences should be highlighted.

gerit / Documentation/intro-change-screen.txt

Patch Set Base 1 2

... skipped 3 common lines ...

```
4 change screen is deprecated and will be discontinued in one of the next Gerrit releases.
5
6 The design spirit of the new change screen is simplicity: only one patch set is
7 presented on the screen. The list of related changes is always visible and
8 optional elements are moved to pop down boxes.
9
10 This is not only a facelift. The main highlights are under the hood:
11
12 * Old style RPC calls are replaced by the REST API
13 * The prettify syntax highlighting library was replaced by CodeMirror
14 * Automatic refresh of open changes
15 * Support to download a patch direct in browser: no local repo is needed
16 * JS API integration: it was never so easy to add change/revision actions or
17   HTML fragments to the UI from a plugin
18
19 This document is intended to help users to switch to the new change screen.
20
21 Further information on the topic can be found in the:
22 link:https://groups.google.com/forum/#topic/repo-discuss/6Ryz9p6AzgB
23 CodeScreen2 thread on the repo-discuss mailing list.
+108 ... skipped 134 common lines ... +108
```

159 is a draft.
160
161 [[draft-comments]]
162 == Highlight draft comments
163
164 If a patch set has draft comments that weren't published yet, then that patch
165 set is marked on the list in the 'Patch Sets' drop down list. In addition a red
166 "draft" prefix appears on the filenames in the file table.
167
168 [[codemirror]]
169 == CodeMirror
170
171 On the user preferences page, 'Side By Side' or 'Unified Diff' view can be
172 configured. Use the "/" key to start the CodeMirror search, like in 'vim'.
173
174 Key bindings are not customizable at the moment. They may be added in the future.
175
176 Range comments are supported on CodeMirror's 'Side By Side' screen. Highlight
177 lines with the mouse and then click the bottom-most line number to create a
178 range comment for the highlighted lines. It can also be used to select a
179 region on one line to emphasize what the comment is related to.
180
181 Key bindings:
182
183 * j / k: next line / previous line
184 * n / p: next diff chunk / previous diff chunk
185 * J / I: next file / previous file
186 * u: up to change
187 * ci: create an inline comment (focus has to be in codemirror view)
188 * or <Ctrl> + f: search in the current file
189 * <Shift> + <Ctrl> + f: search all comments on current line
190 * <Enter> or <Space>: collapse all comments
191 * <Esc> or Cancel: comment edit
192 * i: toggle intraline differences
193 * ,: Show diff preferences
194 * <Ctrl> + s: Save draft comment
195
196 [[reviewers]]
197 == Reviewers
+105 ... skipped 89 common lines ... +105

188 * / or <Ctrl> + f: search in the current file
189 * <Shift> + <Ctrl> + f: collapse all comments on current line
190 * <Shift> or <Space>: expand all comments
191 * <Esc> or Cancel: comment edit
192 * i: toggle intraline differences
193 * ,: Show diff preferences
194 * <Ctrl> + s: Save draft comment
195
196 [[reviewers]]
197 == Reviewers
+100 ... skipped 89 common lines ... +100

287 Key bindings: 'J' & 'K' to navigate between the related changes. 'O' to
288 open the currently selected change on one of the related changes tab base.

Patch Set 1 2

... skipped 3 common lines ...

```
4 change screen is deprecated and will be discontinued in one of the next Gerrit releases.
5
6 The design spirit of the new change screen is simplicity: only one patch set is
7 presented on the screen. The list of related changes is always visible and
8 optional elements are moved to pop down boxes.
9
10 This is not only a facelift. The main highlights are under the hood:
11
12 * Old style RPC calls are replaced by the REST API
13 * The prettify syntax highlighting library was replaced by CodeMirror
14 * Automatic refresh of open changes
15 * Support to download a patch direct in browser: no local repo is needed
16 * JS API integration: it was never so easy to add change/revision actions or
17   HTML fragments to the UI from a plugin
18
19 This document is intended to help users to switch to the new change screen.
20
21 Further information on the topic can be found in the:
22 link:https://groups.google.com/forum/#topic/repo-discuss/6Ryz9p6AzgB
23 CodeScreen2 thread on the repo-discuss mailing list.
+108 ... skipped 134 common lines ... +108
```

159 is a draft.
160
161 [[draft-comments]]
162 == Highlight draft comments
163
164 If a patch set has draft comments that weren't published yet, then that patch
165 set is marked on the list in the 'Patch Sets' drop down list. In addition a red
166 "draft" prefix appears on the filenames in the file table.
167
168 [[codemirror]]
169 == CodeMirror
170
171 On the user preferences page, 'Side By Side' or 'Unified Diff' view can be
172 configured. Use the "/" key to start the CodeMirror search, like in 'vim'.
173
174 Key bindings are not customizable at the moment. They may be added in the future.
175
176 Range comments are supported on CodeMirror's 'Side By Side' screen. Highlight
177 lines with the mouse and then click the bottom-most line number to create a
178 range comment for the highlighted lines. It can also be used to select a
179 region on one line to emphasize what the comment is related to.

Highlighting of Intraline Differences

The diagram illustrates the 'Highlighting of Intraline Differences' feature. A central text block is surrounded by two red circles. One circle is on the left, covering the 'Patch Set Base 1 2' section, and another is on the right, covering the 'Patch Set 1 2' section. Red arrows point from both circles towards the 'Range comments are supported...' line in the middle, which describes how to highlight and manage comments within a single line of code.

- Syntax Highlighting:

Controls whether syntax highlighting should be enabled.

The language for the syntax highlighting is automatically detected from the file extension.

Syntax Coloring for many Languages / File Types

```
// WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
// See the License for the specific language governing permissions and
// limitations under the License.
package com.google.gitlets;
import static com.google.common.base.Preconditions.checkNotNull;
import static javax.servlet.http.HttpServletResponse.SC_INTERNAL_SERVER_ERROR;
import static javax.servlet.http.HttpServletResponse.SC_NOT_FOUND;
import com.google.common.base.Objects;
import com.google.common.base.Optional;
import com.google.common.base.Strings;
import com.google.common.collect.ListMultimap;
import com.google.common.collect.Lists;
import com.google.common.collect.Maps;
import com.google.common.primitives.Longs;
import com.google.gson.reflect.TypeToken;
import org.eclipse.jgit.errors.IncorrectObjectTypeException;
import java.util.Map;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
/** Serves an HTML page with a shortcut for commits and paths. */
public class LogServlet extends BaseServlet {
    private static final long serialVersionUID = 1L;
    private static final Logger log = LoggerFactory.getLogger(LogServlet.class);
    static final String LIMIT_PARAM = "n";
    static final String START_PARAM = "s";
    private static final int DEFAULT_LIMIT = 100;
    private static final int MAX_LIMIT = 10000;
    private final Linkifier linkifier;
    public LogServlet(Renderer renderer, Linkifier linkifier) {
        super(renderer);
        this.linkifier = checkNotNull(linkifier, "linkifier");
    }
    @Override
    protected void doGetHtml(HttpServletRequest req, HttpServletResponse res) throws IOException {
        Repository repo = ServletUtils.getRepository(req);
        GitletsView view = getview(req, repo);
        Paginator paginator = newPaginator(repo, view);
        if (paginator == null) {
            ...
        } else {
            Map<String, Object> params = new HashMap<String, Object>(2);
            params.put("repo", repo);
            params.put("view", view);
            params.put("paginator", paginator);
            params.put("start", Long.parseLong(req.getParameter(START_PARAM)));
            params.put("limit", Integer.parseInt(req.getParameter(LIMIT_PARAM)));
            params.put("logDetail", logDetail);
            params.put("accessFactory", accessFactory);
            params.put("linkifier", linkifier);
            renderer.render(res, "log.html", params);
        }
    }
}
```

• Whitespace Errors:

Controls whether whitespace errors are highlighted.

- Show Tabs:

Controls whether tabs are highlighted.

- Line Numbers:

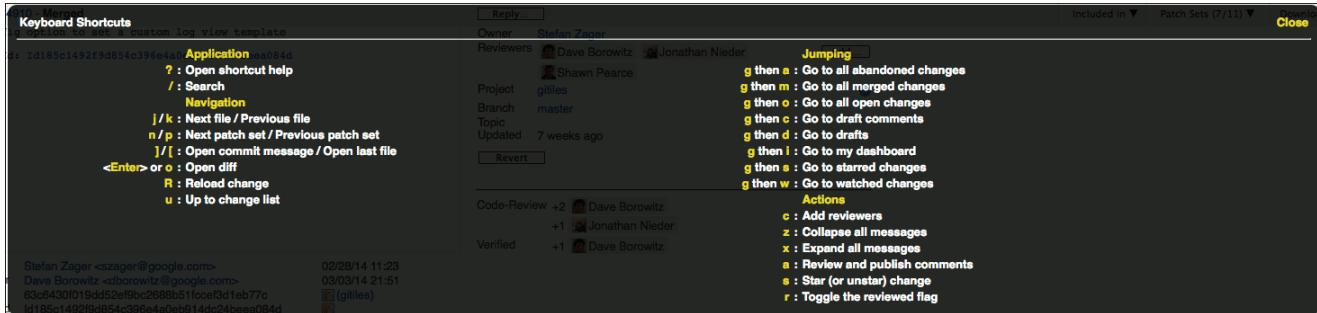
Controls whether line numbers are shown.

- Top Menu:
Controls whether the top menu is shown.
- Mark Reviewed:
Controls whether the files of the patch set should be automatically marked as reviewed when they are viewed.
- Expand All Comments:
Controls whether all comments should be automatically expanded.
- Render:
Controls how patch files that exceed the screen size are rendered.
If `Fast` is selected file contents which are outside of the visible area are not attached to the browser's DOM tree. This makes the rendering fast, but searching by `Ctrl+F` only finds content which is in the visible area.
If `Slow` is selected all file contents are attached to the browser's DOM tree, which makes the rendering slow for large files. The advantage of this setting is that `Ctrl+F` can be used to search in the complete file.

Keyboard Shortcuts

Navigation within the review UI can be completely done by keys, and most actions can be controlled by keyboard shortcuts. Typing `?` opens a popup that shows a list of available keyboard shortcuts:

- Change Screen



- Side-by-Side Diff Screen



In addition, Vim-like commands can be used to [navigate](#) and [search](#) within a patch file.

New Review UI vs. Old Review UI

There are some important conceptual differences between the old and new review UIs:

- The old change screen directly shows all patch sets of the change. With the new change screen only a single patch set is displayed; users can switch between the patch sets by choosing another patch set from the [Patch Sets](#) drop down panel in the screen header.
- On the old side-by-side diff screen, new comments are inserted by double-clicking on a line. With the new side-by-side diff screen double-click is used to select a word for commenting on it; there are [several ways to insert new comments](#), e.g. by selecting a code block and clicking on the popup comment icon.

Limitations of the new review UI:

- The new side-by-side diff screen cannot render images.
- Unified diff view is missing:

By setting `Diff View (New Change Screen)` in the user preferences to `Unified Diff` the new change screen can be configured to open the file diffs in the old unified diff view.

Users preferring the old review UI can [configure the change view](#) in their preferences.

Part of [Gerrit Code Review](#)

Version v2.9