# Презентация спикера

Добрый день!

Меня зовут Александр Казмирчук. В настоящее время работаю системным администратором. Появилось желание изучить программирование на С++.

Изначально занимался самостоятельно по учебным сайтам и книгам. Но без практики невозможно освоить такой сложный язык, а задание сам себе не придумаешь. Поэтому, накопив элементарный багаж теоретических знаний, решил поступить на курсы.

Почему я выбрал именно С++. Как известно, С++ - сложный язык и очень мощный. Позволяет выжимать из кода максимум производительности, и в то же время, есть возможность шаблонного метапрограммирования и рефлексии (особенно с последними стандартами языка), что придаёт языку образность и …

На курсах я получил практический опыт, особенно при написании проекта. Как выяснилось при работе над проектом, самое трудное это грамотно организовать архитектуру программы, чтобы отдельные классы логично взаимодействовали между собой, и в то же время были достаточно независимы.

В дальнейшем, после защиты, продолжу изучение языка и технологий с ним связанных, и буду менять профессию на программиста.

144 слова

Нарисовать образно на 1 слайде рабочее место админа (сеть, сервер ,СХД, виндовс, линукс, маршрутизатор)

Нарисовать образно на 2 слайде, что могу программировать ядро, консольное приложение и оконное

Как-то отметить, что изучил подробно по книге и на курсах Qt/QML

-----------------------------------------------------------------------------------

- кто я сейчас

- чему я научился, что я умею благодаря курсам (зачем мне курс)

- мои цели после обучения, чему я обучался

1 минута, 2 слайда, 100 слов

# Цели и задачи проекта

Цель проекта была разработать локальный поисковый движок по файлам, осуществляющий поиск документов по запросам и выдающий релевантные ответы.

В организациях бывает потребность в поиске информации по базе документов.

Проект решает 4 задачи: ниже описание

Программе указываются в файлах JSON пути документов и запросы. По документам строится база инвертированного индекса, на основе которого определяется релевантность ответа (содержимого документа) для каждого запроса. По завершении поиска результаты выводятся в файл JSON, где для каждого запроса указан документ со степенью релевантности.

явно мало слов, надо добавить что-то. Думаю надо расписать чуть подробнее. Обязательно расписать JSON-файлы.

нарисовать на 1 слайде входящие документы и запросы

нарисовать на 2 слайде, что строится база инвертированного индекса. Что представляет собой инвертированный индекс

нарисовать на 3 слайде, определение релевантности ответа. Что такое релевантность

нарисовать на 4 слайде, что результаты выводятся в файл JSON

------------------------------------------------

- с каким проектом я работал

- какой цели мне надо было добиться

- каких изменений надо было достичь

2 минуты, 4 слайда, 180 слов

# Реализация проекта

На слайде представлены основные классы и общий алгоритм работы программы. Классы сконструированы так, что напрямую между собой никак не связаны. Всей работой программы (взаимодействием между классами) управляет основной класс SearchEngine.

Класс ConverterJSON реализует работу с JSON-объектами. Только он знает, что содержится в объекте, где взять входные данные и в каком виде выдать результат.

Далее класс Documents реализует чтение документов из файлов, предоставленных классом ConverterJSON и хранит документы в базе. Причём, чтение документов происходит в многопоточном режиме.

Далее класс Requests читает запросы, предоставленных классом ConverterJSON и хранит их в базе.

Далее класс InvertedIndex рассчитывает инвертированную индексацию документов на основе предоставленной базой документов классом Documents. Обязательно сказать, что такое инвертированная индексация. Причём, расчёт инвертированного индекса происходит в многопоточном режиме.

Далее класс RelevantResponse определяет релевантность ответов на основе базы инвертированных индексов и запросов, предоставленных классами InvertedIndex и Requests. Сказать, что такое релевантность.

Далее класс ConverterJSON записывает в JSON-объект результаты работы класса RelevantResponse, как ответы.

Чтение и запись текстовых файлов и JSON-файлов осуществляется посредством класса kav::OperationFileAndJSON. Разработан как полноценная библиотека (подробнее сказать). Все возможные ошибки работы с файлами и проверки JSON-структуры осуществляются в нём.

Логирование всех событий и обработка ошибок (работы с файлами) происходит в классе DispatcherOperationValidity. Который для логирования событий использует класс kav::Logger. Во общих чертах описать эти два класса, как дополнительные.

Всей работой управляет класс SearchEngine и …

Получилось 200 слов, надо добавить

- нарисовать на 1 слайде общий алгоритм работы программы (с классами)

- нарисовать на 2 слайде класс ConverterJSON и прописать его функции

- нарисовать на 3 слайде класс Documents и прописать его функции

- нарисовать на 4 слайде класс Requests и прописать его функции

- нарисовать на 5 слайде класс InvertedIndex и прописать его функции

- нарисовать на 6 слайде класс RelevantResponse и прописать его функции

- нарисовать на 7 слайде класс ConverterJSON и, что в результате получаем файл answers.json

- нарисовать на 8 слайде класс OperationFileAndJSON и прописать его функции

- нарисовать на 9 слайде вспомогательный класс DispatcherOperationValidity и прописать его функции

- нарисовать на 10 слайде класс SearchEngine и прописать его функции (поправить в Гитхабе)

- нарисовать на 11 слайде как Logger и прописать его функции

- нарисовать на 7 слайде , что есть класс для работы с файлами и логгер. И что их использует диспетчер.

- на 8 слайде нарисовать, что логгер использует другой поток для записи и другой процесс для вывода на экран. Как организовано межпроцессное взаимодействие.

- зачем я выполнял конкретный этап

- что я сделал

- как я это сделал

- результат выполнения этапа

4 минуты, 8 слайдов, 350 слов

# Результат проекта

- какие изменения произошли, должны произойти в результате применения предложений

- на что это повлияет и как (желательно в цифрах)

3 минуты, 6 слайдов, 260 слов

Выводы: что конкретно делаем? В каких единицах измеряем? За счёт чего достигаем цель? Решение реализуемо? Соответствует цели? Задан конкретный срок реализации?

На 1 слайде нарисовать, как увеличилась производительность при многопоточности

На 2 слайде нарисовать, что имеем три независимых проекта: ядро, две полноценные библиотеки kav\_check и kav\_logger

На 3 слайде нарисовать, как выглядит работа программы с окном консоли, и что позволяет нормально логировать работу

Может что ещё