МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ

ВЫСШЕГО ОБРАЗОВАНИЯ

«НОВОСИБИРСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»

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

ПОЯСНИТЕЛЬНАЯ ЗАПИСКА  
К ДИПЛОМНОМУ ПРОЕКТУ (РАБОТЕ) НА ТЕМУ

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

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

Автор дипломного проекта (работы) \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(подпись студента, выполнившего дипломный проект или работу)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_Торопов Н.А.\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Группа \_\_АВТ-215\_\_\_\_\_\_\_\_\_\_\_\_\_

(фамилия, инициалы студента) (в которой обучался студент)

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

(полное название факультета)

Специальность \_\_\_\_09.03.04 – «Программная инженерия»\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(код и наименование специальности)

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

Руководитель проекта (работы) \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_Романов Е.Л.\_\_\_\_\_\_

(подпись) (инициалы, фамилия)

Новосибирск 2016

Реферат

Оглавление

[Введение. 4](#_Toc452048073)

[Список терминов и сокращений. 6](#_Toc452048074)

[1. Постановка задачи и анализ предметной области. 8](#_Toc452048075)

[1.1 Постановка задачи. 8](#_Toc452048076)

[1.2 Обзор аналогов. 8](#_Toc452048077)

[1.2.1 Расписашка 8](#_Toc452048078)

[1.3 Архитектура Android. 9](#_Toc452048079)

[1.4 Методы авторизации в личном кабинете. 12](#_Toc452048080)

[1.5 Выбор средств разработки. 13](#_Toc452048081)

# 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:

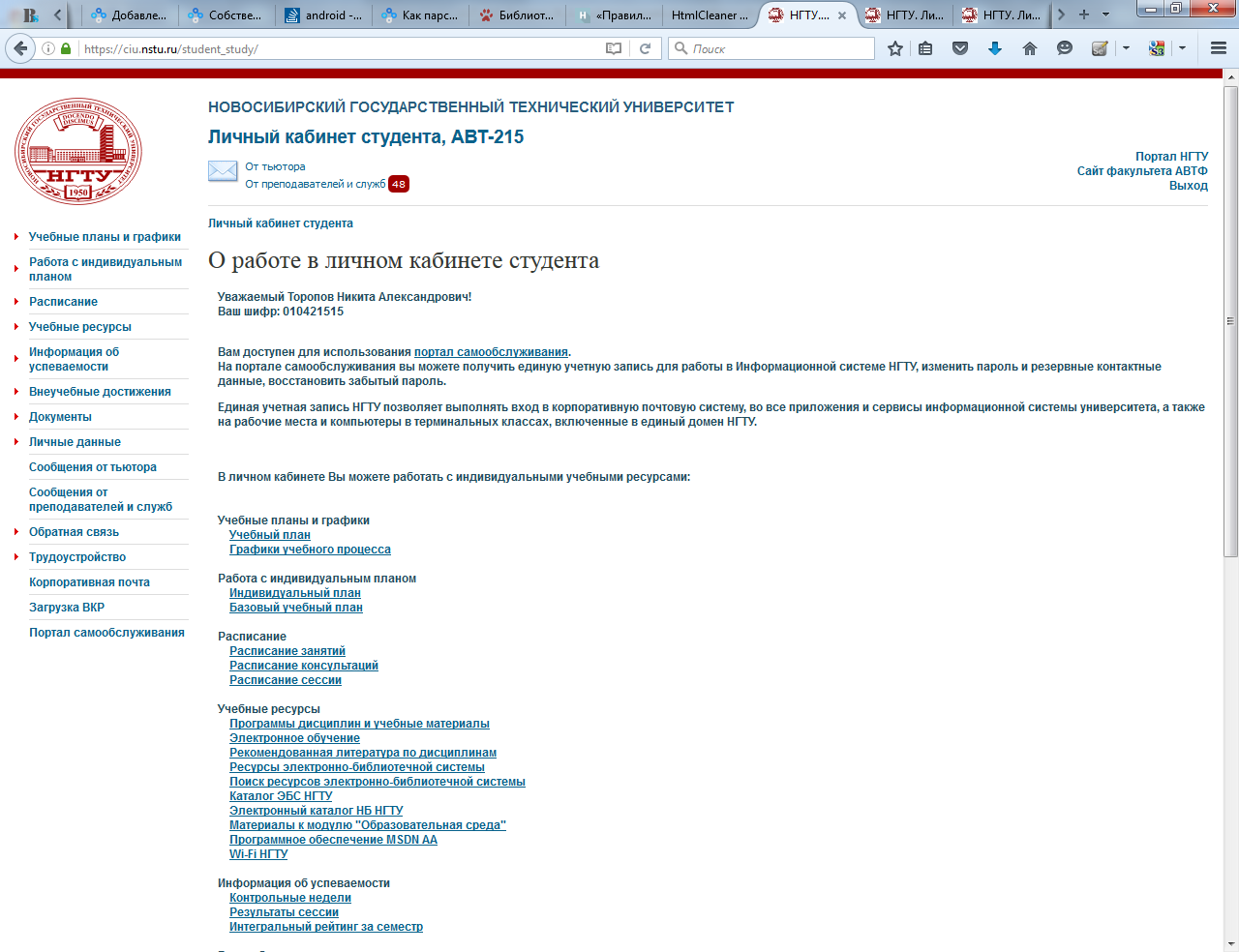


Рисунок 1. Личный кабинет студента.

**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.

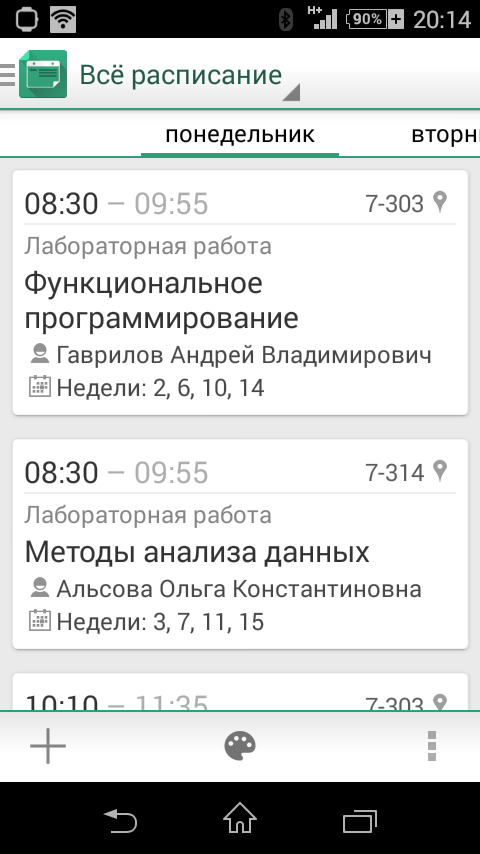


Рисунок 1.1 Внешний вид приложения «Расписашка».

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.*