Ветвления и слияния

№ урока: 6 **Курс:** Основы использования Git

Средства обучения: Git console

Обзор, цель и назначение урока

Научиться работать с ветками, создавать, удалять и переключаться между ними. Выполнять слияние веток, решать конфликты слияния. Также научимся сохранять изменения в ветке не выполняя коммит.

Изучив материал данного занятия, учащийся сможет:

- Создавать и удалять ветки.
- Переключаться между ветками.
- Выполнять слияние или перебазирование.
- Решать конфликты слияния.

Содержание урока

1. Ветвление
2. Создание веток
3. Работа с ветками, переключение, удаление
4. Слияние веток (merge, rebase)
5. Решение конфликтов слияния

Резюме

- **Ветка** — это указатель на коммит. По умолчанию, имя основной ветки в Git — это master. Как только вы начнете создавать коммиты, ветка master будет всегда указывать на последний коммит. Каждый раз, при создании коммита, указатель ветки master будет передвигаться на следующий коммит автоматически.
- Команда **git branch <name>** - создает новую ветку, другими словами - новый указатель на текущий коммит.
- Указатель HEAD почти всегда указывает на одну из веток, при выполнении коммита – он будет добавлен к той ветке, где сейчас находится указатель HEAD.
- Для переключения указателя HEAD на другую ветку используется команда **git checkout <branch_name>**. Git дает возможность создать ветку и переключиться на нее одной командой - **git checkout -b <branch_name>**.
- Для удаления ветки необходимо убедиться, что HEAD сейчас на нее не указывает и после этого выполнить команду **git branch -d <branch_name>**.
- Посмотреть список веток, которые существуют в репозитории, можно также командой **git branch**.
- Для слияния используется команда **git merge <branch_name>** где <branch_name> - имя ветки изменения, которую вы хотите слить в текущую (на которую указывает HEAD при выполнении команды).
- Если коммит сливается с тем, который будет доступен двигаясь по истории прямо, то используется упрощенная процедура - просто перенося указатель ветки вперед, так как нет расхождений в изменениях. Такая операция называется **“fast-forward”.**
- Альтернативой операции merge является операция **rebase** (перебазирование). Она берет изменения текущей ветки и применяет их поверх всего, что есть в указанной.

CyberBionic Systematics ® 2020
19 Eugene Sverstyuk Str., 5 floor
Kyiv, Ukraine

Tel. 0 800 750 312
E-mail: edu@cbsystematics.com
Site: www.edu.cbsystematics.com | itvdn.com

Это работает следующим образом:

1. берется общий родительский снимок двух веток (текущей, и той, поверх которой вы выполняете rebase);
2. определяется дельта каждого коммита текущей ветки и сохраняется во временный файл;
3. текущая ветка устанавливается на последний коммит ветки, поверх которой вы выполняете перебазирование;
4. затем по очереди применяются дельты из временных файлов.

Результат такой же как при слиянии, но история коммитов получается чище. И, в этом случае, не будет конфликтов слияния.

- Недостаток использования rebase, по сравнению с merge. Если перемещать таким образов коммиты, которые уже синхронизированы с удаленным репозиторием, с которым работают другие разработчики – это может привести к серьезным проблемам. И сюда же можно отнести то, что хронология изменений не сохранится.
- После операции rebase необходимо переключится на ветку, в которую он выполнялся и выполнить слияние с веткой, которая перебазировалась. Например:
  1. git rebase master
  2. git checkout master
  3. git merge bugfix

  В результате изменения ветки bugfix применены к ветке master.
- **Конфликт слияния** происходит, когда одну и ту же часть кода меняли в обоих ветках, которые сливаются в одну. Если git самостоятельно не может решить какие изменения будут результатом слияния – необходимо делать это вручную.

Закрепление материала

- Какой командой можно удалить ветку?
- Какая команда позволяет создать ветку и сразу на нее переключиться?
- Какая операция является альтернативой команды слияния?
- Какие коммиты не следует перемещать командой git rebase?

Дополнительное задание

Задание
Создайте в репозитории новую ветку bugfix, сделайте коммит в ветке master, переключитесь на ветку bugfix, убедитесь, что в ней нет последнего коммита в master.

Самостоятельная деятельность учащегося

Задание 1
Создайте новую ветку (feature) и сделайте несколько коммитов в ней, затем переключитесь на ветку master. Выполните несколько коммитов в ветке master. Выполните слияние ветки feature в ветку master. Удалите ветку feature.

Задание 2
Повторите шаги предыдущего задания до слияния, выполните переключение на ветку feature и выполните перебазирование на ветку master.

Рекомендуемые ресурсы

Официальная документация Git
https://git-scm.com/docs

CyberBionic Systematics ® 2020
19 Eugene Sverstyuk Str., 5 floor
Kyiv, Ukraine

Tel. 0 800 750 312
E-mail: edu@cbsystematics.com
Site: www.edu.cbsystematics.com | itvdn.com

Книга Pro Git, Scott Chacon, Ben Straub
https://git-scm.com/book/ru/v2

CyberBionic Systematics ® 2020
19 Eugene Sverstyuk Str., 5 floor
Kyiv, Ukraine

Tel. 0 800 750 312
E-mail: edu@cbsystematics.com
Site: www.edu.cbsystematics.com | itvdn.com