# Технологии разработки программного обеспечения

2017 / 2018, 1 курс, 2 семестр

Пудов Сергей Григорьевич

# Лекция 2

- Git на локальной машине:
  - git add
  - git commit
  - git status
  - git log
  - git diff

# Git на локальной машине: повтор

Книга: https://git-scm.com/book/ru/v2

Помощь:

- *$ git help <verb>*
- *$ git <verb> --help*
- *$ man git-<verb>*

Создание репозитория:

- *$ git init*

Клонирование удаленного репозитория

- *$ git clone*

Добавление файлов под версионный контроль:

- *$ git add <filename>*
- *$ git commit -m 'initial project version'*
- *$ git commit <filename> -m 'initial project version'*

# Git на локальной машине: status

Определение состояния файлов:

- *$ git status*

  *On branch master*

  *nothing to commit, working directory clean*

После создания файла в каталоге:

- *$ git status*

  *On branch master*

  *Untracked files:*

  *(use "git add <file>..." to include in what will be committed)*

  *<filename>*

  *nothing added to commit but untracked files present (use "git add" to track)*

# Git на локальной машине: status

Начать отслеживание файла:

- *$ git add <filename>*
- *$ git status*

  *On branch master*

  *Changes to be committed: (use "git reset HEAD <file>..." to unstage)*

  *new file: <filename>*

```
E:\SERGEY\Lectures\git\test2>git status
On branch master
Changes to be committed:
  (use "git reset HEAD <file>..." to unstage)

        modified:   test2.c

Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git checkout -- <file>..." to discard changes in working directory)

        modified:   test2.c
```

# Git на локальной машине: status

Сокращенная форма вывода:

- $ git status –s

```
M README
MM Makefile
A lib/git.rb
M lib/simplegit.rb
?? LICENSE.txt
```

Игнорирование файлов:

- *$ cat .gitignore*

```
*.[oa]
*~
```

# Git на локальной машине: .gitignore

- # no .a files
- *.a

- # but do track lib.a, even though you're ignoring .a files above
- !lib.a

- # only ignore the root TODO file, not subdir/TODO
- /TODO

- # ignore all files in the build/ directory
- build/

- # ignore doc/notes.txt, but not doc/server/arch.txt
- doc/*.txt

- # ignore all .txt files in the doc/ directory
- doc/**/*.txt

# Git на локальной машине: diff

Чтобы увидеть, что же вы изменили, но пока не проиндексировали, наберите git diff без аргументов:

Эта команда сравнивает содержимое вашего рабочего каталога с содержимым индекса. Результат показывает ещё не проиндексированные изменения.

```
E:\SERGEY\Lectures\git\test2>git diff
diff --git a/test2.c b/test2.c
index 0c9755f..301aab4 100644
--- a/test2.c
+++ b/test2.c
@@ -6,5 +6,5 @@ int main()
 {
        int i;
        cout << "i = " << endl;
-       return 2;
+       return 3;
 }
\ No newline at end of file
```

```
E:\SERGEY\Lectures\git\test2>git diff --staged
diff --git a/test2.c b/test2.c
index ead30a3..0c9755f 100644
--- a/test2.c
+++ b/test2.c
@@ -6,5 +6,5 @@ int main()
 {
        int i;
        cout << "i = " << endl;
-       return 1;
+       return 2;
 }
\ No newline at end of file
```

Если вы хотите посмотреть, что вы проиндексировали и что войдёт в следующий коммит, вы можете выполнить git diff --staged.

Эта команда сравнивает ваши индексированные изменения с последним коммитом:

# Git на локальной машине: commit

- $ git commit

Эта команда откроет выбранный вами текстовый редактор. В нем нужно будет набрать сообщение для коммита.

- $ git commit -m "Story 182: Fix benchmarks for speed"

-m <msg> - сообщение для коммита в виде опции

- $ git commit –a -m "Story 182: Fix benchmarks for speed“

-a

--all

Tell the command to automatically stage files that have been modified and deleted, but new files you have not told Git about are not affected.

# Git на локальной машине: rm/mv

- rm <filename>

если вы выполните команду git rm, удаление файла попадёт в индекс:

- git rm <filename>

удалить файл из индекса, оставив его при этом в рабочем каталоге:

- git rm –cached <filename>

- $ git mv file_from file_to

эквивалентно

- $ mv file_from file_to
- $ git rm file_from
- $ git add file_to

Когда вы переименовываете файл в Git, в нём не сохраняется никаких метаданных, говорящих о том, что файл был переименован

# Git на локальной машине: log

- $ git log

commit ca82a6dff817ec66f44342007202690a93763949
Author: Scott Chacon <schacon@gee-mail.com>
Date: Mon Mar 17 21:52:11 2008 -0700

changed the version number

commit 085bb3bcb608e1e8451d4b2432f8ecbe6306e7e7
Author: Scott Chacon <schacon@gee-mail.com>
Date: Sat Mar 15 16:40:33 2008 -0700

removed unnecessary test

перечисляет коммиты, сделанные в репозитории в обратном к хронологическому порядке

# Git на локальной машине: log

Механизм, которым пользуется Git при вычислении хеш-сумм называется **SHA-1** хеш. Это строка длинной в 40 шестнадцатеричных символов (0-9 и a-f), она вычисляется на основе содержимого файла или структуры каталога. SHA-1 хеш выглядит примерно так:

24b9da6552252987aa493b52f8696cd6d3b00373

Полезные опции для git log:

| | |
|---|---|
| --graph | Отображает ASCII граф с ветвлениями и историей слияний. |
| --oneline | печатает каждый коммит в одну строку |
| --since и --until | опции для ограничения вывода по времени |
| --grep | коммиты, сообщение которых содержит указанную строку |
| -S | коммиты, в которых изменение в коде повлекло за собой добавление или удаление указанной строки |

# Git на локальной машине: отмена

Если нужно переделать коммит, можно запустить commit с параметром --amend (дополнить):

- $ git commit -m 'initial commit'
- $ git add forgotten_file
- $ git commit –amend

Отмена операции add:

- $ git reset <filename>

Откат сделанных изменений:

- $ git checkout -- <filename>

# Продолжение в следующей лекции…