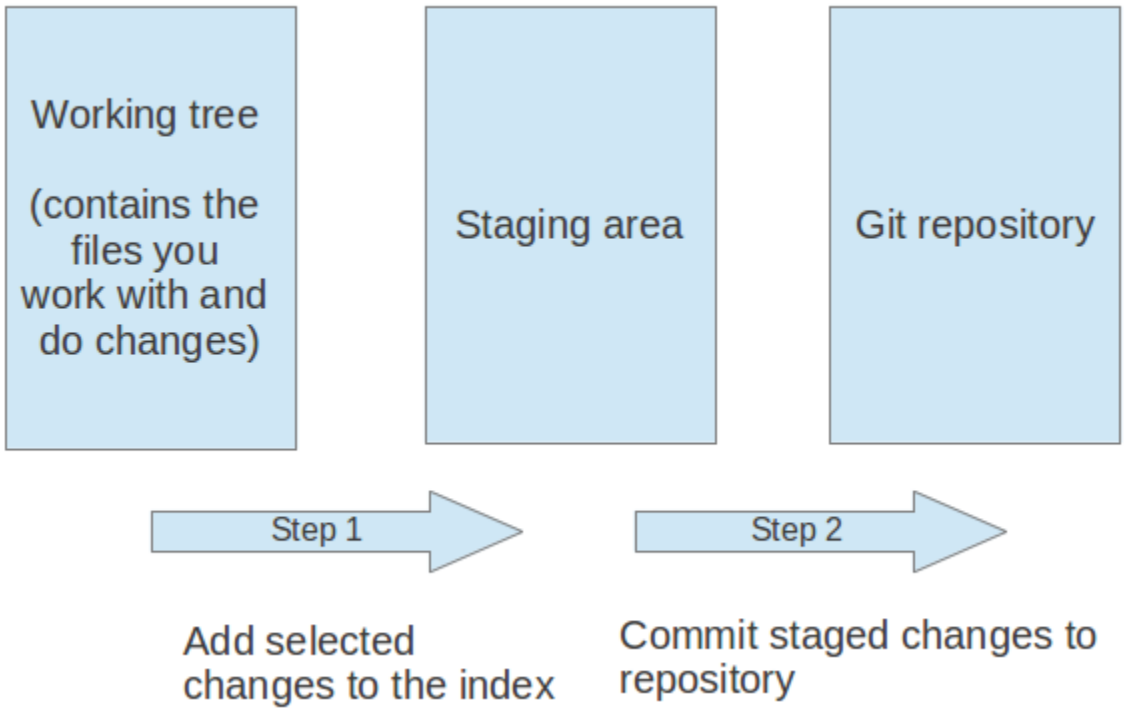
4. The process of adding to a Git repository via staging and committing

4.1. Adding changes to a Git repository

If you modify your *working tree* you need to perform two steps to persist these changes in your local repository. You

- add selected changes to the something called the *staging area* and
- afterwards you commit the changes stored in the *staging area* to the repository

This process is depicted in the following graphic.



4.2. Adding to the staging area

You need to mark changes in the working tree to be relevant for Git. This process is called *staging* or *to add changes to the staging area*.

You add changes in the working tree to the staging area with the `git add` command. This command stores a snapshot of the specified files in the staging area.

The `git add` command allows you to incrementally modify files, stage them, modify and stage them again until you are satisfied with your changes.

Older versions of Git used the term *index* instead of staging area. Staging area is nowadays the preferred term by the Git community. Both terms mean the same thing.

4.3. Committing to the repository

After adding the selected files to the staging area, you can *commit* these files to add them permanently to the Git repository. *Committing* creates a new persistent snapshot (called *commit* or *commit object*) of the staging area in the Git repository. A commit object, like all objects in Git, is immutable.

The *staging area* keeps track of the snapshots of the files until the staged changes are committed.

For committing the staged changes you use the `git commit` command.

4.4. Committing changes

If you commit changes to your Git repository, you create a new *commit object* in the Git repository.