МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ

ВЫСШЕГО ОБРАЗОВАНИЯ

«НОВОСИБИРСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»

Кафедра \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_Вычислительной Техники\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(полное название кафедры)

**ВЫПУСКНАЯ КВАЛИФИКАЦИОННАЯ РАБОТА БАКАЛАВРА**

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_Торопова Никиты Александровича\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(фамилия, имя, отчество студента – автора работы)

\_\_\_\_\_\_\_\_\_Разработка Android-клиента кабинетов студента и преподавателя НГТУ\_\_\_\_\_\_\_\_\_

(тема работы)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_Факультет Автоматики и Вычислительной Техники\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(полное название факультета)

Направление подготовки \_\_\_\_\_\_\_09.03.04 «Программная инженерия»\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(код и наименование направления подготовки бакалавра)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

### Автор выпускной квалификационной работы

\_\_\_\_\_\_\_\_Торопов Н.А.\_\_\_\_\_\_

(фамилия, имя, отчество)

\_\_\_\_\_\_\_АВТФ АВТ-215\_\_\_\_\_\_

(факультет, группа)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(подпись, дата)

### Руководитель от НГТУ

\_\_\_\_\_\_\_\_Романов Е.Л.\_\_\_\_\_\_\_

(фамилия, имя, отчество)

\_\_\_\_\_\_\_\_\_к.т.н. доцент\_\_\_\_\_\_\_\_

(ученая степень, ученое звание)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(подпись, дата)

Новосибирск 2016

Реферат

Оглавление

[Автор выпускной квалификационной работы 0](file:///C:\Users\Admin\Documents\GitHub\Diplom\diplom\Отчет%20по%20диплому.docx#_Toc452151522)

[Руководитель от НГТУ 0](file:///C:\Users\Admin\Documents\GitHub\Diplom\diplom\Отчет%20по%20диплому.docx#_Toc452151523)

[Введение. 3](#_Toc452151524)

[Список терминов и сокращений. 4](#_Toc452151525)

[1. Постановка задачи и анализ предметной области. 6](#_Toc452151526)

[1.1 Постановка задачи. 6](#_Toc452151527)

[1.2 Обзор аналогов. 6](#_Toc452151528)

[1.2.1 Расписашка 6](#_Toc452151529)

[1.3 Архитектура Android. 7](#_Toc452151530)

[1.4 Методы авторизации в личном кабинете. 10](#_Toc452151531)

[1.5 Выбор средств разработки. 11](#_Toc452151532)

[1.5.1 Среда разработки. 11](#_Toc452151533)

[1.5.2 Библиотеки парсинга. 12](#_Toc452151534)

[1.5.3 Программа для записи запросов на сервер. 12](#_Toc452151535)

[1.6 Требования к программному продукту 12](#_Toc452151536)

[2. Структурная разработка 14](#_Toc452151537)

[2.1 Алгоритм получения доступа в личный кабинет. 14](#_Toc452151538)

[2.1.1 Авторизации при помощи *Android WebView* 14](#_Toc452151539)

[2.1.2 Авторизация при помощи запроса на сервер 16](#_Toc452151540)

# Введение.

Пока не представляю, что здесь писать. Возможно, будет текст о мобильных платформах и сайте НГТУ.

# Список терминов и сокращений.

***API*** (интерфейс программирования приложений, интерфейс прикладного программирования) (англ. *application programming interface*) — набор готовых классов, процедур, функций, структур и констант, предоставляемых приложением (библиотекой, сервисом) или операционной системой для использования во внешних программных продуктах. Используется программистами при написании всевозможных приложений.

**Личный кабинет** – *Web*-Сайт для преподавателей и студентов хранящий в себе необходимую для работы и учёбы информацию, доступ к которому, имеет индивидуально каждый студент и преподаватель.

Пример личного кабинета:

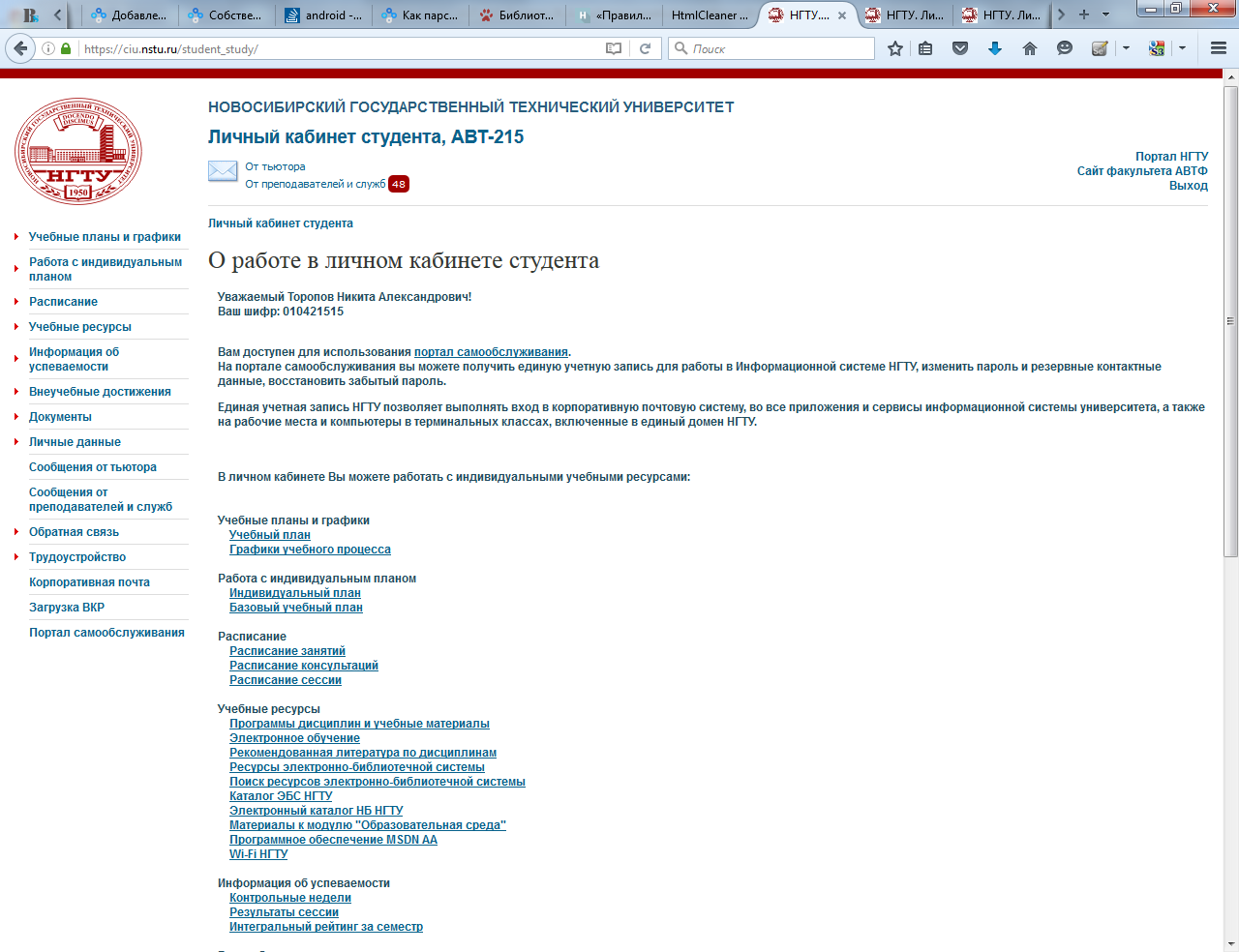


Рисунок 1. Личный кабинет студента.

**Синтаксический анализ «Парсинг»** - процесс сопоставления линейной последовательности лексем (слов, токенов) естественного или формального языка с его формальной грамматикой. Результатом обычно является дерево разбора.

**Токен -** идентификатор доступа выдаваемы после успешной авторизации. Хранится в *Cookie* браузера и передается в заголовке запроса к серверу.

***AJAX -*** *Asynchronous Javascript And Xml* – технология обращения к серверу без перезагрузки страницы;

***IDE***- Интегрированная среда разработки. Система программных средств, используемая программистами для разработки программного обеспечения.

# Постановка задачи и анализ предметной области.

## Постановка задачи.

Основную задачу можно сформулировать следующим образом: необходимо разработать программу обеспечивающую доступ в личный кабинет преподавателя или студента НГТУ с ограниченным функционалом. Такая формулировка включает в себя ряд подзадач:

* Разработка модуля авторизации
* Анализ функциональных возможностей личных кабинетов и выбор основных функций.
* Разработка модуля «личный кабинет» студента
* Разработка модуля «личный кабинет» преподавателя.

Для решения поставленной задачи необходимо провести анализ существующих решений представленных подзадач, выявить их достоинства и недостатки, и на основе результатов сравнения функциональных возможностей выбрать наиболее приемлемый вариант для собственной разработки.

## Обзор аналогов.

На сегодняшний день рынок мобильных приложений «пестрит» всевозможными приложениями любой функциональности. Следовательно, почти любая идея, выбранная для реализации, уже имеет свои аналоги и решения. Так как личный кабинет является частью Web-сайта НГТУ, мое приложение является неким клиентом для работы с данным сайтом. Аналогом можно представить приложение «Расписашка».

### Расписашка

Данное приложение является клиентом для работы с Web-сайтом НГТУ, а именно с расписанием занятий студентов. Приложение предоставляет возможность выбора факультета и группы, для которой необходимо вывести расписание.

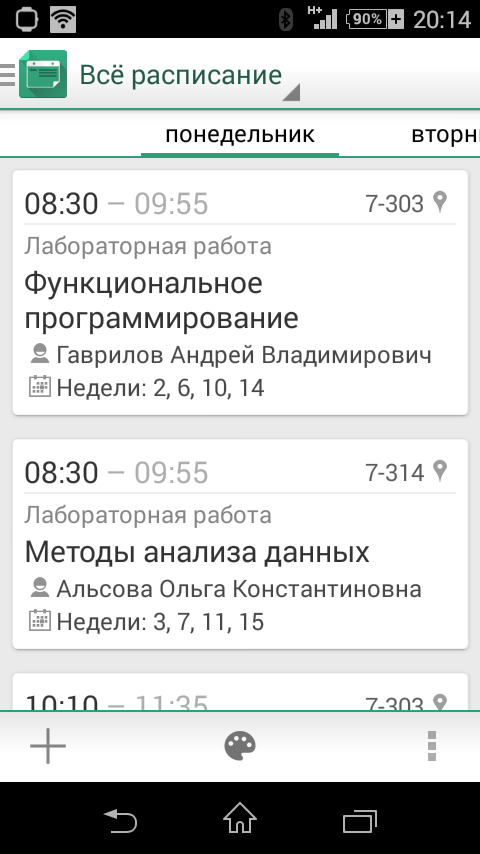


Рисунок 1.1 Внешний вид приложения «Расписашка».

Данное приложение предоставляет доступ к расписанию с Web-сайта НГТУ и не предоставляет доступ в «личный кабинет». Таким образом, можно предположить, что аналогов разрабатываемому приложению нет.

## Архитектура Android.

В настоящее время среди всех мобильных платформ наибольшей популярностью пользуются платформы iOS и Android. Однако, несмотря на то, что iOS более дружелюбна к пользователям, практична в вопросах функциональности и стабильна, нежели Android, в плане разработки она имеет больше минусов, чем плюсов. Средства разработки приложений под iOS, тестирование приложения, доступ к форумам разработчиков являются платными, разрабатывать приложение можно исключительно на компьютере с операционной системой Mac OS. Средства разработки Android, в свою очередь, находятся в свободном доступе и могут быть установлены на любой платформе, а при возникновении тупиковой ситуации всегда можно найти решение проблемы в сети или спросить у более опытных разработчиков. Язык разработки iOS — Objective-C, что тоже является минусом, поскольку я не имею достаточного для реализации задач опыта программирования на данном языке.



Рисунок 1.2 Сравнение iOS и Android.

Архитектура Android представляет собой программный стек для мобильных устройств, включающий в себя операционную систему, связующее программное обеспечение и ключевые приложения.



Рисунок 1.3 Основные компоненты архитектуры Android[3].

Как видно из рисунка 1.3, архитектуру операционной системы Android можно разделить на пять связанных уровней:

* Уровень приложений. Android поставляется с набором стандартных приложений.
* Уровень каркаса приложений (фреймворк). Android предоставляет разработчикам возможность разработки мощных функциональных приложений: они имеют доступ к тем же функциям, структурам и интерфейсам, которые используются при создании встроенных приложений.
* Уровень библиотек. Android включает в себя набор библиотек, реализованный на языке С/С++. Фреймворк позволяет разработчикам использовать эти библиотеки.
* Уровень среды исполнения. Платформа использует регистр-ориентированную виртуальную машину Dalvik. Каждое приложение запускается в собственном процессе и работает с уникальным экземпляром виртуальной машины.
* Уровень ядра Linux. Для работы с основными системными сервисами (управление процессами, памятью, обеспечение безопасности, работа с драйверами) используется Linux версии 2.6.

Самым большим плюсом ОС Android с точки зрения реализации поставленных задач является возможность доступа к методам, функциям и структурам, используемым при разработке стандартных приложений.

Таким образом, по таким критериям как возможности при разработке, доступность телефона с выбранной платформой и доступность средств разработки мною была выбрана платформа Android.

## Методы авторизации в личном кабинете.

Для получения доступа к личному кабинету необходимо иметь токен передаваемый в куки. Для его получения можно использовать следующие варианты:

1. Загрузка страницы авторизации при помощи Android WebView прохождение авторизации, и извлечение куки.
2. Отправка запроса на сервер с логином и паролем пользователя и получение ответа содержащего токен.

В первом случае происходит загрузка всего контента страницы авторизации и это расходует трафик мобильного интернета, а так же загрузка происходит довольно долго, так как существует большое количество перенаправлений и «тяжелый контент». Так же страница авторизации не является адаптивной и не удобна в использовании на мобильных телефонах.

Во втором случае трафик мобильного телефона расходуется гораздо меньше, так как отсутствует какой-либо загружаемый контент. Вдобавок ко всему это ускоряет процесс авторизации, так как нет необходимости ждать загрузки страницы авторизации. Запрос отправляется напрямую на сервер и с него же получает ответ. Так же это позволяет создать более удобную форму для авторизации пользователя.

## Выбор средств разработки.

### Среда разработки.

Разработчик приложения под ОС *Android* имеет довольно большой выбор средств разработки: *IntelliJ IDEA, NetBeans, Eclipse*, а так же комплект *Android SDK* на основе *Eclipse IDE* и комплект *Android Studio* на основе *IntelliJ IDEA IDE*. Выбор среды разработки основывался наследующих требованиях:

* Наличие удобного редактора кода;
* Наличие эффективных средств отладки, позволяющих работать с кодом в пошаговом режиме;
* Наличие графического редактора *XML*-файлов;
* Наличие навыков и опыта использования выбранной среды разработки.

Результаты сравнения интегрированных сред представлены в таблице 1.

Таблица 1 — Сравнение интегрированных средств разработки

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Редактор кода | Средства отладки | *XML*-редактор | Опыт использования |
| *IntelliJ IDEA* | + | + | + | + |
| *NetBeans* | + | + | - | + |
| *Eclipse* | + | + | + | - |

Мною был выбран комплект *Android Studio*, поскольку он обладает всеми перечисленными параметрами, а так же легок и удобен в настройке.

### Библиотеки парсинга.

К сожалению «личный кабинет» не имеет *API* для доступа к нему. В связи с этим для реализации приложения мне придется использовать метод «парсинга» *HTML* страниц для получения необходимой информации.

Вот небольшой перечень библиотек для парсинга *HTML:*

1. *HTML Parser*
2. *Java HTML Parser*
3. *HotSax*
4. *HtmlCleaner*
5. *Jsoup*

Я остановил свой выбор на библиотеке *HtmlCleaner* так как у неё присутствует довольно простое и понятное описание и она включает все необходимые для меня функции. Так же она занимает меньший объем памяти по сравнению с *HTML Parser* и *Java HTML Parser.*

### Программа для записи запросов на сервер.

Для отслеживания отправляемых запросов на сервер пришлось прибегнуть к программе для нагрузочного тестирования *Apach JMeter*, так как отследить запросы стандартными средствами браузера не оказалось возможным. Данная программа позволяет перехватывать отправляемые запросы на серверную сторону. Так же она позволяет просмотреть заголовки и тела запросов.

## Требования к программному продукту

Исходя из поставленной задачи, можно вывести некоторые требования к программному продукту, а именно:

* Корректность извлекаемой информации из личного кабинета.
* Сохранность личных данных пользователя
* Сокращение расхода мобильного трафика
* Упрощение работы с личным кабинетом

# Структурная разработка

## Алгоритм получения доступа в личный кабинет.

Для доступа к информации, хранящейся в личном кабинете, необходимо получить токен выдаваемый сервером при авторизации и хранящийся в заголовке запроса в графе *Cookie.* Для получения данного токена можно воспользоваться следующими алгоритмами:

* Авторизация при помощи *Android WebView*
* Авторизации при помощи запроса на сервер.

### Авторизации при помощи *Android WebView*

Для реализации данного алгоритма пришлось прибегнуть к сторонней библиотеке *Crosswalk Project.* Это свободно распространяемая библиотека, представляющая из себя улучшенный вариант *Android WebView*. Вот перечень предоставляемых улучшений:

* Использование всех возможностей доступных в современных *web* браузерах: HTML5, CSS3, JavaScript.
* Использование экспериментальных API, недоступных в большинстве основных web браузеров.
* Контролирование цикла обновления приложения путем распространения с собственной средой выполнения.

Для открытия *web* страницы на форму устанавливается элемент *XWalkView.* Пример *loginactivity.xml:*

<?xml version="1.0" encoding="utf-8"?>

<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"

android:orientation="vertical" android:layout\_width="match\_parent"

android:layout\_height="match\_parent">

<org.xwalk.core.XWalkView

android:layout\_width="match\_parent"

android:layout\_height="match\_parent"

android:id="@+id/web"/>

</LinearLayout>

Так же создается класс *Login\_By\_WebView* наследующий интерфейс *Activity*. Переопределяется метод onCreate() в котором данному классу в качестве запускаемой активности дается *loginactivity.xml.* Так же в данном методе создаются и определяются объекты для работы с *Cookie* и *XWalkView,* а именно xWalkCookieManager и xWalkView. Загрузка страницы выполняется вызовом у объекта xWalkView метода load с параметрами.

@Override

protected void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

setContentView(R.layout.loginactivity);

XWalkView xWalkView= (XWalkView) findViewById(R.id.web);

XWalkCookieManager xWalkCookieManager=new XWalkCookieManager();

xWalkView.load(url, null);

}

Так же в данном классе создается объект intent для передачи параметров и запуска следующей активности.

Intent intent=new Intent(this,Programm.class);

Так же в методе onCreate() объекту xWalkView устанавливается ресурсный клиент для перехвата *Cookie* и запуска новой активности.

xWalkView.setResourceClient(new MyResourceClient(xWalkView));

В итоге при открытии данной активности открывается страница авторизации на сайте НГТУ. Пройдя авторизацию на ней можно извлечь необходимый токен и передать его далее в программу. Данный метод не является удобным, так как необходимо ожидать загрузки всех компонент находящихся на странице авторизации, что ведёт к расходу мобильного трафика. Так же данная страница не является адаптивной для мобильных устройств, что усложняет её использование. Данный метод был использован для проверки гипотезы о возможности извлечения и передачи токена.

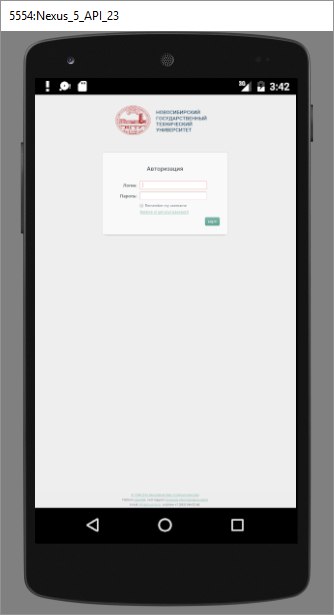


Рисунок 2.1.1.1 Открытие страницы авторизации через *WebView.*

### Авторизация при помощи запроса на сервер

При использовании программы Apach JMeter был обнаружен *POST* запрос отправляемый для авторизации. Самым важным в данном запросе является его тело. Это *JSON* объект, хранящий в себе логин и пароль пользователя, а так же хранящий авторизационный идентификатор необходимый для выдачи авторизационного токена.

Для отправки запроса необходимо запросить у пользователя логин и пароль, а у сервера авторизационный идентификатор, так как он имеет ограниченный срок жизни. Для реализации данного алгоритма был создан класс Login наследующий интерфейс *Activity*. Переопределяется метод onCreate() в котором данному классу в качестве запускаемой активности дается *login\_activity.xml.* В зависимости от выбранной роли на предыдущей активности устанавливается адрес отправки запроса. Так же в данном классе создается объект intent для передачи параметров и запуска следующей активности.

Intent intent=new Intent(this,Programm.class);

В версии Android 4.0 соединение с веб-серверами в основном потоке было запрещено. По этому после ввода логина и пароля на форме и нажатии кнопки войти запускается асинхронный поток, в который передаются адрес отправки запросов, логин и пароль пользователя, для соединения с веб-сервером.

new MyAsyncTask(context)

.execute(url,login.getText().toString(),pass.getText().toString());

В данном потоке выполняется переопределенный метод doInBackground().