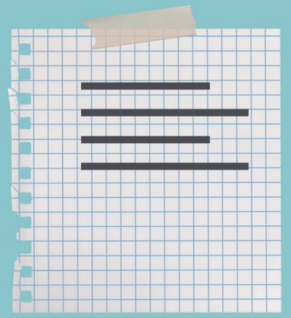
Лекция №5

Git

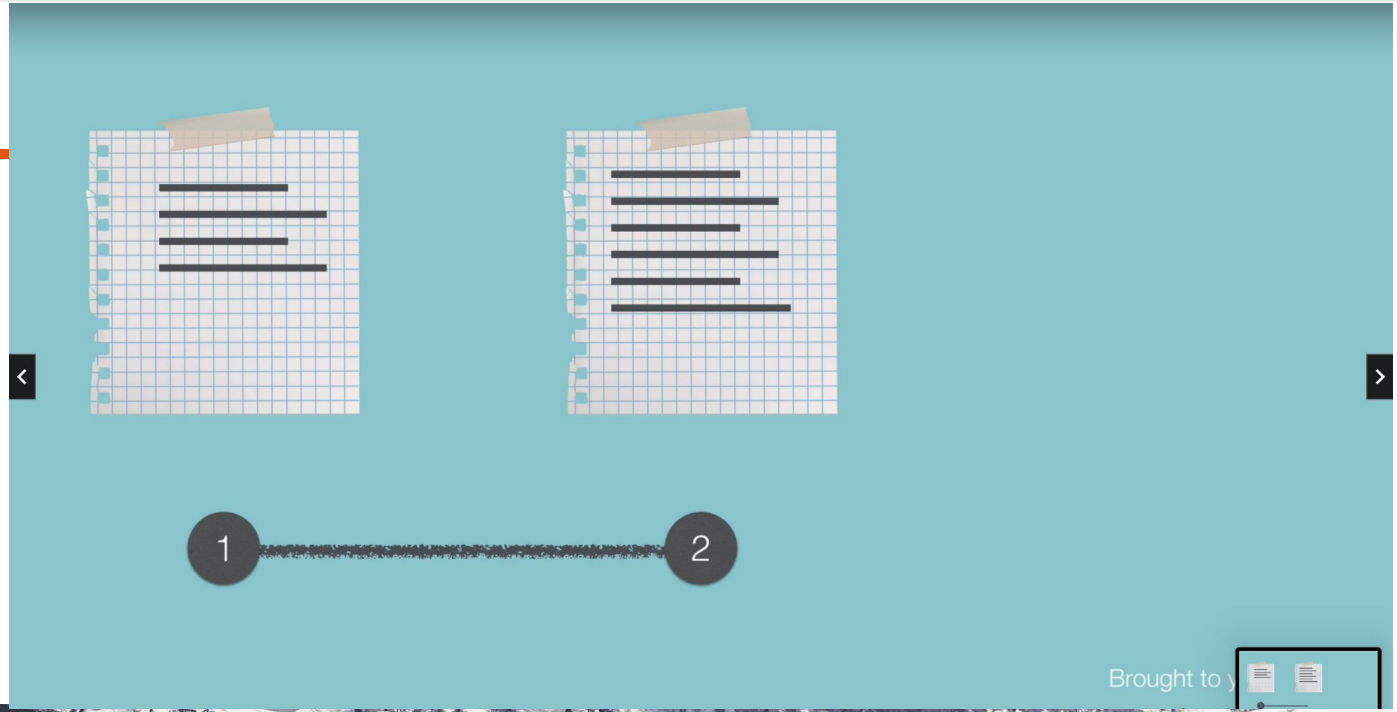
СОДЕРЖАНИЕ

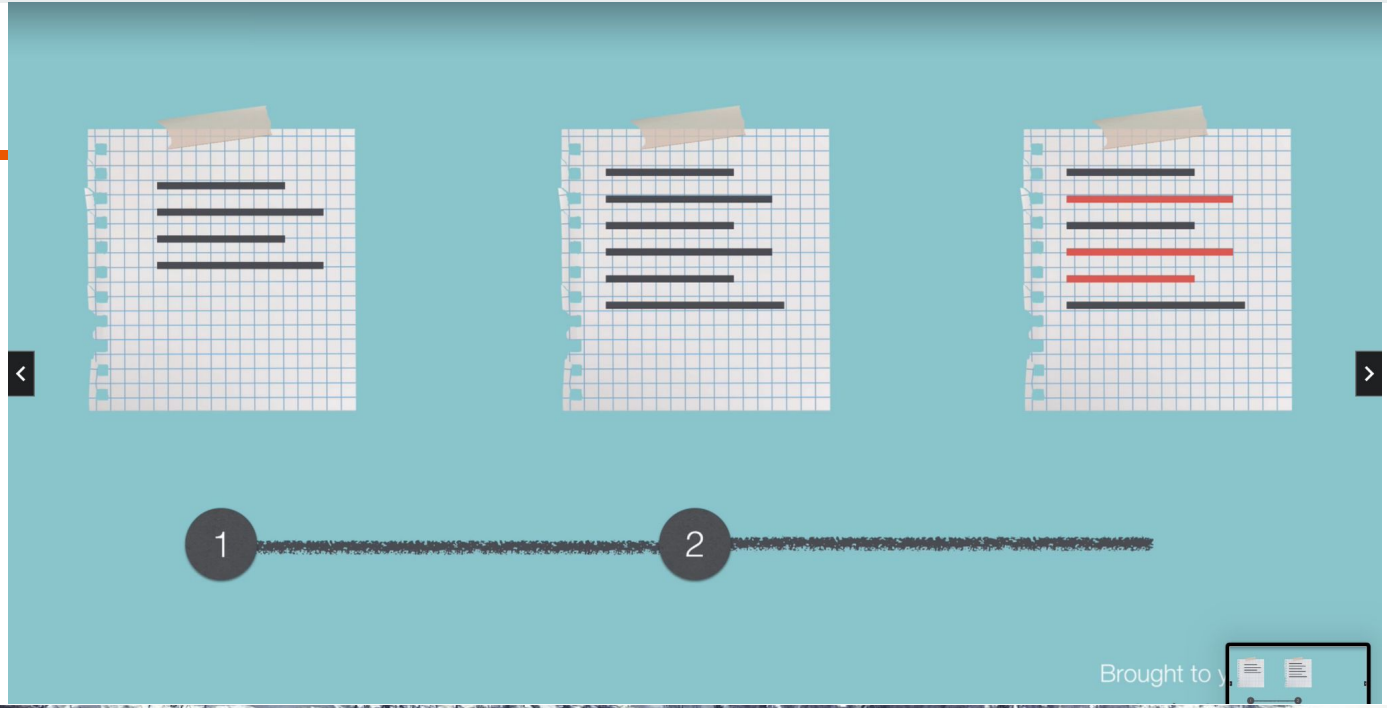

Интро в Git

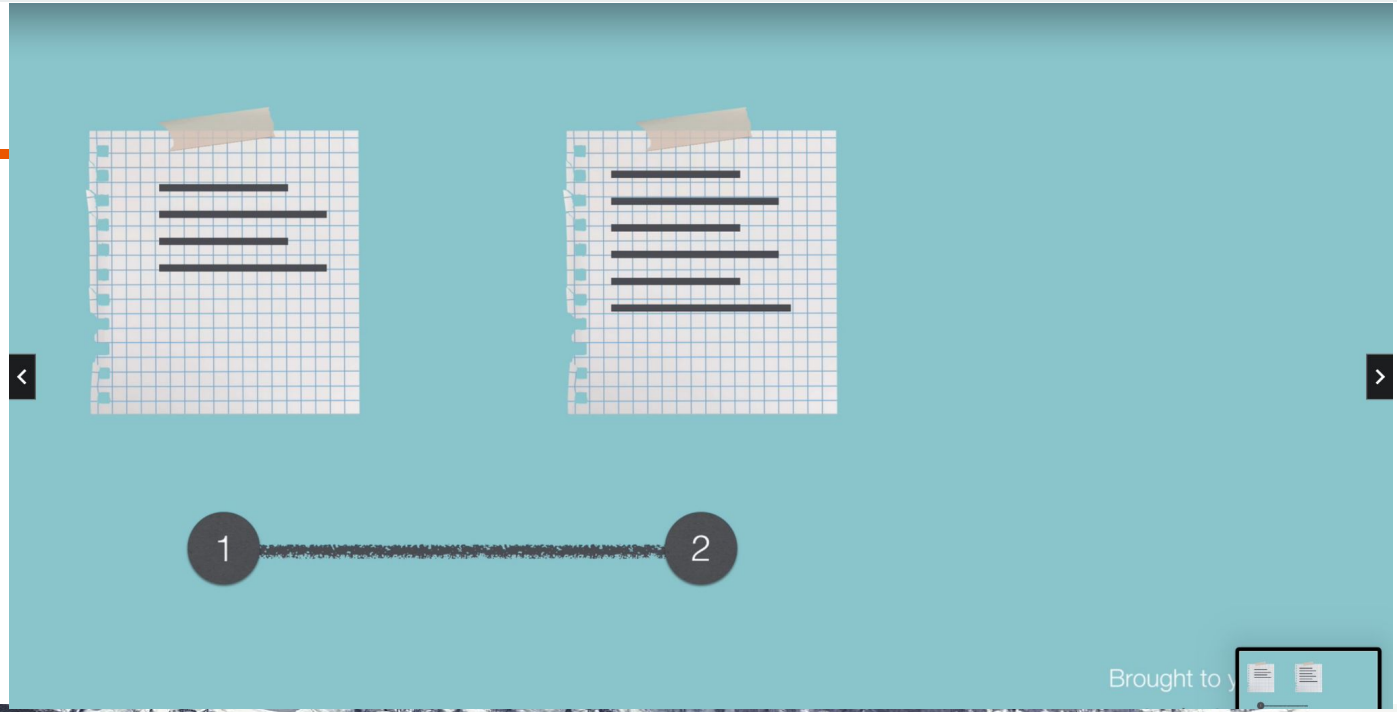


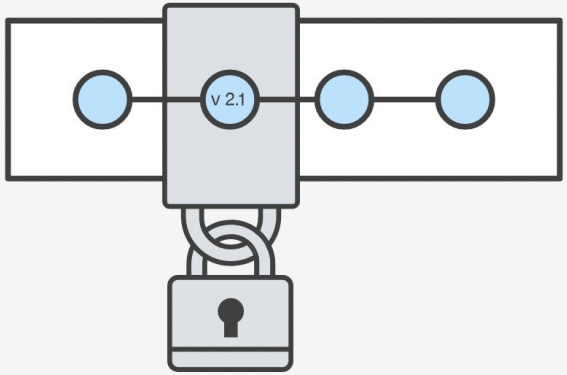
1

Brought to you by


1
2
Brought to y


1
2
Brought to


1
2
Brought to y



Что такое контроль версий?

Что предоставляет гит

1. Полная история изменений каждого файла за длительный период.
2. Ветвление и слияние.
3. Отслеживаемость. Возможность отслеживать каждое изменение, внесенное в программное обеспечение, и связывать его с ПО для управления проектами и отслеживания ошибок

- Пример

- Команды

GIT

- git add - добавляет содержимое рабочей директории в индекс (staging area) для последующего коммита.
- git commit - берёт все данные, добавленные в индекс с помощью git add, и сохраняет их слепок во внутренней базе данных, а затем сдвигает указатель текущей ветки на этот слепок.
- git rm - используется в Git для удаления файлов из индекса и рабочей директории. Она похожа на git add с тем лишь исключением, что она удаляет, а не добавляет файлы для следующего коммита.
- git reset - используется в основном для отмены изменений, убедитесь в серьезности своих намерений прежде чем использовать его.
- git push - вливание локальных изменений в удаленный репозиторий.
- git pull - вливание изменений из удаленного репозитория в локальный. Обмен данными обычно происходит с использованием протокола SSH.

GIT

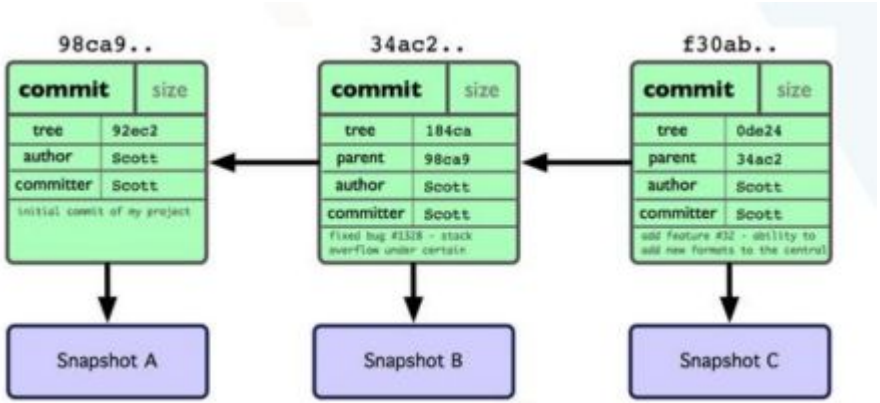
- git checkout - используется для копирования файлов из истории в рабочую директорию. Также она может использоваться для переключения между ветками.
- git branch - делает несколько больше, чем просто создаёт и удаляет ветки. Если вы выполните её без аргументов, то получите простой список имеющихся у вас веток.
- git status - основной инструмент, используемый для определения, какие файлы в каком состоянии находятся.

GIT

- git checkout - используется для копирования файлов из истории в рабочую директорию. Также она может использоваться для переключения между ветками.
- git branch - делает несколько больше, чем просто создаёт и удаляет ветки. Если вы выполните её без аргументов, то получите простой список имеющихся у вас веток.
- git status - основной инструмент, используемый для определения, какие файлы в каком состоянии находятся.

Ветвление

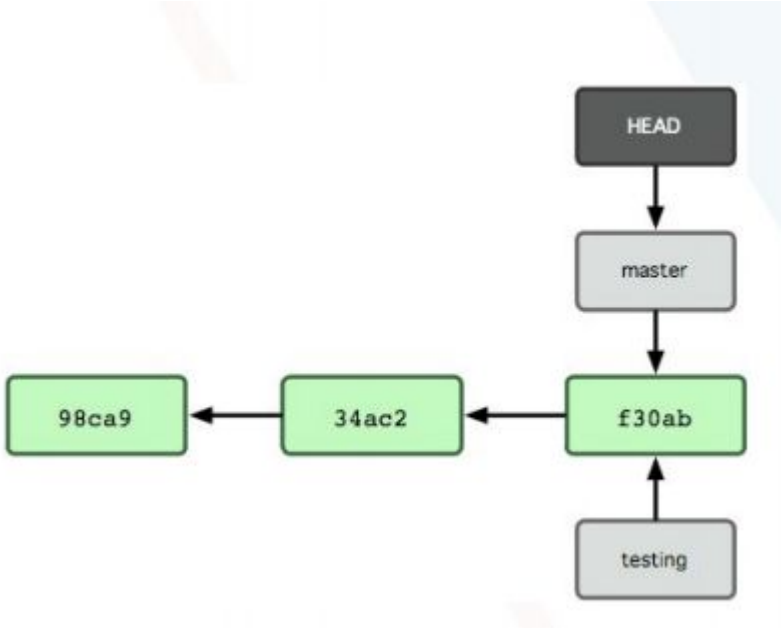
Если вы сделаете некоторые изменения и создадите новый коммит, то следующий коммит сохранит указатель на коммит, который шёл непосредственно перед ним. После следующих двух коммитов история может выглядеть так:



Ветвление

Ветка в Git'е — это просто легковесный подвижный указатель на один из этих коммитов. Ветка по умолчанию в Git'е называется master. Когда вы создаёте коммиты на начальном этапе, вам дана ветка master, указывающая на последний сделанный коммит. При каждом новом коммите она сдвигается вперёд автоматически.

Ветвление



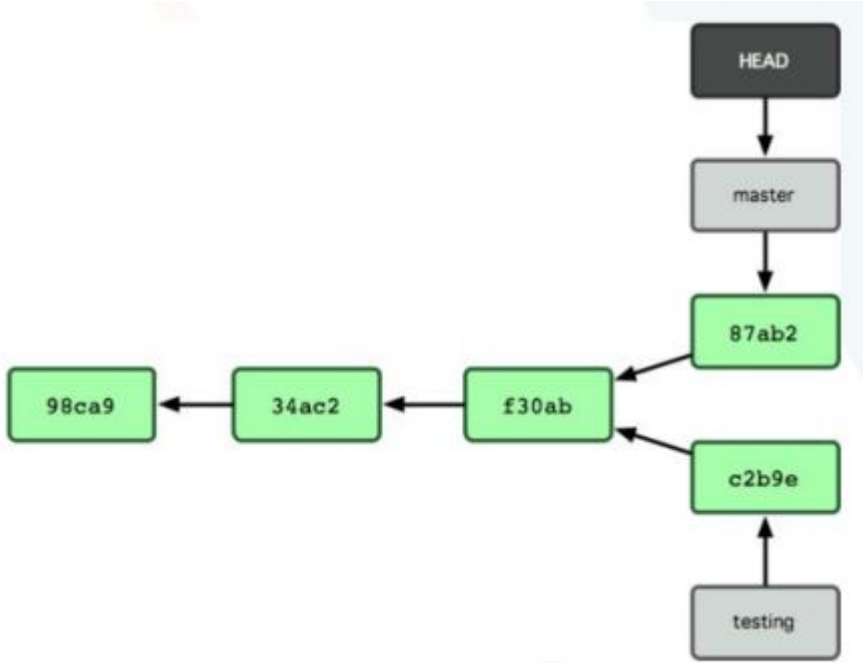
Ветвление

Чтобы перейти на существующую ветку, вам надо выполнить команду git checkout. Давайте перейдём на новую ветку testing:

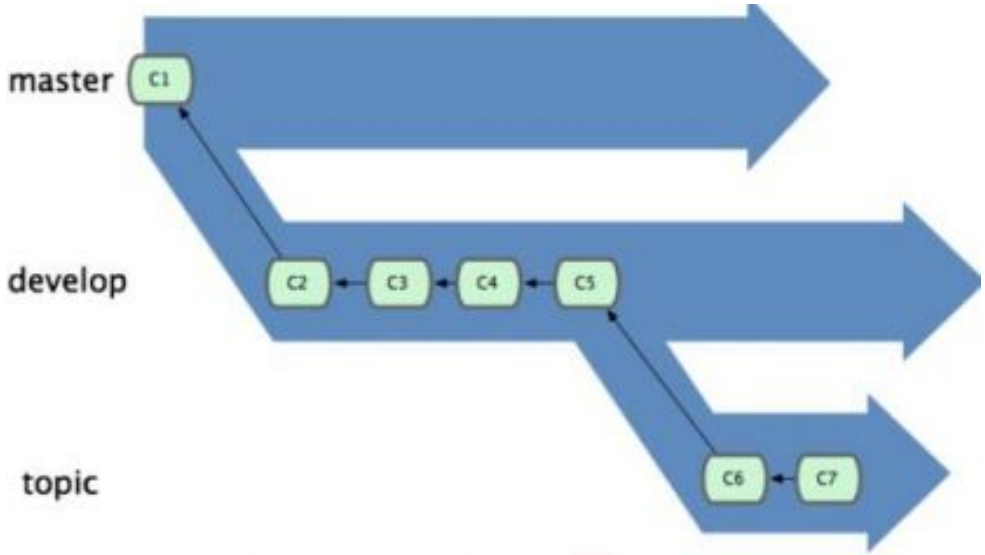
git checkout testing

Это действие передвинет HEAD так, чтобы тот указывал на ветку testing:

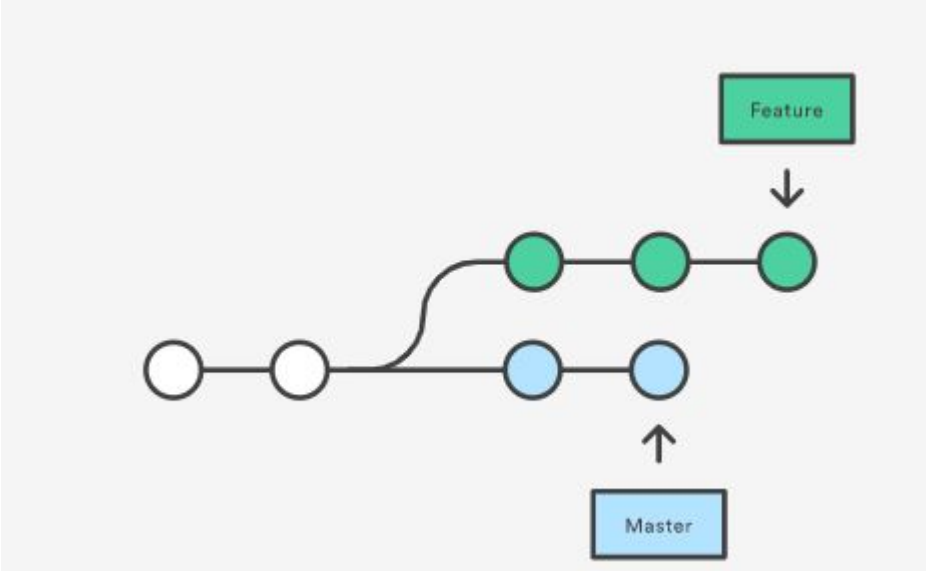
Ветвление



Ветвление



Merge vs rebase

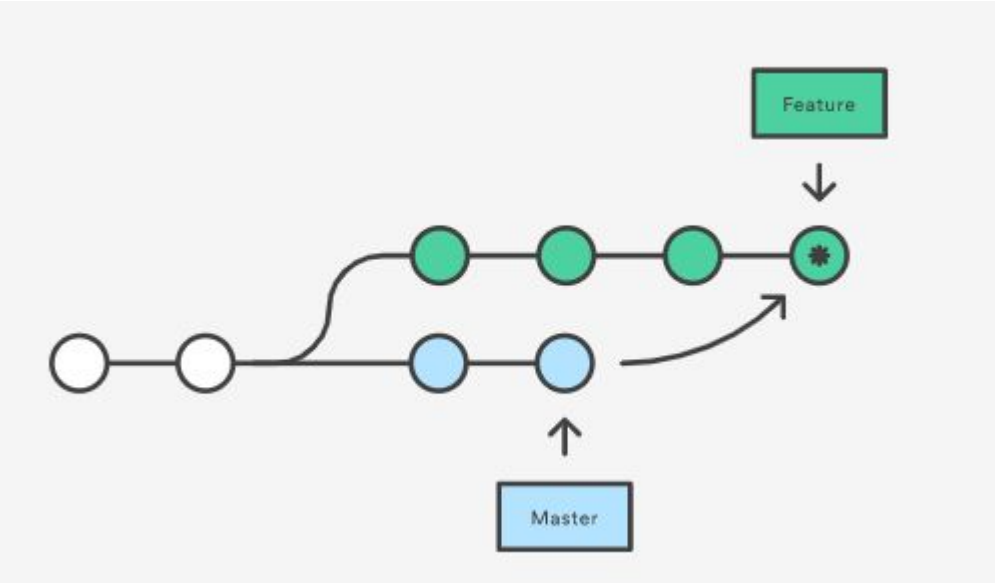


Merge

git checkout feature
git merge master
Или
git merge feature master

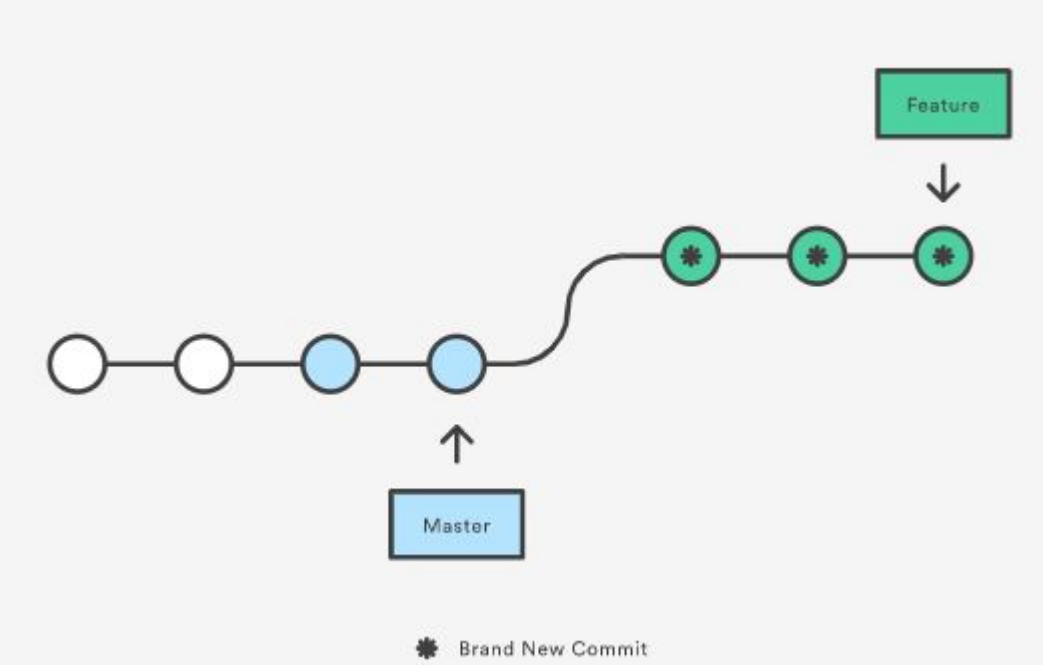
Merge

Эта операция создает в ветке feature новый «коммит слияния», связывающий истории обеих веток. Структура веток будет выглядеть так:



Rebase

git checkout feature
git rebase master



Rebase

Главное преимущество rebase — более чистая история проекта. Во-первых, эта команда устраняет ненужные коммиты слияния, необходимые для git merge

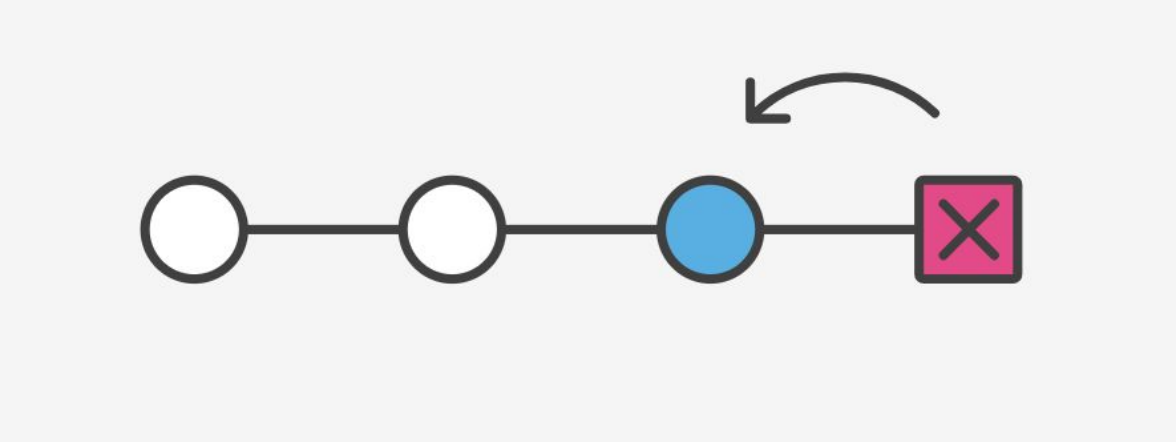
Cherry-pick

git cherry-pick

Команда git cherry-pick берёт изменения, вносимые одним коммитом, и пытается повторно применить их в виде нового коммита в текущей ветке. Эта возможность полезна в ситуации, когда нужно забрать парочку коммитов из другой ветки, а не сливать ветку целиком со всеми внесенными в нее изменениями.

Revert

git revert - Удаляет существующие commits



Git push

git push - отправляет ваши изменения на сервер

Git pull

git pull - стягивает изменения с сервера в ваше локальную копию

.gitignore

К шаблонам в файле .gitignore применяются следующие правила:

- Пустые строки, а также строки, начинающиеся с #, игнорируются.
- Можно использовать стандартные glob шаблоны.
- Можно заканчивать шаблон символом слэша (/) для указания каталога.
- Можно инвертировать шаблон, использовав восклицательный знак (!) в качестве первого символа.

Шаблон	Соответствие
boo?.tmp	book.tmp, boot.tmp, boo1.tmp и др.
boo[tk].tmp	boot.tmp и book.tmp
b*t.t?p	boot.tep, bat.tmp, bt.tnp и др.
[a-c]3.bat	a3.bat, b3.bat и c3.bat

Спасибо!