Оглавление

[. ОС Android для мобильных устройств 1](#_Toc165675724)

[Операционные системы для мобильных устройств 1](#_Toc165675725)

[**ВВЕДЕНИЕ** 1](#_Toc165675726)

[Рынок ОС для мобильных устройств 3](#_Toc165675727)

[**Уровень ядра** 6](#_Toc165675728)

[**Уровень библиотек** 8](#_Toc165675729)

[**Среда выполнения** 11](#_Toc165675730)

[**Уровень каркаса приложений.** 11](#_Toc165675731)

[**Уровень приложений** 13](#_Toc165675732)

[II. Разработка приложения для Google Android 13](#_Toc165675733)

[Вывод 13](#_Toc165675734)

[Создание приложения для Android 15](#_Toc165675735)

[Системные функции Android: 17](#_Toc165675736)

# . ОС Android для мобильных устройств

***https://studopedia.su/13\_44493\_osobennosti-os-dlya-mobilnih-ustroystv.html***

## Операционные системы для мобильных устройств

### **ВВЕДЕНИЕ**

В настоящее время мобильные устройства занимают огромное место в нашей жизни. Люди уже не могут представить себя без мобильника в кармане, без планшета или какой-нибудь портативной приставки.

Целью данной работы является исследование рынка операционных систем для мобильных устройств, их сравнение и выбор наиболее распространенной и удобной для конечного пользователя.

Задачи, которые необходимо решить в ходе работы:

1.Рассмотреть основные операционные системы

2.Дать краткую характеристику каждой из них

3. Рассмотреть устройство наиболее распространенной

4.Реализовать приложение под эту операционную систему

Актуальность данной темы определяется огромным выбором мобильной техники на рынке мобильных устройств и большим количеством компаний.

Мобильные устройства стали неотъемлемой частью повседневной жизни и деятельности большинства людей во всем мире. Поэтому операционные системы для мобильных устройств в настоящее время бурно развиваются. Данная лекция является кратким обзором ОС для мобильных устройств.

К мобильным устройствам принято относить **мобильные телефоны, смартфоны**и **коммуникаторы**. Разработчики ОС для мобильных устройств работают над тем, чтобы приблизить возможности этих ОС к возможностям ОС для настольных и портативных компьютеров. Однако в ОС для мобильных устройств есть своя специфика. Их основные особенности следующие:

**Учет более жестких ограничений по памяти мобильных устройств**. Хотя мобильные устройства активно развиваются, по своим параметрам (объему памяти, быстродействию процессора) они все же пока уступают настольным компьютерам. Поэтому приложения для мобильных устройств, требующие большого объема памяти, воспроизводятся на мобильных устройствах с неполными возможностями. Ряд инструментов, например, Java, также доступны для мобильных устройств в специальных версиях, разработанных с целью экономии памяти, с ограничениями, не свойственными классическим версиям: на мобильных устройствах работает **Java Micro Edition (JME)**,а не полная версия – Java Standard Edition (JSE) для настольных компьютеров. В ней, например, отсутствует вещественная арифметика в Java и ряд других важных возможностей.

**Учет более низкой скорости процессора**. По сравнению с настольными компьютерами, аналоги настольных приложений на мобильных устройствах работают заметно медленнее: например, медленно открывается изображение на экране мобильного телефона, текстовый файл для просмотра и т.д. Это неудобно для пользователей и, по-видимому, будет преодолено в последующих версиях самих мобильных устройств и их ОС.

**Учет особенностей экранов и экранных навигаторов конкретных моделей мобильных устройств**. Многие типы мобильных устройств различных фирм имеют существенно разные экраны и различные виды экранных навигаторов (трэкбол и др.). При разработке ОС и сервисных программ для мобильных устройств эти различия приходится учитывать, что осложняет разработку программного обеспечения.

**Совместимость с основными форматами файлов**: .doc/docx, .ppt/.pptx, .pdf, .jpg и др. При работе на мобильном устройстве необходимо иметь возможность визуализировать, редактировать и создавать файлы тех же привычных форматов, что и на настольных компьютерах. Обеспечение такой совместимости – одна из важных задач ОС и сервисных программ для мобильных устройств.

**Мультимедийные возможности: рисунки, видео, аудио, обмен мультимедийными сообщениями**. Обработка мультимедийной информации для мобильных устройств особенно важна: пользователи должны иметь возможность сделать фотографии, снять видео, просмотреть их на мобильном устройстве, опубликовать в Интернете, послать на настольный компьютер или в виде сообщения своим корреспондентам.

**Поддержка коммуникационных и сетевых технологий**: Wi-Fi / WiMAX, Bluetooth, GPRS, EVDO, GSM, CDMA. Мобильное устройство и его ОС должены обеспечивать для пользователей возможность современных видов коммуникации в беспроводных сетях.

## Рынок ОС для мобильных устройств

В настоящее время на рынке мобильных устройств используется несколько десятков наиболее распространенных ОС. Некоторые из них разработаны на основе свободно распространяяемого ядра Linux. Ведущие фирмы-производители мобильных устройств поддерживают собственные ОС либо ОС, приобретенные вместе с их фирмами-разработчиками. Наиболее распространенные ОС для мобильных устройств следующие:

* Nokia Symbian OS
* Google Android
* Windows Mobile
* Blackberry OS.
* Apple iPhone OS
* Samsung Bada
* PalmOS.

Рассмотрим четыре из них, как самые популярные.

**Windows Mobile**– семейство ОС для мобильных устройств фирмы Microsoft. Оно относится к семейству **Windows CE (Consumer and Embedded)**– Windows для встроенных систем. Ядро ОС Windows Mobile основано на ОС Windows CE.

Текущая версия Windows Mobile (2012) – Windows Mobile 7. В США Windows Mobile - четвертая по популярности ОС для мобильных устройств (после Android OS, iPhone OS и Blackberry OS). Windows Mobile поддерживает следующие виды мобильных устройств: PocketPC, смартфоны, коммуникаторы (например, Qtek). Первая версия Windows Mobile была выпущена в 1996 г.

**Windows Mobile: возможности и ПО**. ОС Windows Mobile предоставляет разнообразный набор возможностей и программного обеспечения:

* **Office Mobile**– аналог Microsoft Office для мобильных устройств; полная совместимость по форматам;
* **Windows Media Player**– мультимедийный проигрыватель, аналог проигрывателя для настольной версии Windows;
* **Internet Explorer Mobile**– Web-браузер, аналог Internet Explorer для настольной версии Windows;
* Программное обеспечение для поддержки Bluetooth и Wi-Fi – современных видов коммуникации;
* Программное обеспечение **Microsoft ActiveSync**для синхронизации данных с настольными компьютерами.
* Windows Mobile поддерживает пользовательский интерфейс с мобильным устройством с помощью касания экрана стайлусом и пальцами, в том числе (в современных версиях) – multi-touch.

Новые версии Windows Mobile поддерживают также .NET Compact Framework, что дает возможность выполнения приложений для платформы .NET на мобильных устройствах.

**Версии Windows Mobile 7**. Начиная с версии 6, Windows Mobile выпускается в трех основных версиях:

* Windows Mobile Classic — для КПК
* Windows Mobile Professional — для коммуникаторов
* Windows Mobile Standard — для смартфонов.

**Перспективы Windows Mobile**. На мой взгляд, ОС Windows Mobile будет вскоре вытеснена с рынка мобильных устройств, все сложнее и сложнее становится конкуренция с другими популярными ОС, прежде всего, с ОС фирм Apple (iPhone OS) и Google (Android). В целях конкуренции фирма Microsoft развивает новый проект – Microsoft KIN, новый смартфон, конкурирующий с Apple iPhone.

**Symbian OS**

**Symbian OS**– в настоящее время четвертая из ОС для мобильных устройств, разработанная консорциумом Symbian (Nokia, Ericsson, Psion, Motorola), основанным в 1998 г. Фирма Nokia финансирует объединение Symbian Foundation, целью которого является разработка и поддержка единой (для мобильных устройств различных компаний) мобильной платформы на основе Symbian OS. Symbian OS разработана на основе ОС Psion EPOC32 (фирмы Psion). Язык реализации системы – C++; имеется также поддержка Java. Последние выпущенные версии – Symbian Anna и Nokia Belle (Symbian Belle).

**Возможности Symbian OS**. Symbian OS поддерживает удобный пользовательский интерфейс и имеет значительное число сервисных программ, в том числе – разработанных фирмой-производителем Nokia. Основные возможности для пользователя следующие:

* Меню с иконками приложений;
* Список контактов;
* Поддержка встроенной фото- и видеокамеры, галереи изображений и видеоклипов;
* Обработка файлов, управление памятью (SmartMedia);
* Web-браузер;
* Электронная почта;
* Обмен сообщениями SMS и MMS;
* Поддержка GPS-навигации;
* Редакторы фото и видео;
* Поддержка Java Micro Edition - загрузки и исполнения мидлетов;
* Библиотека приложений фирмы Nokia.

**Google Android**– стек приложений для мобильных устройств, включающий операционную систему (на базе ядра Linux), промежуточное программное обеспечение (middleware) и сервисные программы. Система Android разработана фирмой Android, Inc., приобретенной компанией Google (2005). В настоящее время (2012 г.) это первая по популярности ОС для смартфонов. Важной особенностью Google Android является то, что сервисные программы и библиотеки этой системы написаны на Java.

**Возможности Google Android**. Прежде всего, Google Android привлекает пользователей своим удобным и эстетичным пользовательским интерфейсом, который разработан с использованием двумерной и трехмерной графики (библиотеки OpenGL). Основные возможности системы следующие:

* СУБД SQLite для хранения данных;
* Поддерживаемые сетевые технологии: GSM/EDGE, IDEN, CDMA, EV-DO, UMTS, Bluetooth, Wi-Fi, WiMAX, Bluetooth 2.0;
* Обмен сообщениями SMS и MMS;
* Web-браузер на базе WebKit Application Framework.

**Поддержка Java**. Фирма Google по принципиальным соображениям использует в системе Android собственную реализацию Java – **Dalvik Virtual Machine**, разработанную специально для мобильных устройств. По мнению специалистов Google, cтандарт Java Micro Edition (JME) устарел, так как рассчитан на устаревшие типы мобильных устройств и их технические возможности. Поэтому в Google Android стандарт JME не поддерживается.

**Поддержка мультимедиа**. В системе Google Android имеются кодеки для всех распространенных мультимедийных стандартов, программное обеспечение для обработки мультимедийных файлов и взаимодействия с видео- и аудиоустройствами.

**Поддержка разработки приложений**. Система Google Android имеет свою собственную интегрированную среду для разработки приложений - **Android SDK**, включающий эмулятор мобильных устройств, средства отладки, профилирования, а также plug-in к популярной среде Eclipse для разработки Java-приложений.

Согласно докладу Comscore, 97.9 млн. человек в США владеют смартфонами по данным на последний квартал 2011 года, что составляет 40% всех абонентов мобильной связи. Google Android также возглавила список рынка смартфонов, захватив 47.3% акций. Компания Apple имеет вторую позицию с 29.6%. RIM занимает третье место с 16%, за которой следует Microsoft с 4.7% и Symbian с 1.4%.

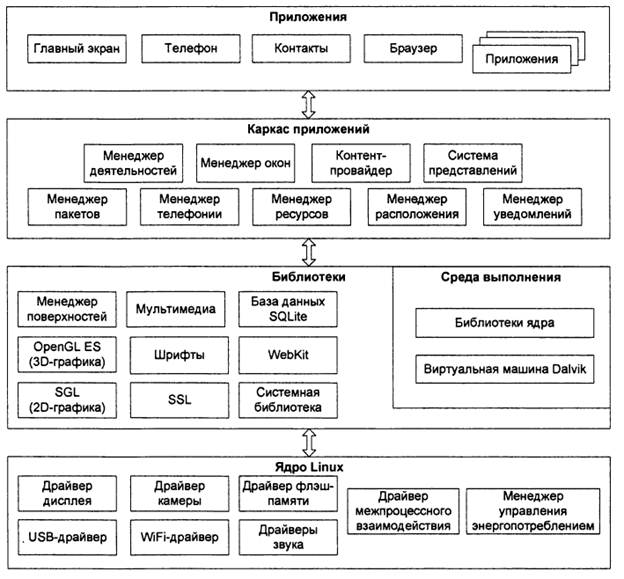
Во время проведения исследований было решено разрабатывать наше приложение для мобильных устройств работающих под управлением платформы android, т.к. данная платформа по мнению аналитиков и экспертов все больше захватывает рынок мобильных устройств, постепенно вытесняя с него общепризнанных лидеров.

Система Android – это программный стек для мобильных устройств, который включает операционную систему, программное обеспечение промежуточного слоя (middleware), а также основные пользовательские приложения (e-mail-клиент, календарь, карты, браузер, контакты и др.).

### **Уровень ядра**

Ядро является слоем абстракции между оборудованием и остальной частью программного стека. На этом уровне располагаются основные службы типа управления процессами, распределения памяти и управления файловой системой.

Ядро android основано на ядре Linux 2.6, но сама система Android не является Linux-системой в чистом виде, имеет некоторые отличия и содержит дополнительные расширения ядра, специфичные для Android, - свои механизмы распределения памяти, взаимодействие между процессами и др.



**Рис. 1** Архитектура системы Android

Основные компоненты уровня ядра:

* Драйвер межпроцессорного взаимодействия (ipc driver);
* Драйвер управления питанием (android power management);
* Набор драйверов для оборудования, входящего в состав мобильного устройства.

Рассмотрим теперь кратко основные компоненты ядра android.

**Драйвер IPC.** Приложения и службы могут работать в защищенных отдельных процессах, которые должны общаться между собой и иметь доступ к общим данным. Платформа android обеспечивает механизм ipc (inter-process communication), который является основным механизмом взаимодействия между процессами.

Драйвер IPC обеспечивает следующую функциональность:

* Взаимодействие процессов;
* Создание и обработку пулов потоков в процессах;
* Подсчет и отображение ссылок на объекты в других процессах;
* Синхронные запросы между процессами.

**Управление энергопотреблением.** Система управления энергопотреблением (android power management) разработана на основе стандартного драйвера управления питанием linux, но оптимизирована для мобильных устройств с учетом их специфических особенностей.

Основная функция системы управления энергопотреблением – экономное использование батареи мобильного устройства. Драйвер переводит системы в «спящий режим» с минимальным потреблением мощности процессором, если приложения и службы не используются.

**Драйверы оборудования.** Программный стек android разработан с учетом необходимой гибкости, включая работу со многими дополнительными компонентами, имеющимися в мобильных устройствах. Эти компоненты в значительной степени полагаются на доступность определенных аппаратных средств на данном устройстве. Они предоставляют дополнительную фунциональность для мобильных устройств (сенсорный экран, камера, gps, акселерометр и т.д.).

### **Уровень библиотек**

Следующий уровень над ядром Linux является набором библиотек c/c++ типа openGL, webkit, freetype, ssl, библиотеки поддержки libc, базы данных SQLite и мультимедиабиблиотек (media framework). Системная библиотека базируется на berkeley software distribution (bsd) и разработана для мобильных устройств на основе Linux.

Следующий уровень над ядром Linux включает набор библиотек c/c++, используемых различными компонентами ос. Для разработчиков доступ к функциям этих библиотек реализован через использование application framework – каркаса приложений. Библиотеки этого уровня по своему функциональному назначению можно разделить на следующие группы:

* Системная библиотека с;
* Менеджер поверхностей;
* Функциональные библиотеки C/C++

**Системная библиотека libc.** Компания Google разработала собственную с-библиотеку (libc) – bionic. Это было необходимо по следующим причинам:

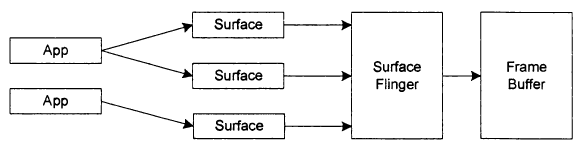
· Библиотека будет загружаться в каждый процесс и, следовательно, должна иметь маленький размер. Библиотека bionic имеет размер около 200 кбайт, что в два раза меньше размера glibc;

· Ограниченная мощность центрального процессора мобильного устройства. Это означает, что библиотека должна быть оптимизирована для максимального быстродействия.

Библиотека bionic имеет встроенную поддержку важных для android системных служб и регистрацию системных событий. Библиотека bionic не поддерживает определенные функциональности, например исключения с++, и несовместима с gnu libc и стандартом posix.

**Менеджер поверхностей.** Система android использует композитный менеджер поверхностей, похожий на compiz (композитный менеджер для x window system, использующий для ускорения 3d-графику opengl). Вместо того чтобы рисовать непосредственно в буфер экрана, команды рисунка входят за кадром в битовые массивы, которые потом объединяются с другими битовыми массивам, чтобы сформировать изображение, которое видит пользователь. Это позволяет системе создавать все виды интересных эффектов, например прозрачные окна и причудливые переходы.

Менеджер поверхностей обрабатывает весь рендеринг поверхностей на фреймовый буфер. Менеджер может объединить 2d- и 3d-поверхности и поверхности от нескольких приложений. Поверхности передаются как буферы компоновкой ipc-запросов. Менеджер поверхностей использует двойную буферизацию, используя транспонирования страницы. Системные интеграторы могут подключать аппаратное 2d-ускорение, используя плагины khronos. Обработка графической информации менеджером поверхностей представлена на следующем рисунке



**Рис. 2.** Обработка графической информации менеджером поверхностей

**Функциональные библиотеки.** Android включает ряд библиотек с/с++, используемых различными компонентами системы. Далее приводятся основные функциональные библиотеки системы.

· Мультимедиа (media framework). Эти библиотеки ответственны за регистрацию и воспроизведение аудио- и видеоформатов. Основаны на packetvideo opencore и предназначены для поддержки популярных аудио- и видеоформатов (mpeg4, h.264, mp3 и др.).

· SQLite – процессор баз данных, доступный всем приложениям. SQLite не является отдельно работающим процессом, с которым взаимодействует программа, а представляет библиотеку, с которой программа компонуется, и движок становится составной частью программы. Таким образом, в качестве протокола обмена используются вызовы функций (api) библиотеки sqlite. Такой подход уменьшает накладные расходы, время отклика и упрощает разработку приложений для работы с данными. Sqlite хранит всю базу данных (включая определения, таблицы, индексы и данные) в единственном стандартном файле на том компьютере, на котором исполняется программа.

· OpenGL es – движок для работы с 3d-графикой, основанный на api-версии openGL es 1.0. OpenGL es – это упрощенная версия спецификации openGL, позволяющая мобильным устройствам работать с тяжеловесными в графическом отношении приложениями. Библиотека использует аппаратный 3d-акселератор (если он доступен на этом устройстве) или встроенное высоко оптимизированное трехмерное программное обеспечение для работы с растровой графикой.

· Freetype – библиотека шрифтов, предназначенная для работы с растровыми и векторными шрифтами.

· Webkit – библиотека, предназначенная для функционирования встроенного в android web-браузера. Webkit поддерживает css, javascript, dom, ajax.

· SGL – движок для работы с 2d-графикой. Android также поддерживает собственную графическую 2d-библиотеку skia, которая написана на языке c и с++ (skia также формирует ядро популярного браузера google chrome).

· SSL – библиотека предназначена для работы с сокетами, основанная на протоколе ssl: sslv3.0 или tslv1.2.

### **Среда выполнения**

Среда выполнения обеспечивает библиотеки ядра dalvik virtual machine (виртуальная машина dalvik), которые предоставляют требуемую функциональность для java-приложений.

**Dalvik virtual machine.**Прикладное программное обеспечение, запускаемое на мобильном устройстве, исполняет виртуальная машина dalvik, которая хоть и является аналогом виртуальной машины java, существенно от нее отличается. Dalvik относится к классу регистровых машин (регистры процессора используются как первичные модули хранения данных), идеально подходящих для работы на процессорах risc-архитектуры, к которым относятся и процессоры arm, применяемые в мобильных устройствах, тогда как стандартная виртуальная машина java компании sun microsystems – стековая. В результате использования регистровой виртуальной машины Google надеется на 30 процентов уменьшить количество команд по сравнению со стековыми машинами.

Созданные с помощью стандартного java-компилятора class-файлы и преобразуются в байт-код dalvik (\*.dex) транслятором dx, входящим в состав sdk. Изнутри работающий Android выглядит как набор виртуальных машин dalvik, в каждой из которых исполняется прикладная задача.

Виртуальная машина dalvik, на которой построена вся операционная система Google Android, дает разработчикам приложений удобный механизм для написаний приложений, которым не принципиален объем используемой памяти и мощность процессора.

**Core libraries.** Включает набор основных библиотек, которые предоставляют функциональность для java. Библиотеки ядра обеспечивают слой API и являются основной платформой разработки java-приложений для Android.

### **Уровень каркаса приложений.**

Уровень каркаса приложений находится на вершине системных библиотек, функциональных библиотек и dalvik vm. На этом уровне находятся основные службы Android для управления жизненным циклом приложений, пакетами, ресурсами и т.д.

Программист имеет полный доступ к тем же API, которые используются основными приложениями. Архитектура этих приложений разработана с целью упрощения многократного использования компонентов. Любое разрабатываемое приложение может использовать возможности базовых приложений и, соответственно, любое стороннее приложение может использовать возможности вашего приложения (с учетом установленных разрешений). Этот же самый механизм позволяет многократно использовать уже разработанные компоненты.

Службы Android – это службы, которые являются основными для всех приложений, работающих на устройстве. К ним относятся:

* Менеджер деятельностей (activity manager) – управляет жизненным циклом приложений и предоставляет систему навигации по истории работы с деятельностями (стеку деятельностей);
* Менеджер пакетов (package manager) – управляет установкой и развертыванием пакетов прикладных программ, которые находятся на устройстве;
* Менеджер окон (window manager) – сохраняет окна приложения. Если разработчик предусмотрел вывод экрана, а затем переключение на другой экран, первый будет сохранен операционной системой и поставлен в режим ожидания. Это, в свою очередь, позволяет с помощью клавиши <back> мобильного устройства просматривать уже использовавшиеся экраны подобно тому, как это делается в web-браузере;
* Менеджер ресурсов (resource manager) – предназначен для доступа к строковым, графическим и другим типам ресурсов;
* Контент-провайдеры (content providers) – службы, которые позволяют приложениям получать доступ к данным других приложений, а также предоставлять сторонним приложениям доступ к своим данным;
* Система представлений (view system) – система с расширяемой функциональностью, которая служит для создания внешнего вида приложений, включающего такие компоненты, как списки, таблицы, поля ввода, кнопки, встроенный web-браузер и многое другое;
* Телефонный менеджер – обеспечивает слой api, контролирующий основную телефонную информацию, такую как сетевой тип и статус подключения, а также предоставляет различные утилиты для управления телефонными номерами;
* Менеджер местоположения – навигационные службы, которые позволяют приложениям получать периодические обновления географического местоположения устройства или запускать определенное приложение;
* Менеджер уведомлений – позволяет любому приложению отображать пользовательские уведомления в строке состояния.

### **Уровень приложений**

Мобильное устройство Android поставляется с набором основных приложений, включая почтового клиента, программу для работы с sms, календарь, навигационные карты, браузер, контакты и др.

Что интересно, платформа Android не делает разницы между основными приложениями телефона сторонним программным обеспечением – таким образом, ключевые приложения, входящие в стандартный набор программного обеспечения, можно заменить при желании альтернативными приложениями. Программы для Android пишутся на языке java.

При разработке приложений программисты имеют полный доступ ко всей функциональности операционной системы. Архитектура приложений построена так, чтобы было легко использовать основные компоненты предоставляемые системой. Также есть возможность создавать свои компоненты и предоставлять их в открытое использование.

## II. Разработка приложения для Google Android

## Вывод

По экспертным оценкам, наиболее распространенной ОС на мобильных устройствах является Google Android – около 47%. А следовательно, почти у каждого второго есть в кармане смартфон с установленной на нем ОС от Google. Что говорит о ее удобстве и качестве, потребитель не соврет.

Для того чтобы начать разработку, нам необходимо подготовить все необходимое программное обеспечение. В первую очередь, на компьютер устанавливается Java Runtime Environment.

Java Runtime Environment (сокр. JRE) — минимальная реализация виртуальной машины, необходимая для исполнения Java-приложений, без компилятора и других средств разработки. Состоит из виртуальной машины — Java Virtual Machine и библиотеки Java-классов.

JRE распространяется свободно и для большинства платформ может быть загружена с сайта Oracle. Свежий пакет установки находится по адресу http://www.java.com/inc/BrowserRedirect1.jsp?locale=en

После установки JRE необходимо скачать и распаковать пакет Eclipse Classic по адресу **www.eclipse.org/downloads.** Eclipse — свободная интегрированная среда разработки модульных кроссплатформенных приложений. Развивается и поддерживается Eclipse Foundation.

Наиболее известные приложения на основе Eclipse Platform — различные «Eclipse IDE» для разработки ПО на множестве языков (например, наиболее популярный «Java IDE», поддерживавшийся изначально, не полагается на какие-либо закрытые расширения, использует стандартный открытый API для доступа к Eclipse Platform).

Когда Eclipse распакован, с сайт [**http://developer.android.com/sdk/index.html**](http://developer.android.com/sdk/index.html)необходимо скачать и распаковать Android SDK.

В распакованной папке находим SDK Manager и запускаем его. В диалоговом окне уже отмечены Tools – инструменты, необходимые нам для работы с SDK.

* Android SDK Tools
* Android SDK Platform Tools

Также обязательно нужно отметить платформу, для которой мы будем разрабатывать приложение – я выбрал платформу Android 2.2 (API 8). Отмечаем нужное нам ПО (Google API и Galaxy Pad нам, допустим, не нужен). Таким образом, после установки у нас должны быть установлены инструменты (Tools) и платформа 2.2 со следующим ПО:

* SDK Platform
* Samples for SDK

Теперь у нас есть все инструменты для разработки, осталось только связать их воедино.

По умолчанию, Eclipse не работает с Android. Поэтому, необходимо установить ADT-плагин, который настраивает среду разработки Eclipse для работы с Android SDK.

Чтобы установить его, открываем Eclipse, указываем рабочее пространство (workspace) и переходим в меню Help – Install new software… В появившемся диалоговом окне нажимаем Add.. в правом верхнем углу, в поле name вводим Android Development Tools, в поле Location **https://dl-sl.google.com/android/eclipse/** и нажимаем OK.

Когда в окне появится Development Tools отмечаем ее и нажимаем Next. После просмотра устанавливаемых компонентов:

* Android DDMS
* Android Development Tools
* Android Hierarchy Viewer
* Android Traceview

Нажимаем Next, прочитываем лицензионное соглашение и нажимаем Finish.

После установки всех компонентов, перезапускаем Eclipse и указываем папку с распакованным SDK Tools. Среда разработки для Android-устройств готова к использованию.

## Создание приложения для Android

Перед тем как создать своё первое приложение, можно создать виртуальное Android-устройство, чтобы быстро тестировать на нём свой свеженаписанный софт. Android Virtual Device (AVD) - это виртуальный смартфон с ОС Android, на котором с легкостью можно запускать созданную нами программу. Как можно заметить, преимущество Android Virtual Device состоит в том, что наглядно можно увидеть как будет работать наша программа на разных смартфонах с Android, а не покупать весь модельный ряд и тестировать приложение на каждом из них.

Приступим к созданию виртуального устройства Android. Следуем в меню по пути **Window → Android SDK and AVD Manager**.

Выбираем New в правой части окна, в появившемся окне вписываем имя виртуального устройства, платформу (допустим Android 2.1), размер карты памяти (допустим 512 Mb), тип дисплея (допустим HVGA). Затем жмем внизу на кнопку Create AVD.

Теперь создадим проект. Для этого идем по пути в меню **File → New → Other**, в появившемся списке **Android → Android Project**.

* Project Name - имя которое будет отображаться в проводнике. Но не на самом устройстве.
* Contents - контент. Здесь мы задаем место где будет храниться проект. Так как этот проект новый надо установить переключатель на Create new project in workspace. Также вы можете выбрать путь где будет хранится наш проект сняв галочку с Use default location.
* Build Target - здесь мы можем установить для какой версии Android будет скомпилировано наше приложение. Я всегда беру версию Android 2.1. Так как её будут поддерживать все устройства работающие на этой версии и выше.
* Application Name - это название приложения будет отображаться на телефоне.
* Package name - надо ввести имя пакета. Так как мы будем программировать на Java мы должны ввести имя пакета удовлетворяющего пространству имен этого языка. Выглядит это так : com.название компании.название приложения. Вместо названия компании и приложения можно ввести что угодно (любые слова).
* Create Activity - создать деятельность.Он служит для создания класса запуска Activity по умолчанию.

\*Деятельность - это центральный компонент платформы Android. Это как задачи на Windows. Каждое приложение выполняет определённую задачу на определённом экране пользовательского интерфейса. У нас пока будет только одна деятельность YourActivity

В проводнике появился наш проект. Раскроем его и все его вкладки.

* AndroidManifest.xml - основной файл (файл манифеста) приложения. В нем находится и редактируется конфигурация приложения
* default.propretis - это файл сборки который использует Eclipse. Его нельзя редактировать
* /src - папка где хранится исходный год приложения
* /src/com.feel.you/youactivity.java - главная точка входа приложения (запуск по умолчанию) назначенная в файле манифеста.
* /gen/com.feel.you/R.java - этот файл генерируется автоматически. Служит для управления ресурсами телефона при работе приложения. Его не нужно редактировать.
* /assets - папка для хранения нескомпилированных файлов ресурсов проекта
* /res - папка в которой находятся все ресурсы приложения. Иконки, картинки, анимация, музыкальные файлы и т.д.
* /res/drawble - ldpi - находятся маленькие графические ресурсы приложения; mdpi -находятся средние графические ресурсы и hdpi - находятся большие графические ресурсы приложения
* /res/layout/main.xml - файл макета экрана. Используется деятельностью для прорисовки экрана
* /res/values/strings.xml - строковые ресурсы приложения

Файл манифеста:

* Manifest - Манифест. На этой вкладке определяются общие параметры приложения. Версия приложения например.
* Application - Приложение. На этой вкладке мы можем изменить имя и значок приложения. Какие деятельности могут выполняться. Устанавливаются разрешения. Устанавливаются возможности и сервисы, обеспечиваемые приложениями.
* Permissions - Разрешения. на этой вкладке определяются права приложения.
* Instrumentation - Инструментарий. Эта вкладка используется для тестирования компонентов приложения с помощью Android SDK
* AndroidManifest.xml - это редактор исходного кода вручную.Точнее всего файла манифеста.

Компоненты:

## Системные функции Android:

У Android есть одна ключевая особенность. Он может "заимствовать" некоторые элементы другого приложения. Каждое запущенное приложение Android работает в собственном секторе процессора Android. Это нужно для реализации многозадачности. Процессор выкидывает менее важные приложения чтобы освободить память для работы более важных приложений и процессов. Каждому приложению в Android OS присваивается уникальный Linux ID. По этому ID приложение определяется в системе и ему выставляются определённые разрешения и права. Каждый процесс запускается и имеет собственную виртуальную машину, поэтому приложение выполняется отдельно от других приложений. Приложения Android не имеют единственной точки входа (начала) приложения. Поэтому приложения состоят из так называемых КОМПОНЕНТОВ, код которых система может запускать и выполнять в любом порядке по необходимости.

Рассмотрим четыре компонента:

**Activities**- Деятельности - Одна деятельность = Одна задача. Деятельность представляет собой интерфейс приложения. Приложение может состоять из одной деятельности или более, но все деятельности идущие после основной становятся субклассами базового класса Activity

**Views** - Виды - Виды обеспечивают взаимодействии с пользователем. Они принимают информацию и отправляют её пользователю. Это кнопки, загрузчики, поле для ввода т.д.

**Intent**- Интент (Цель, Намерение) - Это цель выполнить определённое

действие. С помощью интента можно :

· запустить деятельность

· запустить службу

· ответить или набрать номер

· передать широковещательное сообщение

**Services** - Службы - работают в фоновом режиме. У каждой службы есть своя цель в приложении. Они запускаются по запросу пользователя или приложения на неопределенный промежуток времени. Например у нас играет плеер. Мы хотим закрыть плеер и посмотреть E-mail. Поэтому деятельность вызывает службу фонового воспроизведения и плеер играет в фоне. Каждая служба наследуется от базового класса Service.

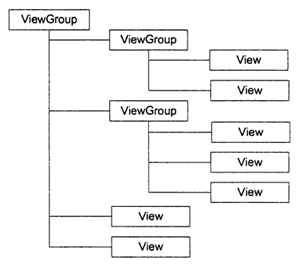
**Broadcast Recivers** - Подписчики широковещательных сообщений - это сообщения очень похожи на СМС. Когда кончается заряд батареи, например, выскакивает сообщение "Включить режим энергосбережения?". Но у этих сообщений есть одна особенность они могут запускать деятельности. Приложение может иметь любое число подписчиков. Все подписчики наследуются от базового класса BroadcastReceiver.

**Content Providers** - Поставщики содержимого - делают специальный набор данных приложении доступным для других приложений. Данные могут храниться файловой системе, в SQLite базе данных, или в любой другой подходящей форме. Поставщики наследуются от базового класса ContentProvider, реализующего стандартный набор методов для получения и сохранения данных заданного типа. Однако, приложение не может вызывать эти методы напрямую. Вместо этого они используют ContentResolver объект и его методы.

На платформе android необходимо определить пользовательский интерфейс для каждой деятельности, используя иерархии узлов view и viewgroup, как показано на рисунке 4.

Объекты view – основные модули отображения интерфейса пользователя на платформе android. Объект view – структура данных свойства которой сохраняют параметры разметки и содержание для определенной прямоугольной области экрана. Как объект в интерфейсе пользователя, объект view является точкой взаимодействия пользователя и программы.

Класс viewgroup служит ядром для подклассов, называемых разметками (layouts), которые формируют расположение элементов пользовательского интерфейса на форме, использую различные виды архитектуры разметки – фреймовый, линейный, табличный и относительный.



**Рис. 1** пример дерева представлений для деятельностей