



Системы управления кодом

# Что?

- хранение кода
- обмен изменениями
- история изменений

# Зачем?

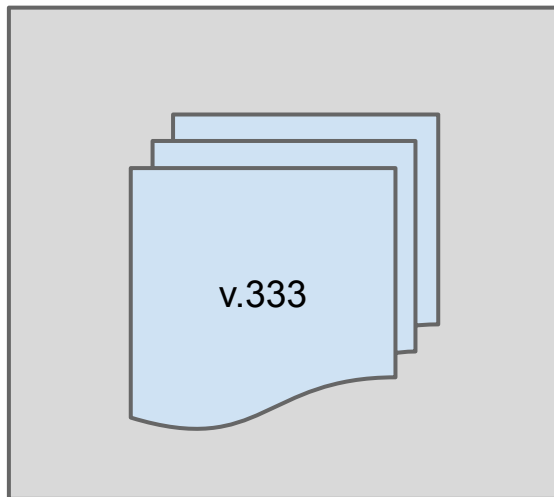
- сохранность кода
- работа в команде
- поиск изменений
- авторство кода
- откат изменений
- СНИМКИ

# Без **VCS**

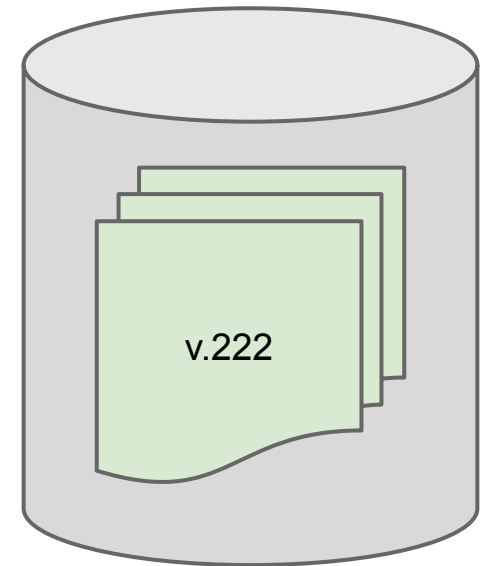
- сетевой диск
- обмен архивами
- обмен патчами
- отсутствие полной истории
- велика вероятность испортить
- нужны дополнительные согласования

# Основные понятия

Рабочая копия

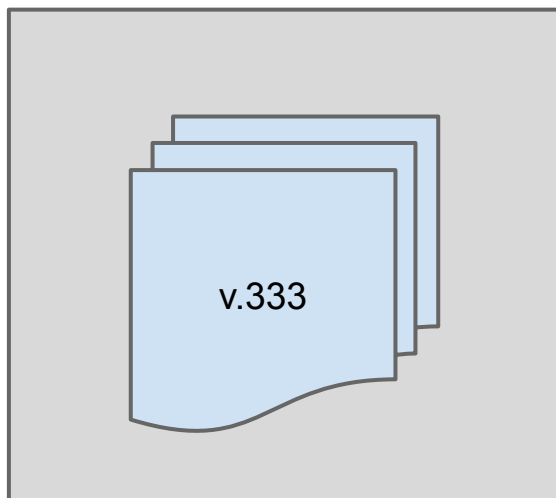


Репозиторий



# Основные понятия

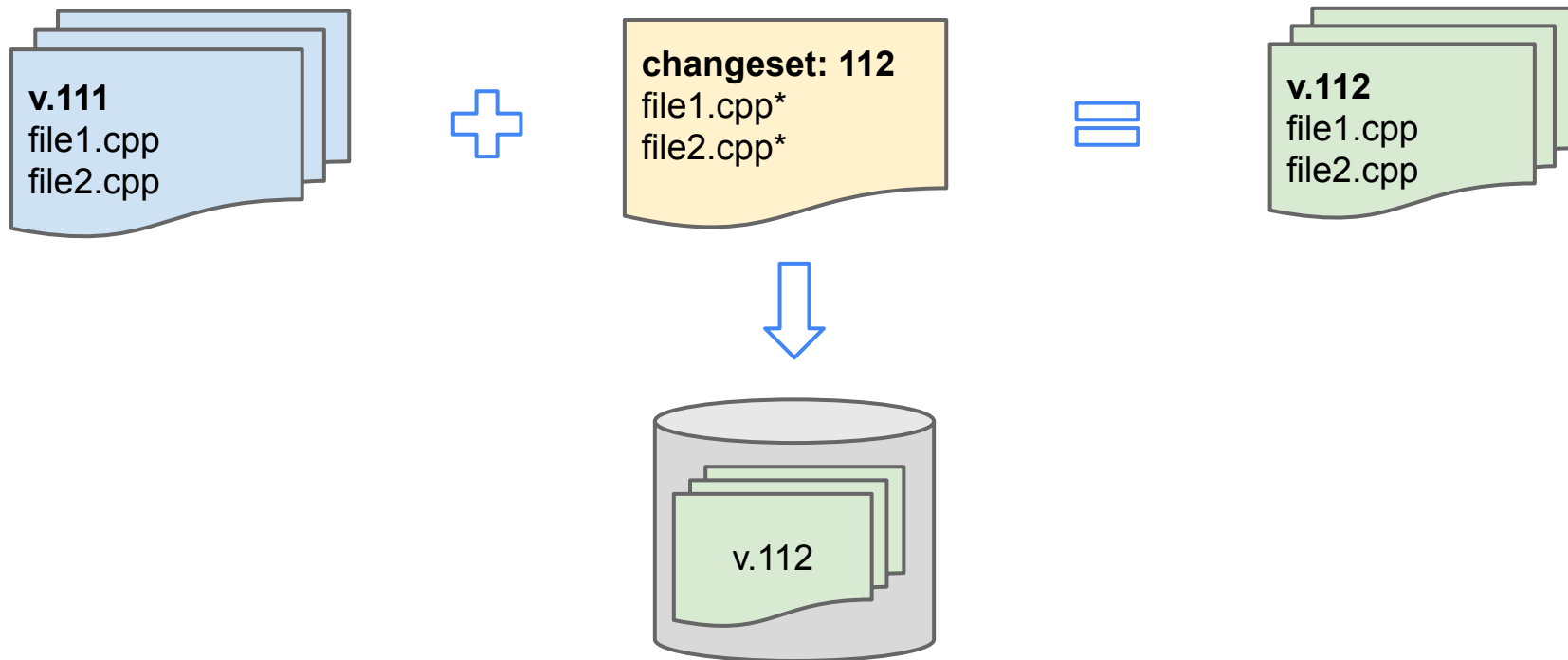
## Версия



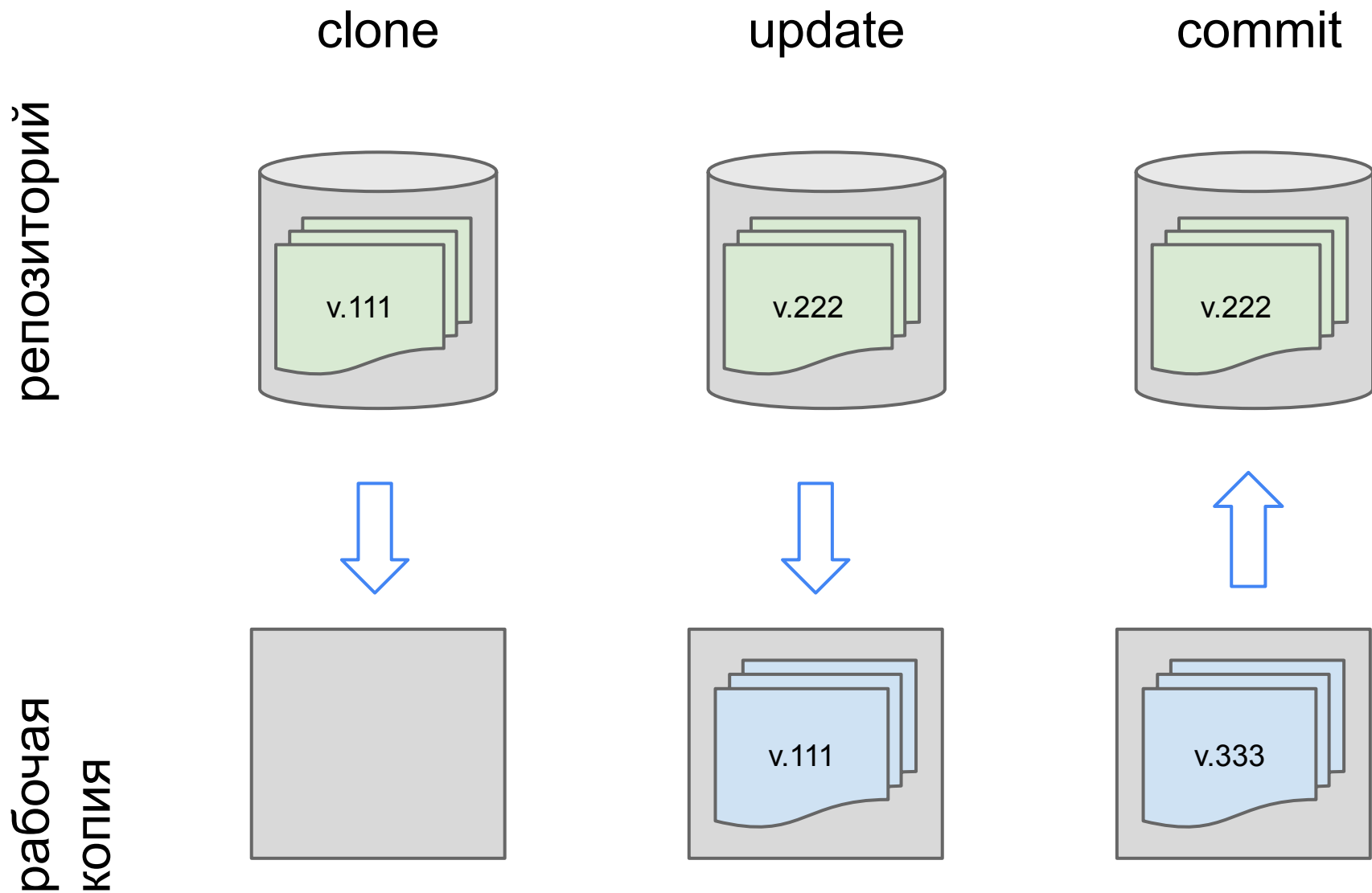
- номер
  - локальный
  - глобальный
- автор
- дата
- изменения (патч)

# Изменения

- имя ревизии
- список файлов
- текущая ветка
- сообщение
- автор изменений
- дата коммита

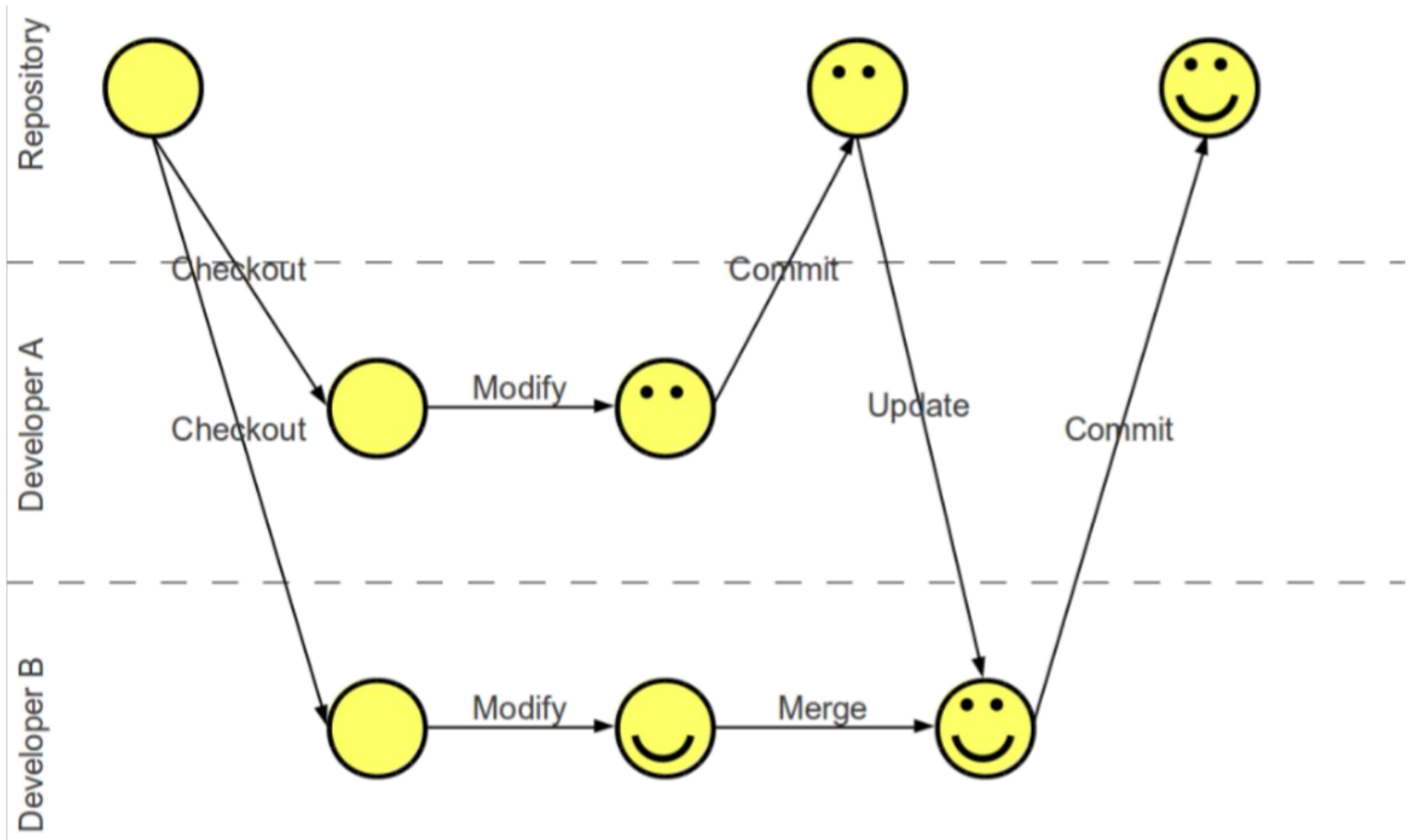


# Действия





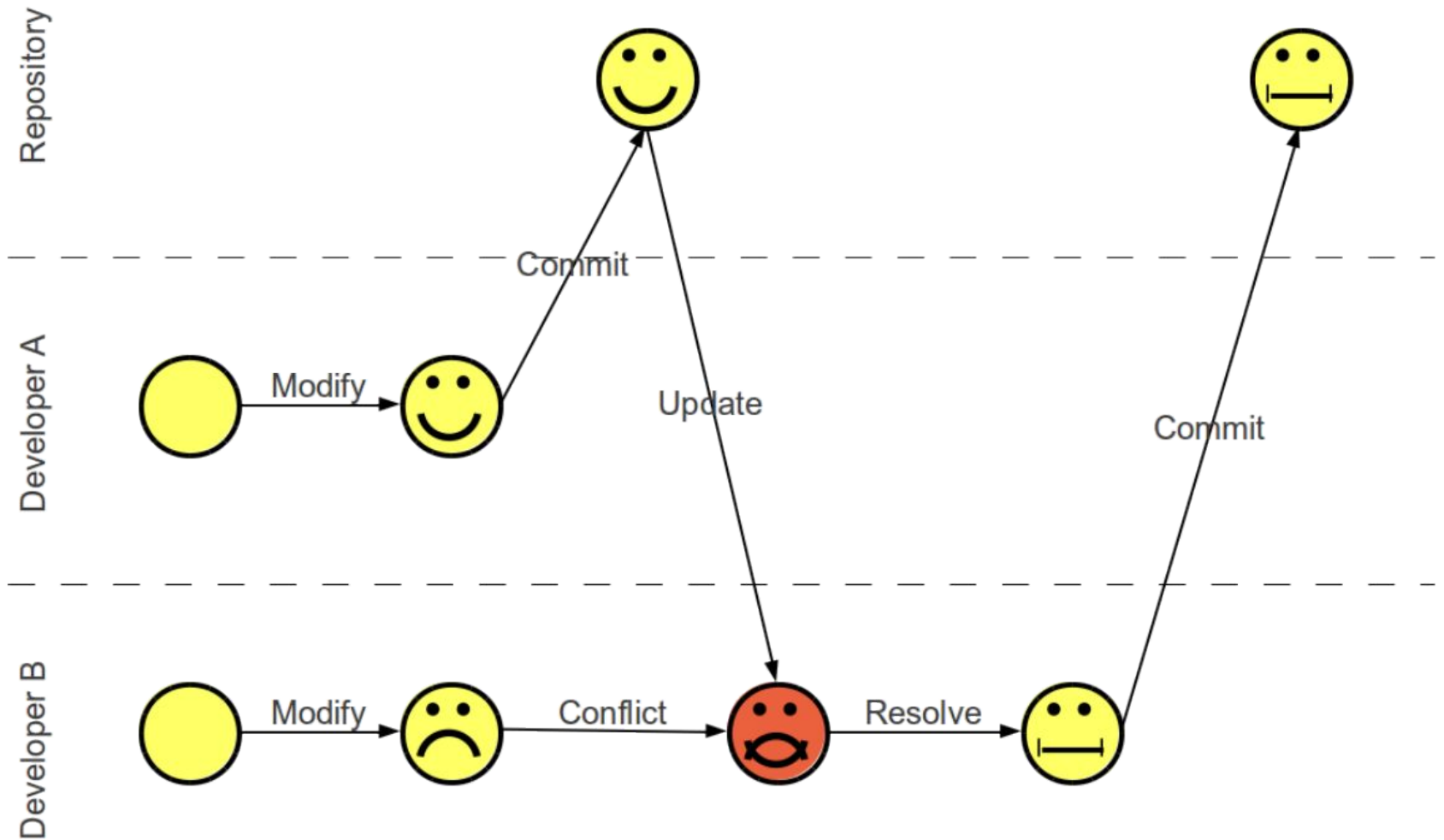
# Совместная работа



# Конфликты

- Если два разработчика изменяли один и тот же файл
- Становятся видны при update/pull
- Разрешаются тем разработчиком, который внес изменения (сделал commit/push) позднее
- Разрешать конфликты аккуратно – не затирать бездумно чужие изменения

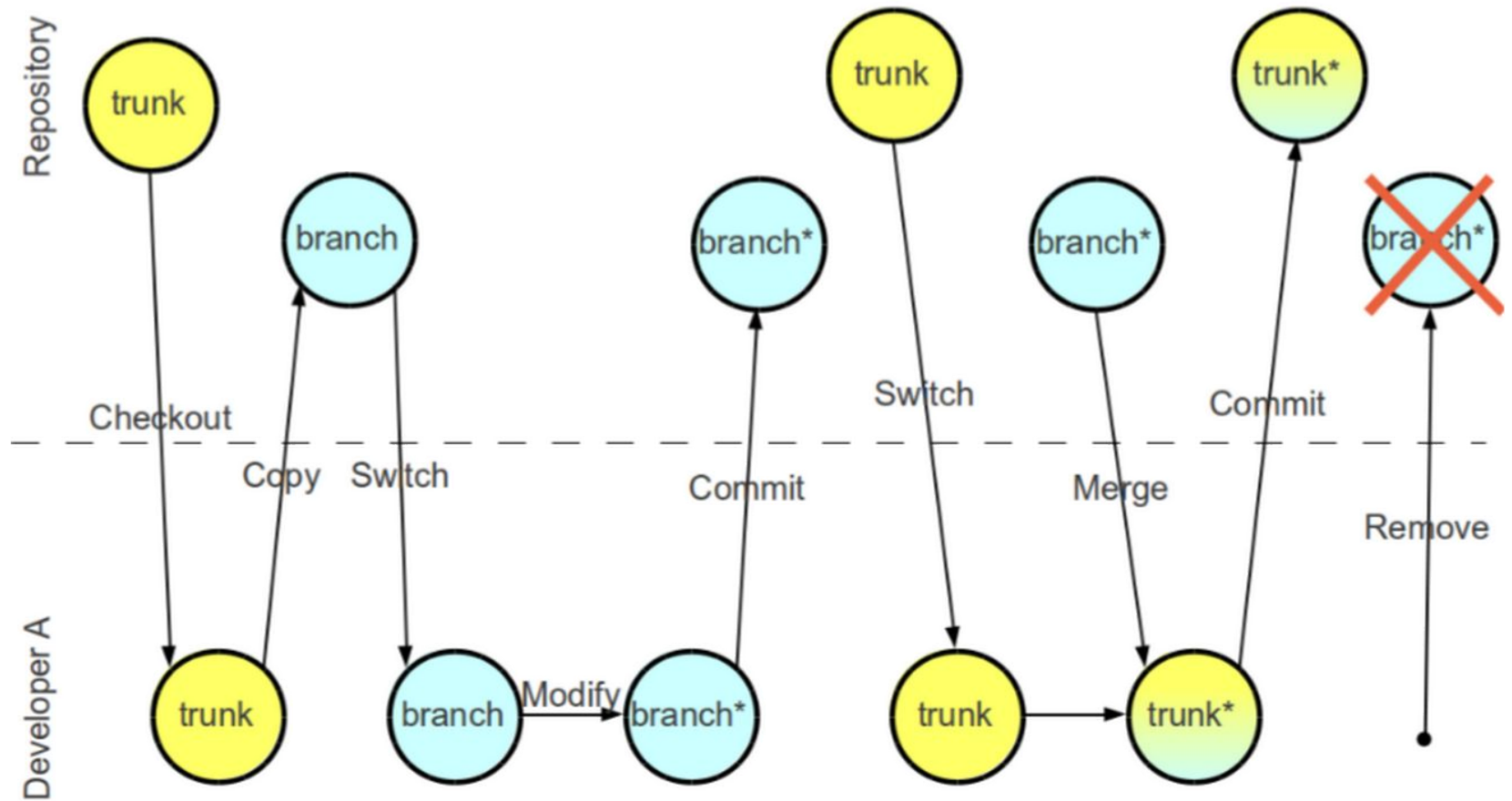
# Конфликты



# Ветки

- самостоятельная ветвь кода, изменения в которой делаются независимо от других ветвей
- управляется отдельными командами
- для внесения сложных и длительных изменений
- для сопровождения различных версий кода
  - исправление ошибок в старой версии
  - реализация фич в текущей версии
  - эксперименты для будущей версии
- ветка по-умолчанию: trunk, master, tip

# Ветки



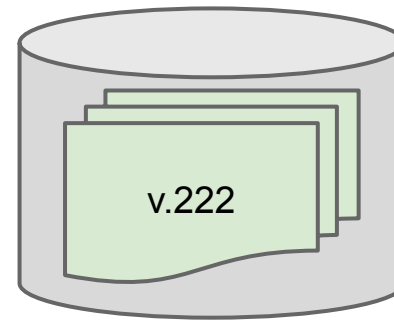
# Слияние (**merge**)

- слияние изменений из репозитория в локальную копию
- вливание изменений из одной ветви в другую
- слияние двух и более веток в одну

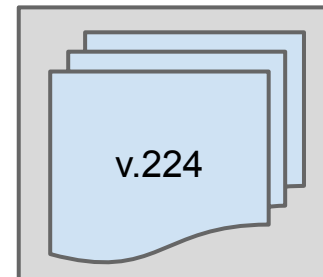
# Централизованная модель

**Примеры: SVN, CVS**

Центральный  
репозиторий

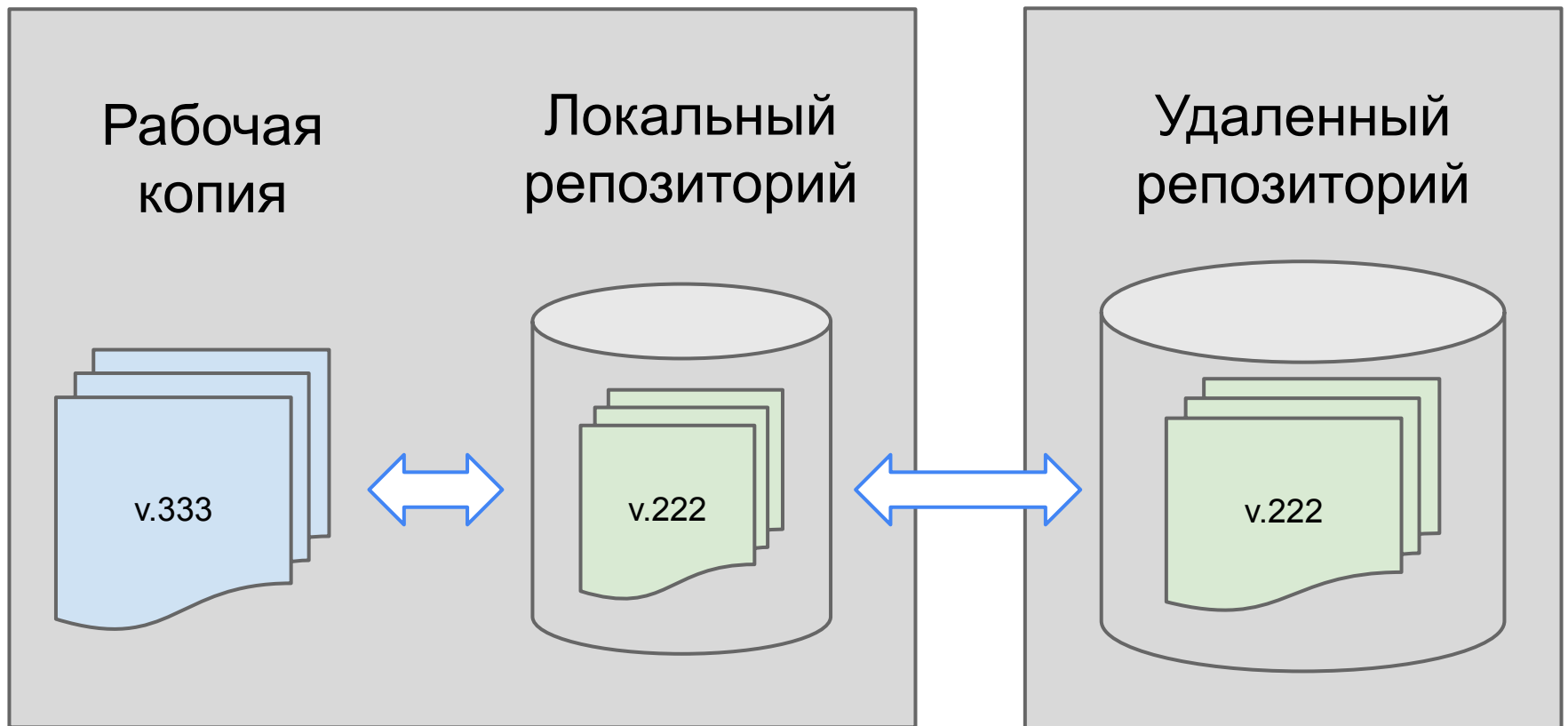


Рабочие  
копии



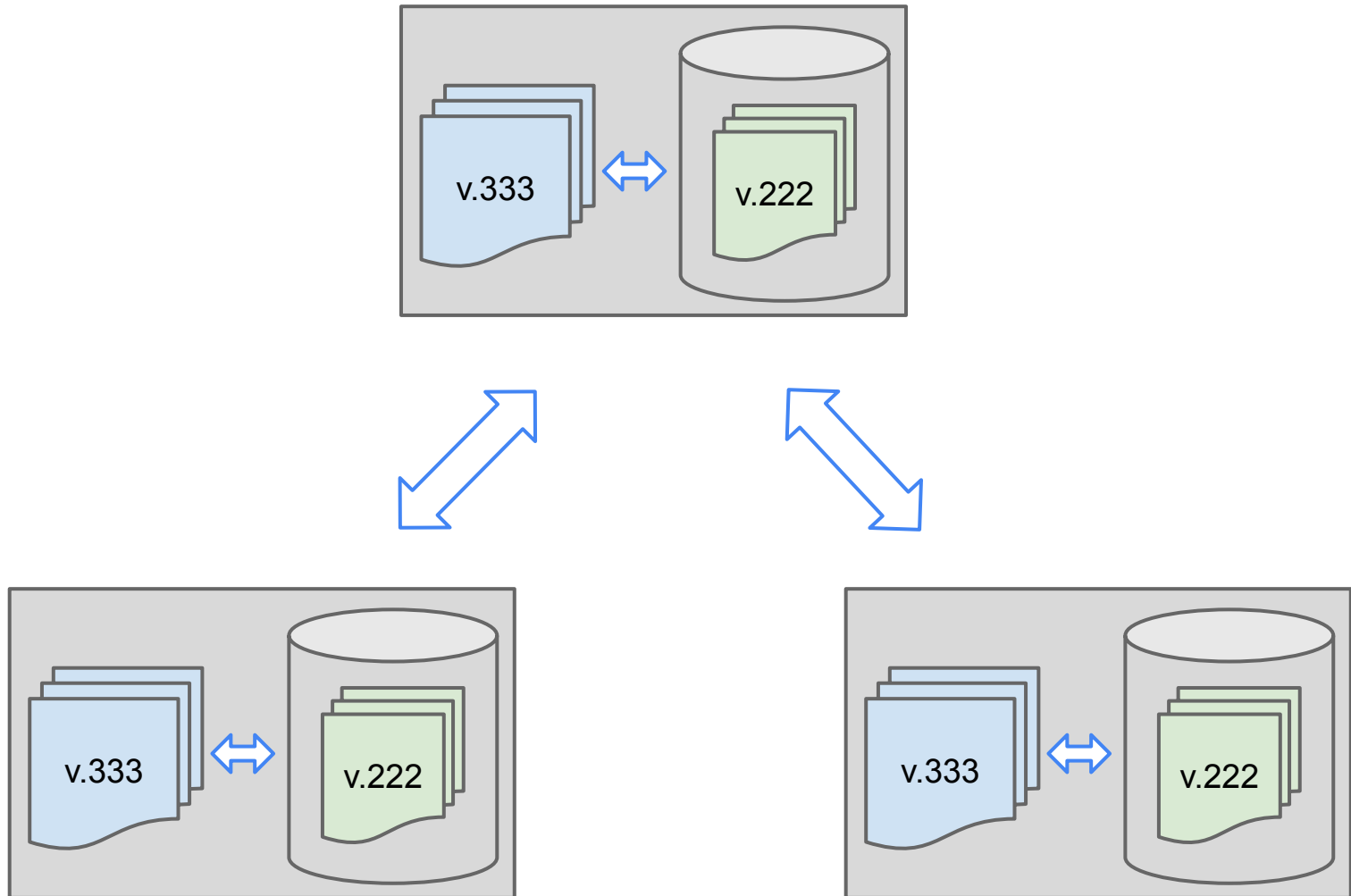
# Распределенная модель

**Примеры:** Git, Mercurial, Bazaar, Darcs





# Распределенная модель



# Почему распределенные?

## Централизованные

- Нужен доступ к репозиторию для коммита
- Нужен доступ к репозиторию для истории
- Каждый коммит немедленно виден всем, должен быть безопасным

## Распределенные

- Оффлайновые коммиты
- Оффлайновый доступ к истории (быстрее)
- Коммиты могут быть меньше и чаще, коммиты отделены от публикации изменений
- Сохранность



- распределенная VCS
- реализация: c++
- хорошая интеграция с IDE, ОС и др.

Используется:

- Google
- facebook
- Eclipse
- PostgreSQL

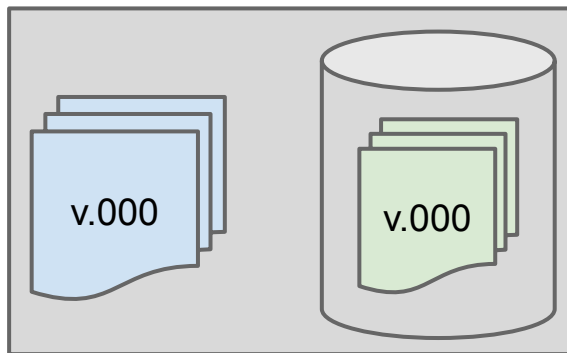


## Ревизия

- идентификатор: 1ef7872431f9c64908c732f0bcd4db5700b4cb70
  - можно указывать начальную часть
- ТЭГ: release-1.2
  - бывает локальный и общий

# Новый репозиторий

**git init**



- создает локальный репозиторий и новую рабочую копию

# Новый репозиторий

```
akoval@AK-PC /d/work/projects/школа разработчика/test
$ git init
Initialized empty Git repository in d:/work/projects/школа разработчика/test/.git/

akoval@AK-PC /d/work/projects/школа разработчика/test (master)
$
$ git log
fatal: bad default revision 'HEAD'

akoval@AK-PC /d/work/projects/школа разработчика/test (master)
$ git status
On branch master

Initial commit

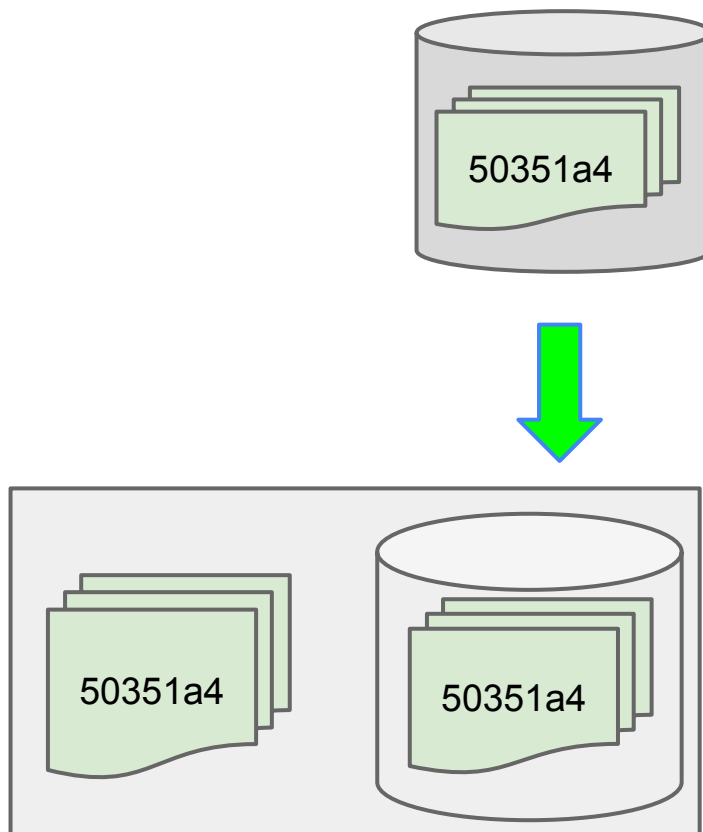
nothing to commit (create/copy files and use "git add" to track)

akoval@AK-PC /d/work/projects/школа разработчика/test (master)
$
```

# Копия репозитория

`git clone ssh://git@bitbucket.org:akoval/test.git`

`git clone https://akoval@bitbucket.org/akoval/test.git`



- создает локальный репозиторий и новую рабочую копию
- копирует содержимое удаленного репозитория в локальный

## Копия репозитория

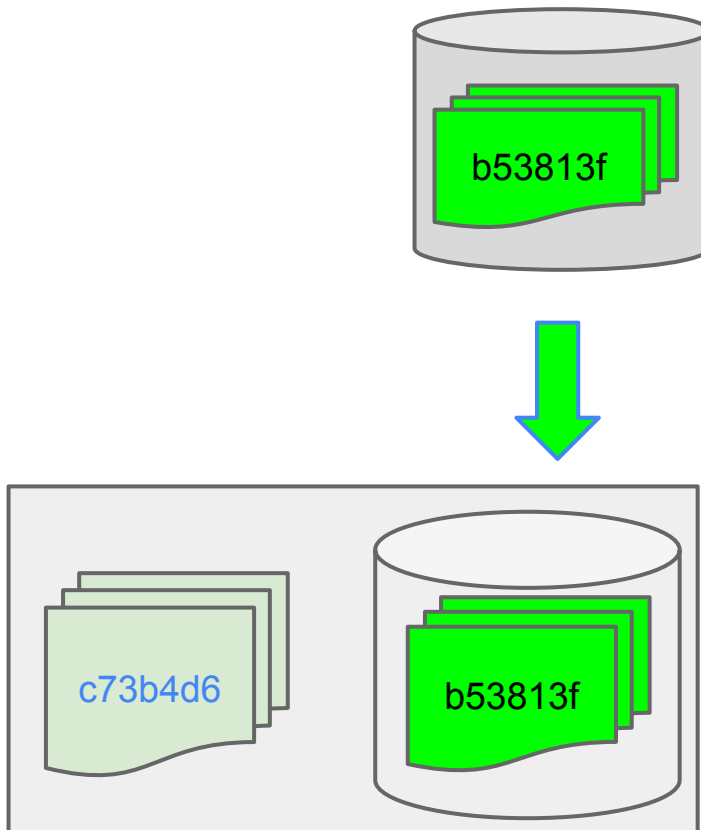
```
akoval@AK-PC /d/work/projects/школа разработчика/test (master)
$ git clone https://akoval@bitbucket.org/akoval/rest-example.git
Cloning into 'rest-example'...
Password for 'https://akoval@bitbucket.org':
remote: Counting objects: 69, done.
remote: Compressing objects: 100% (43/43), done.
remote: Total 69 (delta 14), reused 0 (delta 0)
Unpacking objects: 100% (69/69), done.
Checking connectivity... done.

akoval@AK-PC /d/work/projects/школа разработчика/test (master)
$
```



# Загрузка изменений

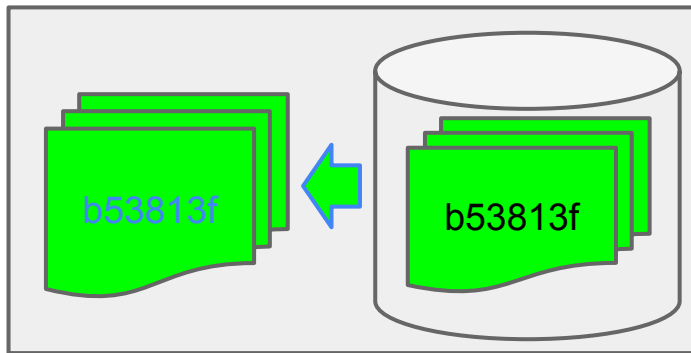
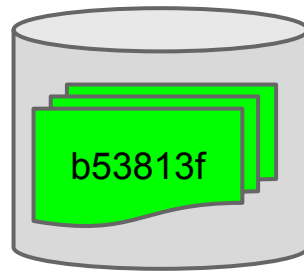
**git fetch**



- копирует изменения с удаленного репозитория в локальный
- ВОЗМОЖНЫ КОНФЛИКТЫ

# Применение изменений

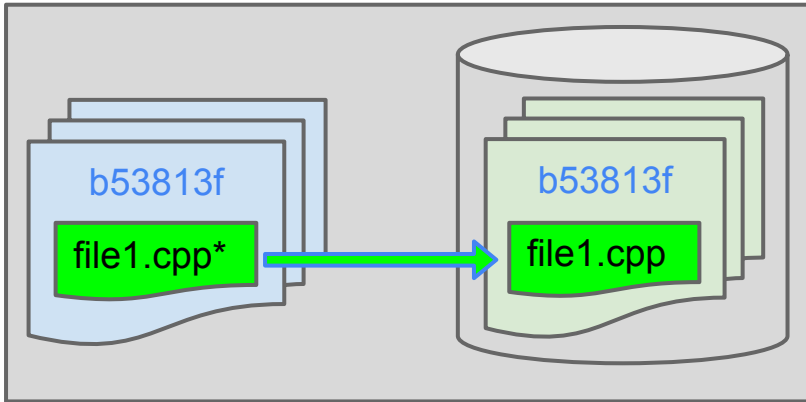
**git merge origin/master**



- применяет последние изменения из локального репозитория к рабочей копии
- ВОЗМОЖНЫ конфликты

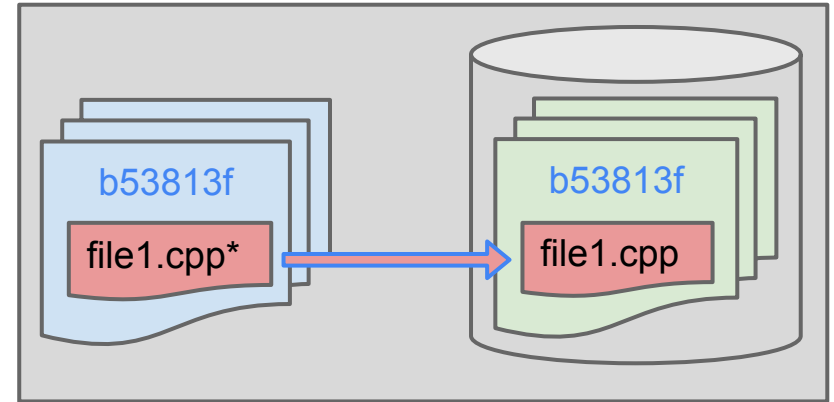
# Добавление/удаление файлов

**git add file1.cpp**



добавляет файлы к  
локальному  
репозитрию/коммиту

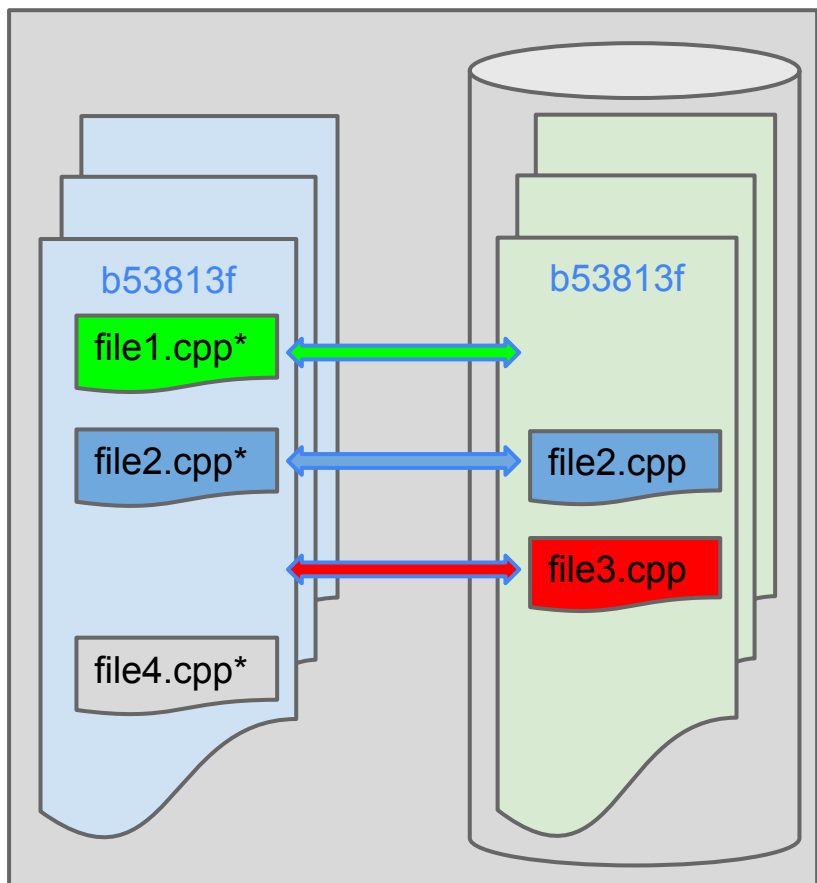
**git rm file1.cpp**



удаляет файлы из  
локального  
репозитория/коммита

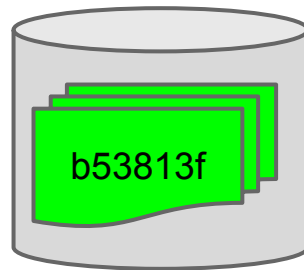
# Состояние рабочей копии

git status

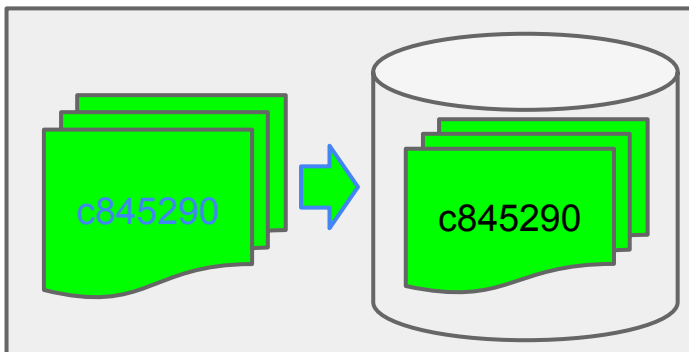


# Коммит изменений

**git commit -m "Refs #12345 Added NPE check"**

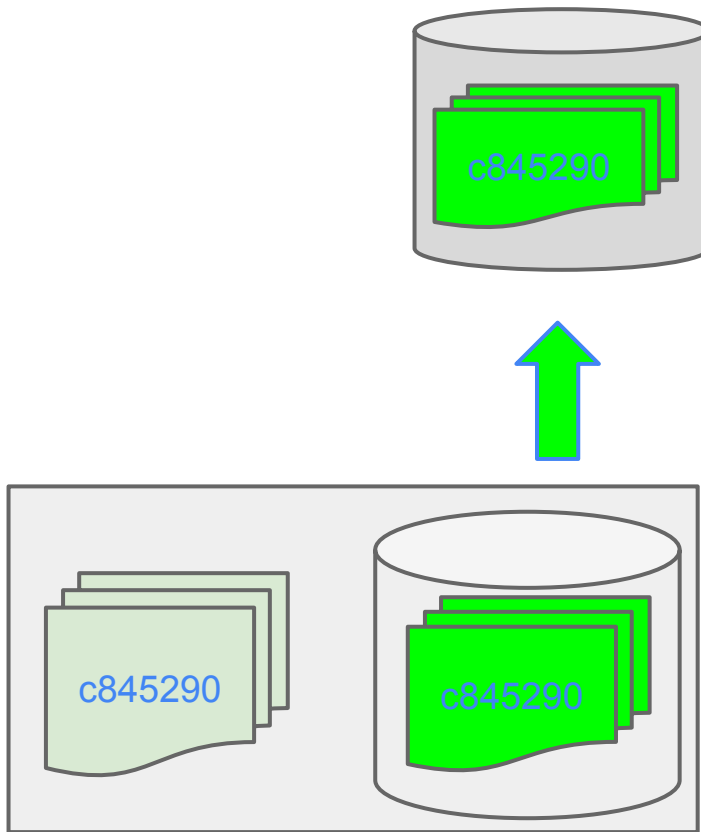


- сохраняет изменения из рабочей копии в локальный репозиторий
- возможны конфликты



# Выгрузка изменений

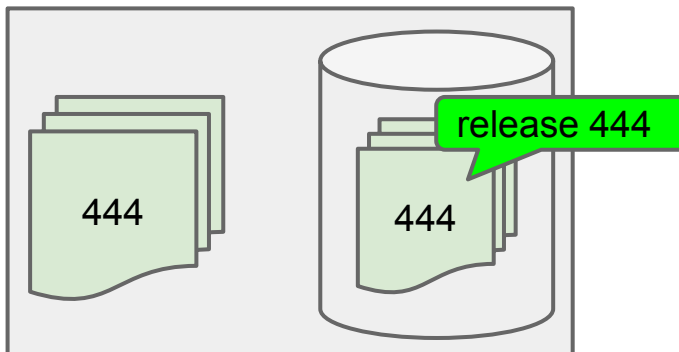
**git push**



- копирует изменения с локального репозитория в удаленный
- необходима синхронизация с удаленным репозиторием
- возможны конфликты

# Тэги

**git tag -m "release 444" 444**

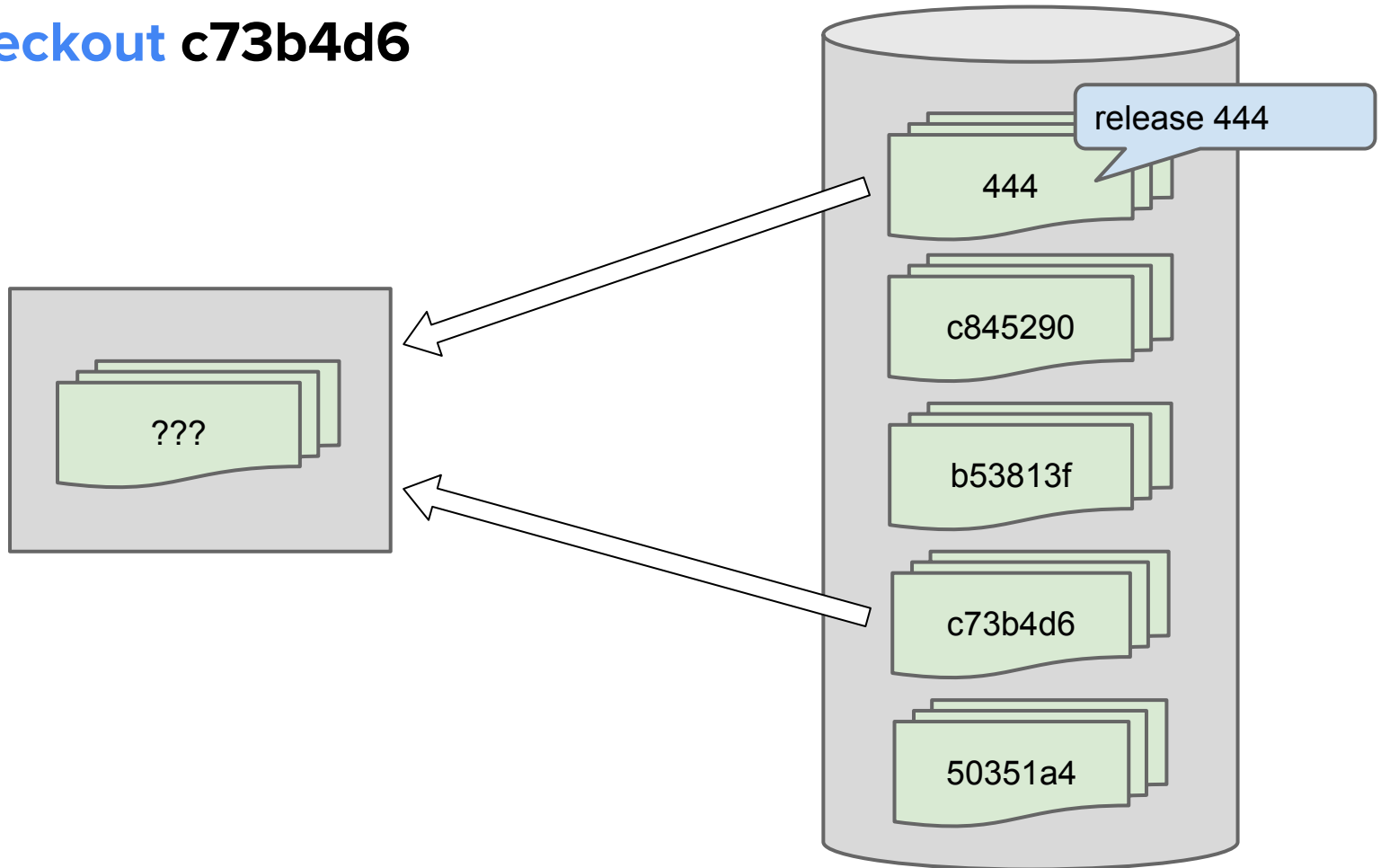


- пометка состояния репозитория в важный момент времени
  - релизы
  - синхронизация с другим репозиторием
- создается новый КОММИТ

# Версии

**git checkout 444**

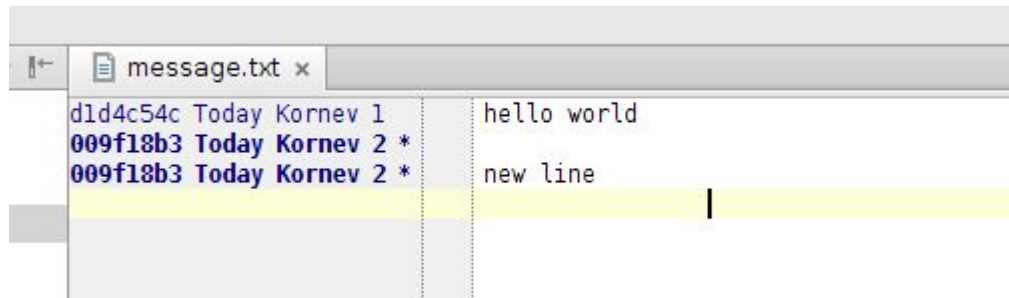
**git checkout c73b4d6**





# Построковая аннотация

**git show file.java**



```
message.txt x
d1d4c54c Today Kornev 1 | hello world
009f18b3 Today Kornev 2 * |
009f18b3 Today Kornev 2 * | new line
```