Merge/Rebase Examples

???????? ???????

Схема коммитов до объединения веток

```
* efa552b - (2026-06-25 13:32:25 +0300) v6: develop - shion (develop)
* c828a46 - (2026-06-25 13:32:25 +0300) v5: develop - shion
* e17a9c1 - (2026-06-25 13:32:25 +0300) v4: develop - shion
| * 10834a6 - (2026-06-25 13:32:08 +0300) v6: main - shion (HEAD -> main)
| * ecb682a - (2026-06-25 13:32:08 +0300) v5: main - shion
| * 3725452 - (2026-06-25 13:32:08 +0300) v4: main - shion
|/
* 6dca252 - (2026-06-25 13:32:08 +0300) v3: main - shion
* 6630ea4 - (2026-06-25 13:32:08 +0300) v2: main - shion
* 6b1466c - (2026-06-25 13:32:08 +0300) v1: main - shion
```

code.txt [Main ver.]

```
01_main
02_main
03_main
04_main
05_main
06_main
```

code.txt [Develop ver.]

```
01_main
02_main
03_main
04_develop
05_develop
06_develop
```

Git Merge

??????? 1. ?????????????? Main

```
git checkout main
# Включаем режим union-слияния для файла code.txt
echo "code.txt merge=union" > .gitattributes
# Слияние ветки develop с приоритетом изменений main
git merge -Xours develop -m "Merge branch 'develop'"
rm .gitattributes
```

> Ветка develop вливается в ветку main. Файлы объединяются построчно
> В конфликтных местах (строки 04-06) побеждает версия из main
> Не конфликтующие строки (07-09 из develop) добавляются в результирующий файл

```
*   6a3d02b - (2026-06-25 13:34:55 +0300) Merge branch 'develop' - shion (HEAD -> main)
|\
| * efa552b - (2026-06-25 13:32:25 +0300) v6: develop - shion (develop)
| * c828a46 - (2026-06-25 13:32:25 +0300) v5: develop - shion
| * e17a9c1 - (2026-06-25 13:32:25 +0300) v4: develop - shion
* | 10834a6 - (2026-06-25 13:32:08 +0300) v6: main - shion
* | ecb682a - (2026-06-25 13:32:08 +0300) v5: main - shion
* | 3725452 - (2026-06-25 13:32:08 +0300) v4: main - shion
|/
* 6dca252 - (2026-06-25 13:32:08 +0300) v3: main - shion
* 6630ea4 - (2026-06-25 13:32:08 +0300) v2: main - shion
* 6b1466c - (2026-06-25 13:32:08 +0300) v1: main - shion
```

code.txt [Merge]

01_main
02_main
03_main
04_main
05_main
06_main
04_develop
05_develop
06_develop

??????? 2. ????????????? Develop

```
git checkout develop
# Включаем режим union-слияния для объединения строк файла code.txt
echo "code.txt merge=union" > .gitattributes
# Слияние ветки main с приоритетом изменений develop
git merge -Xtheirs main -m "Merge branch 'main' into develop"
rm .gitattributes
```

> Ветка main вливается в ветку develop. Файлы объединяются построчно
> В конфликтных местах (строки 04-06) побеждает версия из develop (-Xours)
> Не конфликтующие строки (04-06 из main) добавляются в результирующий файл

```
*   47739bd - (2026-06-25 13:39:07 +0300) Merge branch 'main' into develop - shion (HEAD ->
develop)
|\
| * 10834a6 - (2026-06-25 13:32:08 +0300) v6: main - shion (main)
| * ecb682a - (2026-06-25 13:32:08 +0300) v5: main - shion
| * 3725452 - (2026-06-25 13:32:08 +0300) v4: main - shion
* | efa552b - (2026-06-25 13:32:25 +0300) v6: develop - shion
* | c828a46 - (2026-06-25 13:32:25 +0300) v5: develop - shion
* | e17a9c1 - (2026-06-25 13:32:25 +0300) v4: develop - shion
|/
* 6dca252 - (2026-06-25 13:32:08 +0300) v3: main - shion
* 6630ea4 - (2026-06-25 13:32:08 +0300) v2: main - shion
* 6b1466c - (2026-06-25 13:32:08 +0300) v1: main - shion
```

code.txt [Dev ver.]

01_main
02_main
03_main
04_develop
05_develop
06_develop

Git Rebase

??????? 1. ?????????? Main

```
git checkout develop
git rebase main
```

> **Суть:** Коммиты из ветки develop переносятся в конец ветки main и пересоздаются с новыми hash
> **Файлы:** При перебазировании каждый коммит из develop по очереди применяется поверх текущего состояния main
> **Хронология:** Ветка develop перестраивается поверх main, её коммиты получают новые hash
> Конфликты разрешаются в пользу ветки develop

> Rebase main (+develop) — подобен процессу, когда вы синхронизируете свою инфраструктуру (main)
> с последними изменениями с тестового стенда (develop), перестраивая историю develop поверх main,
> но сохраняя все свои экспериментальные наработки (develop) в конфликтных местах
> Вы поднимаете main на новый уровень, содержащий все изменения develop
> и история main становится линейным продолжением истории develop

```
* f7f88b7 - (2026-06-25 15:53:56 +0300) v6: develop - shion (HEAD -> develop)
* d5f3c38 - (2026-06-25 15:53:56 +0300) v5: develop - shion
* ce362ac - (2026-06-25 15:53:56 +0300) v4: develop - shion
* 10834a6 - (2026-06-25 13:32:08 +0300) v6: main - shion (main)
* ecb682a - (2026-06-25 13:32:08 +0300) v5: main - shion
* 3725452 - (2026-06-25 13:32:08 +0300) v4: main - shion
* 6dca252 - (2026-06-25 13:32:08 +0300) v3: main - shion
* 6630ea4 - (2026-06-25 13:32:08 +0300) v2: main - shion
* 6b1466c - (2026-06-25 13:32:08 +0300) v1: main - shion
```

code.txt [Rebase]

01_main
02_main
03_main
04_develop
05_develop
06_develop

??????? 2. ?????????? Develop

```
git checkout main
git rebase develop
```

> **Суть:** Commits из ветки main переносятся в конец ветки develop и пересоздаются с новыми hash
> **Файлы:** При перебазировании каждый commit из main по очереди применяется поверх текущего состояния develop
> **Хронология:** Ветка main перестраивается поверх develop, её коммиты получают новые hash
> Конфликты разрешаются в пользу ветки main

> Rebase develop (+main) - подобен процессу, когда вы обновляете свой тестовый стенд(develop),
> пересобирая его на основе последней версии инфраструктуры (main), но с сохранением всех кастомных доработок.
> Теперь develop базируется на свежей стабильной версии, а его история является логичным продолжением истории main.

```
* 94f8376 - (2026-06-25 16:12:33 +0300) v6: main - shion (HEAD -> main)
* daea4c3 - (2026-06-25 16:12:33 +0300) v5: main - shion
* 98ceae4 - (2026-06-25 16:12:33 +0300) v4: main - shion
* efa552b - (2026-06-25 13:32:25 +0300) v6: develop - shion (develop)
* c828a46 - (2026-06-25 13:32:25 +0300) v5: develop - shion
* e17a9c1 - (2026-06-25 13:32:25 +0300) v4: develop - shion
* 6dca252 - (2026-06-25 13:32:08 +0300) v3: main - shion
* 6630ea4 - (2026-06-25 13:32:08 +0300) v2: main - shion
* 6b1466c - (2026-06-25 13:32:08 +0300) v1: main - shion
```

code.txt [Rebase]

01_main
02_main
03_main
04_main
05_main
06_main

??????? REBASE

>

Если в develop больше строк, (например 07-09) и они не конфликтуют с main, то они сохраняются и после rebase
Конфликт возникает только там, где строки пересекаются (04-06)

ЗАПРЕЩЕНО производить git rebase с ветками которые вы уже отправили в общий репозиторий !

Revision #95
Created 2026-06-22 15:30:24 UTC by sa
Updated 2026-07-13 20:11:35 UTC by sa