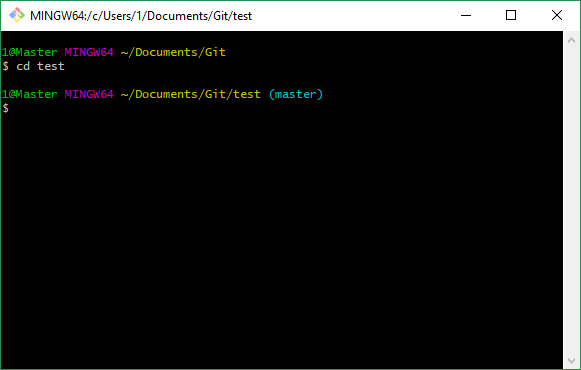
1. Запустите Git в папке проекта с помощью команды GitBash Here контекстного меню. Перейдите в папку test с помощью команды

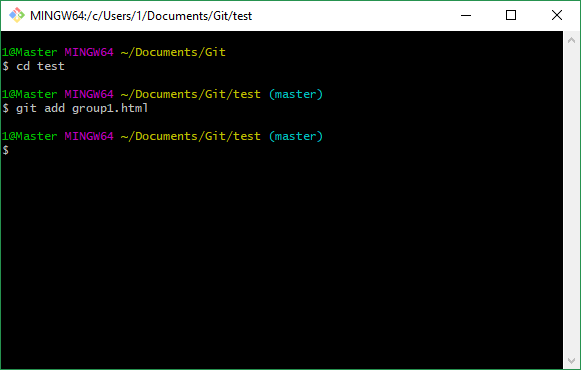
***cd test***

Если всё сделано верно, то в конце приглашения в терминале появится голубая надпись (master):



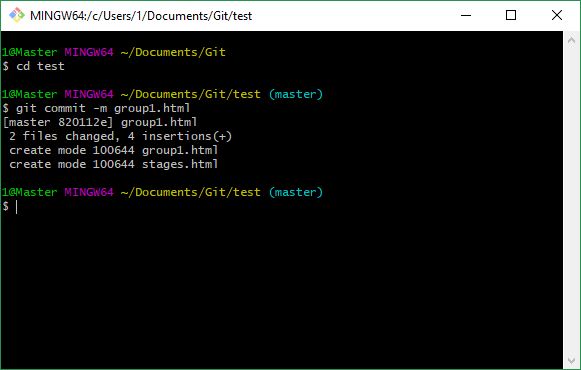
2.Создайте в вашей папке новый файл формата .html. В нём укажите заголовок разрабатываемой вашей группой страницы. Добавьте файл в git командой

**git add group1.html** (вместо group1 укажите имя созданного вами файла)



3.Выполните коммит (сохранение состояния) с помощью команды

**git commit –m “group1.html added”**



4.Выполните загрузку файла в глобальный репозиторий:

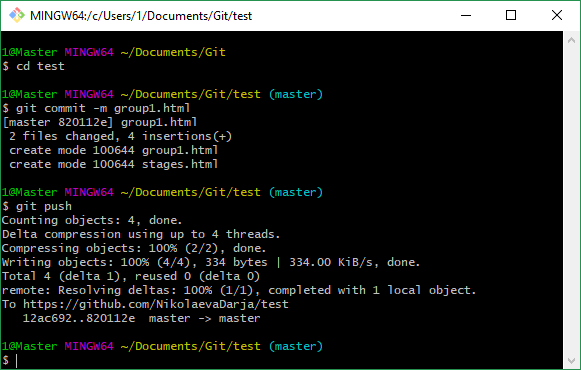
Сначала получите обновления из глобального репозитория с помощью команды

**git pull**

Затем загрузите свои файлы в глобальный репозитория с помощью команды

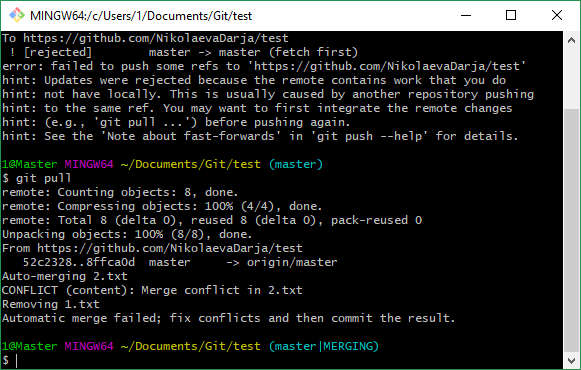
**git push**

Если всё сделано верно, вы увидите сообщения следующего вида, описывающие процесс передачи файлов в глобальный репозиторий:

****

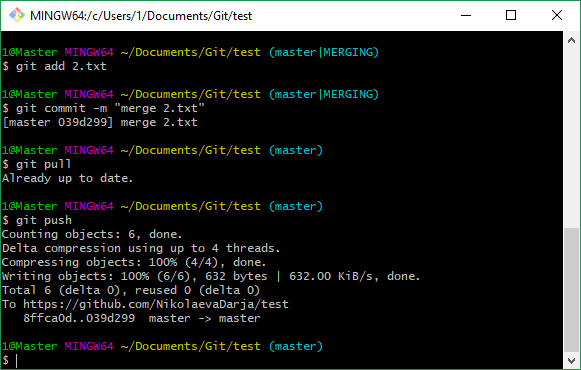
В дальнейшем всегда будем использовать эти команды в следующей последовательности: сначала git pull, затем git push.

Возможно, после выполнения команд git push или git pull вы увидели сообщение следующего вида и приглашение, оканчивающееся голубыми словами (master|MERGING):

****

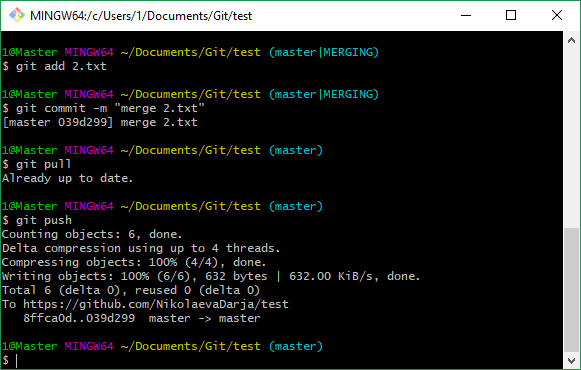
Это обозначает, что произошла ***коллизия***. То есть, вы изменили чужой файл (например, task1.txt), и теперь есть две разных версии этого файла. Речь о коллизиях пойдёт на уроке 6. Однако, чтобы не задерживать процесс изучения распределённой разработки приложений, укажем, что следует вернуть файл к изначальному состоянию, а затем выполнить следующую команду:

**git commit –m “merge task1.txt”**

**(вместо task1 указываем имя своего файла)**

Если всё сделано верно, то в конце строки вновь появится привычное голубое слово (master).

Теперь осталось внести выполненные изменения в глобальный репозиторий командой

**git push **