**МИНСТЕРСТВО ПО РАЗВИТИЮ ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ И КОММУНИКАЦИЙ РЕСПУБЛИКИ УЗБЕКИСТАН**

**ФЕРГАНСКИЙ ФИЛИАЛ**

**ТАШКЕНТСКОГО УНИВЕРСИТЕТА ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ им.МУХАММАД АЛ-ХОРАЗМИЙ**

**ФАКУЛЬТЕТ «ПРОГРАММНЫЙ ИНЖИНИРИНГ»**

**ПРЕДМЕТ:**

**«Разработка мобильных приложений»**

**Самостоятельная работа**

**ТЕМА:**

**«Разработка мобильного приложения “расписание”»**

Выполнил студент 654-20 группы: *Мустафин Фаррух*

Приняла преподаватель*: Хаджаев С.И.*

**СОДЕРЖАНИЕ**

ВВЕДЕНИЕ

ГЛАВА I. ПРИНЦИП РАБОТЫ ANDROID OS И ANDROID ПРИЛОЖЕНИЙ

1.1 КОМПОНЕНТЫ ПРИЛОЖЕНИЙ В ANDROIDГРАММА ПРИЛОЖЕНИЕ

ГЛАВА II

2.1 РЕСУРСЫ. РЕАЛИЗАЦИЯ ФАЙЛОВ. XML В ANDROID

2.2 ВЫБОР ИНСТРУМЕНТА ДЛЯ ПОЛУЧЕНИЯ ДАННЫХ

2.3 ЧТО ТАКОЕ JSOUP

2.4 ПОЛУЧЕНИЕ ДАННЫХ

# Введение

Главное направление совершенствования мобильных телефонов можно определить одним понятием: конвергенция технологий. Мобильные телефоны объединили в себе едва ли не всё, что можно и разумно объединять под корпусом одного устройства.

Основные задачи телефона прошлого - совершать и принимать звонки, писать SMS. Сегодня эти задачи дополнилась работой с интернет - ресурсами, прослушиванием музыки, фотосъемкой, использованием игр и приложений. Мобильный будущего, скорее всего, добавит к списку основных задач просмотр телепрограмм, управление различной техникой, функции контроля и возможно многое другое.

Эксперты обращают внимание потребителей на тот факт, что сейчас на рынке доступно несколько мобильных платформ, стремительную популярность среди которых в последнее время приобретает iOS и Android. Программная платформа Android на мировом рынке смартфонов достигла рекордных 87,5 %. Иными словами, примерно девять из десяти аппаратов используют именно эту ОС.

Что бы ни говорили, а мобильность сегодня - явление глобальное, проникающее во многие сферы нашей жизни. Ныне потребитель ищет не просто мобильный телефон, он внимательно изучает условия, соотнося их с требованиями современного рынка. Именно поэтому можно утверждать, что развитие мобильных технологий будет только ускоряться.

Целью данной работы является разработка приложение для ОС Android, назначением которого будет доступ к расписанию Бурятского государственного университета Пользователями данного приложения будут студенты университета.

Для достижения поставленной цели требуется решить следующие задачи:

1) Изучить архитектуру, инструменты и особенности разработки приложений для ОС Android;

2) Создать макет - прототип будущей программы;

3) Разработать приложение согласно макету;

4) Разработать интерфейс пользователя;

5) Протестировать разработанное приложение.

Практическая значимость данной работы состоит в том, что результат достижения поставленной цели имеет возможность применения в деятельности конкретного учебного заведения.

Android Studio - это [интегрированная среда разработки](https://ru.wikipedia.org/wiki/%D0%98%D0%BD%D1%82%D0%B5%D0%B3%D1%80%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%BD%D0%B0%D1%8F_%D1%81%D1%80%D0%B5%D0%B4%D0%B0_%D1%80%D0%B0%D0%B7%D1%80%D0%B0%D0%B1%D0%BE%D1%82%D0%BA%D0%B8" \o "Интегрированная среда разработки) (IDE) для работы с платформой [Android](https://ru.wikipedia.org/wiki/Android" \o "Android).

Android Studio, основанная на программном обеспечении [IntelliJ IDEA](https://ru.wikipedia.org/wiki/IntelliJ_IDEA" \o "IntelliJ IDEA) от компании [JetBrains](https://ru.wikipedia.org/wiki/JetBrains" \o "JetBrains), официальное средство разработки Android приложений. Данная среда разработки доступна для [Windows](https://ru.wikipedia.org/wiki/Windows" \o "Windows), [OS X](https://ru.wikipedia.org/wiki/OS_X" \o "OS X) и [Linux](https://ru.wikipedia.org/wiki/Linux" \o "Linux).

# Глава I. Принцип работы Android OS и Android приложений

Приложения под операционную систему Android разрабатываются в основном с использованием Java. Скомпилированный программный код (вместе со всеми файлами ресурсов и прочей необходимой информацией) упаковывается в специальный файл-архив, Android Package. Этот файл имеет расширение \*. apk и упаковывается специальной утилитой aapt tool. Именно он в дальнейшем распространяется как программа и инсталлируется на мобильные устройства. Один такой файл связан с кодом одного приложения. И каждое приложение в Android живет в своем собственном мире - в такой машине. По умолчанию, каждая программа выполняется в своем собственном процессе, управлением которого занимается ядро Linux, которое также осуществляет менеджмент памяти. Таким образом, чаще всего код приложения выполняется в изоляции от всех других приложений. Android стартует процесс, когда возникает необходимость выполнить какой-нибудь программный код и завершает его, когда в нем больше нет необходимости и системные ресурсы требуются другим приложениям. По умолчанию, каждому приложению присваивается свой уникальный ID Linux-пользователя. Права доступа устанавливаются таким образом, чтобы файлы приложения были видны только этому пользователю и данному приложению. Хотя, есть способы, позволяющие экспортировать их в другие приложения. Например, существует возможность "разделения" одного и того же пользовательского ID между двумя приложениями. В таком случае, они смогут видеть файлы друг друга. Для того, чтобы экономить системные ресурсы, приложения с одинаковым ID можно также договориться запускать в одном и том же Linux-процессе, разделяя одну и ту же виртуальную машину.

android приложение операционная система

# 1.1 Компоненты приложений в Androidграмма приложение

Интересной чертой операционной системы Android является тот факт, что приложение в ней может использовать для своего функционала элементы других приложений, если, конечно, они предоставляют такой доступ. Например, если нашему приложению требуется отобразить прокручиваемый список изображений, а другое приложение уже имеет реализованный подходящий скроллер, открытый для других приложений, то можно просто вызвать его для осуществления этого действия и не разрабатывать свой собственный. При этом код нашего приложения не смешивается с кодом другого и не компонуется с ним. Скорее, он просто запускает некоторый кусок другого файла, когда возникает такая необходимость. Для обеспечения такого принципа работы, система должна быть способна запускать процесс приложения тогда, когда потребуется любая его часть и создавать экземпляры java-объектов именно для этой части. Поэтому приложения Android не имеют единой точки входа, как это принято в большинстве систем. Вместо этого их код представляет собой набор некоторых отдельных целостных сущностей, компонент, из которых система по мере необходимости может создавать экземпляры и использовать их.

Можно выделить четыре типа таких сущностей-компонент. Поговорим о каждой в отдельности.

Activities

Activity представляет собой внешний пользовательский интерфейс для одной операции, которую может совершить пользователь. Если упростить, то это просто один текущий экран как некоторая единица активности, своего рода кадр с одним пользовательским действием. Здесь и далее будет использоваться термин activity без перевода, как некоторое имя собственное. Хотя дословный перевод, "активность" или "действие", хорошо передает общий смысл компонента.

Например, activity может предоставить список пунктов меню, которые может выбрать пользователь или отобразить фотографии с их подписями. Или другой пример - приложение для мгновенного обмена сообщениями может использовать одно activity для того чтобы отобразить лист контактов, другое - чтобы создать сообщение для выбранного контакта, третье - чтобы посмотреть историю сообщений или выполнить настройки и так далее.

Приложение может состоять из всего одного activity или сразу из нескольких, как упомянутый ранее в качестве примера мессенджер. Какими именно будут activity и сколько их будет, зависит от конкретного приложения и его дизайна. Как правило, одно из activity помечается как первое, это означает, что оно будет предоставлено пользователю при запуске приложения. Одно activity может запускать другое. Таким образом, переход от одного activity к другому осуществляется тогда, когда текущее activity вызывает следующее.

Каждое activity предоставляет окно по умолчанию. Обычно окно создается в полноэкранном виде, но оно также может и не занимать весь экран и находиться поверх других окон. Activity также может задействовать дополнительные окна - например, всплывающее диалоговое для взаимодействия с пользователем в процессе работы activity, или окно для предоставления текущей информации при выборе какой-нибудь важной опции.

Визуальное содержание окна строится с помощью иерархии визуальных компонентов (или представлений) - объектов, производных от базового класса View. Каждый компонент представляет собой просто прямоугольное пространство внутри окна. Родительские компоненты содержат дочерние и организуют их расположение. Иерархию компонент можно представить в виде дерева, а те элементы, которые находятся в самой нижней его части (“листья”) и не имеют дочерних компонент, отрисовывают прямоугольные области и ожидают действий пользователя на этом участке. Для того чтобы поместить в окно такую иерархию, нужно вызвать метод Activity. setContentView (). Параметром метода является экземпляр класса View, лежащий в корне иерархии.

Services (сервисы) представляют из себя компоненты, которые работают в фоновом режиме. Он, как правило, требуется для длительных операций или для обеспечения работы удаленных процессов, но в общем случае это просто режим, который функционирует, когда приложение не в фокусе. Примером такого процесса может стать прослушивание музыки в то время, когда пользователь делает что-то другое или получение данных по сети без блокирования текущей активности. Сервис сам по себе не предоставляет пользовательского интерфейса, то есть с пользователем не взаимодействует, а запускается, управляется и связан с другими компонентами, например, activity. Также может запускаться вместе с системой.

Content providers Данный компонент управляет наборами данных, которые приложения предоставляют другим. Эти данные могут храниться в файловой системе, базах данных SQLite, в сети, или в любом другом постоянном месте, к которому приложение может иметь доступ. Посредством content provider другое приложение может запрашивать данные и, если выставлены соответствующие разрешения, изменять их. Например, система Android содержит content provider, который управляет пользовательской информацией о контактах. Он позволяет любому приложению, обладающему соответствующими правами вызывать составляющие этого компонента для того, чтобы считывать, записывать или изменять информацию о конкретном человеке.

Broadcast receivers. Этот компонент отвечает за распространение общесистемных сообщений, отслеживание и реагирование на действия. Многие оповещения идут от системы, например, сообщения о том, что заряд батареи мал или экран выключен. Приложения также могут инициировать такие оповещения, например, сигнализировать о том, что информация загружена на устройство и доступна к использованию. Как и сервисы, broadcast receiver не предоставляет пользовательского интерфейса, однако, он способен создавать уведомления в строке состояния, чтобы предупреждать пользователя о том, что произошло какое-то событие.

Внешний вид приложения.

"По сравнению с другими ОС, Android - одна из наиболее простых и одновременно комплексных платформ, вся система скачивается за один раз".

Большинство студентов используют телефоны с операционной системой Android, что позволяет сделать разрабатываемое приложение более доступным и тем самым обеспечивает студентам доступ к расписанию вне института.

Задачи, подлежащие автоматизации:

1. Информирование об изменениях в расписании;

2. Информирование о текущих предметах.

Данное приложение может работать как в режиме online, так и в offline. В режиме offline приложение выполняет функцию предоставления материалов с возможностью просматривать ранее сохраненные данные (предмет, время, аудитория и т.п.). В режиме online появляется возможность полного информирования.

В его основе лежит система баз данных SQLite. "SQLite" - мощная и легковесная, отличается от многих обычных движков баз данных отсутствием типизации каждого столбца.

Когда приложение использует SQLite, их связь производится с помощью функциональных и прямых вызовов файлов, содержащих данные, а не кого-то интерфейса, что повышает скорость и производительность.

# Глава II

Приложение с интуитивно понятным интерфейсом быстро и без особого труда позволит студентам учебного заведения получать:

1. актуальное расписание занятий учебной группы на неделю либо на отдельно взятый день;
2. расписание экзаменов, зачетов;
3. расписание лабораторных работ и практических занятий;
4. получить быстрый, простой и оперативный доступ к информации, касающейся учебного процесса - расписание занятий, экзаменов, преподавателя, аудитория, время.

В результате данное приложение упрощает повседневную жизнь студентов. Простой интерфейс делает процесс обучения более комфортным, информативным и более доступным.

При входе в приложение пользователю показывается окно (рис. 1):

Рис. 1. Главное окно приложения

Главное окно приложения включает в себя элементы: "Расписание", "Дни недели", "Группа", кнопка "Подтвердить". При нажатии на каждую из них происходит переход на соответствующую вкладку программы. При каждом выборе определенной группы происходит обновления расписания путем http-запроса на сервер и получение в ответ http-страницы.

Вкладка "Расписание" содержит элемент "List View", с элементами с каждой из которых представляет собой вкладку с расписанием на конкретный день недели. При нажатии на вкладку происходит показ ее содержимого, которое представляет собой набор вкладок, разбитых по учебным планам (рис.2):

Виджет ListView представляет собой прокручиваемый список элементов. Очень популярен на мобильных устройствах из-за своего удобства.

При большом количестве информации, которую надо поместить на экране необходимо использовать полосы прокрутки. В Android существуют специальные компоненты ScrolView и HorisontalView, которые считаются контейнерными элементами и наследуются от ViewGroup.

При большом количестве информации, которую нужно поместить на экране приходится использовать полосы прокрутки. В Android существуют специальные компоненты ScrollView и HorizontalScrollView, которые являются контейнерными элементами и наследуются от ViewGroup. Их использование с TextView способен улучшить вид приложения и повышает удобство работы для пользования

На панели инструментов компоненты можно найти в разделе Containers.

Виджет ScrollView, поддерживает только вертикальную прокрутку, поэтому для создания вертикальной и горизонтальной прокрутки необходимо использовать ScrollView в сочетании с HorizontalScrollView.

Рис. 2. Кнопка Подтверждения и вкладка с расписанием.

Структура навигации в приложении выглядит следующим образом (рис. 3):

Рис. 3. Схема навигации мобильного приложения

**2.1 РЕСУРСЫ. РЕАЛИЗАЦИЯ ФАЙЛОВ. XML В ANDROID**

Android представляет собой платформу с открытым кодом для разработки приложений для мобильных устройств. С ее помощью можно получить доступ ко всем компонентам устройства, на котором выполняется эта ОС, начиная от низкоуровневого программирования графики и заканчивая использования встроенной камеры.

Ресурсы - один из основных компонентов, с которыми придется работать. В Android принято держать объекты, как изображения, строковые константы, цвета, стили и т.п. за пределами исходного кода. В основном они хранятся в виде xml-файлов в каталоге res с подкаталогами values, drawable-ldpi, drawable-mdpi, drawable-hdpi, layout, но также бывают другие типы ресурсов.

Самыми распространенными ресурсами являются строки (string), цвета (color), и графические рисунки (bitmap).

Цвета, как и все остальные ресурсы, следует определять в разметке xml. Обычно цвета определяются в файле с названием colors. xml, который создается многими шаблонами приложений Android Studio (рис. 4).

Рис.4. Res / values / colors. xml.

Android позволяет хранить в ресурсах массивы строк. Для этого используется тег string-array, который содержит внутри элементы item с конкретными строками.

Элементы массива инициализируются, как обычные строки, а элементы item содержат только ссылку на них такая инициализация удобна. Само описание массива при этом удобно вынести в отдельный ресурсный файл (рис.5).

Рис.5. Res / values / string. xml.

Ресурсы размеров

В Android используются следующие единицы измерения: пикселы, дюймы, точки. Все они могут входит в состав ХМL-шаблонов и кода Jаvа. Данные единицы измерения также можно использовать в качестве ресурсов при помощи тега <dimen> (обычно используют файл dimens. xml):

Рис.6. Res / values / attrs. xml.

Часто используемый список format (тип свойства):

• dimension - может быть значение типа 10px, 10dip;

• integer - может быть значение типа 10, 5, 2;

• string - просто текстовое значение типа "Hello World".

Как и другие ресурсы, ресурс dimension определяется в корневом элементе <resources>. Тег <dimen> обозначает ресурс и в качестве значния принимает некоторое значение размера в одной из принятых единиц измерения (dp, sp, pt, px, mm, in).

Так, добавим в Android Studio в папку res/values новый файл, который назовем dimens. xml. Определим в нем следующее содержимое:

Рис.7. Res / values / dimens. xml.

Здесь определены два ресурса для отступов activity\_horizontal\_margin и activity\_vertical\_margin, которые хранят значение 1 dp и 8 dp, и атрибут divider, который отвечает за внешний вид разделителя. Названия ресурсов могут быть произвольными.

Экран имеет такие физические значения как диагональ и разрешение. Диагональ - это расстояние между противоположными углами экрана, обычно измеряется в дюймах. Разрешение - количество точек по горизонтали и вертикали, которое экран способен отобразить, измеряется в пикселях.

Layout windth и Layout heigth.

На них можно указывать абсолютные значения, а можем использовать константы.

Абсолютные значения:

dp и dip - Dencity-indenperendent Pixels. Абстракная ЕИ, позволяющая приложениям выглядеть одинаково на разных экранах и разрешениях.

px - пиксель, не рекомендуется использовать так как на разных экранах приложение будет выглядеть по-разному.

mm - миллиметр, определяется по физическому размеру экрана.

in - дюйм, определяется по физическому размеру экрана.

Константы:

math\_parent (fill\_parent) - означает, что элемент займет всю доступную ему в родительском элементе ширину и высоту.

wrap\_content - ширина и высота элемента будет определятся его содержимым.

Расположение View-элементов на экране зависит от ViewGroup (Layout), в которой они находятся. LinerLayout - отображает View - элементы в виде одной строки (Horizontal) или одного столбца (Vertical). Этот вид ViewGroup по умолчанию предлагается при создании новых layout-файлов. Он достаточно удобен и гибок, чтобы создавать экраны различной сложности.

<LinearLayout xmlns: android="http://schemas. android.com/apk/res/android" xmlns: tools="http://schemas. android.com/tools" android: id="@+id/activitty\_main" android: layout\_width="match\_parent" android: layout\_height="match\_parent" android: orientation="vertical" android: paddingBottom="@dimen/activity\_vertical\_margin" android: paddingLeft="@dimen/acivity\_horizontal\_margin" android: paddingRight="@dimen/activity\_horizontal\_margin" android: paddingTop="@dimen/activity\_vertical\_margin" android: background="@color/background" tools: context="com. a2210. alive. mydailyscheduller. MainActivity"> <ListView android: layout\_width="match\_parent" android: layout\_height="match\_parent" android: background="@color/background" android: dividerHeight="@dimen/divider" android: divider="@color/colorPrimary" android: id="@+id/li"/> </LinearLayout>

Res / layout / activity\_main. xml.

Суть приложения: считывание информации о занятиях с сайта университета. Приложение должно посылать запрос который содержит сведения о группе на сайт БГУ и получать ответ в виде файла с расписанием.

**2.2 ВЫБОР ИНСТРУМЕНТА ДЛЯ ПОЛУЧЕНИЯ ДАННЫХ**

Для получения данных парсеры взаимодействуют с разметкой документов, написанной на языке разметки HTML (либо XML). Существует 2 вида парсеров:

1) SAX (англ. Simple API for XML) - способ последовательного чтения и/или записи HTML/XML-файлов. В ходе последовательного чтения SAX-парсер сообщает вызвавшему приложению о встреченных элементах разметки, а программист в своюочередь описывает нужную обработку данных событий.

2) DOM (англ. Document Object Model) позволяет представить любой документ известной структуры в виде дерева узлов, каждый узел которого представляет собой элемент, атрибут или иной объект. Узлы связаны между собой отношениями "родитель - потомок". DOM - парсеры сначала загружают весь документ разметки, а на выходе представляют дерево, из которого можно получить доступ к содержимому документа.

**2.3 ЧТО ТАКОЕ JSOUP**

Jsoup это open-source Java библиотека для работы с реальным HTML. Она обеспечивает очень удобный API для извлечения и манипулирования данными, используя лучшие DOM, CSS, и JQuery-подобные методы. Jsoup реализует WHATWG HTML5 спецификацию, и разбирает HTML в ту же модель DOM, как это делают современные браузер вроде Chrome и Firefox. Вот некоторые из полезных функций Jsoup библиотеки:

Jsoup может очистить и разобрать HTML из URL, файла или строки.

Jsoup может найти и извлечь данные используя обход DOM или CSS селекторы.

Jsoup позволяет манипулировать HTML элементами, атрибутами и текстом.

"Аккуратный" HTML на выходе: при разборе HTML-текста библиотека исправляет ошибки наподобие незакрытых тегов.

Jsoup предназначен для работы с различными видами HTML существующими в реальном мире, включая должным образом подтвержденный HTML c неполным неподтвержденным набором тэгов. Одно из основных преимуществ Jsoup это его надежность. С помощью него очень просто разобрать HTML, все что Вам нужно это вызвать статический метод Jsoup. parse () и передать в него Вашу HTML строку. Jsoup предоставляет несколько перегруженных методов parse () для чтения HTML из строки, файла, из базового URI, из URL и из InputStream. Вы также можете указать кодировку, для корректного чтения HTML файла.

**2.4 ПОЛУЧЕНИЕ ДАННЫХ**

Для получения данных используется сторонняя Java-библиотека jsoup. Кроме того, необходимо предоставить приложению доступ в Интернет. Для этого в файл AndroidManifest. xml нужно дописать следующую строку:

<uses-permission android: name="android. permission. INTERNET"/>

В самом классе Parser надо прописать импорт следующих классов:

import java. util. List;

import org. jsoup. nodes. Document;

import org. jsoup. nodes. Element;

import org. jsoup. nodes. Node;

import org. jsoup. parser. CharacterReader;

import org. jsoup. parser.htmlTreeBuilder;

import org. jsoup. parser. ParseError;

import org. jsoup. parser. ParseErrorList;

import org. jsoup. parser. ParseSettings;

import org. jsoup. parser. Tokeniser;

import org. jsoup. parser. TreeBuilder;

import org. jsoup. parser. XmlTreeBuilder;

Информация о расписании университета берется с сайта "bsu.ru", и для начала работы необходимо подключитьсяк нужной странице и получить объект класса Document: получаем данные методом GET

org. jsoup. nodes. Document doc = Jsoup. connect ("http://bsu.ru/rasp/? g="+groupeNum). get ();

После подключения нужно извлечь требуемую информацию:

Elements elements = doc. select ("div. week"); /\*\*разбиваем по неделям\*\*/

Метод select () позволяет получить нужные теги.

Далее в цикле для каждого элемента списка извлекается нужная информация:

for (Element element: elements) {

Elements el = element. select ("table. rasp\_week");

for (Element element1: el) {

Elements elements1 = element1. select ("td");

for (Element element2: elements1) {

Elements day = element2. select ("h3. rasp\_day"); /\*получаем день\*\*/

for (Element element3: day) {d = element3. text (); }

Elements time = element2. select ("td. rasp\_time"); /\*\*время\*\*/

for (Element element3: time) {t = element3. text (); }

Elements subj = element2. select ("td. rasp\_subj"); /\*\*предмет\*\*/

for (Element element3: subj) {s = element3. text (); }

Elements type = element2. select ("td. rasp\_subj\_type"); /\*\*тип\*\*/

for (Element element3: type) {st = element3. text (); }

Elements aud = element2. select ("td. rasp\_aud"); /\*\*номер аудит\*\*/

for (Element element3: aud) {a = element3. text (); }

Elements name = element2. select ("td. rasp\_aud"). next (); /\*\*следующий элемент после номера аудитории - имя преподователя\*\*/

for (Element element3: name) {n = element3. text ();

/\*\*если день недели совподает то вывести то формируем список\*\*/

if (dayOfWeek. equals (d)) {scheduler = w + "\r\n" +

"Время: " + t + "\r\n" + "Название предмета: " + s + "\r\n" + "Тип пары: " + st +

" Аудитория: " + a + "\r\n" + "Имя преподавателя: " + n + "\r\n\r";

my. add (scheduler); /\*\*добавляем строку в список строк \*\*/