ЛАБОРАТОРНА РОБОТА №4
РОБОТА З ВІДДАЛЕНИМ РЕПОЗИТОРІЄМ GIT

4.1 Мета роботи

Вивчити основні команди по роботі з віддаленим репозиторієм GIT.

4.2 Основні теоретичні відомості

Чтобы иметь возможность совместной работы над каким-либо Git-проектом, необходимо знать, как управлять удалёнными репозиториями. Удалённые репозитории — это модификации проекта, которые хранятся в интернете или ещё где-то в сети. Их может быть несколько, каждый из которых, как правило, доступен для вас либо только на чтение, либо на чтение и запись. Совместная работа включает в себя управление удалёнными репозиториями и помещение (push) и получение (pull) данных в и из них тогда, когда нужно обменяться результатами работы. Управление удалёнными репозиториями включает умение добавлять удалённые репозитории, удалять те из них, которые больше не действуют, умение управлять различными удалёнными ветками и определять их как отслеживаемые (tracked) или нет и прочее. Данный раздел охватывает все перечисленные навыки по управлению удалёнными репозиториями.

Отображение удалённых репозиториев

Чтобы просмотреть, какие удалённые серверы у вас уже настроены, следует выполнить команду git remote. Она перечисляет список имён-сокращений для всех уже указанных удалённых дескрипторов. Если вы склонировали ваш репозиторий, у вас должен отобразиться, по крайней мере, origin — это имя по умолчанию, которое Git присваивает серверу, с которого вы склонировали:

```
$ git clone git://github.com/schacon/ticgit.git
Initialized empty Git repository in /private/tmp/ticgit/.git/
remote: Counting objects: 595, done.
remote: Compressing objects: 100% (269/269), done.
remote: Total 595 (delta 255), reused 589 (delta 253)
Receiving objects: 100% (595/595), 73.31 KiB | 1 KiB/s, done.
Resolving deltas: 100% (255/255), done.
$ cd ticgit
$ git remote
```

```
origin
```

Чтобы посмотреть, какому URL соответствует сокращённое имя в Git, можно указать команде опцию -v:

```
$ git remote -v
origin  git://github.com/schacon/ticgit.git (fetch)
origin  git://github.com/schacon/ticgit.git (push)
```

Если у вас больше одного удалённого репозитория, команда покажет их все. Например, мой репозиторий Grit выглядит следующим образом.

```
$ cd grit
$ git remote -v
bakkdoor  git://github.com/bakkdoor/grit.git
cho45     git://github.com/cho45/grit.git
defunkt   git://github.com/defunkt/grit.git
koke      git://github.com/koke/grit.git
origin    git@github.com:mojombo/grit.git
```

Это означает, что мы легко можем получить изменения от любого из этих пользователей. Но, заметьте, что origin — это единственный удалённый сервер прописанный как SSH-ссылка, поэтому он единственный, в который я могу помещать свои изменения.

Добавление удалённых репозиториев

В предыдущих разделах мы упомянули и немного продемонстрировали добавление удалённых репозиториев, сейчас мы рассмотрим это более детально. Чтобы добавить новый удалённый Git-репозиторий под именем-сокращением, к которому будет проще обращаться, выполните git remote add [сокращение] [url]:

```
$ git remote
origin
$ git remote add pb git://github.com/paulboone/ticgit.git
$ git remote -v
origin  git://github.com/schacon/ticgit.git
pb  git://github.com/paulboone/ticgit.git
```

Теперь вы можете использовать в командной строке имя pb вместо полного URL. Например, если вы хотите извлечь (fetch) всю

информацию, которая есть в репозитории Павла, но нет в вашем, вы можете выполнить git fetch pb:

```
$ git fetch pb
remote: Counting objects: 58, done.
remote: Compressing objects: 100% (41/41), done.
remote: Total 44 (delta 24), reused 1 (delta 0)
Unpacking objects: 100% (44/44), done.
From git://github.com/paulboone/ticgit
 * [new branch]      master     -> pb/master
 * [new branch]      ticgit     -> pb/ticgit
```

Ветка master Павла теперь доступна локально как pb/master. Вы можете слить (merge) её в одну из своих веток или перейти на эту ветку, если хотите её проверить.

Fetch и Pull

Как вы только что узнали, для получения данных из удалённых проектов, следует выполнить:

```
$ git fetch [имя удал. сервера]
```

Данная команда связывается с указанным удалённым проектом и забирает все те данные проекта, которых у вас ещё нет. После того как вы выполнили команду, у вас должны появиться ссылки на все ветки из этого удалённого проекта. Теперь эти ветки в любой момент могут быть просмотрены или слиты. (В главе 3 мы перейдём к более детальному рассмотрению, что такое ветки и как их использовать.)

Когда вы клонируете репозиторий, команда clone автоматически добавляет этот удалённый репозиторий под именем origin. Таким образом, git fetch origin извлекает все наработки, отправленные (push) на этот сервер после того, как вы склонировали его (или получили изменения с помощью fetch). Важно отметить, что команда fetch забирает данные в ваш локальный репозиторий, но не сливает их с какими-либо вашими наработками и не модифицирует то, над чем вы работаете в данный момент. Вам необходимо вручную слить эти данные с вашими, когда вы будете готовы.

Если у вас есть ветка, настроенная на отслеживание удалённой ветки, то вы можете использовать команду git pull. Она автоматически извлекает и затем сливает данные из удалённой ветки в вашу текущую ветку. Этот способ может для вас оказаться более простым или более удобным. К тому же по умолчанию команда git clone автоматически настраивает вашу локальную ветку master на отслеживание удалённой ветки master на сервере, с которого вы клонировали (подразумевается, что на удалённом сервере есть ветка master). Выполнение git pull, как

правило, извлекает (fetch) данные с сервера, с которого вы изначально склонировали, и автоматически пытается слить (merge) их с кодом, над которым вы в данный момент работаете.

Push

Когда вы хотите поделиться своими наработками, вам необходимо отправить (push) их в главный репозиторий. Команда для этого действия простая:

git push [удал. сервер] [ветка].

Чтобы отправить вашу ветку master на сервер origin (повторимся, что клонирование, как правило, настраивает оба этих имени автоматически), вы можете выполнить следующую команду для отправки наработок на сервер:

$ git push origin master

Эта команда срабатывает только в случае, если вы клонировали с сервера, на котором у вас есть права на запись, и если никто другой с тех пор не выполнял команду push. Если вы и кто-то ещё одновременно клонируете, затем он выполняет команду push, а затем команду push выполняете вы, то ваш push точно будет отклонён. Вам придётся сначала вытянуть (pull) их изменения и объединить с вашими. Только после этого вам будет позволено выполнить push.

Инспекция удалённого репозитория

Если хотите получить побольше информации об одном из удалённых репозиториев, вы можете использовать команду git remote show [удал. сервер]. Если вы выполните эту команду с некоторым именем, например, origin, вы получите что-то подобное:

```
$ git remote show origin
* remote origin
  URL: git://github.com/schacon/ticgit.git
  Remote branch merged with 'git pull' while on branch master
    master
  Tracked remote branches
    master
    ticgit
```

Она выдаёт URL удалённого репозитория, а также информацию об отслеживаемых ветках. Эта команда любезно сообщает вам, что если вы, находясь на ветке master, выполните git pull, ветка master с удалённого сервера будет автоматически влита в вашу сразу после получения всех необходимых данных. Она также выдаёт список всех полученных ею ссылок.

Если вы используете Git более интенсивно, вы можете увидеть гораздо большее количество информации от git remote show:

```
$ git remote show origin
* remote origin
  URL: git@github.com:defunkt/github.git
  Remote branch merged with 'git pull' while on branch issues
    issues
  Remote branch merged with 'git pull' while on branch master
    master
  New remote branches (next fetch will store in remotes/origin)
    caching
  Stale tracking branches (use 'git remote prune')
    libwalker
    walker2
  Tracked remote branches
    acl
    apiv2
    dashboard2
    issues
    master
    postgres
  Local branch pushed with 'git push'
    master:master
```

Данная команда показывает какая именно локальная ветка будет отправлена на удалённый сервер по умолчанию при выполнении git push. Она также показывает, каких веток с удалённого сервера у вас ещё нет, какие ветки всё ещё есть у вас, но уже удалены на сервере. И для нескольких веток показано, какие удалённые ветки будут в них влиты при выполнении git pull.

Удаление и переименование удалённых репозиториев

Для переименования ссылок в новых версиях Git'a можно выполнить git remote rename, это изменит сокращённое имя, используемое для удалённого репозитория. Например, если вы хотите переименовать pb в paul, вы можете сделать это следующим образом:

```
$ git remote rename pb paul
$ git remote
origin
paul
```

Стоит упомянуть, что это также меняет для вас имена удалённых веток. То, к чему вы обращались как pb/master, стало paul/master.

Если по какой-то причине вы хотите удалить ссылку (вы сменили сервер или больше не используете определённое зеркало, или, возможно, контрибьютор перестал быть активным), вы можете использовать git remote rm:

$ git remote rm paul
$ git remote
origin

4.3 Завдання на лабораторну роботу

1. Зарегистрируйтесь на Bitbucket (https://bitbucket.org) (рис.1)

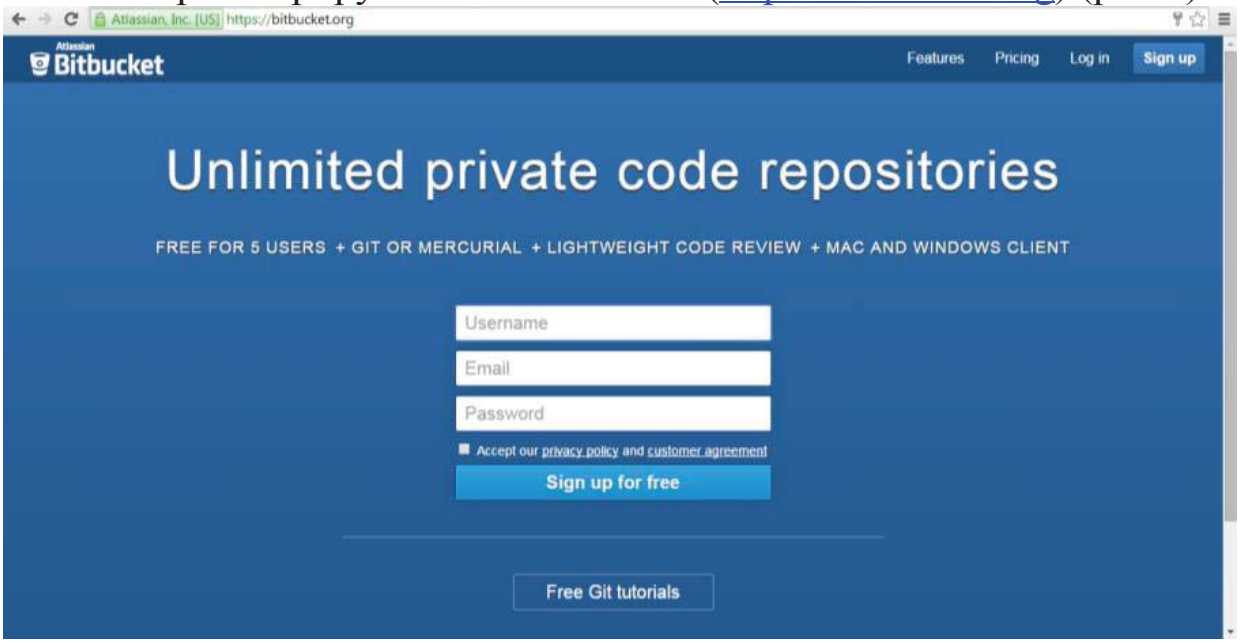

Рисунок 1. Регистрация на Bitbucket

2. Установите и настройте Git-клиент (для Windows – msysgit (скачать можно по адресу http://msysgit.github.io/)).

3. После установки, запускаем "Git Bash" и вводим следующие команды (для вставки используйте клавиши Shift+Insert):

- задаём глобальное имя и e-mail:

git config --global user.email "my_email@mail.com"
git config --global user.name "my_nickname"

- устанавливаем метод по-умолчанию для push в simple.

git config --global push.default simple

Краткое описание методов push:

nothing - "git push" не будет работать без явно указанных имени и ветки репозитория

current - обновляется удалённая ветка, имя которой совпадает с текущей веткой

upstream - обновляется upstream ветка, т.е. ветка, из которой выполняется "pull"

simple - то же, что и "upstream", только дополнительно выполняется проверка на совпадение имён веток (самый безопасный метод)

matching - обновляет все ветки по имени (метод по-умолчанию в Git < 2.0)

Если не устанавливать метод, то при "git push" будут появляться warning сообщения.

4. Копирование проекта из удаленного репозитория

Создайте fork указанного проекта и скачайте проект на локальную машину двумя способами в разные папки - с помощью командной строки и с помощью qt creator.

Внимание! Для доступа к удаленному репозиторию через proxy-server пропишите следующую команду:

git config --global http.proxy http://proxyuser:proxypass@10.0.2.1:8080

4.1. Через командную строку

Запускаем "Git Bash" и переходим в директорию для будущего проекта:

cd /c/Projects/My_project (или cd C:\\Projects\\My_project)

Затем, в корневой папке проекта, выполняем следующие команды:

- Инициализируем пустой локальный репозиторий.

git init

- Копируем файлы в локальный репозиторий, предварительно сделав fork из репозитория https://bitbucket.org/tata007/zntulabs (рис. 2)

git clone https://usernamebitbucket@bitbucket.org/tata007/zntulabsfork.git

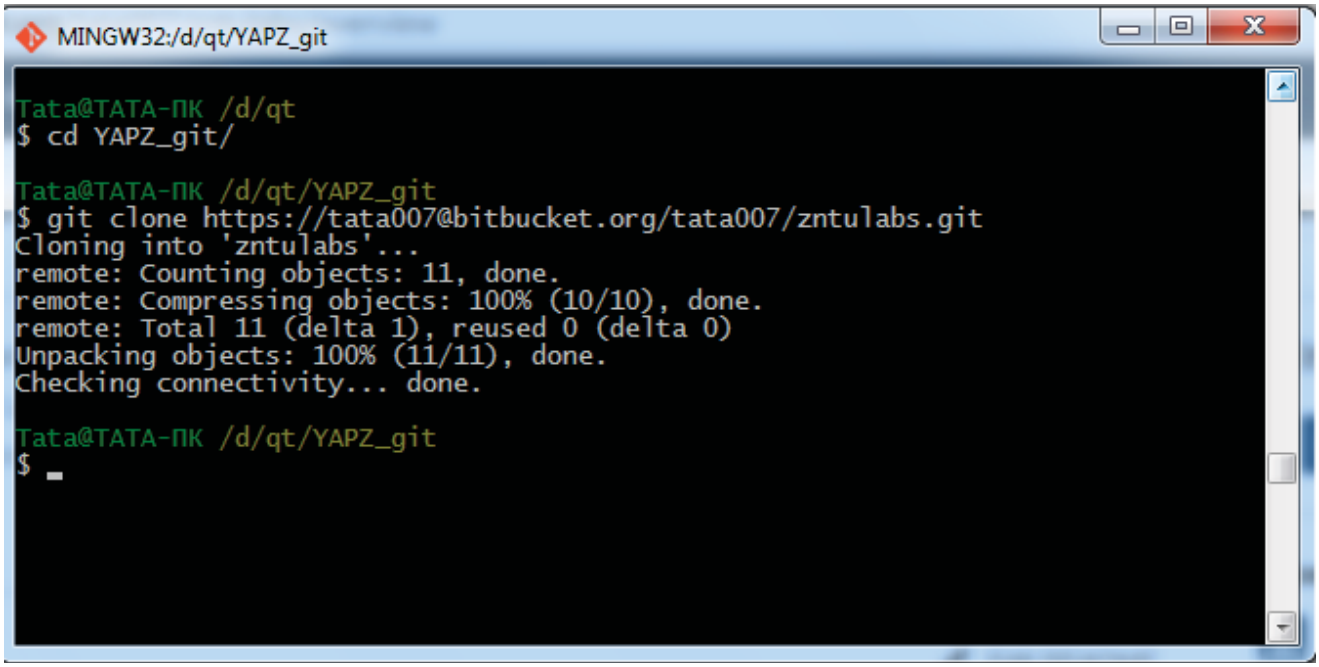

Рисунок 2. Копирование проекта из удаленного репозитория через командную строку

4.2. С использованием qt creator

При создании проекта выбираете вариант "Импортировать проект / Git Repository Clone" (рис. 3).

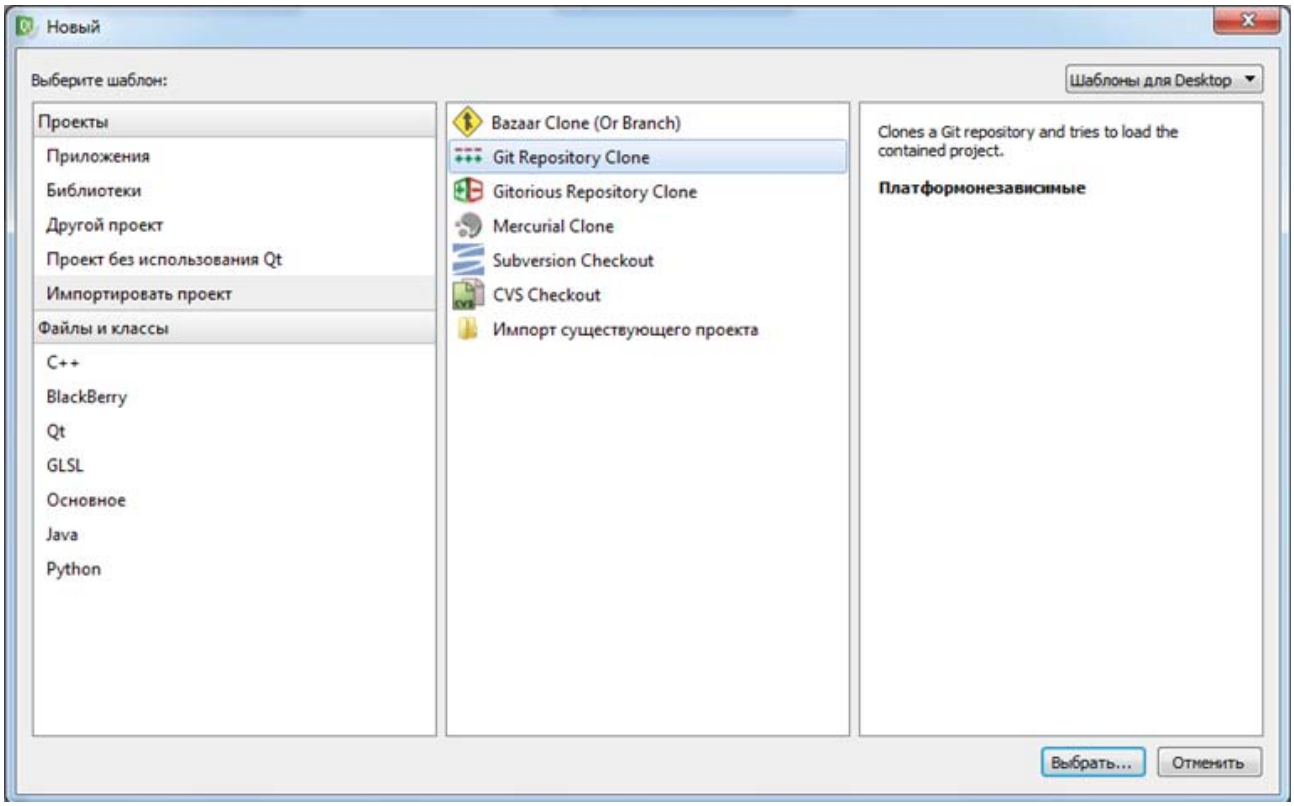

Рисунок 3. Выбор шаблона при копировании проекта из удаленного репозитория в qt creator

Следующий шаг – выбор репозитория (клонируйте из созданного fork’a https://bitbucket.org/tata007/zntulabs) (рис. 4).

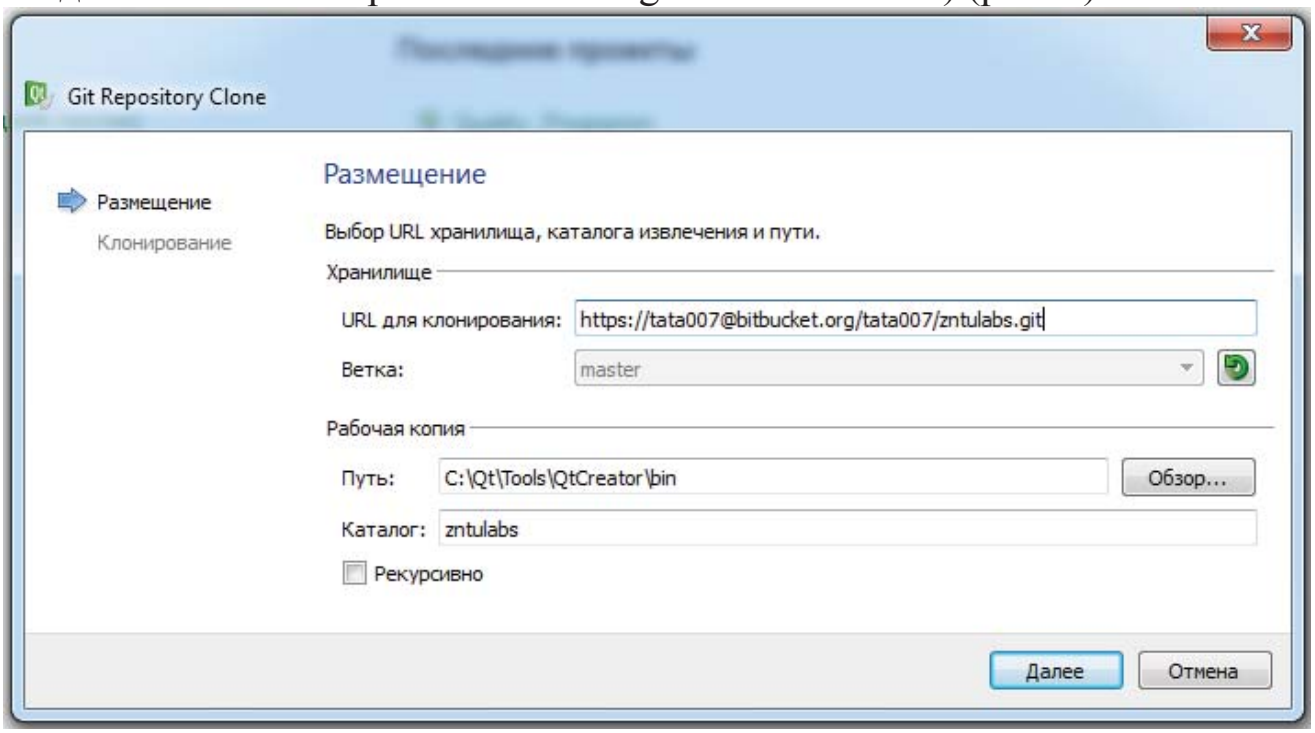

Рисунок 4. Выбор хранилища при копировании проекта из удаленного репозитория в qt creator

5. Для каждой версии проекта создать свою ветку, внести изменения в обе версии и зафиксировать их commit'ом

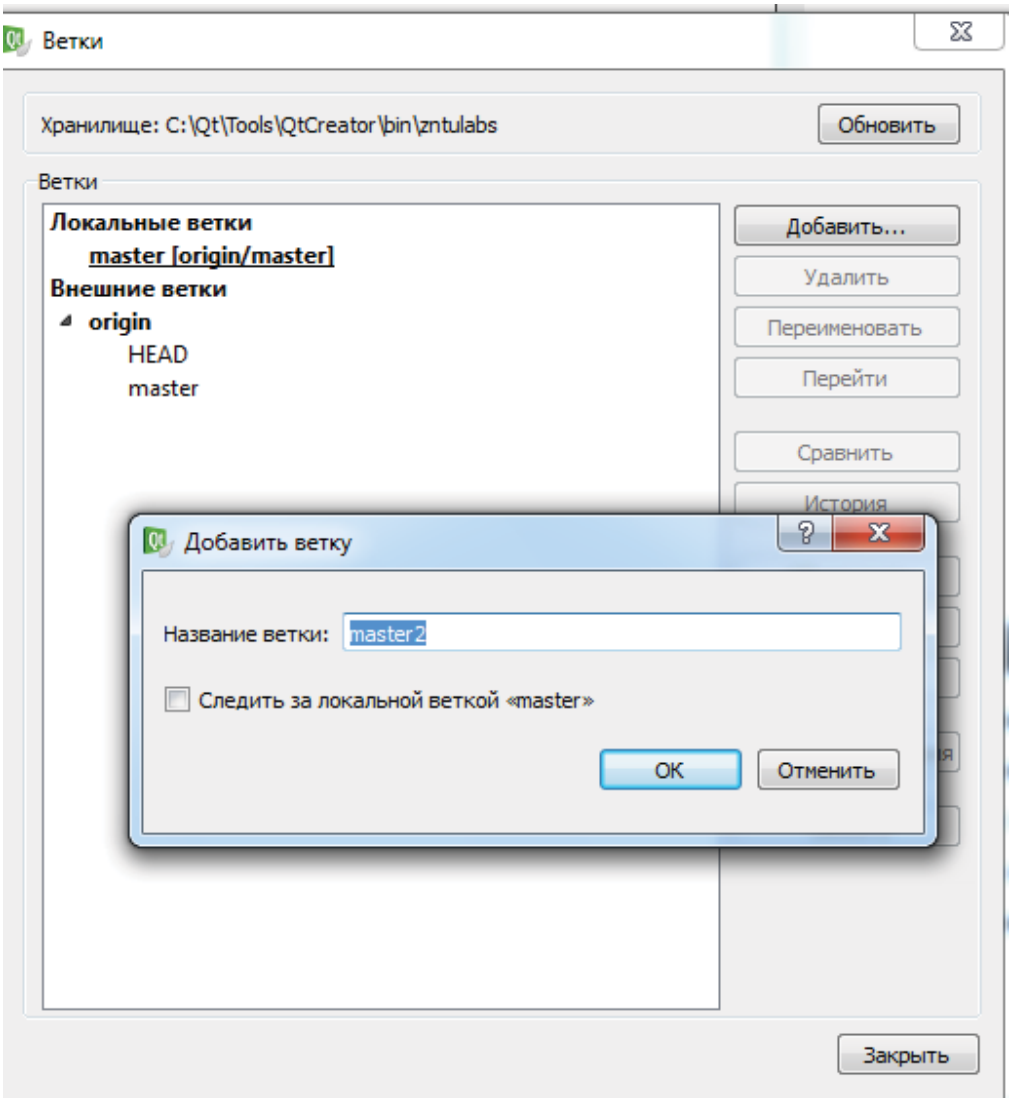

Рисунок 5. Создание ветки в qt creator

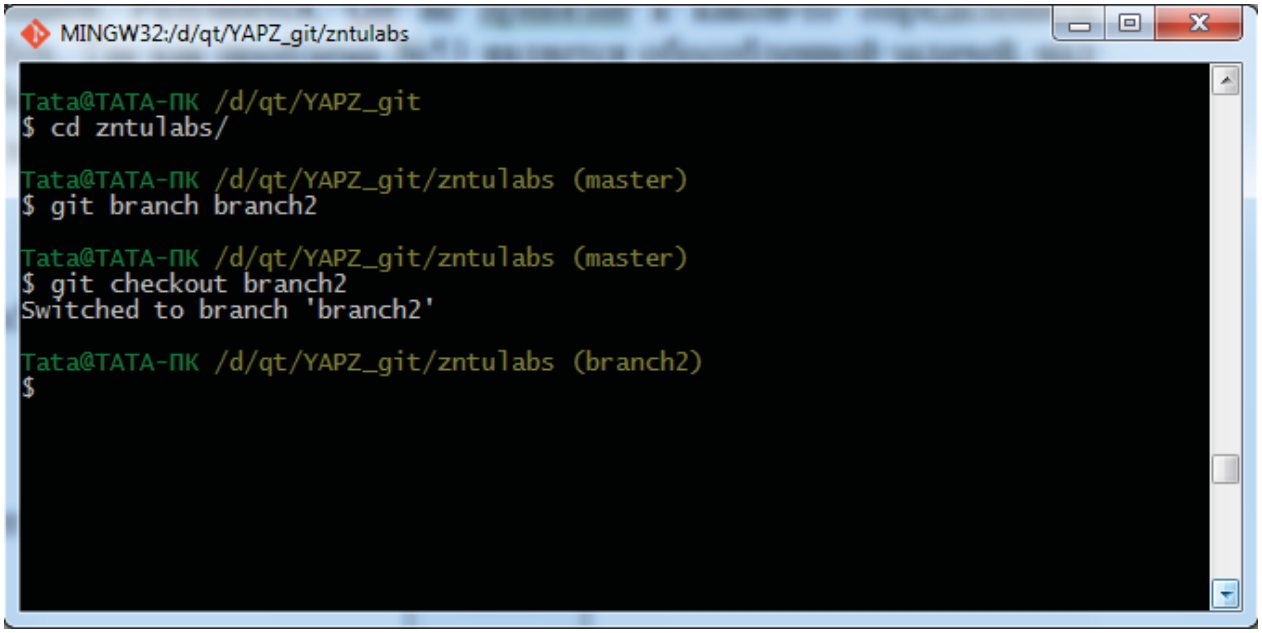

Рисунок 6. Создание ветки с помощью командной строки

6. Изучить команды git, которые предоставляет qt creator (рис. 7)

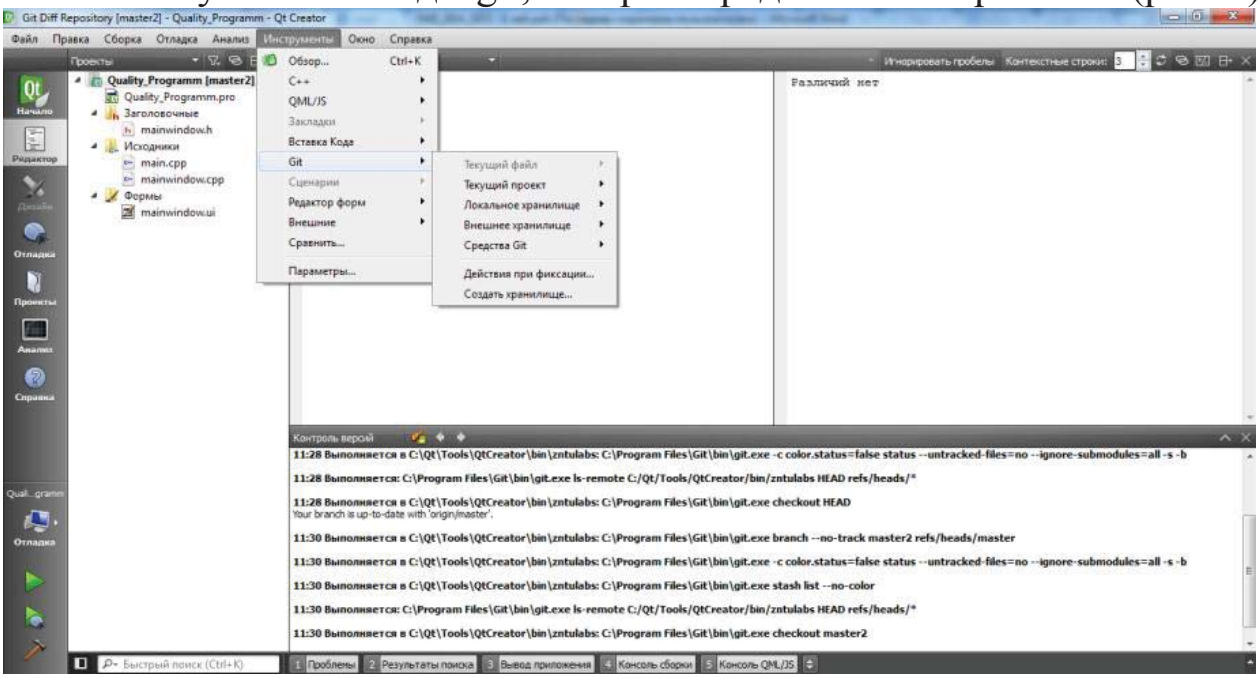

Рисунок 7. Инструментарий для работы с git в qt creator

7. Залить созданные ветки на fork https://bitbucket.org/tata007/zntulabs в ветки, названные "фамилия+qt" и "фамилия+bash" (рис. 8 и 9)

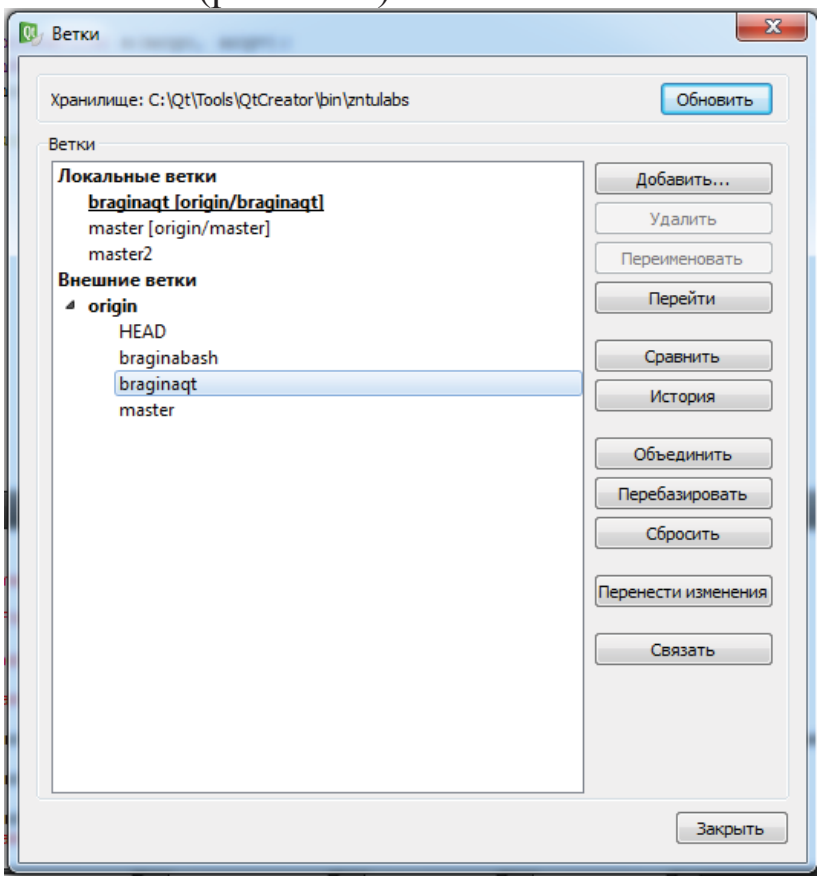

Рисунок 8. Создание удаленной ветки в qt creator

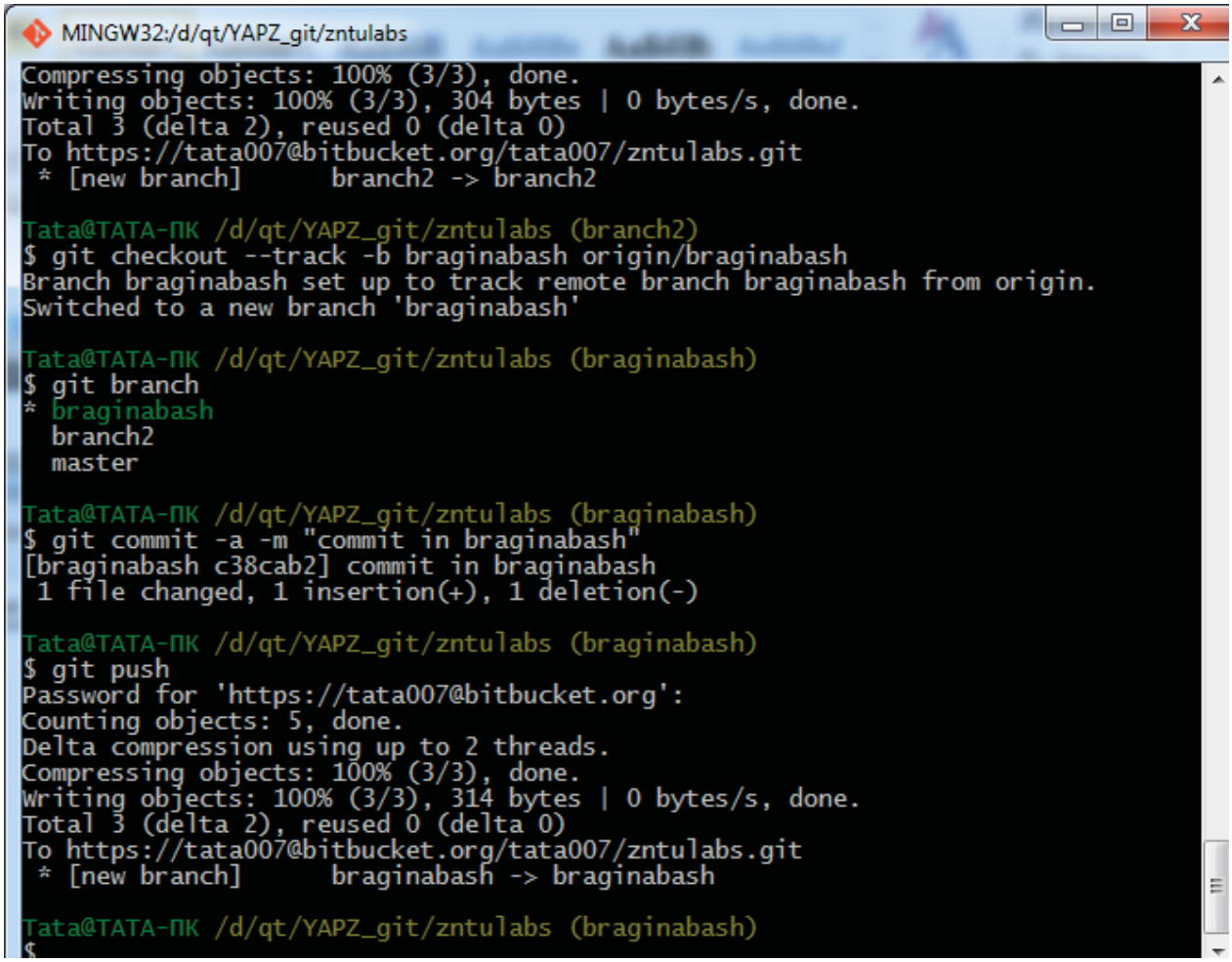

Рисунок 9. Создание удаленной ветки в командной строке

8. Сделайте pull request в репозиторий https://bitbucket.org/tata007/zntulabs

2.5 Контрольні запитання

4.5.1 Як копіювати віддалений Git-репозиторій?

4.5.2 Як відправити зроблені в локальною репозиторії зміни до віддаленого Git-репозиторію?