## АННОТАЦИЯ

Отчет о курсовой работе: \*\*\* с., \*\*\* рис., \*\*\* табл., \*\*\* приложение, \*\*\* источников.  
 Объект исследования – матрицы и операции над ними.

Цель работы – разработка веб-сайта для произведения математических операций над матрицами.

Метод исследования – изучение принципов разработки веб-приложений средствами фреймворка Django.

В работе были использованы технологии: HTML, CSS, Java Script, Pyhton, Django, PyCharm, AJAX.

В результате решения задачи был разработан сайт-калькулятор для произведения математических операций над матрицами разной размерности.

WEB-ПРИЛОЖЕНИE, JQUERY, HTML, CSS, JAVA SCRIPT, PYHTON, DJANGO, PYCHARM, AJAX

|  |  |  |
| --- | --- | --- |
| СОДЕРЖАНИЕ | | |
|  | ВВЕДЕНИЕ | 5 |
| 1 | Анализ предметной области | 6 |
|  | 1.1 Состояние вопроса | 6 |
|  | 1.2 Актуальность и цель работы | 8 |
| 2 | Техническое задание | 9 |
|  | 2.1 Общие требования к продукту | 9 |
|  | 2.2 Позиционирование продукта | 10 |
|  | 2.3 Функции продукта | 21 |
|  | 2.4 Сценарии использования продукта | 22 |
|  | 2.5 Функциональные требования | 26 |
| 3 | Проектирование программного продукта | 30 |
|  | 3.1 Обоснование средств разработки | 30 |
|  | 3.2 Проектирование математической модели |  |
|  | 3.3 Проектирование взаимодействия клиентской и серверной части |  |
| 4 | Разработка ПО | 41 |
|  | 4.1 Разработка пользовательского интерфейса |  |
|  | 4.2 Разработка серверной части | 41 |
|  | 4.3 Разработка взаимодействия клиентской и серверной части | 49 |
| 5 | Описание программного продукта | 56 |
|  | 5.1 Структура проекта | 56 |
|  | 5.2 Описание объектов и их взаимодействия | 58 |
|  | 5.3 Описание запросов | 60 |
| **6** | **Тестирование и внедрение** | **62** |
|  | 6.1 Тестирование разработанного ПО | **62** |
|  | **6.2** Установка программы | **64** |
| **ЗАКЛЮЧЕНИЕ** | | **68** |
| **СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ** | | **69** |
| **ПРИЛОЖЕНИЕ А (обязательное) Фрагменты листинга** | | **70** |
| **ПРИЛОЖЕНИЕ Б (обязательное) Экранные формы** | |  |
|  | |  |
|  | |  |
|  | |  |

## ВВЕДЕНИЕ

Термин «матрица» имеет много значений. Основное значение термин «матрица» имеет в математике. Ма́трица — математический объект, записываемый в виде прямоугольной таблицы элементов кольца или поля (например, целых или комплексных чисел), которая представляет собой совокупность строк и столбцов, на пересечении которых находятся её элементы. Количество строк и столбцов матрицы задают размер матрицы. Хотя исторически рассматривались треугольные матрицы, в настоящее время говорят исключительно о матрицах прямоугольной формы, так как они являются наиболее удобными и общими. Матрицы широко применяются в математике для компактной записи систем линейных алгебраических или дифференциальных уравнений. В этом случае, количество строк матрицы соответствует числу уравнений, а количество столбцов — количеству неизвестных. В результате решение систем линейных уравнений сводится к операциям над матрицами. Поэтому, целью курсовой работы стало создание веб-приложение для произведения операций над матрицами.

Исходя из указанной цели, можно выделить частные задачи, поставленные в курсовой работе:

1. Выделить основные операции над матрицами.
2. Проанализировать существующие аналоги и их реализацию.
3. В качестве результата анализа составить ТЗ для веб-приложения.
4. Изучить технологии: HTML, CSS, Java Script, Pyhton, Django, PyCharm, AJAX.
5. Разработать структуру и функционал веб-приложения, соответствующий техническому заданию.
6. Разработать клиентскую и серверную части программы.
7. Проанализировать разработанное веб-приложение.

**1 Анализ предметной области**

* 1. Состояние вопроса

Впервые матрицы упоминались ещё в древнем Китае, называясь тогда «волшебным квадратом». Основным применением матриц было решение линейных уравнений. Так же, волшебные квадраты были известны чуть позднее у арабских математиков, примерно тогда появился принцип сложения матриц. После развития теории определителей в конце 17-го века, Габриэль Крамер начал разрабатывать свою теорию в 18-ом столетии и опубликовал «правило Крамера» в 1751 году. Примерно в этом же промежутке времени появился «метод Гаусса». Теория матриц начала своё существование в середине XIX века в работах Уильяма Гамильтона и Артура Кэли. Фундаментальные результаты в теории матриц принадлежат Вейерштрассу, Жордану, Фробениусу. Термин «матрица» ввел Джеймс Сильвестр в 1850 г.

С развитием технического прогресса теория матриц находит все новые применения. Например, в программировании матрица – это двумерный массив, в электронике – набор проводников, которые можно замкнуть в точках их пересечений и так далее. Также, актуальность решения операций над матрицами возникает в сфере образования для проверки рений.

С целью выявления недостатков и достоинств, уточнения терминологии, были проанализированные некоторые из существующих аналогов: «Образовательные веб-сервисы» [\*\*\*], «Онлайн-калькулятор Math» [\*\*\*]. Данные аналоги были выбраны из общей массы, по причине того, что они являются одними из самых рейтинговыми в поисковых сетях Google и Яндекс.

|  |  |
| --- | --- |
| Критерий | Работа с матрицами |
| Название | Образовательные веб-сервисы |
| Страна | Россия |
| Функционал | На сайте собраны калькуляторы для решения матриц онлайн: нахождение определителя, умножение и сложение матриц, нахождение обратной матрицы и многие другие. |
| Достоинства | 1. У каждой программы есть описание, информация о данном действии над матрицей и формула, используемая программой для решения. 2. Онлайн калькуляторы подробно расписывают ход решения, что позволяет не только получить результат, но и научиться решать матричные задачи, найти и исправить ошибки или проверить правильность своего решения. |
| Недостатки | 1. Функционал данного сервиса разбит на отдельные веб-приложения. 2. При каждом вводе данных (выбор ранга и ячеек матрицы) веб-приложение перенаправляет пользователя на отдельную веб-страницу. |

|  |  |
| --- | --- |
| Критерий | Работа с матрицами |
| Название | Онлайн-калькулятор Math |
| Страна | Россия |
| Функционал | Используется для проверки своего решения по многим математическим и экономическим дисциплинам. |
| Достоинства | 1. У каждой программы есть описание, информация о данном действии над матрицей и формула, используемая программой для решения. 2. Результат решения - это отчет в формате Word (и Excel при необходимости), содержащий ход решения с комментариями, исходные формулы и выводы. |
| Недостатки | 1. Функционал данного сервиса разбит на отдельные веб-приложения. 2. После ввода данных происходит перезагрузка страницы. 3. Если в браузере включена блокировка рекламы, то решение появляется только через 30 секунд. |

В ходе анализа были обнаружены недостатки систем, связанные с тем, что для каждой операции существует отдельный сервис, а также с повышенным трафиком на интернет, так как происходит перезагрузка страницы.

1.2 Актуальность и цель работы

Создание разрабатываемого веб-приложения, актуально, так как актуальность произведения операций над матрицами растет, а текущие аналоги обладают не всеми нужными характеристиками для удовлетворения нужд пользователей. Внедрение разрабатываемого веб-приложения позволит:

1. Создавать квадратные и прямоугольные матрицы произвольных рангов
2. Производить основные операции над матрицами

Целью курсовой работы является разработка веб-приложения, которое позволит производить основные математические операции над матрицами.

Исходя из указанной цели, можно выделить частные задачи, поставленные в курсовой работе:

1. Выделить основные операции над матрицами.
2. Проанализировать существующие аналоги и их реализацию.
3. В качестве результата анализа составить ТЗ для веб-приложения.
4. Изучить технологии: HTML, CSS, Java Script, Pyhton, Django, PyCharm, AJAX.
5. Разработать структуру и функционал веб-приложения, соответствующий техническому заданию.
6. Разработать клиентскую и серверную части программы.
7. Проанализировать разработанное веб-приложение.

**2 Техническое задание**

2.1 Общие требования к продукту

Основная задача веб-приложения – произведение основных математических операций с матрицами. Из анализа существующих аналогов можно выделить следующие требуемые особенности веб-приложения:

1. Все операции и ввод данных должны производится в рамках одной веб-страницы
2. Результаты вычислений должны выводится на той же веб-странице без перезагрузки
3. В силу особенностей некоторых операций, следует ограничить пользователя при введении рангов матриц и данных

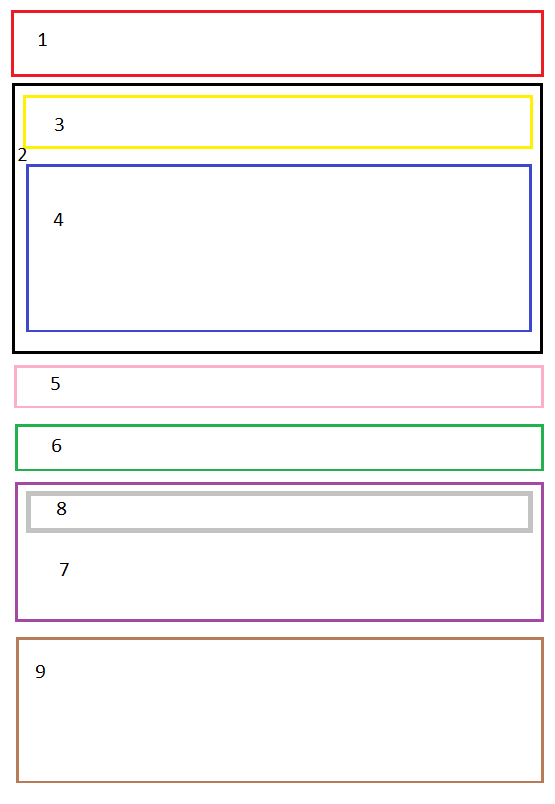
Интерфейс программы должен быть простым и понятным, неиспользуемые поля для некоторых операций следует скрыть, а нужные отображать с подсказкой об их функциональности.

2.2 Позиционирование продукта

2.2.1 Требование к пользовательским интерфейсам

Интерфейсом веб-приложения является веб-страница, отображаемая при входе на сайт. При ее создании в дизайне следует придерживаться простоты и понятности, а также текущих стандартов интерфейса веб-страниц.

Элементы веб-приложения следует разместить в вертикальной последовательности с подписями к каждому из них, как показано на рисунке \*\*\*.



На рисунке \*\*\* показаны следующие элементы интерфейса:

1. «Шапка» сайта с названием веб-приложения и кратким описанием
2. Блок для ввода начальной матрицы
3. Поля для ввода размеров матрицы
4. Таблица, содержащая поля для ввода значений ячеек матрицы
5. Элемент для выбора операции из списка
6. Поле для ввода числа (для операции «Умножение матрицы на число»)
7. Блок для ввода матрицы-операнда (для операций сложения, вычитания и умножения матриц)
8. Блок с элементом для ввода ширины матрицы-операнда (для операции «Умножение матриц»)
9. Блок с таблицей результирующей матрицы.

Для всех блоков, которые предназначены для некоторых операций реализовать их автоматическое появление и скрытие при выборе операций в соответствующем блоке. Вывод результата операции «Определитель матрицы» выводить в качестве модального окна.

2.2.2 Требования к программным интерфейсам

Код программы должен быть понятным, содержать комментарии, не содержать повторяющегося кода и быть готовым для дальнейшей разработки. Архитектура веб-приложения должна быть реализована в соответствии со стандартами MVC, а операции с матрицами должны быть реализованы в качестве отдельного python-модуля.

* + 1. Требования к пользователям продукта
* Подключение к сети Интернет.
* Базовый навык работы с компьютером.

2.3 Функции продукта

Требуемый функционал продукта:

* Создание матрицы произвольного размера
* Заполнение созданной матрицы
* Выполнение основных математических операций над матрицами

2.4 Сценарии использования продукта

Начальный сценарий: пользователь открывает веб-приложение в браузере.

Сценарии выбора математической операции:

Сценарий 1: Пользователь выбирает операцию сумма/вычитание матриц. На веб-странице ниже появляется матрица, такого же размера, как и исходная матрица.

Сценарий 2: Пользователь выбирает операцию умножение матриц. На веб-странице ниже появляется матрица, такой же высоты, как и у исходной матрицы, а над ней появляется элемент для выбора ширины матрицы-операнда.

Сценарий 3: Пользователь выбирает операцию умножение матрицы на число. На веб-странице ниже появляется блок для ввода числа, при этом блок для ввода второй матрицы пропадает

Сценарий 4: Пользователь выбирает операцию нахождения определителя или транспонирования матрицы. Все элементы для ввода вторых операндов пропадают.

Сценарии при нажатии кнопки подсчета:

Сценарий 1: Выбрана операция нахождения определителя. В результате появляется модальное окно с найденным определителем.

Сценарий 2: Выбрана операция отличная от нахождения определителя. В результате ответ помещается в блок с результирующей таблицей.

Сценарий 3: Выбрана любая операция, но выбраны не корректные данные. Появляется модальное окно, оповещающее об ошибке.

**3. Реализация программного продукта**

3.1 Обоснование средств разработки

3.1.1 HTML5

HTML5 (англ. HyperText Markup Language, version 5) — язык для структурирования и представления содержимого всемирной паутины. Это пятая версия HTML. Хотя стандарт был завершён (рекомендованная версия к использованию) только в 2014 году (предыдущая, четвёртая, версия опубликована в 1999 году), ещё с 2013 года браузерами оперативно осуществлялась поддержка, а разработчиками — использование рабочего стандарта (англ. HTML Living Standard). Цель разработки HTML5-улучшение уровня поддержки мультимедиа-технологий с одновременным сохранением обратной совместимости, удобочитаемости кода для человека и простоты анализа для парсеров.

В HTML5 реализовано множество новых синтаксических особенностей. Например, элементы <video>, <audio> и <canvas>, а также возможность использования SVG и математических формул. Эти новшества разработаны для упрощения создания и управления графическими и мультимедийными объектами в сети без необходимости использования сторонних API и плагинов. Другие новые элементы, такие как <section>, <article>, <header> и <nav>, разработаны для того, чтобы обогащать семантическое содержимое документа (страницы). Новые атрибуты были введены с той же целью, хотя ряд элементов и атрибутов был удалён. Некоторые элементы, например, <a>, <menu> и <cite>, были изменены, переопределены или стандартизированы.

* + 1. CSS3

CSS3 (англ. Cascading Style Sheets 3 — каскадные таблицы стилей третьего поколения) — активно разрабатываемая спецификация CSS. Представляет собой формальный язык, реализованный с помощью языка разметки. Самая масштабная редакция по сравнению с CSS1, CSS2 и CSS2.1. Главной особенностью CSS3 является возможность создавать анимированные элементы без использования JS, поддержка линейных и радиальных градиентов, теней, сглаживания и многое другое.

CSS3 преимущественно используется как средство описания и оформления внешнего вида веб-страниц, написанных с помощью языков разметки HTML и XHTML, но может также применяться к любым XML-документам, например, к SVG или XUL.

В отличие от предыдущих версий спецификация разбита на модули, разработка и развитие которых идёт независимо. CSS3 основан на CSS2.1, дополняет существующие свойства и значения и добавляет новые нововведения, начиная с малых, вроде закругленных углов блоков, заканчивая трансформацией (анимацией) и, возможно, введением переменных.

3.1.3 JavaScript

JavaScript — прототипно-ориентированный сценарный язык программирования. Является реализацией языка ECMAScript (стандарт ECMA-262).

JavaScript обычно используется как встраиваемый язык для программного доступа к объектам приложений. Наиболее широкое применение находит в браузерах как язык сценариев для придания интерактивности веб-страницам.

На JavaScript оказали влияние многие языки, при разработке была цель сделать язык похожим на Java, но при этом лёгким для использования непрограммистами. Языком JavaScript не владеет какая-либо компания или организация, что отличает его от ряда языков программирования, используемых в веб-разработке.

3.1.4 Twitter Bootstrap

Пользовательским интерфейсом является всплывающая, при нажатии на иконку расширения, HTML-страница. Для создания данной страницы использовался фреймворк Bootstrap 3 [\*\*\*]. Он предоставляет набор интрументов для создания сайтов и различных веб-страниц. Данный фреймворк включает в себя HTML и CSS шаблоны, а также включает в себя расширения Javascript.

Bootstrap позволяет создать адаптивный сайт с помощью резиновой системы разметки, которая масштабируется до 12 столбцов на различных устройсвах. Подобный масштаб позволяет создавать как простые варианты разметки, так и более сложные макеты.

3.1.5 Django framework

Django (Джанго) — свободный фреймворк для веб-приложений на языке Python, использующий шаблон проектирования MVC. Сайт на Django строится из одного или нескольких приложений, которые рекомендуется делать отчуждаемыми и подключаемыми. Это одно из существенных архитектурных отличий этого фреймворка от некоторых других (например, Ruby on Rails). Один из основных принципов фреймворка — DRY (англ. Don't repeat yourself)

Также, в отличие от других фреймворков, обработчики URL в Django конфигурируются явно при помощи регулярных выражений, а не выводятся автоматически из структуры моделей контроллеров.

Для работы с базой данных Django использует собственный ORM, в котором модель данных описывается классами Python, и по ней генерируется схема базы данных.

3.1.6 PyCharm IDE

PyCharm — интегрированная среда разработки для языка программирования Python. Предоставляет средства для анализа кода, графический отладчик, инструмент для запуска юнит-тестов. PyCharm разработана компанией JetBrains на основе IntelliJ IDEA.

Основным критерием выбора данной IDE является то, что она поддерживает веб-разработку на Django. Среди возможностей PyCharm, помогающих в разработке проекта можно выделить следующие:

* Создание каркаса проекта Django
* Запуск тестового сервера Django, нажатием одной кнопки
* Отображение лога сервера в интегрированной в IDE консоли

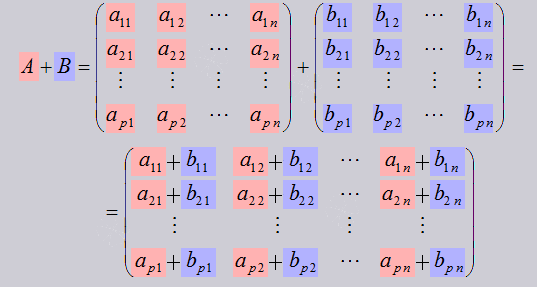
3.2 Проектирование математической модели

Матрицей называется прямоугольная таблица из чисел, содержащая некоторое количество m строк и некоторое количество n столбцов. Числа m и n называются порядками матрицы. В случае, если m = n, матрица называется квадратной, а число m = n - ее порядком.

3.2.1 Основные операции над матрицами.

Основными арифметическими операциями над матрицами являются умножение матрицы на число, сложение и умножение матриц.

Операция сложения определена только для матриц одного порядка. Сумма двух матриц А и B - это матрица, элементы которой равны сумме соответствующих элементов матриц А и В, как показано на рисунке \*\*\*.

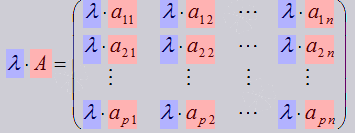


Из определения суммы матриц вытекает, что операция сложения матриц обладает теми же свойствами, что и операция сложения вещественных чисел, а именно:

1. Для матриц А, В и С одного порядка характерно свойство ассоциативности сложения А+(В+С) = (А+В)+С.
2. Для матриц данного порядка существует нейтральный элемент по сложению, которым является нулевая матрица. То есть, справедливо свойство А+О=А.
3. Для ненулевой матрицы А данного порядка существует матрица (–А), их суммой является нулевая матрица: А+(-А) = О.
4. Для матриц А и В данного порядка справедливо свойство коммутативности сложения А+В=В+А.

Операцию вычитания можно вывести из третьего свойства операции сложения (A-B = A+(-B), т.к. B = - (-B)).

Произведение матрицы и действительного (или комплексного) числа - это матрица, элементы которой получаются умножением соответствующих элементов исходной матрицы на число (рисунок \*\*\*). Операция умножения матрицы на число определена для матриц любого порядка.

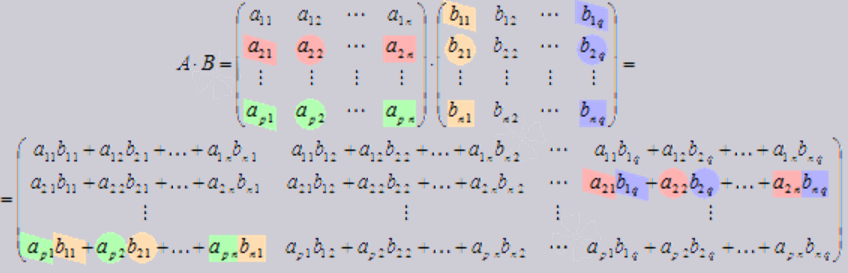


Свойства операции умножения матрицы на число:

1. Для матриц одного порядка А и В, а также произвольного действительного (или комплексного) числа справедливо свойство дистрибутивности умножения относительно сложения.
2. Для произвольной матрицы А и любых действительных (или комплексных) чисел выполняется свойство дистрибутивности.
3. Для произвольной матрицы А и любых действительных (или комплексных) чисел справедливо свойство ассоциативности умножения.
4. Нейтральным числом по умножению на произвольную матрицу А является единица.

Из свойств операции умножения матрицы на число следует, что умножение нулевой матрицы на число ноль даст нулевую матрицу, а произведение произвольного числа и нулевой матрицы есть нулевая матрица.

Произведение матрицы А порядка m\*n и матрицы В порядка n\*k - это такая матрица С m\*k, каждый элемент которой равен сумме произведений элементов i-ой строки матрицы А на соответствующие элементы j-ого столбца матрицы В, как на рисунке \*\*\*. Операция умножения двух матриц, A и B определяется только для случая, когда число столбцов матрицы A равно числу строк матрицы B.



Если матрицы А, В и С подходящих порядков, то справедливы следующие свойства операции умножения матриц.

1. Свойство ассоциативности умножения матриц A\*(B\*C) = (A\*B)\*C.
2. Два свойства дистрибутивности (A+B)\*C = A\*C + B\*C и A\*(B+C) = A\*B+A\*C.
3. В общем случае операция умножения матриц некоммутативна.
4. Единичная матрица Е порядка n на n является нейтральным элементом по умножению, то есть, для произвольной матрицы А порядка p на n справедливо равенство A\*E = A, а для произвольной матрицы А порядка n на p – равенство E\*A = A.

**4. Разработка ПО**

4.1 Создание структуры расширения

Создадим папку для расширения и добавим в нее файл «Manifest.json». Для описания расширения Google Chrome используется специальный файл Manifest.json, в котором в котором указаны все ресурсы расширения, его права доступа, название и так далее. Обязательными полями для этого файла являются:

* "manifest\_version" - в последних версиях браузера поддерживается только вторая версия манифеста
* "version". Данное поле должно иметь формат версии (например, «1.1» или «1.1.9» и т.д.).
* Поле «name» содержит название приложения.

По мимо обязательных есть еще и рекомендованные поля, ими являются:

* "default\_locale" – это поле для указания языка по умолчанию в мульти язычных расширениях.
* "description" – описание расширения.
* "icons": {...} – список с расположениями изображений иконок для расширений.

Приложение может работать как на фоне всего браузера, так и отдельно для каждой вкладки, для этого нужно указать поля "background"."scripts" и "browser\_action"."default\_popup" соответственно. Также, для того, чтобы использовать функциональность Chrome API, следует сперва указать необходимые права доступа (например, "contextMenus" или "tabs") в поле "permissions".

Для хранения будущих скриптов и изображений создадим папки scripst и images соответственно, а в них пустые файлы, чтобы можно было указать на них в файле описания, а в дальнейшей разработке наполнить их функционалом.

После заполнения необходимых полей, файл Manifest.json будет выглядеть, как на картинке \*\*\*.

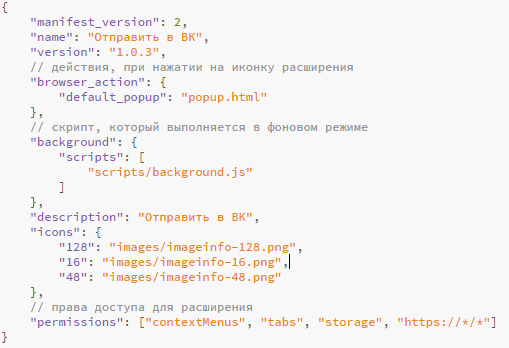
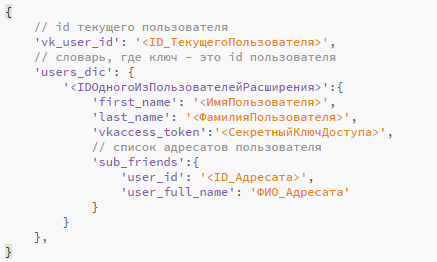
Назначение выбранных прав доступа можно узнать из таблицы \*\*\*, их список может дополнится при дальнейшей разработке.

Таблица \*\*\* – Используемые права доступа в расширении.

|  |  |
| --- | --- |
| Название | Описание |
| contextMenus | Доступ к контекстному меню браузера. |
| tabs | Доступ к вкладкам и их содержимому. |
| storage | Доступ к хранилищу Chrome. |
| https://\*/\* | Доступ к сторонним сайтам по https. |

Обязательной частью приложения является структура данных, которую оно хранит. Для хранения данных в Chrome API существует специальное хранилище, в котором можно хранить данные в формате JSON. С учетом требуемой функциональности можно создать следующую JSON структуру для хранения данных приложения (рисунок \*\*\*):



4.2 Создание пользовательского интерфейса

Для создания интерфейса расширения использовался фреймворк Twitter Bootstrap. Он предоставляет набор интрументов для создания сайтов и различных веб-страниц. Данный фреймворк включает в себя HTML и CSS шаблоны, а также включает в себя расширения Javascript.

Bootstrap позволяет создать адаптивные веб-страницы с помощью резиновой системы разметки, которая масштабируется до 12 столбцов на различных устройсвах. Подобный масштаб позволяет создавать как простые варианты разметки, так и более сложные макеты. Для верстки сайта использовались три основных класса, описанные в таблице \*\*\*.

Таблица \*\*\* – Основные используемые классы при верстке

|  |  |
| --- | --- |
| Класс | Описание |
| nav-tabs | Класс для создания переключателей вкладок |
| row | Базовый блок для колонок |
| col | Блок колонки |
| img-circle | Класс круглого изображения |

Система разметки Bootstrap также имеет собственные правила размещения элементов, которые рассмотрим ниже:

1. Элементы .row всегда помещаются в .container (с фиксированной шириной) или в .container-fluid (полная ширина) для корректного выравнивания и наполнения.
2. Строки используются для создания горизонтальной группы столбцов.
3. В столбцах помещается контент, только колонки могут быть непосредственными потомками строк.
4. Для быстрого создания сетки макетов используются стандартные классы сетки (например .row или .col-xs-4).
5. Колонки имеют собственный внутренний отступ (padding). Для первой и последней колонки он убран с помощью отрицательного отступа margin.
6. Столбцы сетки планируются с учетом максимального ограничения – 12 столбцов. Следовательно, чтобы создать три блока, нужно использовать класс .col-md-4
7. Если ограничение превышено – каждая следующая колонка переносится на новый ряд.

Класс вкладок требует, чтобы элемент имел и базовый класс nav, а также нужно использовать JavaScript-плагин для вкладок, который содержится в файле tab.js. Для корректной работы вкладок необходимо на каждый tab (вкадку) добавить атрибут data-toggle="tab" и атрибут href с идентификатором панели, которая будет отображаться при нажатии на этот tab (вкладку). Кроме этого, необходимо для каждой панели добавить атрибуты id="ИдентификаторПанели" и class="tab-pane". После этого все панели необходимо заключить в контейнер div с атрибутом class="tab-content".

Всплывающая страница приложения не должна быть статична и единственный способ ее генерировать, это можно сделать в скрипте на события загрузки страницы и переключения аккаунтов. Шаблонизатором списков была выбрана библиотека Mustache.js по причине своей простоты. С методами данной библиотеки можно ознакомится в таблице \*\*\*. Для того, чтобы указать HTML-элемент как шаблон, нужно указать атрибут type как "text/x-custom-template", при этом чтобы при отображении данный элемент игнорировался можно установить его как тег <script>.

Таблица \*\*\* – Методы библиотеки Mustache.js

|  |  |
| --- | --- |
| Метод | Описание |
| render(template, data, [partials]) | Подставляет данные из параметра data в HTML-шаблон из параметра templatе. |
| clearCache() | Очистка кеша. |
| compile(template, tags) | Компилирует шаблон в выполняемую функцию. |
| compilePartial (name, template, tags) | Компиляция подшаблона. |

На картинке \*\*\* приведен фрагмент разметки выпадающего окна расширения с использованием Bootstrap и Mustache. В нем можно увидеть разметку панели вкладок, а также вкладку «Аккаунты».



4.3 Разработка логики программы

4.3.1 Выделение необходимых методов

Логика программы должна хранится только в фоновом скрипте расширения, а файл popup.js должен отвечать за обновление содержимого всплывающей страницы и за вызов необходимых методов из файла логики. Например, при изменении состояния элемента checkbox во вкладке «Адресаты» должно изменятся контекстное меню и т.д.

Для начала, определим функции, которые помогут для отладки расширения при разработке, данные функции будет работать только при значении глобальной переменной isDebug равной true:

* Log(msg) – вывод сообщения в консоль браузера.
* LogObg(msg, obj) – вывод объекта с сообщением в консоль.

На основе технического задания можно выделить основные функции фонового скрипта:

* UpdateContextMenu() – обновляет контекстное меню браузера. В качестве параметров будут выступать значения переменных из локального хранилища данных.
* ClickHandler(usr, action) – основная функция расширения, которая выполняет отправку данных. В качестве параметров в функцию приходят id пользователя социальной сети и тип данных, который следует отослать.
* vkAuthorizationDialog() – функция, открывающая окно для получения прав доступа расширению от пользователя.

В качестве дополнительный функций можно выделить функции, связанные с составлением запросов, чтению параметров из адресной строки и т.д. Также, ради чистоты и читабельности кода сделаем работу со сторонним VK API в качестве отдельных методов.

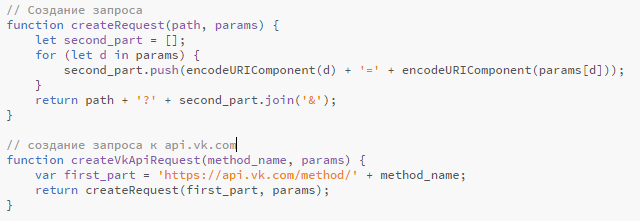
4.3.2 Разработка взаимодействия с используемыми API

Chrome API обеспечивает доступ ко всему, что связанно с браузером. Его методы можно использовать прямо в скриптах, поскольку они будут выполнятся в области видимости браузера. Основные методы, необходимые для написания расширения рассмотрены в таблице \*\*\*.

Таблица \*\*\* – Методы Chrome APIs

|  |  |  |
| --- | --- | --- |
| API | Метод | Описание |
| storage.local | get(params,callback) | Вызывает функцию callback с запрашиваемыми параметрами. |
| set(dict,callback) | Добавляет или изменяет данные в хранилище, после чего вызывает функцию callback. |
| contextMenus | removeAll(callback) | Удаляет все пользовательские меню. |
| create(menu) | Создает элемент контекстного меню и возвращает его id. |
| Tabs | getSelected(wnd,callback) | Возвращает текущую вкладку по id окна браузера в функию callback. |
| onUpdated. addListener | Добавляет функцию, которая будет выполнятся при обновлении вкладки. |

Для вызова методов VK API требуется выполнение https-запросов по специальным адресам и со специальными параметрами, поэтому, для облегчения работы с VK API и запросами в целом были созданы вспомогательные функции, которые можно наблюдать на рисунке \*\*\*.



 Далее, чтобы обратиться к функциям VK API достаточно соблюдать простой синтаксис, пример которого можно наблюдать в одной из основных функций расширения – отправка сообщения другу (рисунок \*\*\*).

Основные методы VK API, необходимые для написания расширения рассмотрены в таблице \*\*\*:

Таблица \*\*\* – Методы VK APIs

|  |  |  |
| --- | --- | --- |
| API | Метод | Описание |
| Messages | send | Отправляет сообщение другу. |
| Friends | Get | Удаляет все пользовательские меню. |
| Photos | GetMessagesUploadServer | Возвращает адрес сервера для загрузки фото в личное сообщение и другие параметры. |
| SaveMessagesPhoto | Сохраняет изображение после его загрузки на сервер и возвращает данные для прикрепления данного файла к сообщению. |
| Account | GetProfileInfo | Возвращает информацию о текущем аккаунте |

4.3.3 Разработка взаимодействия скриптов

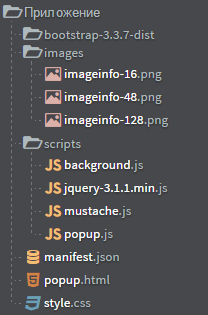
При открытии всплывающего окна расширения выполняется скрипт popup.js. В нем, при событии window.onload создается объект подключения с определенным именем, который виден во всей области видимости Chrome API. Такой же файл мы создадим и при запуске background.js. Далее, в обоих скриптах создадим функцию прослушивания данного порта следующего вида, как на рисунке \*\*\*. Данная структура позволит обрабатывать разнообразные сообщения, при этом, в объекте msg могут передаваться и другие данные (например, список аккаунтов или адресатов).



**5. Описание программного продукта**

* 1. Структура проекта

На рисунке \*\*\* изображена файловая структура проекта.



В целом, данная структура схожа со стандартной и часто рекомендуемой структурой веб-сайта. Скрипты и изображения расширения перенесены в отдельные папки для того, чтобы файловая структура была проще и логичнее, а единственный html-файл хранится в корне проекта вместе с файлом css, который в нем используется.

Также, в папке проекта содержится каталог с файлами фреймворка Bootstrap. Данный каталог содержит три подкаталога:

* css – подкаталог с файлами стилей.
* Fonts – подкаталог с шрифтами.
* Js – подкаталог с вспомогательными скриптами.
  1. Описание объектов и их взаимодействия

С учетом структуры расширения и используемых API можно составить общую схему взаимодействия объектов в расширении, как на рисунке \*\*\*:



Как можно заметить на рисунке \*\*\*, расширение выполняется внутри процесса браузера и содержит два основных скрипта, которые взаимодействуют между собой через объект extension. По мимо этого, фоновый скрипт расширения взаимодействует с VK API и локальным хранилищем браузера.

* 1. Описание запросов

Запросы в расширении были использованы для работы с VK API. Для создания запросов была использована функция createVkApiRequest (рисунок \*\*\*), а для работы с ними был использован обьект XMLHttpRequest.

Объект XMLHttpRequest (или, сокращенно, XHR) дает возможность браузеру делать HTTP-запросы к серверу без перезагрузки страницы. Несмотря на слово XML в названии, XMLHttpRequest может работать с данными в любом текстовом формате, и даже c бинарными данными. Его можно использовать как синхронным, так и асинхронным способом. За это отвечает третий параметр в функции send. Асинхронный вариант работы функции позволяет уменьшить «заморозки» в работе расширения. Пример работы данного обьекта можно наблюдать на рисунке \*\*\*.

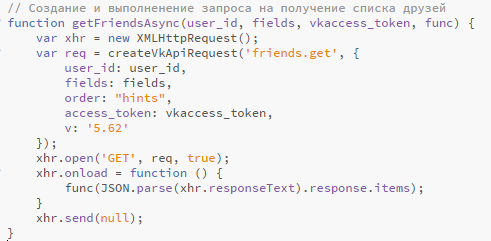
Рассмотрим детальнее запросы к VK API. Общими параметрами при любом запросе являются:

* «v» - версия API, к которой происходит запрос.
* «access\_token» - уникальный ключ пользователя, который позволяет определить личность отправителя.

Запрос на получение друзей пользователя вызывается вместе с параметрами:

* «user\_id» - id пользователя.
* «fields» - массив из названий полей, которые нужно извлечь (в нашем случае это только поле photo\_50).
* «vkacces\_token» - уникальный ключ пользователя, который позволяет определить личность отправителя.
* «order» - порядок, в котором будет отсортирован список из друзей («hints» - порядок по популярности).

Реализация данного запроса изображена на рисунке \*\*\*.



Далее рассмотрим загрузку изображения в сообщение. Первый этап – это загрузка изображения в формате blob и получение адреса сервера Вконтакте для дальнейшей загрузки с помощью метода «getMessagesUploadServer». Второй этап – это загрузка данных на сервер Вконтакте и при удачной загрузке – сохранение результатов с помощью метода «saveMessagesPhoto». После успешного сохранения изображения его можно прикреплять к сообщениям, как строку вида photo<idВладельца>\_<IdТранзакцииСохранения> к полю «attachment».

**6. Тестирование и внедрение**

6.1 Тестирование разработанного ПО

Назначение, область применения и требования к надежности программного продукта показаны в таблице \*\*\*, а результаты анализа надежности в таблице \*\*\*.

Таблица \*\*\* – Назначение, область применения и требования к надежности

|  |  |  |
| --- | --- | --- |
| Назначение | Область применения | Требования к надежности |
| Отправка данных из веб-страниц браузера Google Chrome в социальную сеть «Вконтакте». | Коммуникации | Сохранность данных, стабильная работа |

Таблица \*\*\* - Анализ надежности программного продукта

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| № Теста | Данные для теста  (вводимая информация) | Время функционирования программы | Результат (успешно\отказ) | Результат |
|  | Установка приложения | 2 секунды | успешно | Появление значка расширения в браузере |
|  | Добавление текущего авторизированного «Вконтакте» аккаунта | Меньше секунды | успешно | Появление окна авторизации в социальной сети |

Продолжение таблицы \*\*\*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Подтверждение прав доступа к данным социальной сети | Менее секунды | успешно | Закрытие окна авторизации, обновление вкладки «Аккаунты» |
|  | Отказ доступа к данным социальной сети | Менее секунды | успешно | Закрытие окна авторизации |
|  | Удаление аккаунта социальной сети из расширения | Моментально | успешно | обновление вкладки «Аккаунты» |
|  | Выбор адресата | Моментально | успешно | Изменение состояния элемента checkbox, напротив адресата |
|  | Клик по иконке расширения | Менее трех секунд | успешно | Открытое всплывающее окно расширения с открытой вкладкой «Адресаты» |
|  | Отправка данных | Менее секунды | успешно | Отправка сообщения в социальной сети |

* 1. Установка программы

Все, что нужно для расширения – это установленный браузер Google Chrome. Для его установки разработчики рекомендуют следующие параметры системы:

* Windows 7, Windows 8, Windows 8.1, Windows 10 или более поздней версии.
* Процессор Intel Pentium 4 или более поздней версии с поддержкой SSE2.

В данный момент расширение еще не внедрено и находится на стадии разработки, поэтому для его установки потребуются следующие действия:

1. Зайти в настройки браузера и перейти по меню Дополнительные инструменты -> Расширения.
2. Включить режим разработчика.
3. Загрузить расширение, нажав на кнопку «Загрузить распакованное расширение» и выбрав в файловом диалоговом окне папку с расширением.

После проделанных действий приложение будет загружено и установлено в браузер в течении пары секунд. Если в дальнейшем приложение будет модифицировано, то не нужно его перезагружать, а можно воспользоваться кнопкой «Обновить», после чего Google Chrome сам перезагрузит расширение. Также, в меню расширений можно вызвать инструменты разработчика для фоновой страницы расширения.

**ЗАКЛЮЧЕНИЕ**

В результате работы над курсовой работой выполнены все поставленные задачи. Разработано работоспособное расширение для браузера Google Chrome, которое позволяет передавать информацию в социальную сеть «Вконтакте». Проанализированы существующие аналоги, выделены базовые критерии оценки. Был изучен ряд технологий для создания проекта: C HTML, CSS, Java Script, Chrome JS APIs, VK API, Brackets, json. Также, было написано обоснование выбора данных технологий в курсовом проекте. Функционал расширения был утвержден согласно разработанной структуре программы. На основе разработанной функциональной схемы проекта составлен программный продукт, обладающий интерфейсной частью и работой со сторонними API.

На данный момент при работе с расширением доступны следующие возможности: добавление аккаунта социальной сети в список аккаунтов расширения, изменение текущего активного аккаунта и удаление аккаунтов из списка расширения, изменение списка адресатов для текущего аккаунта, отправка разных типов данных в социальные сети.

Дальнейшее развитие программы связано с расширением ее возможностей, улучшением уровня безопасности, добавление ее в магазин Google Extensions. Тестирование программы показало ее работоспособность и высокую степень надежности.

**СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ**

1. Интернет-магазин Chrome

URL: https://chrome.google.com/webstore/ (дата обращения: 14.02.2017)

1. Официальная страница фреймворка Bootstrap 3

URL: http://getbootstrap.com/ (дата обращения: 15.02.2017)  
 3. Wikipedia – информационный ресурс:

URL: https://ru.wikipedia.org/wiki/HTML5

(дата обращения: 15.02.2017)

URL: https://ru.wikipedia.org/wiki/JavaScript

(дата обращения: 15.02.2017)

URL: https://en.wikipedia.org/wiki/Brackets\_(text\_editor)

(дата обращения: 15.02.2017)

4. Документация API Вконтакте

URL: https://vk.com/dev/manuals (дата обращения: 20.02.2017)

5. JavaScript. Подробное руководство / Дэвид Флэнаган / Год: 2013

6. w3school: информационный ресурс

URL: https://www.w3schools.com/html/default.asp  
(дата обращения: 21.02.2017)

URL: https://www.w3schools.com/css/default.asp  
(дата обращения: 22.02.2017)

URL: https://www.w3schools.com/js/default.asp  
(дата обращения: 25. 02.2017)  
 7. Chrome Developer // [ресурс с материалами для разработки] //

URL: https://developer.chrome.com/extensions (дата посещения: 03.03.2017)

## ПРИЛОЖЕНИЕ А

(обязательное)

**Фрагменты листинга**

Листинг А.1 Файл background.js

var debug = true,

vkCLientId = '5886692',

vkRequestedScopes = 'docs,offline,messages,wall,photos',

vk\_default\_redirect\_uri = 'https://oauth.vk.com/blank.html',

files\_to\_share = [{

title: "Изображение",

context: ["image"]

},

{

title: "Видео",

context: ["video"]

},

{

title: "Ссылка",

context: ["link"]

},

{

title: "Страница",

context: ["all"]

},

{

title: "Текст",

context: ["selection"]

}];

function log(msg) {

if (debug)

console.log(msg);

}

function logObj(msg, obj) {

if (debug)

console.log(msg, obj);

}

chrome.extension.onConnect.addListener(function (port) {

log("Connected .....");

port.onMessage.addListener(function (msg) {

log("message recieved " + msg.message);

switch (msg.message) {

case "getFriends":

{

chrome.storage.local.get([

'users\_dic',

'vk\_user\_id'

], function (items) {

if (items.vk\_user\_id != undefined) {

log(items.vk\_user\_id);

log(items.users\_dic);

getFriendsAsync(items.vk\_user\_id, "photo\_50", items.users\_dic[items.vk\_user\_id].vkaccess\_token, function (friends) {

var mod\_friends = friends.map(function (friend) {

items.users\_dic[items.vk\_user\_id].sub\_friends.forEach(function (sub\_friend) {

if (friend.id == sub\_friend.user\_id) {

friend.checked = "checked";

}

//log(friend.user\_id + sub\_friend.user\_id)

});

return friend;

});

msg = {

message: "returnFriends",

friends: mod\_friends,

}

port.postMessage(msg);

});

} else {

msg = {

message: "returnFriends",

friends: {},

}

port.postMessage(msg);

}

});

}

break;

case "getAccounts":

{

chrome.storage.local.get([

'users\_dic',

'vk\_user\_id'

], function (items) {

var accounts = [];

for (var key in items.users\_dic) {

if (items.vk\_user\_id == key) {

items.users\_dic[key].checked = "checked";

}

items.users\_dic[key].id = key;

accounts.push(items.users\_dic[key]);

}

var response = {

message: "returnAccounts",

accounts: accounts,

}

port.postMessage(response);

});

}

break;

case "ChangeFriend":

{

log(msg.user\_id + " " + msg.checked);

chrome.storage.local.get([

'users\_dic',

'vk\_user\_id'

], function (items) {

if (msg.checked == true) {

items.users\_dic[items.vk\_user\_id].sub\_friends.push({

'user\_id': msg.user\_id,

'user\_full\_name': msg.user\_full\_name

});

} else {

items.users\_dic[items.vk\_user\_id].sub\_friends = items.users\_dic[items.vk\_user\_id].sub\_friends.filter(function (e) {

return JSON.stringify(e) !== JSON.stringify({

'user\_full\_name': msg.user\_full\_name,

'user\_id': msg.user\_id

});

});

}

chrome.storage.local.set({

'users\_dic': items.users\_dic,

}, function () {

log("Friend sub list changed to - " + items.users\_dic[items.vk\_user\_id].sub\_friends.toString());

UpdateContextMenu();

});

});

}

break;

case "SetCurrentUser":

{

log(msg.user\_id + " " + msg.checked);

chrome.storage.local.set({

'vk\_user\_id': msg.user\_id,

}, function () {

log("Current user changed to - " + msg.user\_id);

UpdateContextMenu();

var response = {

message: "userChange",

}

port.postMessage(response);

});

}

break;

case "refreshUser":

{

vkAuthorizationDialog(function () {

UpdateContextMenu();

var response = {

message: "userRefresh",

}

port.postMessage(response);

});

}

break;

case "deleteUser":

{

log(msg.user\_id);

chrome.storage.local.get(['users\_dic', 'vk\_user\_id'],

function (result) {

delete result.users\_dic[msg.user\_id];

log(result.vk\_user\_id);

if (result.vk\_user\_id == msg.user\_id) {

chrome.storage.local.remove('vk\_user\_id');

}

log(result);

chrome.storage.local.set({

'users\_dic': result.users\_dic

}, function () {

UpdateContextMenu();

var response = {

message: "userDelete",

}

port.postMessage(response);

});

});

}

break;

}

});

});

// Вытягиваем параметры из адресной строки

function getUrlParameterValue(url, parameterName) {

"use strict";

var urlParameters = url.substr(url.indexOf("#") + 1),

parameterValue = "";

urlParameters = urlParameters.split("&");

for (var index = 0; index < urlParameters.length; index += 1) {

var temp = urlParameters[index].split("=");

if (temp[0] === parameterName) {

return temp[1];

}

}

return parameterValue;

}

// Создание запроса

function createRequest(path, params) {

let second\_part = [];

for (let d in params) {

second\_part.push(encodeURIComponent(d) + '=' + encodeURIComponent(params[d]));

}

return path + '?' + second\_part.join('&');

}

// создание запроса к api.vk.com

function createVkApiRequest(method\_name, params) {

var first\_part = 'https://api.vk.com/method/' + method\_name;

return createRequest(first\_part, params);

}

Листинг А.2 Файл popup.html

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<title>Document</title>

<!-- Latest compiled and minified CSS -->

<link rel="stylesheet" href="bootstrap-3.3.7-dist/bootstrap-3.3.7-dist/css/bootstrap.css">

<!-- jQuery library -->

<script src="scripts/jquery-3.1.1.min.js"></script>

<!-- Latest compiled JavaScript -->

<script src="bootstrap-3.3.7-dist/bootstrap-3.3.7-dist/js/bootstrap.js"></script>

<link rel="stylesheet" href="style.css">

<script src="scripts/mustache.js"></script>

<script src="scripts/popup.js"></script>

</head>

<body>

<ul class="nav nav-tabs">

<li class="active"><a data-toggle="tab" href="#accounts\_tab">Аккаунты</a></li>

<li><a data-toggle="tab" href="#friends\_tab">Адресаты</a></li>

<!-- <li><a data-toggle="tab" href="#settings\_tab">Настройки</a></li> -->

</ul>

<div class="tab-content">

<div id="accounts\_tab" class="tab-pane fade in active">

<button id="refresh\_user" type="button" class="btn btn-primary btn-block"> Добавить текущий аккаунт </button>

<hr>

<script id="account\_template" type="text/x-custom-template">

<label>

<input class="check\_account" type="radio" name="account\_radio" value="{{id}}" style="width:50px;height:50px" {{checked}} hidden>

<div class="account\_row">

<div class="account row">

<div class="col-xs-9">

<p>{{first\_name}}</p>

<p>{{last\_name}}</p>

</div>

<div class="col-xs-3">

<button class="delete\_account btn btn-danger" value="{{id}}" style="width:50px;height:50px;margin-top:4px;">X</button>

</div>

</div>

<div>

</label>

<hr>

</script>

<div id="accounts">

</div>

</div>

<div id="friends\_tab" class="tab-pane fade">

<script id="friend\_template" type="text/x-custom-template">

<label>

<div class="friend row">

<div class="col-xs-3">

<img src="{{photo\_50}}" class="img-circle">

</div>

<div class="col-xs-6">

<p>{{first\_name}}</p>

<p>{{last\_name}}</p>

</div>

<div class="col-xs-3">

<input class="check\_friend" type="checkbox" value="{{last\_name}} {{first\_name}} {{id}}" style="width:50px;height:50px" {{checked}}>

</div>

</div>

</label>

<hr>

</script>

<div id="friends">

</div>

</div>

<div id="settings\_tab" class="tab-pane fade">

</div>

</div>

</body>

</html>

Листинг А.3 Файл popup.js

window.onload = function () {

var port = chrome.extension.connect({

name: "Sample Communication"

});

port.onMessage.addListener(function (msg) {

switch (msg.message) {

case "returnFriends":

{

var friends\_list = document.getElementById("friends");

var friend\_template = document.getElementById("friend\_template");

friends\_list.innerHTML = "";

var friendAdd = function (target, template, friend) {

var friend\_li = Mustache.render(template.innerHTML, friend);

target.innerHTML += (friend\_li);

};

msg.friends.forEach(function (friend, i) {

friendAdd(friends\_list, friend\_template, friend);

});

$(".check\_friend").change(function (target) {

var full\_name = this.value.split(' ').slice(0, 2).join(' '),

user\_id = this.value.split(' ')[2];

var msg = {

message: "ChangeFriend",

user\_full\_name: full\_name,

user\_id: user\_id,

checked: this.checked,

}

port.postMessage(msg);

});

}

break;

case "returnAccounts":

{

var accounts\_list = document.getElementById("accounts");

var account\_template = document.getElementById("account\_template");

accounts\_list.innerHTML = "";

msg.accounts.forEach(function (account) {

var account\_li = Mustache.render(account\_template.innerHTML, account);

accounts\_list.innerHTML += (account\_li);

});

getFriends();

$(".check\_account").change(function (target) {

user\_id = this.value;

var msg = {

message: "SetCurrentUser",

user\_id: user\_id,

}

port.postMessage(msg);

});

}

break;

case "userDelete":

case "userRefresh":

case "userChange":

{

getAccounts();

}

break;

}

});

var getAccounts = function () {

port.postMessage({

message: "getAccounts"

});

}

var getFriends = function () {

port.postMessage({

message: "getFriends"

});

}

getAccounts();

$("body").on("click", ".delete\_account", function (target) {

user\_id = this.value;

port.postMessage({

message: "deleteUser",

user\_id: user\_id

});

});

$("#refresh\_user").click(function () {

port.postMessage({

message: "refreshUser"

});

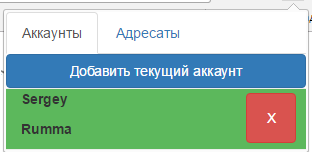
});

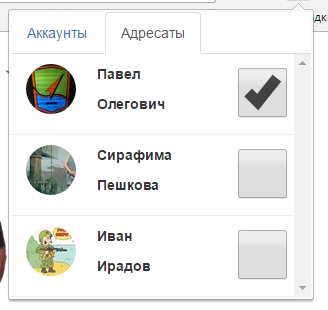
}

## ПРИЛОЖЕНИЕ Б

(обязательное)

**Экранные формы**



****