Разработка методики работы с системой контроля версий разрабатываемого продукта

Обучающегося 4 курса
очной формы обучения
Елистратова Ивана Владимировича

Актуальность исследования.

На сегодняшний день практически во всех сферах из области разработки программного обеспечения применяются системы контроля версий, поэтому владение навыками работы с системой контроля версий является одним из базовых знаний на пути разработчика.

Предмет исследования

Методика работы с основными инструментами предоставляемыми системой контроля версий.

Теоретическая значимость

Заключается в рассмотрении каждого отдельно взятого инструмента системы контроля версий.

Практическая значимость

Заключается в разработке методики работы с системой контроля версий для разрабатываемого продукта, при помощи которой начинающие разработчики смогут овладеть основными инструментами системы контроля версий.

Этапы разработки программного продукта

аналитика;

техническое задание;

проектирование и дизайн;

разработка;

тестирование и стабилизация;

запуск приложения;

поддержка;

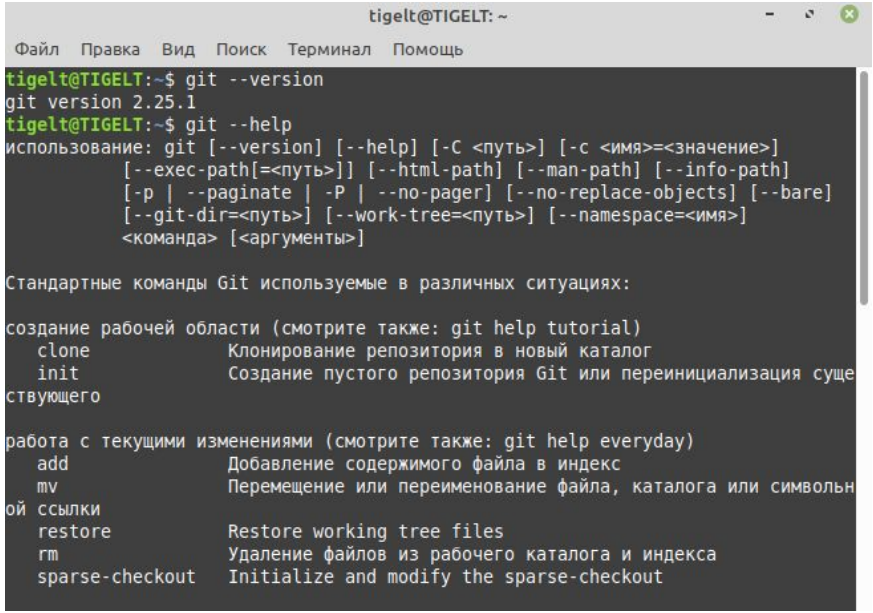
Технологии разработки приложений

- Progressive web application (PWA)
- Multi-page application (MPA)
- Single-page application (SPA)

Базовая методика работы с системой

1. Начало работы с проектом
2. Ежедневный цикл работы
3. Ветвления
4. Слияние версий
5. Конфликты и их разрешение
6. Блокировки
7. Версии проекта, теги

Основные команды git



Заключение

Обобщая методику использования системы контроля версий, а в частности гита, стоит отметить не какие то жесткие требования, а некоторые практики, которых стоит придерживаться при работе.

Во первых что касается работы в ветках, не стоит работать в ветке мастер, если нам нужно сделать какой то новый функционал, стоит отвести для этого отдельную ветку, закоммитить туда свои правки, если нужно сделать несколько коммитов, протестировать то что у вас получилось, а потом уже стоит делать сливать нашу ветку с основной.

Во вторых что касается сообщений сопровождающих коммит, их стоит писать императивно, как можно короче, но при этом, чтобы они содержали в себе как можно больше информации, это стоит делать для того, чтобы при просмотре коммитов разработчику было понятно, что конкретно было сделано в данном коммите, не открывая сам коммит и не просматривая сами правки.

В третьих, в истории коммитов стоит придерживаться линейности.