**3.4 Практика по теме «Работа с удалёнными ветками»**

Описание

В этом курсе мы предусмотрели опциональные мини-задания для самостоятельной практики. Их не нужно сдавать на проверку. Нажав на кнопку внизу страницы, вы найдёте алгоритм и советы по выполнению этого задания с перечнем использованных команд. Постарайтесь решить задание самостоятельно, но если возникнет вопрос, то вы всегда сможете посмотреть в эту подсказку.

В предыдущем видео вы узнали, что такое ветки в Git-репозитории, как их создавать и переключаться между ними.

В этом задании вам необходимо, используя связанные локальный и удалённый репозитории:

1. создать новую ветку в локальном репозитории, внести в ветку изменения и закоммитить их;
2. создать новую ветку в удалённом репозитории через интерфейс веб-хостинга, внести в ветку изменения и закоммитить их на хостинге;
3. убедиться, что в локальном репозитории нет ветки из удалённого, как и в удалённом нет локальной;
4. скопировать себе все новые ветки из удалённого репозитория;
5. отправить локальную ветку на удалённый репозиторий;
6. посмотреть на все ветки вашего репозитория и то, к каким удалённым веткам они привязаны.

Для выполнения этого задания воспользуйтесь проектом из предыдущего задания для самостоятельного выполнения или создайте новый проект с использованием Git и отправьте его на удалённый репозиторий.

Эталонное решение

1. Создайте новую ветку в локальном репозитории и перейдите в неё одной командой - git checkout -b “название ветки X”. Держите в голове, что эта команда — просто удобная форма для двух: git branch “X” & git checkout “X”.
2. Внесите в ветку изменения и закоммитьте их.  
   Вы можете заметить, что, если сделать изменения без первого коммита, а потом попробовать переключиться обратно, скажем, на ветку master командой git checkout master, вы это сделаете без проблем. Но сделать коммит в новую ветку, а потом внести изменения без коммита и переключиться обратно в master не получится.
3. Создайте новую ветку в удалённом репозитории через интерфейс веб-хостинга. Внесите пару изменений и закоммитьте на хостинге.  
   Такой способ создания веток может пригодиться в будущем, если вы, например, захотите создать ветку по задаче, которую поставил руководитель. Такая ветка автоматически свяжется с задачей. Далее в этом модуле вы узнаете о pull/merge request веток. Сейчас скажем, что если завершить pull/merge request из веток, автоматически связанных с задачей, то завершатся и сами задачи.
4. Убедитесь, что в локальном репозитории нет ветки из удалённого, командой git branch. Далее убедитесь, что в удалённом репозитории не появилась ветка из локального. Чтобы репозитории узнали о существовании веток друг у друга, нужно поделиться ветками.
5. Выполните git pull в локальном репозитории, чтобы скопировать себе всё новое из удалённого репозитория, в частности новые ветки.
6. Отправьте локальную ветку на удалённый репозиторий. Имена одной ветки могут различаться локально и удалённо, поэтому нужно явно указать в локальном репозитории, какая ветка в локальном репозитории соответствует ветке в удалённом. Для этого используйте команду git push -u origin “название ветки X”*.*Аналогичную команду мы использовали в первом пуше в удалённый репозиторий из ветки master.
7. Посмотрите на все ветки вашего репозитория и то, к каким удалённым веткам они привязаны, с помощью команды git branch -vv.