TeamCity Control

Andrea Gennari

Contents

1 Introduction

This project provides a tool to efficiently identify and classify differences between local and remote Git branches, highlighting modified, deleted, and renamed files relative to a common merge base. It simplifies tracking changes and resolving conflicts in collaborative Git workflows.

2 Test

2.1 Files

- **Modified Files**
  This test was just to check if a file on *branchA* and *branchB* had been modified in content by comparing the SHAs. For this test, `edited_only()`, we excluded the case in which a file was modified on one branch and removed on the other.

- **Deleted File**
  The first edge case encountered is when a file was modified on the remote branch *branchA* and deleted on the local branch *branchB*, or viceversa. We managed this with the `deleted_to_edited()` function.

- **Renamed and Modified File**
  With the `renamed_to_edited()` function, we deal with a situation in which a file has been renamed, keeping the same content, on one branch, and modified on the other branch.

- **Renamed and Deleted File**
  Here we handle a file renamed on one branch and deleted on the other, using the `renamed_to_deleted()` function.

2.2 Output

Listing 1: Output example from the Git Diff tool

```
Files modified both remotely (branchA) and locally (branchB), still
    existing in both:
 - file1.txt

Files deleted in one branch and modified in the other:
 - file5.txt -> deleted in branchB, modified in branchA
 - file4.txt -> deleted in branchA, modified in branchB

Files renamed or added in one branch, but modified in the other:
 - file3.txt -> file6.txt: renamed in branchA, modified in branchB as
     file3.txt

Files renamed or added in one branch and deleted in the other:
 - file2.txt -> file7.txt: renamed in branchB, deleted in branchA
```

When we have a file with the same SHA as the one at the merge base, we cannot tell if it is a new file with the same content or the file has been renamed, but for simplicity we assume that it has been renamed.

The algorithm is correctly not listing the file `file8.txt` in branchA, even if it has the same content as `file4.txt` in branchB, because it did not existed at the merge base.