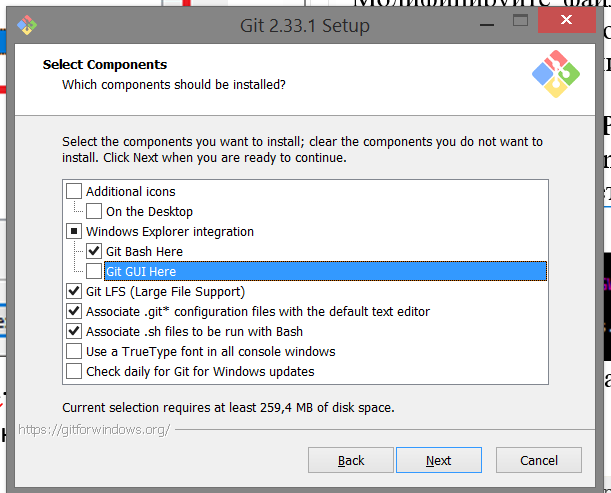
**Лабораторная работа 6**

**Системы контроля версий. Основы работы в Git**

**Цель работы:** ознакомление с основными возможностями системы контроля версий Git.

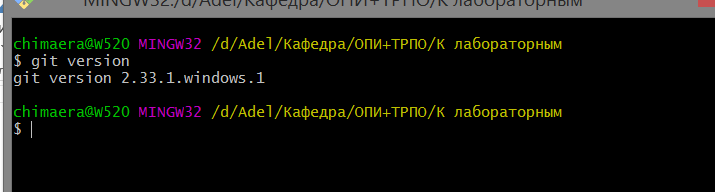
1. Используйте при выполнении лабораторной работы материал лекции 10 (часть1).
2. Отчет по лабораторной работе оформить в виде документа в MS Word.
3. Скачайте с сайта http://git-scm.com/downloads/ и установите консольного клиента (***gitbush***):

Установить интеграцию с Проводником, выбрав чекбокс: Git Bash Here



***НЕ РЕКОМЕНДУЕТСЯ*** изменять настройки для среды.

1. Проверьте установленную версию и зафиксируйте ее в отчете:



1. Получите справку по **Git**. Ознакомьтесь со списком командам. Какой командой для получения справки вы воспользовались? Описание поместите в отчет.
2. Создайте локальный репозиторий. Опишите последовательность действий в отчете.
3. Выполните конфигурацию гит. Для этого настройте конфигурационный файл, используя:

$ git config --global user.name "XXX"

$ git config --global user.email [XXX@XXX.com](mailto:XXX@XXX.com)

1. Проверить настройки конфигурационного файла командой git config --list
2. Инициализируйте локальный репозиторий в рабочей папке (см. лекцию), где будет он будет размещен. Исследуйте изменения, которые произошли в папке. Зафиксируйте их в отчете.
3. С помощью блокнота создайте в папке текстовый файл Hello.txt.
4. Научитесь получать информацию о статусе (status) репозитория. Какая информация отображается в терминальном окне GitBash? Прокомментируйте вывод команды в отчете.
5. Проиндексируйте файл Hello.txt Какие изменения произошли в репозитории?
6. Зафиксируйте изменения индексированных файлов в репозитории. Какие изменения произошли в репозитории? Как Git информирует об успешном создании нового коммита? В какую ветку добавлен файл?
7. Просмотрите журнал с помощью команды git log
8. Научитесь получать информацию о проиндексированных и неиндексированных файлах. Для этого измените существующий файл Hello.txt, добавьте в репозиторий новый файл и исследуйте состояние репозитория. Зафиксируйте изменения в репозитории.
9. Научитесь пользоваться командами управления и фиксации изменений (add, commit, status, log).
10. Ознакомиться с шаблонами .gitignore и настройте этот файл для вашего репозитория.
11. Создайте (branch) новую ветку, например, test и переключитесь (checkout) в нее.
12. Создайте новый текстовый файл test.html и зафиксируйте его (commit).
13. Просмотрите список всех веток и определите текущую ветку.
14. Выполните слияние (merge) веток.
15. По очереди отредактируйте файл test.html в ветках master и test. Выполните commit. Выполните слияние в одной и второй ветках. Разрешите ситуацию конфликта слияний.
16. Научитесь пользоваться командами **Git** для работы с ветками.

**Вопросы.**

1. Что такое система контроля версий, для чего ее используют?
2. Какой принцип хранения файлов использует Git?
3. В чем отличие Git от других систем контроля версий?
4. В каких трех основных состояниях файлы могут находиться в Git ?
5. Что такое индексация файла? Какой командой она выполняется?
6. Что такое фиксация файла? Какой командой она выполняется?
7. Продемонстрируйте команду проверки выбранных настроек.
8. Как инициализировать репозиторий в существующей папке?
9. Как указать файлы, за которыми должна следить система?
10. Как выполнить фиксацию изменений?
11. Какой командой определить состояния файлов?
12. Для чего создается файл .gitignore? Поясните его структуру.
13. Как используется команда git log?
14. Что такое ветвление?
15. Как используется команда git checkout?
16. Что такое указатель HEAD?
17. Продемонстрируйте создание новых веток в Git?
18. Какая команда осуществляет переключение веток?
19. Для чего используется команда merge?
20. Какая команда позволяет отобразить историю репозитория?
21. Как выполнить слияние изменений из одной ветки в другую?
22. Какие проблемы могут быть при слиянии и как они разрешаются?