Инструкция по GitHub с возможными проблемами и способами решения

Основные команды

Клонируем репозиторий на локальный компьютер (эта команда выполняется только при необходимости подтянуть состояние репозитория с GitHub, если у вас на вашем рабочем компьютере итак состояние актуальное, т.е. соответствует состоянию с GitHub, то эта команда пропускается)

`git clone ссылка-до-ВАШЕГО-репозитория`

Переключаем ветку из master на вашу рабочую ветку (если это ещё не сделано)

1. Если ветка уже у вас была создана, то просто переходим в неё

   `git checkout имя-рабочей-ветки`

2. Если у вас нет другой рабочей ветки и только master, то создаёте рабочую ветку

   `git branch имя-рабочей-ветки`

   и переходите в неё

   `git checkout имя-рабочей-ветки`

   или одной командой создаём новую ветку и переходим в неё

   `git checkout -b имя-рабочей-ветки`

Находясь в рабочей ветке, пишите/редактируете код. Следите, что вы ушли из master.

Чтобы сделать «checkpoint» вашего текущего состояния кода, необходимо создать коммит. Для этого нужно проделать две вещи: 1) добавить файлы, 2) сформировать коммит из добавленных файлов.

Если в папке лежат только файлы, которые вы хотите добавить к коммиту, можно использовать команду

`git add .`

В противном случае вместо точки указывать конкретные файлы, например:

`git add HW/HWSolution/Task0/main.c`

`git add HW/HWSolution/ HWSolution.sln`

Папки Debug мы на GitHub не заливаем!

Затем сформировать коммит командой, добавив комментарий, какие изменения проделаны в добавленных файлах

`git commit –m "fix problem with magic numbers, add comments"`

Можно сделать сразу несколько коммитов, то есть повторить действия:

1) ушли писать дальше код,
2) решили, что хотите сохранить данное состояние файлов,
3) выполнили git add .,

4) создали коммит с комментарием git commit –m “…”.

Когда вам понадобится залить все созданные коммиты (1 и больше) на GitHub, необходимо выполнить команду

`git push origin имя-рабочей-ветки`

или команду с ключом

`git push –u origin имя-рабочей-ветки`

Замечание. Если сделать git push ветки без ключа -u, git не свяжет локальную ветку с веткой удалённого репозитория. Смысл использовать ключ -u есть только при пуше новых веток, для существующих (связанных с удалёнными) веток каждый раз перезаписывать связку необязательно.

Топ проблем с их решениями

Ситуация 0. Не могу залить на GitHub в классе, потому что git привязался к другому пользователю.

Решение 0. Зайти в Панель управления -> Учётные записи пользователей -> Диспетчер учётных данных -> Учётные данные Windows. Удалить учётные данные для github.

Ситуация 1. На этапе git clone я указал не свою ссылку, а вашу. Теперь при git push он мне пишет, что у меня нет прав доступа к репозиторию пользователя UsovaMA.

Решение 1. Выполнить команду

```
git remote set-url origin ссылка-до-ВАШЕГО-репозитория
```

попробовать повторить

```
git push origin название-рабочей-ветки
```

Ситуация 2. Я пытаюсь сделать пуш, но git пишет, что на GitHub есть изменения в этой ветке, которые отсутствуют в вашей локальной копии на компьютере.

```
hint: Updates were rejected because the remote contains work that you do
hint: not have locally. This is usually caused by another repository pushing
hint: to the same ref. You may want to first integrate the remote changes
hint: (e.g., 'git pull ...') before pushing again.
hint: See the 'Note about fast-forwards' in 'git push --help' for details.
```

Решение 2. Можно попробовать залить изменения принудительно

```
git push -f origin название-рабочей-ветки
```

Либо попробовать подкачать изменения с GitHub (актуализировать ветку), а потом снова попробовать залить изменения. Если коммиты при этом исчезнут, придётся создать их снова.

```
git pull origin название-рабочей-ветки
```

```
git push origin название-рабочей-ветки
```

Ситуация 3. Забыл создать новую ветку и работал в master.

Решение 3. Если Вы закоммитили (git commit –m "…") уже кучу файлов, можно переместить все эти изменения в новую ветку можно с помощью следующих трёх/двух команд (действия выполняются в ветке master)

```
git branch название-рабочей-ветки
```

```
git reset HEAD~ --hard
```

```
git checkout название-рабочей-ветки
```

Первая команда выполняется, если ветка еще не была создана.

(!) Обратите внимание на то, что если вы не воспользовались, в применении к изменениям, командами commit, они будут утеряны.

Ситуация 4. Добавил к коммиту лишние файлы (например, папку Debug).

Решение 4. Если коммит ещё не был создан достаточно команды

```
git reset имя-файла/папки-с-путём
```

Если изменение было закоммичено, то и это поправимо. Придётся воспользоваться ещё несколькими командами

```
git reset --soft HEAD~1
git reset имя-файла/папки-с-путём
rm имя-файла/папки-с-путём
git commit
```

Команда rm удалит данный файл/папку!

Про менее частные проблемы можно прочитать здесь:

Шпаргалка по Git. Решение основных проблем — Блог HTML Academy