

Отчет по прохождению практики
Параллельные вычисления

Выполнил студент
группы Р3310
Шаймарданов Р. Р.

Руководитель
Соснин В. В.

Содержание

1	Введение	2
2	Материалы для ознакомления	3
2.1	Система компьютерной верстки $\text{\TeX}(\text{\LaTeX})$	3
2.1.1	Описание	3
2.1.2	Сравнение с другими программными средствами . . .	3
2.1.3	Выбор инструмента редактирования	5
2.2	Система контроля версий Git	6
2.2.1	Описание	6
2.2.2	Сравнению с другими системами контроля версий . .	7
2.2.3	Основные команды	8
2.2.4	GitHub	9
3	Исследование обучающих материалов	10
3.1	Описание	10
3.2	Критерий сравнения	10
3.3	Иностранные материалы	10
3.4	Российские материалы	10
4	Используемая литература	11

1 Введение

Здесь будут слова о том, чем я занимаюсь на своей практике

2 Материалы для ознакомления

2.1 Система компьютерной верстки \TeX (\LaTeX)

2.1.1 Описание

\TeX — это низкоуровневый язык разметки и программирования, созданный Дональдом Кнудом для единообразной вёрстки документов. Кнут начал разрабатывать систему набора текста \TeX в 1977 году для исследования потенциальных возможностей оборудования цифровой печати, которое в то время начинало проникать в издательское дело. Главным образом он надеялся улучшить качество печатной продукции, которое расстраивало в его собственных книгах и статьях. После выпуска в 1989 году поддержки восьмибитных символов разработка \TeX приостановилась, только иногда выходили версии с исправленными ошибками.

\LaTeX — основанный на \TeX пакет макросов, созданный Лесли Лампортом. Основная цель — упростить вёрстку текста, особенно в документах с математическими формулами. Значительно позднее авторы разработали для \LaTeX расширения, которые называются пакетами или стилями. Некоторые из них распространяются вместе с большинством дистрибутивов $\text{\TeX}/\text{\LaTeX}$.

Так как \LaTeX содержит часть команд \TeX , то создание документа в \LaTeX — тоже программирование: создаётся текстовый файл в \LaTeX разметке, макросы \LaTeX обрабатывают его и производят конечный документ.

2.1.2 Сравнение с другими программными средствами

Подход \LaTeX к созданию документа называется WYSIWYM¹: во время набора текста Вы не видите окончательный вариант документа, толь-

¹What You See Is What You Mean (То, что ты видишь, есть то, что ты имеешь в виду)

ко логическую структуру этого документа. Оформлением занимается сам \LaTeX . Такой подход имеет как достоинства, так и недостатки по сравнению с WYSIWYG² программами, такими как Openoffice.org Writer или Microsoft Word.

Достоинства:

- Файлы с исходными текстами можно просмотреть в любом текстовом редакторе, они понятнее в отличие от сложных бинарных файлов и форматов XML, используемых WYSIWYG программами.
- Вы полностью сосредотачиваетесь на структуре и содержании документа и забываете о том, как будет выглядеть печатный вариант.
- Не нужно вручную настраивать шрифты, размер текста, высоту строк или читаемость текста.
- Легко скопировать структуру документа в другой документ, в WYSIWYG программах не всегда ясно, какое именно было использовано форматирование.
- Разметка, шрифты, таблицы и т.д. согласованы во всём документе.
- Легко набирать математические формулы.
- Легко создаются алфавитные указатели, сноски, ссылки и библиографические списки.
- Так как исходный документ содержит просто текст, с помощью программных средств на любом языке программирования можно создать таблицы, рисунки, формулы и т.д.

Недостатки:

²What You See Is What You Get (Что видишь, то и получишь)

- Во время редактирования документа нельзя (обычно) увидеть его окончательный вариант.
- Необходимо знать нужные команды разметки \LaTeX .
- Иногда сложно получить требуемый вид документа.

Документ \LaTeX — обычный текстовый файл, в котором указано содержание документа вместе с дополнительной разметкой. При обработке исходного файла макросами \LaTeX можно получить документ в разных форматах. Изначально \LaTeX поддерживает форматы DVI и PDF, но при использовании другого ПО можно легко получить PostScript, PNG, JPG и т.д.

2.1.3 Выбор инструмента редактирования

В ходе изучения всех возможных вариантов работа с \LaTeX для создания данного отчета, была выбрана программа Textmaker

Выбор Textmaker'а обусловлен следующими его особенностями:

- Автоматическая подсветка синтаксиса
- Функция автодополнения команд \LaTeX
- Соккрытие блоков кода (Code folding)
- Быстрая навигация по структуре документа
- Указание на строку с ошибкой, для быстрой отладки
- Интегрированный просмотр PDF

2.2 Система контроля версий Git

2.2.1 Описание

Система управления версиями — программное обеспечение для облегчения работы с изменяющейся информацией. Система управления версиями позволяет хранить несколько версий одного и того же документа, при необходимости возвращаться к более ранним версиям, определять, кто и когда сделал то или иное изменение, и многое другое.

Git — это гибкая, распределённая система управления версиями. Проект был создан Линусом Торвальдсом для управления разработкой ядра Linux, первая версия выпущена 7 апреля 2005 года. На сегодняшний день его поддерживает Джунио Хамано. Программа является свободной и выпущена под лицензией GNU GPL версии 2.

У каждого разработчика, использующего Git, есть свой локальный репозиторий, позволяющий локально управлять версиями. Затем, сохранёнными в локальный репозиторий данными, можно обмениваться с другими пользователями. Часто при работе с Git создают центральный репозиторий, с которым остальные разработчики синхронизируются. В этом случае все участники проекта ведут свои локальные разработки и беспрепятственно скачивают обновления из центрального репозитория. Когда необходимые работы отдельными участниками проекта выполнены и отлажены, они, после удостоверения владельцем центрального репозитория в корректности и актуальности проделанной работы, загружают свои изменения в центральный репозиторий. Работа над версиями проекта в Git может вестись в нескольких ветках, которые затем могут с лёгкостью полностью или частично объединяться, уничтожаться, откатываться и разрастаться во все новые и новые ветки проекта.

2.2.2 Сравнению с другими системами контроля версий

Достоинства:

- Надежная система сравнения ревизий и проверки корректности данных, основанные на алгоритме хеширования Secure Hash Algorithm 1.
- Гибкая система ветвления проектов и слияния веток между собой.
- Наличие локального репозитория, содержащего полную информацию обо всех изменениях, позволяет вести полноценный локальный контроль версий и заливать в главный репозиторий только полностью прошедшие проверку изменения.
- Высокая производительность и скорость работы.
- Удобный и интуитивно понятный набор команд.
- Множество графических оболочек, позволяющих быстро и качественно вести работы с Git'ом.
- Возможность делать контрольные точки, в которых данные сохраняются без дельта компрессии, а полностью. Это позволяет уменьшить скорость восстановления данных, так как за основу берется ближайшая контрольная точка, и восстановление идет от нее. Если бы контрольные точки отсутствовали, то восстановление больших проектов могло бы занимать часы.
- Широкая распространенность, легкая доступность и качественная документация.
- Гибкость системы позволяет удобно ее настраивать и даже создавать специализированные контрольные системы или пользовательские интерфейсы на базе git.
- Универсальный сетевой доступ с использованием протоколов http, ftp, rsync, ssh и др.

Недостатки:

- Unix – ориентированность. На данный момент отсутствует зрелая реализация Git, совместимая с другими операционными системами.
- Возможные (но чрезвычайно низкие) совпадения хеш - кода отличных по содержанию ревизий.
- Не отслеживается изменение отдельных файлов, а только всего проекта целиком, что может быть неудобно при работе с большими проектами, содержащими множество несвязных файлов.
- При начальном (первом) создании репозитория и синхронизации его с другими разработчиками, потребуется достаточно длительное время для скачивания данных, особенно, если проект большой, так как требуется скопировать на локальный компьютер весь репозиторий.

2.2.3 Основные команды

add: Добавляет содержимое рабочей директории в индекс для последующего коммита.

status: Показывает состояния файлов в рабочей директории и индексе: какие файлы изменены, но не добавлены в индекс; какие ожидают коммита в индексе. Вдобавок к этому выводятся подсказки о том, как изменить состояние файлов.

diff: Используется для вычисления разницы между любыми двумя Git деревьями.

difftool: Запускает внешнюю утилиту сравнения для показа различий в двух деревьях, на случай если вы хотите использовать что-либо отличное от встроенного просмотрщика `git diff`.

commit: Берёт все данные, добавленные в индекс с помощью `git add`, и сохраняет их слепок во внутренней базе данных, а затем сдвигает указатель текущей ветки на этот слепок.

reset: Используется в основном для отмены изменений. Она изменяет указатель HEAD и, опционально, состояние индекса.

rm: Используется в Git для удаления файлов из индекса и рабочей директории. Она похожа на `git add` с тем лишь исключением, что она удаляет, а не добавляет файлы для следующего коммита.

mv: Удобный способ переместить файл, а затем выполнить `git add` для нового файла и `git rm` для старого.

clean: Удаление мусора из рабочей директории. Это могут быть результаты сборки проекта или файлы конфликтов слияний.

2.2.4 GitHub

GitHub — крупнейший веб-сервис для хостинга IT-проектов и их совместной разработки. Основан на системе контроля версий Git и разработан на Ruby on Rails и Erlang компанией GitHub, Inc (ранее Logical Awesome).

Для выполнения практической работы создан репозиторий в аккаунте [RandomRuslan](#) на GitHub'e.

3 Исследование обучающих материалов

3.1 Описание

3.2 Критерий сравнения

3.3 Иностранные материалы

3.4 Российские материалы

4 Используемая литература

1. <https://ru.wikibooks.org/wiki/LaTeX>
2. <https://git-scm.com/book/ru/v2>