Find the Differences Between Two Git Branches

Last updated: February 6, 2024

Written by: baeldung (https://www.baeldung.com/ops/author/baeldung)

Git (https://www.baeldung.com/ops/category/git)

Git Branches (https://www.baeldung.com/ops/tag/git-branches)

1. Overview

In this tutorial, we'll discover ways of finding differences between two *git (/git-guide#what-is-git)* branches. We are going to explore the *git diff* (https://git-scm.com/docs/git-diff) command and use it in branch comparison.

2. Compare Branches in a Single Command

git diff is a useful command that allows us to compare different types of git objects, such as files, commits, branches, and many more. **This makes *git diff* a good choice when we need to compare the differences between two branches.**

To compare the branches, we specify both branches' names after the *git diff* command:

```
$ git diff branch1 branch2
diff --git a/file1.txt b/file1.txt
index 3b18e51..c28f4fa 100644
--- a/file1.txt
+++ b/file1.txt
@@ -1 +1 @@
-hello world
+hello from branch2
```

Let's examine the output. The first line *diff --git a/file1.txt b/file1.txt* shows which files are different in the branches. In our example, it is the file *file1.txt,* which is represented as *a/file1.txt* for *branch1,* and *b/file1.txt* is the same file for *branch2*.

The next line shows index cache numbers for the *file1.txt* in both the branches. Then, there are several lines that show the file names with added content, and the ones with removed content.

And finally, the most important lines are at the end of the output, where we can see the exact modified lines. We can see that the line *"hello world"* in the file *file1.txt* from *branch1* was replaced by the line *"hello from branch2"* in *branch2*.

3. Showing Only the Names of Files

Although we found the differences between the two branches in the example above, it can be inconvenient to display changes for every single line. Often, we want to see just the names of the changed files.

To display only the names of files that are different between two branches, we use the *--name-only* option in the *git diff* command:

```
$ git diff branch1 branch2 --name-only
file1.txt
```

Now, the output shows just the name of files that are different in both the branches. In our case, it's just a single file *file1.txt*.

4. An Alternative Way to Compare Branches

In the examples above, we used a single *git diff* command to find the differences between the branches. However, we had to specify both branches' names in this command. Sometimes **it feels more intuitive if we do the comparison from within one of the branches.**

To check out to a branch, we use the *git checkout (https://git-scm.com/docs/git-checkout)* command. After that, we only need to *diff* the other branch and see all the changes relative to our current branch.

For example, let's compare *branch1* and *branch2* from within *branch2*:

```
$ git checkout branch2
$ git diff branch1
```

We should now be able to see the same output as earlier.

5. Finding the Differences From a Common Ancestor

So far we were comparing the two branches in regard to their latest states. However, sometimes we need to compare a branch with a common ancestor of another branch.

In other words, we may need to find all differences in the branch compared to another one since they were branched off. For illustration, let's see the tree below:

```
---A---B---C---D  <== branch1
        \
         E---F    <== branch2
```

We'd like to find the differences between *branch2* (current commit position F) and *branch1* at its commit position B. As we can see, at position B, the two branches are forked out and then they develop separately from each other. In this case, position B is the common ancestor of the two branches.

To find the diff between *branch2* and *branch1* common ancestor, we need to use three dots between the branch names:

```
git diff branch1...branch2
```

The output will be in a similar format as earlier. However, now the comparison is happening between *branch2* and a common ancestor of *branch1.*

6. Finding the Differences Between Commit Hashes

It's **possible to display differences not only between the branches, but also between the commits**.

The syntax for this is similar to the branch comparison. However, instead of branches, we need to specify the commit hashes that we want to compare:

```
git diff b94a88bac17318fb3c3cc881d657c04de9fd7901
73ea8956375c10fe41c669ba8c6f6f9e01490452
```

The output will be of a similar format to the examples above.

7. Conclusion

In this article, we learned how to find the differences between two git branches using the *git diff* command.

We first looked at comparing the branches using a single command. Then, we found out how to compare branches from within one of the branches using the *git checkout* command.

Lastly, we saw how to compare branches to a common ancestor, along with finding the differences between commits.