
Front matter

title: "Отчёт по лабораторной работе № 1" subtitle: "Работа с Git и Markdown" author: "Абу Сувейлим Мухаммед Мунифович"

Generic options

lang: ru-RU toc-title: "Содержание"

Bibliography

bibliography: bib/cite.bib csl: pandoc/csl/gost-r-7-0-5-2008-numeric.csl

Pdf output format

toc: true # Table of contents toc-depth: 2 lof: true # List of figures lot: true # List of tables fontsize: 12pt
linestretch: 1.5 papersize: a4 documentclass: scrreprt

l18n polyglossia

polyglossia-lang: name: russian options: - spelling=modern - babelshorthands=true polyglossia-otherlangs:
name: english

l18n babel

babel-lang: russian babel-otherlangs: english

Fonts

mainfont: PT Serif romanfont: PT Serif sansfont: PT Sans monofont: PT Mono mainfontoptions: Ligatures=TeX
romanfontoptions: Ligatures=TeX sansfontoptions: Ligatures=TeX,Scale=MatchLowercase monofontoptions:
Scale=MatchLowercase,Scale=0.9

Biblatex

biblatex: true biblio-style: "gost-numeric" biblatexoptions:

- parenttracker=true
- backend=biber
- hyperref=auto
- language=auto
- autolang=other*
- citestyle=gost-numeric

Pandoc-crossref LaTeX customization

figureTitle: "Рис." tableTitle: "Таблица" listingTitle: "Листинг" lofTitle: "Список иллюстраций" lotTitle:
"Список таблиц" lolTitle: "Листинги"

Misc options

indent: true header-includes:

- `\usepackage{indentfirst}`
- `\usepackage{float} # keep figures where there are in the text`
- `\floatplacement{figure}{H} # keep figures where there are in the text`

Цель работы

- Во-первых, ознакомились с Git и Markdown.
- Сделайте отчёт по предыдущей лабораторной работе в формате Markdown.
- В качестве отчёта просьба предоставить отчёты в 3 форматах: pdf, docx и md (в архиве, поскольку он должен содержать скриншоты, Makefile и т.д.)

Задание

- Создать новую репозиторию на GitHub
- Сделайте отчёт по предыдущей лабораторной работе в формате Markdown.
- В качестве отчёта просьба предоставить отчёты в 3 форматах: pdf, docx и md (в архиве, поскольку он должен содержать скриншоты, Makefile и т.д.)


Теоретическое введение

Git — абсолютный лидер по популярности среди современных систем управления версиями. Это развитый проект с активной поддержкой и открытым исходным кодом. Система Git была изначально разработана в 2005 году Линусом Торвальдсом — создателем ядра операционной системы Linux. Git применяется для управления версиями в рамках колоссального количества проектов по разработке ПО, как коммерческих, так и с открытым исходным кодом. Система используется множеством профессиональных разработчиков программного обеспечения. Она превосходно работает под управлением различных операционных систем и может применяться со множеством интегрированных сред разработки (IDE).

Выполнение лабораторной работы

1. Во-первых, я установил Git и gh на моем ПК.
2. Выполнял следующие команды, чтобы git узнал мое имя и электронную почту.

```
git config --global user.name "Mukhammed"
git config --global user.email "mohammedswelem13@gmail.com"
```

 1.1.1{#fig:001 width=70%}

3. Настрой core.autocrlf с параметрами true, чтобы сделать все переводы строк текстовых файлов в главном репозитории одинаковыми.

```
git config --global core.autocrlf true
git config --global core.safecrlf true
```

 1.1.2{#fig:001 width=70%}

4. Чтобы избежать нечитаемых строк, я установил соответствующий флаг.

```
git config --global core.quotepath off
```

 1.1.3{#fig:001 width=70%}


5. Создал пустой каталог hello и файл hello.html

 1.2.1{#fig:001 width=70%}  1.2.1{#fig:001 width=70%}

6. Я инициализировал git, чтобы потом создать репозиторий.

 1.2.3{#fig:001 width=70%}

7. Проверил текущее состояние репозитория.

 1.2.4{#fig:001 width=70%}

8. Добавил HTML-теги к нашему приветствию.


```
<h1>Hello, World!</h1>
```

 1.3.1{#fig:001 width=70%}

9. В первой строке ввел комментарий: «Added h1 tag».

 1.4.1{#fig:001 width=70%}


10. Изменил страницу «Hello, World», чтобы она содержала стандартные теги html и body. Далее, добавил заголовки HTML (секцию head) к странице «Hello, World».


 1.4.2a{#fig:001 width=70%} 1.4.2b{#fig:001 width=70%}

11. Я Получил список произведенных изменений


 1.4.3{#fig:001 width=70%}

12. Возвращался назад в историю используя команду `checkout` и вернитея к последней версии в ветке `master`.


1.4.4a{#fig:001 width=70%}

1.4.4b{#fig:001 width=70%}

13. Далее, я создал новые теги `v1` и `v1-beta`.

1.4.5{#fig:001 width=70%}


14. Изменил в файл `hello.html` в виде нежелательного комментария


1.6.1a{#fig:001 width=70%}

15. Выполнил сброс буферной зоны

1.6.3{#fig:001 width=70%}

16. Далее, я изменил файл `hello.html` и сделал коммит


1.7.2a{#fig:001 width=70%}

1.7.2{#fig:001 width=70%}

17. Я сделал коммит с новыми изменениями, отменяющими предыдущие

1.7.3{#fig:001 width=70%}

18. Я сбросил коммит к предшествующим коммиту `Oops`

1.8.4{#fig:001 width=70%}


19. Я удалил тега `oops`

1.9.1{#fig:001 width=70%}


20. После того как я добавил комет, я решил добавить еще мой номер телефона, но без нового комета.

1.10.3{#fig:001 width=70%}

21. Создал каталог `lib` и в нем переносил страницу.

1.11.1{#fig:001 width=70%}

22. Я нашел инфо про тагов и веток.


1.14.5a{#fig:001 width=70%}

1.14.5b{#fig:001 width=70%}


1.14.5c{#fig:001 width=70%}

1.14.5d{#fig:001 width=70%}


23. Вывол последний коммит с помощью SHA1 хэша

1.15.2{#fig:001 width=70%}


24. Поиск дерева

1.15.3{#fig:001 width=70%}


25. Вывод каталога lib


1.15.4{#fig:001 width=70%}

26. Вывод файла hello.html


1.15.5{#fig:001 width=70%}

27. Создал ветку style и файл lib/style.css


1.16.1{#fig:001 width=70%}

1.16.2{#fig:001 width=70%}

28. Просмотрил текущие ветки

1.19.2{#fig:001 width=70%}


29. Слияние веток

1.20.1{#fig:001 width=70%}


30. Слияние master с веткой style

1.22.1a{#fig:001 width=70%}


31. Сделал коммит решения конфликта

1.22.3{#fig:001 width=70%}


32. Сбросил ветку style

1.23.1{#fig:001 width=70%}


33. Перебазирование

1.25a{#fig:001 width=70%}


34. Создать клон репозитория hello

1.27.2{#fig:001 width=70%}


35. Просмотрим историю репозитория

1.28.2{#fig:001 width=70%}


36. Слейте извлеченные изменения в локальную ветку master

 1.32.1{#fig:001 width=70%}


37. Проверка файла README.md

 1.31.3{#fig:001 width=70%}

38. Создал чистый репозиторий

 1.35{#fig:001 width=70%}

39. Добавление удаленного репозитория

 1.36{#fig:001 width=70%}

40. Извлечение общих изменений

 1.38{#fig:001 width=70%}

Выводы

Git is harder than you would initially be allowed to think. This mainly stems from what seems more abstraction (compared to doing creating repos and files manually on GitHub) but it is also the reason to why it works best when dealing with a significant number of files and directories. Managing big systems is best suited for linux based OS compared to Windows or MacOS.

Список литературы{.unnumbered}

Репозиторий: <https://github.com/yamadharma/course-directory-student-template>.