

Отчет по лабораторной работе №4

Язык разметки Markdown

Виктор Максимович Кадров

Содержание

1	Цель работы	4
2	Задание	5
3	Теоретическое введение	6
4	Выполнение лабораторной работы	7
5	Задания для самостоятельной работы	9
6	Ответы на вопросы для самопроверки	10
7	Выводы	11

Список иллюстраций

4.1	Обновление репозитория	7
4.2	Компиляция отчета	7
4.3	Шаблон в Markdown	8
4.4	Заполнение отчета	8
4.5	Компиляция и выгрузка на github	8
5.1	Заполнение отчета	9

1 Цель работы

Целью работы является освоение процедуры оформления отчетов с помощью легковесного языка разметки Markdown.

2 Задание

1. Создание шаблонов отчета при помощ Makefile
2. Заполнение шаблона
3. Загрузка отчетов на github

3 Теоретическое введение

Markdown — облегчённый язык разметки, созданный с целью обозначения форматирования в простом тексте, с максимальным сохранением его читаемости человеком, и пригодный для машинного преобразования в языки для продвинутых публикаций (HTML, Rich Text и других).

Чтобы создать заголовок, используйте знак #, например:

```
# This is heading 1
## This is heading 2
### This is heading 3
#### This is heading 4
```

Чтобы задать для текста полужирное начертание, заключите его в двойные звездочки:

```
This text is **bold**.
```

Чтобы задать для текста курсивное начертание, заключите его в одинарные звездочки:

```
This text is *italic*.
```

Синтаксис Markdown для встроенной ссылки состоит из части [link text], представля name.md) – URL-адреса или имени файла, на который дается ссылка: [link text](file name.md) или [link text](http://example.com/ "Необязательная подсказка")

4 Выполнение лабораторной работы

Обновляем локальный репозиторий (рис. 4.1). После компилируем шаблон с использованием Makefile (рис. 4.2) и проверяем корректность выполнения (рис. 4.3). После этого заполняем отчет (рис. 4.4), компилируем и выгружаем на github (рис. 4.5).

```

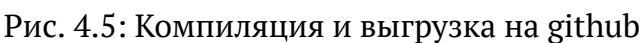
Title: unknabovskiy@kali:~/work/study/2022-10-18/kompyuterya-smyeterya/ssh-pc
1: unknabovskiy@kali:~/work/study/2022-10-18/kompyuterya-smyeterya/ssh-pc
$ ssh-keygen
Generating public/private rsa key pair.
Enter file in which to save the key (/home/unknabovskiy/.ssh/id_rsa):
Created new key, and wrote the keypair to file /home/unknabovskiy/.ssh/id_rsa.
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/unknabovskiy/.ssh/id_rsa.
Your public key has been saved in /home/unknabovskiy/.ssh/id_rsa.pub.
The key fingerprint is:
SHA256:1234567890123456789012345678901234567890123456789012345678901234
The key's randomart image is:
[+]

```

Рис. 4.1: Обновление репозитория

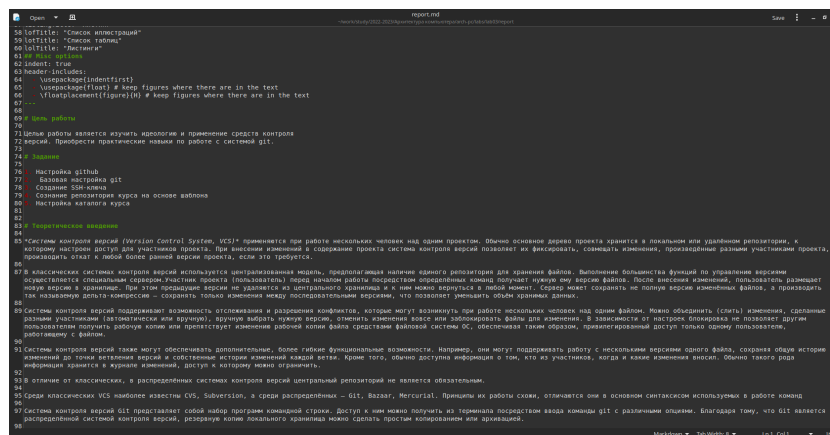
[illegible]

Рис. 4.2: Компиляция отчета



5 Задания для самостоятельной работы

Заполняем отчет по лабораторной работе №3 (рис. 5.1). Затем загружаем его на github.



```
56 \otitle: "Ссылка на инструкцию"
59 \otitle: "Ссылка на проект"
60 \otitle: "Дистанция"
61
62 \index: true
63 \header{\indexfirst}
64 \usepackage{float} # keep figures where there are in the text
65 \floatplacement{figure} # keep figures where there are in the text
66
67
68
69 % Начиная работу
70
71 Целью работы является изучить идеологию и применимые средства контроля
72 версий. Пройти практические навыки по работе с системой git.
73
74 % Задачи
75
76 Настройка github
77 Базовая настройка git
78 Создание репозитория
79 Создание репозитория курса на основе шаблона
80 Настройка каталога курса
81
82
83 % Теоретическое введение
84
85 Системы контроля версий (Version Control System, VCS) применяются при работе нескольких человек над одним проектом. Обычно отладочные версии проекта хранятся в локальном или удаленном репозитории, к
86 которому настроен доступ для участников проекта. При внесении изменений в содержание проекта система контроля версий позволяет не потерять внесенные изменения, отслеживать действия участников проекта,
87 возвращать файл к любой более ранней версии проекта, если это требуется.
88
89 Классические системы контроля версий используются централизованные модели, предполагающие наличие одного репозитория для хранения файлов. Выполнение большинства функций по управлению версиями
90 осуществляется специальным сервером. Участники проекта (пользователи) перед началом работы посредством определенных команд получают копию нужных им версий файлов. После внесения изменений, пользователь размещает
91 новую версию в репозитории. При этом происходит версия не удаляется из центрального репозитория и в нем вновь добавляется в новый момент. Сервер может сохранять не только версии измененных файлов, а производить
92 так называемую дельта-компрессию – сохранять только изменения между последовательными версиями, что позволяет уменьшить объем хранимых данных.
93
94 Системы контроля версий поддерживают возможность отслеживания и разрешения конфликтов, которые могут возникнуть при работе нескольких человек над одним файлом. Можно объединить (сшить) изменения, сделанные
95 разными участниками (автоматически или вручную), вручную выбрать нужную версию, отменить изменения в базе или закоммитовать файлы для изменений. В зависимости от настроек безопасности не позволяет другим
96 пользователям получить рабочую копию или протестировать изменения рабочей копии файла средствами файловой системы ОС, обеспечивая таким образом, приватизированный доступ только одному пользователю,
97 работающему с файлом.
98
99 Системы контроля версий также могут обеспечивать дополнительные, более гибкие функциональные возможности. Например, они могут поддерживать работу с несколькими версиями одного файла, сохраняя историю
100 изменений до точки ветвления версий и собственные истории изменений каждой ветки. Кроме того, обычно доступна информация о том, кто из участников, когда и какие изменения вносил. Обычно такого рода
101 информация хранится в журнале изменений, доступ к которому может быть ограничен.
102
103 отличие от классических, в распределенных системах контроля версий центральный репозиторий не является обязательным.
104
105 Среди классических VCS наиболее известны CVS, Subversion, а среди распределенных – Git, Bazaar, Mercurial. Принципы их работы схожи, отличаются они в основном синтаксисом используемых в работе команд
106
107 Система контроля версий Git представляет собой набор программ командной строки. Доступ к ним можно получить из терминала посредством ввода команды git с различными опциями. Благодаря тому, что Git является
108 распределенной системой контроля версий, резервную копию локального хранилища можно сделать простым копированием или архивацией.
109
```

Рис. 5.1: Заполнение отчета

6 Ответы на вопросы для самопроверки

1. *Markdown* — облегчённый язык разметки, созданный с целью обозначения форматирования в простом тексте, с максимальным сохранением его читаемости человеком, и пригодный для машинного преобразования в языки для продвинутых публикаций (HTML, Rich Text и других).
2. Чтобы задать для текста полужирное начертание, нужно заключить его в двойные звездочки, а чтобы задать для текста курсивное начертание — заключить его в одинарные звездочки.
3. Упорядоченный список можно отформатировать с помощью соответствующих цифр. Чтобы вложить один список в другой, добавьте отступ для элементов дочернего списка. Неупорядоченный (маркированный) список можно отформатировать с помощью звездочек или тире.
4. В Markdown вставить изображение в документ можно с помощью непосредственного указания адреса изображения.
5. Внутритекстовые формулы делаются аналогично формулам LaTeX. Например, формула $\sin^2(x) + \cos^2(x) = 1$ запишется как

`$\sin^2 (x) + \cos^2 (x) = 1$`

7 Выводы

В ходе лабораторной работы были освоены процедуры оформления отчетов с помощью легковесного языка разметки Markdown.