**git rebase master**:

* When you run **git rebase master** from your feature branch, it means you want to bring in changes from the **master** branch into your feature branch.
* This is commonly used when you want to keep your feature branch up to date with the latest changes in the **master** branch.
* During this rebase, Git will replay your feature branch's commits on top of the current **master** branch tip.
* This results in a linear commit history where your feature branch appears to be "based on" the latest **master** branch commit.
* Example:

**(master) A---B---C---D---E**

**\**

**(feature) X---Y**

After **git rebase master**:

**(master) A---B---C---D---E**

**\**

**(feature) X'---Y'**

**git rebase feature**:

Running **git rebase feature** from your **master** branch would mean you want to incorporate changes from your feature branch into the **master** branch.

This is less common and is usually done when you want to bring specific features or changes from a feature branch back into the **master** branch.

During this rebase, Git will replay your **master** branch's commits on top of the latest commit in your feature branch.

It can be useful when your feature branch contains a substantial change or feature that you want to include in the **master** branch.

Example:

(master) A---B---C---D---E

\

(feature) X---Y

After **git rebase feature**:

In summary, the target branch in **git rebase** determines which branch's changes will be replayed on top of the other. It's essential to choose the right direction based on your workflow and intentions. Typically, you'd use **git rebase master** to keep your feature branch up to date with **master** and **git rebase feature** less frequently when you want to merge feature changes into **master**.

|  |  |
| --- | --- |
| Merge | Rebase |
| Git Merge lets you merge different Git branches. | Git Rebase allows you to integrate the changes from one branch into another. |
| Git Merge logs show you the complete history of commit merging. | Git Rebase logs are linear. As the commits are rebased, the history is altered to reflect this. |
| All the commits on a feature branch are combined into a single commit on the master branch. | All commits are rebased, and the same number of commits are added to the master branch. |
| Merge is best used when the target branch is supposed to be shared. | Rebase is best used when the target branch is private. |
| Merge preserves history. | Rebase rewrites history. |