ПРАВИТЕЛЬСТВО РОССИЙСКОЙ ФЕДЕРАЦИИ

Пермский филиал

федерального государственного автономного образовательного

учреждения высшего профессионального образования

"Национальный исследовательский университет

"Высшая школа экономики"

Факультет бизнес-информатики

Кафедра информационных технологий в бизнесе

УДК 004.9

**ТЕМА РАБОТЫ**

Проектирование приложения мобильной печати для ОС Android

|  |  |
| --- | --- |
|  | Работу выполнил студент  группы БИ-09-1  4 курса факультета бизнес-информатики  Гришин Иван Олегович  Научный руководитель:  Румянцев Александр Николаевич  “07” мая 2013 г. |

Пермь 2013

Оглавление

[Введение 3](#_Toc356213413)

[Глава 1. 5](#_Toc356213414)

[1.1 Классификация мобильной печати 5](#_Toc356213415)

[1.2 Обзор существующих решений мобильной печати на ОС Android 5](#_Toc356213416)

[1.3 Обзор процесса печати 6](#_Toc356213417)

[1.3.1 Процесс поиска принтера 6](#_Toc356213418)

[1.3.2 SNMP. Обзор 7](#_Toc356213419)

[1.3.3 Программное обеспечение Bonjour. Обзор 7](#_Toc356213420)

[1.3.4 Установка соединения с принтером 8](#_Toc356213421)

[1.3.5 IPP. Обзор 8](#_Toc356213422)

[1.3.5 LRP. Обзор 9](#_Toc356213423)

[1.3.5 BJNP. Обзор 10](#_Toc356213424)

[1.4 Выбор драйвера 10](#_Toc356213425)

[1.4.1 Язык описания страниц 11](#_Toc356213426)

[1.4.2 PCL (Printer Control Language) 11](#_Toc356213427)

[1.4.3 PostScript 12](#_Toc356213428)

[1.4.4 Ghostscript 14](#_Toc356213429)

# Введение

Начиная с 2011 года мобильные технологии с каждым днем становятся все популярнее и популярнее. По данным Deloitte Canada Research, именно в этом году, впервые объем продаж смартфонов и планшетных компьютеров превысил объем продаж персональных компьютеров. Этот факт не является столь удивительным в наше время, ведь мобильные устройства стали настолько мощными и доступными, что немногие могут отказаться от приобретения смартфона или планшета.

Сейчас люди могут себе позволить делать все тоже самое на мобильных устройствах, что они привыкли делать на персональных компьютерах. Вместе с ростом продаж мобильных устройств наблюдается тенденция роста мобильности рабочей силы. По прогнозам, в 2014 году мобильная рабочая сила будет составлять 35% от всей мировой рабочей силы. Мобильность рабочей силы достигается средствами мобильных устройств, облачных технологий, развивающихся мобильных приложений. Многие предприниматели заинтересованы в возможностях внедрения мобильных технологий в рабочий процесс и применении их в решении нетрадиционных задач. Использование мобильных устройств в бизнесе позволяет приобрести конкурентное преимущество.

Предоставление документов в печатном виде является одной из неотъемлемых частей рабочего процесса. Однако, когда процесс печати с персональных компьютеров является простым и понятным, возникают трудности, когда речь заходит о печати с мобильных устройств. В настоящее время существует широкий диапазон приложений для решения этой проблемы, однако все из них для завершения печати требуют отдельный сервер или персональный компьютер. На данный момент не существует такого приложения, которое позволило бы печатать напрямую с мобильного устройства с помощью Wi-Fi, Bluetooth, USB, поддерживая большинство принтеров различных фирм.

Таким образом целью данной работы является увеличение продуктивности работы пользователей путем предоставления готового решения, позволяющего распечатывать контент различного типа с мобильных устройств напрямую на принтеры.

Для достижения данной цели было поставлено несколько задач:

1. Анализ имеющихся решений
2. Поиск и изучение литературы о методах соединения с принтерами
3. Изучение возможных протоколов печати
4. Изучение процесса соединения и передачи данных на принтер
5. Создание пользовательского интерфейса мобильного приложения
6. Проектирование приложения

Предметом дипломной работы является сам процесс печати с мобильного устройства, который заключается в поиске доступных принтеров, установления с ними соединения, и передачи данных в доступным для них виде.

Результатом данной работы должно являться проект мобильного приложения для ОС Android, позволяющее пользователю подключаться к принтерам по Wi-Fi, Bluetooth и USB и распечатывать как со смартфонов, так и планшетных компьютеров изображения, документы, электронные письма и другого вида контент. Так же пользователю должна предоставляться возможность выбора различных параметров печати, таких как: выбор страниц для печати, количество копий, размер бумаги, качество печати, выбор лотка принтера.

# Глава 1.

## 1.1 Классификация мобильной печати

Мобильная печать является новым и развивающимся рынком с рядом различных реализаций и без каких-либо четких стандартов. В результате, сегодня существуют различные способы классификации имеющихся решений в области мобильной печати. Во всех случаях, задание на печать отправляется с мобильного устройства на принтер по беспроводному соединению. Некоторые классификации способов печати имеющихся решений:

* По методам отправления документа на печать
  + Напрямую из мобильных офисных приложений с помощью кнопки «Печать»
  + Из приложений мобильной печати
  + Из почтового клиента, как специальное вложение или печать тела письма.
* По месту, откуда пользователь печатает
  + Внутри инфраструктуры организации
  + Вне организации
* Приложения привязанные и непривязанные к фирме-производителю печатающих устройств
* По месту рендеринга и отправлению данных
  + Мобильные устройства
  + Печатающий сервер
  + Персональный компьютер
  + Облачный сервис
* По типам документов, которые могут быть распечатаны

## 1.2 Обзор существующих решений мобильной печати на ОС Android

На официальном маркете приложений Google Play предоставлен широкий выбор приложений мобильной печати. Рассмотрим самые популярные предложения:

1. Cloud Print – позволяет печатать с мобильного устройства используя Google Cloud Print сервис, поддерживаемый некоторыми современными принтерами, и принтерами подключенными к компьютеру с включенным Google Chrome
2. Brother iPrint&Scan, Canon Easy-PhotoPrint, HP iPrint Photo, Samsung MobilePrint – Как можно догадаться из названий, эти приложения разработаны компаниями производителями принтеров, и поддерживают принтеры только соответствующего производителя доступные по локальной сети.
3. Print2Ext – Печать с помощью программы-посредника на компьютере, которую можно скачать с [www.externalprint.com](http://www.externalprint.com/). Приложение на Android-устройстве посылает письма на определенный ящик с определенной темой, а программа-посредник проверяет почтовый ящик и печатает вложения.
4. Breezy Print – Печать посредством компьютера с подключенным принтером. Программу-посредника можно скачать с [www.breezyprint.com](http://www.breezyprint.com/), предварительно зарегистрировавшись. Печатает любой документ, который может быть распечатан с компьютера, и именно так, как будто он и был распечатан с компьютера.

Можно заметить, что практически все предлагаемые решения используют либо облачную технологию, либо компьютер в качестве посредника между мобильным устройством и устройством печати, что означает невозможность печати без постоянно включенного компьютера или без Интернет-соединения. Также те решения, которые не используют посредников при печати, предоставляются только для принтеров того же производителя, что и приложения. Более того эти приложения не поддерживают Bluetooth и USB соединение. В качестве вывода можно сказать, что данная работа является актуальной и целесообразной.

## 1.3 Обзор п**роцесса установки принтера**

### 1.3.1 Процесс поиска принтера

Для того, чтобы функциональность приложения мобильной печати не отличалась от возможностей персональных компьютеров и для максимальной комфортности использования приложения, в первую очередь необходима реализация поиска принтера и автоматического выбора наиболее подходящего драйвера без участия пользователя. Если же с поиском принтеров по Bluetooth и по USB не возникает особых проблем в связи с простотой обнаружения устройств, то с обнаружением сетевых устройств в локальной сети возникают некие трудности. Многие существующие решения на рынке не предоставляют возможности сканирования сети на наличие сетевых принтеров, а предлагают вводить вручную сетевой адрес, однако не многие пользователи догадываются, как и где взять этот адрес. Поэтому в рамках данной работы было решено реализовать автоматический поиск всех доступных принтеров в сети. Данная возможность может быть внедрена с помощью простого протокола сетевого управления (SNMP) и [протокол](http://ru.wikipedia.org/wiki/%D0%9F%D1%80%D0%BE%D1%82%D0%BE%D0%BA%D0%BE%D0%BB_%D0%BF%D0%B5%D1%80%D0%B5%D0%B4%D0%B0%D1%87%D0%B8_%D0%B4%D0%B0%D0%BD%D0%BD%D1%8B%D1%85)а автоматического [обнаружения сервисов](http://ru.wikipedia.org/wiki/%D0%9E%D0%B1%D0%BD%D0%B0%D1%80%D1%83%D0%B6%D0%B5%D0%BD%D0%B8%D0%B5_%D1%81%D0%B5%D1%80%D0%B2%D0%B8%D1%81%D0%BE%D0%B2) Bonjour**.**

### 1.3.2 SNMP. Обзор

SNMP — стандартный интернет-протокол для управления устройствами в IP-сетях на основе архитектур [UDP](http://ru.wikipedia.org/wiki/UDP)/[TCP](http://ru.wikipedia.org/wiki/TCP). К поддерживающим SNMP устройствам относятся маршрутизаторы, коммутаторы, серверы, рабочие станции, принтеры, модемные стойки и другие. Протокол обычно используется в системах сетевого управления для контроля подключенных к сети устройств на предмет условий, которые требуют внимания администратора. SNMP определен Инженерным советом интернета (IETF) как компонент TCP/IP. Он состоит из набора стандартов для сетевого управления, включая протокол прикладного уровня, схему баз данных и набор объектов данных.

SNMP предоставляет данные для управления в виде переменных, описывающих конфигурацию управляемой системы. Эти переменные могут быть запрошены (а иногда и заданы) управляющими приложениями.

### 1.3.3 Программное обеспечение Bonjour. Обзор

Предоставляющий простой подход к обнаружению сервисов локальной сети, Bonjour широко введен в эксплуатацию по средствам Mac OS X. Программное обеспечение позволяет пользователю создавать локальные сети без хлопотной настройки. В данный момент, Bonjour широко применяется для поиска принтеров и серверов, которые настроены на общий доступ. Bonjour использует совершенно новый протокол для обнаружения сетевых служб, а также поддерживает все прикладные протоколы на базе TCP/IP или UDP/IP. Bonjour работает в рамках одного домена широковещательной передачи по небольшой площади без каких-либо специальных доменных имен (DNS). Разработчики Apple опубликовали исходный код Bonjour для создания различных сборок на базе DNS, покрывающих большинство существующих платформ, включая Mac OS 9, OS X, Linux, Windows и многие другие. Совсем не многие люди знают о возможностях программного обеспечения Bonjour. Начнем с того, что необычный протокол Bonjour применяется не только для обнаружения информации, но и для контроля состояния информации, которая имеет свойство меняться. Bonjour умеет различать статистическую и динамическую информацию и уведомляет об изменениях всех клиентов, которые присутствуют в локальной сети, о случившемся изменении.

В случае с принтерами, Bonjour Print Service позволяет открыть и запустить принтер с помощью Мастера Настройки Принтеров Bonjour (Bonjour Printer Wizard). Эта служба работает с локальными принтерами по средствам Wi-Fi или Ethernet.

### 1.3.4 Установка соединения с принтером

Найти в сети принтер и получить его сетевой адрес – это половина пути по подключению принтера. Для того, чтобы иметь возможность передавать данные на сетевой принтер, необходимо установить с ним соответствующее соединение. Каждый принтер поддерживает некоторые протоколы сетевой печати, по которым он может принимать задание на печать. Сегодня наиболее популярными протоколами являются: IPP (*Internet Printing Protocol* — «протокол межсетевой печати», «протокол печати через Интернет»), LPR (*Line Printer Remote* — протокол «построчной печати на удалённом принтере»), **BJNP – (протокол используемый принтерами производства Canon). Реализуя данные протоколы, можно установить соединение с большинством сетевых принтеров. Что же касается Bluetooth и USB, то для передачи данных на принтер будет достаточно внутренней реализации в ОС Android. Для Bluetooth будет использоваться реализация** [SPP](https://ru.wikipedia.org/w/index.php?title=Bluetooth_Serial_Port_Profile&action=edit&redlink=1) (*Serial Port Profile*), который в свою очередь использует протокол [RFCOMM](https://ru.wikipedia.org/wiki/RFCOMM).

### 1.3.5 IPP. Обзор

IPP (Internet Printing Protocol) — протокол печати через Интернет, описывает и поддерживает стандартные способы пересылки по Интернету заданий на печать. Он позволяет пользователям получать сведения о готовности принтеров к печати независимо от их месторасположения, а администраторам — следить за статусом принтеров не только в пределах локальной рабочей группы. Использование IPP позволяет получить ряд преимуществ по сравнению с факсимильной связью, включая более высокое качество печати (разрешающая способность факсимильных аппаратов 200 точек/дюйм, принтеров — от 600 до 1200 точек/дюйм с полноценной передачей цвета), быстродействие (в результате использования высокоскоростных принтеров) и выигрыш в стоимости.

### 1.3.5 LRP. Обзор

LRP— сетевой протокол прикладного уровня для передачи документов на печать, является стандартом де-факто для UNIX-систем, предоставляющий базовые возможности печати. В отличие от Windows-печати, оперирует не «принтерами», а «очередями» — в простейшем случае очередь отождествляется с именем порта принтера: например, очередь *lpt2* будет обозначать принтер, подключённый к порту *LPT2* на сервере печати.

Исторически для печати в UNIX-системах существовали две системы печати: LPD, разработанная для Berkeley UNIX, и AT&T Line Printer system. Эти системы печати были созданы в 70-х годах для печати текстов на построчно-печатающих (линейных) принтерах. Принимая во внимание, что аппаратные средства печати с тех пор существенно изменились, можно было бы предположить, что существенно переработаны и программные средства для управления печатью. Однако, этого не произошло. Хотя и были созданы различные улучшенные системы печати, например, LPRng, однако ни одна из этих новых разработок не изменяла фундаментальные возможности этих систем.

Из-за того, что принтеры являются относительно медленными устройствами, с целью исключить задержки в программах система печати использует фоновую печать с кешированием данных. Файл данных, предназначенный для печати на принтере, сначала помещается во временную область (временный каталог на диске), которая называется областью спулинга или каталогом спула. Фоновый процесс — демон печати — периодически сканирует область спулинга в ожидании файлов, предназначенных для печати. Для каждого принтера, подключенного к системе, используется свой подкаталог в области спулинга. Таким образом, область спулинга представляет собой набор очередей заданий на печать, дожидающихся того момента, когда освободится соответствующий принтер и демон печати отправит данное задание на печать (в фоновом режиме).

### 1.3.5 BJNP. Обзор

Протокол BJNP является частью более старых моделей принтеров Canon Pixma. В настоящее время новые версии протокола поддерживают сетевую установку и печать. С новым технологиям, внедренным в принтерах Canon, несколько пользователей могут иметь возможность печатать документы с помощью сетевого принтера без необходимости устанавливать программное обеспечение на каждом из отдельных компьютеров, связанных с определенной сети. Пользователи всегда жаловались на ограничения печати в соответствии с старым протоколом BJNP. Для некоторых, это просто большая проблема установить несколько программ и драйверов на разные компьютеры только для печати простых документов. Для новых принтеров, которые могут быть сконфигурированы для доступа в сети, печать выполняется проще и удобнее.

## 1.4 Выбор драйвера

Следующим этапом в процессе установки принтера является выбор драйвера. Драйвер принтера – в данной работе является алгоритм обработки растрового изображения в тот вид, который принтер требует. Если быть точнее, то это преобразование массива байтов изображения с добавлением различных команд принтера. Так же при установке драйвера инициализируются такие параметры принтера, как доступные размеры бумаги для печати, доступные виды качества печати, виды лотков, и возможность дуплексной печати. На данный момент существует неисчислимое количество принтеров разных производителей и разных моделей, каждый из которых принимает дынные только в том виде, в котором он требует. И реализовать поддержку каждого из них практически невозможно, тем более для мобильных устройств. Однако существуют некоторые стандарты, которые поддерживаются во многих современных принтерах. Таким стандартом является язык описания страниц PCl и PostScript. Однако, что делать с теми принтерами, которые не поддерживают эти языки. Оказывается, существует такое решение, как Ghostscript, которое позволяет отправлять задание на печать в приемлемом виде для большинства популярных принтеров. В рамках данной работы будет использоваться Ghostscript IJS - Inkjet and other raster devices – программное обеспечение, которое позволяет отправлять задание на печать в виде растрового изображения.

### **1.4.1 Язык описания страниц**

Лазерные и струйные принтеры называют страничными, поскольку они формируют образ целой страницы в памяти перед перемещением его на бумагу. Это основное отличие лазерных и струйных принтеров от матричных, которые являются символьными. Для “общения” компьютера со страничным принтером применяется специализированный язык описания страницы (page description language — PDL). Это средство кодирования каждой части печатаемого документа в поток данных, который может быть передан на принтер. После получения принтером кодов языка описания страницы встроенное программное обеспечение принтера преобразует код в шаблон точек, которые переносятся на бумагу. В настоящее время существует два языка описания страниц, ставших фактическим стандартом в компьютерной индустрии, — PCL и PostScript. О них речь пойдет далее в этом разделе. Принтеры, не поддерживающие язык описания страниц, используют последовательность escape-кодов (escape code sequence) для управления свойствами принтера в комбинации со стандартным текстом ASCII для передачи содержимого документа. За это “отвечает” драйвер принтера, который распознает передаваемые символы — escape-коды или язык описания страниц. При печати документа неважно, в какой программе он был создан и в каком формате файла был сохранен; данные для печати должны быть преобразованы в поток данных языка описания страниц или поток ASCII-текста с escape-кодами.

1.4.2 PCL (Printer Control Language)

Язык описания страниц PCL разработан фирмой Hewlett-Packard в начале 80-х годов для использования в принтерах собственного производства. Когда Hewlett-Packard завоевала значительную часть рынка принтеров, язык PCL стал стандартом, который эмулируют многие производители. Кроме текста, который необходимо напечатать, поток данных языка PCL содержит множество команд, разработанных для управления принтером. Эти команды можно разделить на четыре категории:

* Управляющие коды. Стандартные коды ASCII, которые представляют собой функцию (например, возврат каретки (CR) или перевод строки (LF)), а не символы.
* Команды PCL. В основном состоят из последовательности escape-кодов, которые используются в матричных принтерах. Эти команды составляют значительную часть управляющего кода PCL-файла и включают специфичные для каждого принтера эквиваленты параметров документа (например, форматирование страницы и используемый шрифт).
* Команды HP-GL/2 (Hewlett-Packard Graphics Language — язык графики Hewlett-Packard). Служат для печати векторной графики составного документа. Они состоят из двухбуквенных мнемоник и одного параметра (или нескольких), определяющего  
  процесс выполнения команды принтером.
* Команды PJL (Printer Job Language — язык выполнения печати). Позволяют принтеру “общаться” с компьютером по двунаправленной линии для обмена информацией о состоянии, процессе печати и других параметрах.

С развитием возможностей принтеров совершенствовался и язык PCL. Первые версии языка применялись в струйных и портативных принтерах Hewlett-Packard в начале 80-х годов и не содержали языка описания страниц. В первой модели лазерного принтера LaserJet, выпущенной в 1984 году, использовался язык PCL 3, а последние модели лазерных принтеров поддерживают PCL 6.

### 1.4.3 PostScript

Этот язык описания страниц разработан фирмой Adobe и впервые использован в принтере Apple LaserWriter в 1985 году. Первые версии языка PostScript уже обладали такими возможностями, как масштабируемые шрифты и поддержка векторной графики, в то время как в язык PCL они были добавлены относительно недавно. Поэтому PostScript быстро стал (и остается поныне) промышленным стандартом для настольных издательских систем и графических программ. Фирма Adobe лицензировала язык PostScript многим производителям принтеров, включая производителей фотонаборного оборудования, широко используемого в допечатной подготовке газет, журналов, книг и другой печатной продукции.

Язык PostScript не поддерживает последовательность escape-кодов, как PCL; он больше напоминает стандартный язык программирования. PostScript называют объектно-ориентированным языком программирования, поскольку на принтер отправляется не изображение, а геометрические объекты. Для того чтобы напечатать текст определенным шрифтом, драйвер принтера должен указать последнему контур шрифта и его размер. Контур шрифта служит шаблоном для создания символов любого размера. Принтер генерирует изображение символа из его контура, а не загружает из памяти. Этот тип изображения, который генерируется индивидуально для каждой страницы, называется векторной графикой, в отличие от растровой графики, которая отправляется на принтер в виде готового набора точек. Возможность масштабирования шрифтов была добавлена только в пятую версию PCL, появившуюся в 1990 году.

При использовании контуров процесс печати шрифтов упрощается. Большинство принтеров содержат встроенные шрифты любого размера. Растровые шрифты должны быть загружены в принтер из компьютера. Различие между векторными и растровыми объектами можно заметить в напечатанном образце. Поскольку векторное изображение генерируется в принтере, его качество определяется возможностями принтера. Векторное изображение, напечатанное на принтере с разрешением 600 dpi, выглядит намного лучше, чем аналогичное изображение, напечатанное на принтере с разрешением 300 dpi.

Первые модификации языка PostScript отражали развитие возможностей лазерных принтеров Apple. Незначительные модификации языка легли в основу новой версии, которую фирма Adobe выпустила в 1992 году и назвала PostScript Level 2. На этом развитие языка не остановилось, и в 1997 году появилась его следующая версия— PostScript Level 3. В этих обновленных версиях языка PostScript поддерживается большая производительность PostScript-принтеров, а также реализована возможность использования большего объема установленной памяти принтера и нескольких лотков с бумагой. Однако революционных изменений, подобных происходящим в языке PCL, PostScript не переживает. Он обладает всеми необходимыми свойствами еще со дня своего появления, и его существующие версии остаются обратно совестимыми.

### 1.4.4 Ghostscript

Ghostscript — набор программного обеспечения, позволяющего интерпретировать язык PostScript и документы PDF.

Ghostscript может осуществлять конвертирование файлов PostScript в файлы различных графических форматов, выводить на дисплей интерпретированное содержимое и осуществлять печать на принтерах не поддерживающих PostScript; часто используется как «виртуальный принтер» для создания документов в формате PDF или PostScript из программ, не поддерживающих конвертирование в эти форматы.

Может использоваться как процессор растровых изображений (RIP) для растровых компьютерных принтеров — например, как входной фильтр для LPD — или как механизм RIP для программ просмотра PDF или PostScript.

Поскольку Ghostscript является интерпретатором языка, то он может также использоваться в качестве универсальной среды программирования. Ghostscript был перенесён на многие операционные системы, включая Unix, Linux, Mac OS, OpenVMS, Microsoft Windows, MS-DOS, FreeDOS, OS/2 и AmigaOS.

# Глава 2. Постановка задачи, выбор методов реализации и описание технического задания.

## 2.1 Постановка задачи

Задачей автора данной работы является проектирование приложения мобильной печати. Это влечет за собой выполнение нескольких подзадач:

* Выбор используемых технологий;
* Проектирование интерфейса мобильного приложения;
* Изучение публичного API и сторонних JAVA библиотек для реализации функционала приложения;
* Создание технического задания на разработку мобильного приложения;
* Проектирование мобильного приложения;

## 2.2 Выбор используемых технологий

Разработка приложений для мобильных устройств — это процесс, при котором приложения разрабатываются для небольших портативных устройств таких как КПК, смартфоны или сотовые телефоны. Эти приложения могут быть предустановленны на устройство в процессе производства, загружены пользователем с помощью различных платформ для распространения ПО или являться веб-приложениями, которые обрабатываются на стороне клиента или сервера (JavaScript).

В перспективе – реализуемое приложение должен быть представлен на всех популярных мобильных платформах, но в рамках данной работы была выбрана одна платформа наиболее подходящая для выхода на рынок. При выборе приоритетной платформы автор опирался на следующие критерии:

* Популярность платформы
* Удобство существующих инструментов разработчика
* Доступность разработки
* Наличие технических средств

### 2.2.2 Платформы для разработки

Каждая из платформ для мобильных приложений имеет интегрированную среду разработки, предоставляющую инструменты, позволяющие разработчику программировать, тестировать и внедрять приложения на целевую платформу. На текущий момент наиболее популярными платформами в мире являются Windows Phone, Android, iOS и BlackBerry. Однако BlackBerry практически не представлен на российском рынке, поэтому не рассматривается как перспективная платформа для создаваемого приложения.

Android — портативная (сетевая) операционная система для коммуникаторов, планшетных компьютеров, электронных книг, цифровых проигрывателей, наручных часов, нетбуков и смартбуков, очков Google основанная на [ядре Linux](http://ru.wikipedia.org/wiki/%D0%AF%D0%B4%D1%80%D0%BE_Linux). Изначально разрабатывалась компанией Android Inc., которую затем купила Google. Впоследствии Google инициировала создание альянса Open Handset Alliance, который сейчас занимается поддержкой и дальнейшим развитием платформы. Android позволяет создавать Java-приложения, управляющие устройством через разработанные Google библиотеки. Android Native Development Kit позволяет портировать библиотеки и компоненты приложений, написанные на Си и других языках.

В 75 % смартфонов, проданных в третьем квартале 2012 года, была установлена операционная система Android.

В качестве программной платформы для разработки по операционную систему Android могут быть использованы следующие IDE:

* + ADT (плагин Android для Eclipse)
  + Kenai (плагин Android для NetBeans)
  + IntelliJ IDEA

Лицензия разработчика Google Play покупается одноразово и стоит 20 долларов.

## 2.3 Изучение публичного API

## Разработка пользовательского интерфейса

Интерфейс пользователя (UI) - это часть программы, которая находится на виду у пользователя и призвана обеспечивать отображение данных, управление или диалог с пользователем. При разработке программного обеспечения программисты не должны оставлять дизайн интерфейса пользователя "на потом", считая, что в программе только код должен быть краеугольным камнем. Время от времени у пользователей возникает чувство раздражённости например даже из-за неудачно подобранных шрифтов, неудобного расположения элементов управления, отображения данных. Для удобства пользователей должно быть затрачено n-ое количество часов, так как это серьёзная работа. Это следует учитывать при озвучивании сроков разработки программы.

Удобство пользовательского интерфейса входит в такое понятие как "Эргономика", которое включается в процессы разработки и тестирования программного продукта как часть системы качества. Разработка пользовательского интерфейса (ПИ) должна вестись соместно с дизайном программного продукта в целом.

Все основные правила разработки ползовательского интерфейса описаны в User Interface Guidelines for Android. Этот документ появился вместе с выходом на рынок четвертой версии операционной системы. Руководствуясь им при разработке приложения, можно быть уверенным в том, что полученная разработка будет проста и понятна пользователю.

Для разработки прототипа интерфейса можно воспользоваться следующими сервисами и программами:

1. gotiggr.com
2. gliffy.com
3. Pencil
4. Axure
5. MS Visio
6. InDesign
7. LiquidUI

Для данной работы гораздо целесообразнее воспользоваться бесплатным программным обеспечением. Полностью бесплатен из вышеуказанного списка только программный продукт LiquidUI, доступный для всех операционных систем. LiquidUI предоставляет различные наборы фигур и иконок для рисования различных видов пользовательского интерфейса, как для настольных решений, так и для мобильных платформам. Начиная с версии 2.0.2, LiquidUI предоставляет встроенную поддержку Android Ice Cream Sanwich. Более того, LiquidUI расширяем, т.е. при необходимости можно подключить собственный набор интерфейсных элементов.

## 2.4 Создание технического задания на разработку приложения

Техническое задание — исходный документ на проектирование технического объекта. ТЗ устанавливает основное назначение разрабатываемого объекта, его технические характеристики, показатели качества и технико-экономические требования, предписание по выполнению необходимых стадий создания документации (конструкторской, технологической, программной и т. д.) и её состав, а также специальные требования.

В данной работе техническое задание разрабатывалось на основе ГОСТ 19.201-78. Полученный результат можно посмотреть в Приложении №6.

## 2.5 Проектирование

### 2.5.1 Архитектура OS Android

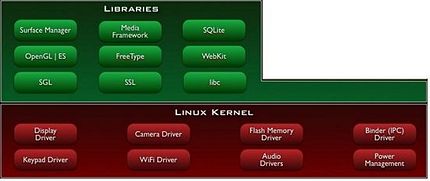
Android - это программный стек для мобильных устройств, который включает операционную систему, программное обеспечение промежуточного слоя (middleware), а также основные пользовательские приложения (e-mail-клиент, календарь, карты, браузер, контакты и другие).

Архитектура Android построена на основе ядра Linux версии 2.6. Оно отвечает за такие системные службы, как управление безопасностью, памятью, процессами, включает сетевой стек и модель драйверов. Кроме того, это ядро также играет роль абстрактного слоя между аппаратным уровнем и остальной частью программного стека.

[](http://wiki.androidfan.ru/%D0%A4%D0%B0%D0%B9%D0%BB:Arch_1.jpg)

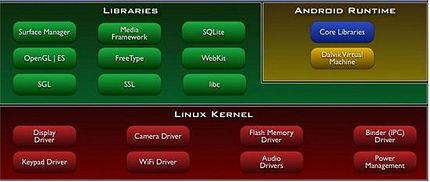
**Рис. 1** Ядро Linux

Следующий уровень в иерархической системе - библиотеки, написанные на C/C++. В медиабиблиотеках (Media Libraries) заложена поддержка основных форматов и кодеков, включая MPEG4, H.264, MP3, WMA, WAV, AAC, AMR, JPG, PNG. Графические библиотеки (3D Libraries) поддерживают API OpenGL ES 1.0 и обеспечивают поддержку аппаратных 3D-ускорителей. Как отмечается, в одном приложении можно одновременно использовать двух- и трехмерную графику. Еще одной примечательной библиотекой является LibWebCore (WebKit) - движок для web-браузера Android, используемый также в популярном браузере Safari от Apple.

[](http://wiki.androidfan.ru/%D0%A4%D0%B0%D0%B9%D0%BB:Arch_2.jpg)

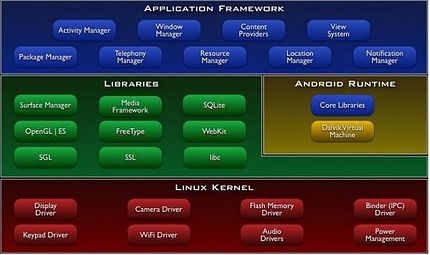
**Рис. 2** Уровень библиотек

Одной из важнейших частей стека является Android Runtime, состоящая из виртуальной Java-машины Dalvik и набора библиотек ядра. Dalvik позволяет поддерживать одновременную работу нескольких приложений и выполняет файлы в специальном формате .dex, оптимизированном для устройств с малым количеством памяти. Библиотеки ядра написаны на языке Java и включают большой набор классов, которые поддерживают широкий диапазон функциональных возможностей.

[](http://wiki.androidfan.ru/%D0%A4%D0%B0%D0%B9%D0%BB:Arch_3.jpg)

**Рис. 3** Виртуальная Java-машина Dalvik и набор библиотек ядра

Следующий уровень - Application Framework (каркас приложений). Этот уровень фактически представляет собой инструментарий, которым пользуются все приложения. Здесь стоит выделить такой компонент, как Content Providers (провайдеры данных), который является уникальной особенностью Android. Провайдеры данных позволяют, например, объединять информацию из интернета с данными телефона - контактной информацией или географической локацией, - чтобы реализовать новые функции. Кроме того, данные могут совместно использоваться разными приложениями.

[](http://wiki.androidfan.ru/%D0%A4%D0%B0%D0%B9%D0%BB:Arch_4.jpg)

**Рис. 4** Уровень каркаса приложений

И, наконец, на вершине пирамиды - уровень приложений (Applications). Что интересно, Android не делает разницы между основными приложениями телефона и сторонним программным обеспечением - таким образом, ключевые компоненты, такие как набор номера, рабочий стол или почтовый клиент GMail, можно заменить альтернативными аналогами. Приложения для Android пишутся на языке Java.

[](http://wiki.androidfan.ru/%D0%A4%D0%B0%D0%B9%D0%BB:Arch_5.jpg)

**Рис. 5** Уровень приложений

Прежде всего, система Android предназначена для мобильных устройств. Это означает, что наивысший приоритет отдается сохранению энергии батареи и эффективному управлению ограниченными ресурсами памяти. В системе Android можно выделить пять слоев:

Ядро Linux, работающее на архитектуре ARM (Acorn RISC Machine), является базой для остальных слоев. Linux - проверенная и высоконадежная технология, а семейство процессоров ARM известно своей высокой производительностью при низком энергопотреблении.

Набор библиотек предоставляет общий повторно используемый код низкого уровня для основных функций, таких как кодеки, позволяющие кодировать и декодировать цифровой звук и видео, функции для отображения качественной графики на маленьких дисплеях, поддержка зашифрованного трафика TCP/IP в облаке, поддержка компонентов для Web-серфинга (WebKit), функции SQL-СУБД (SQLite) и стандартная для Linux-систем функциональность библиотек C.

Интерпретатор байт-кода реального времени Dalvik, имеющий большое сходство с интерпретатором Java™, отличается несколькими свойствами, обеспечивающими Android уникальную безопасность и минимизацию потребления энергии. Каждое запущенное в текущий момент приложение имеет свой собственный пользовательский идентификатор и собственную копию интерпретатора, что обеспечивает строгое разграничение процессов для повышения надежности и безопасности.

Структура приложений Android позволяет использовать и заменять компоненты так, как вы считаете нужным. Эти интегрируемые компоненты являются высокоуровневыми Java-классами, составляющими Android API.

Стандартные приложения Android включают в себя браузер WebKit, календарь Google, клиент Gmail, приложение Gmaps, SMS-мессенджер и e-mail клиент. Приложения для Android написаны на Java, и вы всегда можете загрузить дополнительные приложения из магазина Android.

Далее, каждое приложение Android можно разделить на следующие функциональные модули:

* Activities (Действия)
* Intents (Намерения)
* Services (Службы)
* Security model (Модель безопасности)

Действия (Activities)

Действия (Activities) - это компоненты приложения Android, расширяющие базовый класс Activity и определяющие интерфейс, в который входит Представление (View), отвечающее на различные События (Events). Если приложение состоит из трех окон (например, окно авторизации, окно отображения текста и окно просмотра файлов), каждое из них, как правило, представляется отдельным классом Activity.

Android сохраняет историю действий для каждого приложения, запущенного со стартового экрана, и вы можете с помощью кнопки Back передвигаться назад по этой истории.

Намерения (Intents)

Намерения (Intents), подобно действиям, являются специальными классами в коде приложения, которые определяют и описывают запросы приложения на выполнение каких-либо операций. Намерения добавляют слой, позволяющий оперировать компонентами с целью их повторного использования и замещения. Например, приложение может иметь кнопку Клиенты, которая, будучи нажатой, отображает список клиентских контактов. С помощью Намерения вы можете использовать любой компонент для их отображения вместо стандартного.

В некоторых случаях это может быть очень мощным средством интеграции приложений. Например, топографическая карта в определенных случаях может быть лучше, чем изображение карты по умолчанию.

Классы наподобие BroadcastReceiver определяют код, который выполняется после возникновения внешних событий. Так можно отслеживать события типа срабатывания таймера или входящего звонка. Как правило, такой код не отображает никакого окна, но с помощью класса NotificationManager можно сообщить пользователю информацию, требующую его внимания.

Службы (Services)

Службой является приложение, запущенное в фоновом режиме и не имеющее графического интерфейса. Обычно подобные программы работают в фоне в течение долгого времени. Отличным примером является медиаплеер, проигрывающий треки из списка. Несмотря на то, что у такого приложения есть интерфейс, позволяющий пользователям составлять свои плейлисты, в процессе проигрывания песен оно может работать в автоматическом режиме службы.

Модель безопасности (Security model)

Модель безопасности Android позволяет программам иметь собственную область данных. Если разработчик хочет обобществить данные между несколькими различными программами, он может использовать поставщики данных (content providers).

Пакет android.provider содержит набор классов и интерфейсов, обеспечивающих доступ к многочисленным встроенным базам данных Android. Приложения с подобными интерфейсами могут с легкостью работать с контактами, фотографиями и музыкой.

Силу модели Open Source, как и силу свободного творчества, нельзя недооценивать. Не ограниченная закрытыми API и интересами собственника, которые зачастую препятствуют прогрессу программного обеспечения, платформа Android сформировала большое и очень активное сообщество, суммарная ценность которого больше ценности его отдельных частей.

Сердцем Android является ARM Linux. Одно это должно вселять уверенность в возможность быстрого роста этой платформы. Linux является быстрой и защищенной операционной системой, с которой знакомы тысячи программистов. Многие основанные на Linux системы работают без отказов на протяжении многих лет, будучи подключенными к облаку и обслуживая его запросы — и это является лучшим доказательством их надежности.

### 2.5.2 Структура приложения и интерфейс

Приложения для Android состоят из компонентов, который система может запускать и управлять так, как ей необходимо. Для этого система должна быть в состоянии запустить процесс для приложения, в котором находятся требуемые компоненты, и инициализировать нужные ей объекты. Одним из компонентов Android-приложения является деятельность (activity).

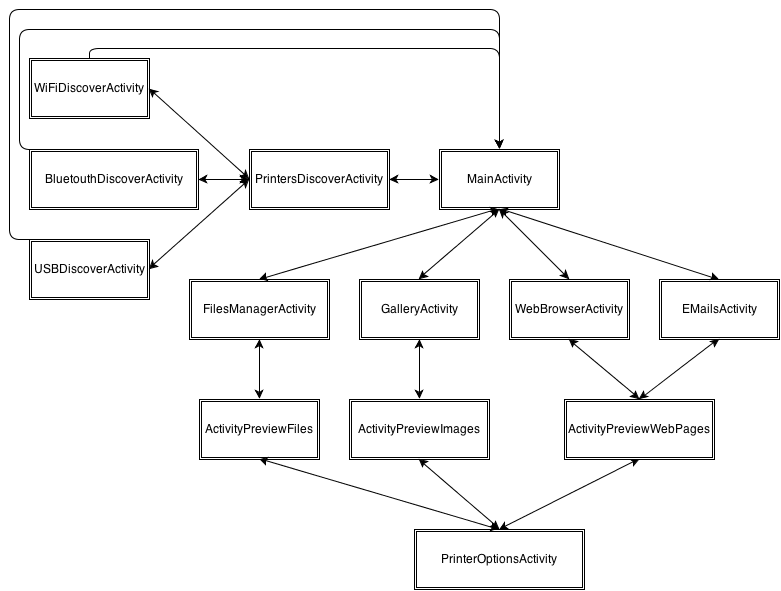
Activity представляет собой визуальный интерфейс (отдельный экран) для одного действия, которое пользователь может совершить.

Приложение может состоять из одного activity или из нескольких. Это зависит от типа приложения и его дизайна. Одно activity может вызвать другое.

Каждое activity задаёт окно для отображения, которое, обычно, занимает весь экран, но может быть меньше и плавать поверх других окон. Activity может использовать дополнительные окна, например, всплывающий диалог, который требует промежуточного ответа пользователя, или окно, которое отображает пользователям важную информацию при выборе элемента, заслуживающего особого внимания.

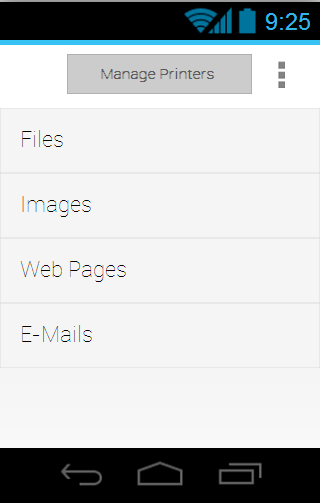
Визуальный интерфейс строится на основе иерархии визуальных компонентов — объектов, производных от базового класса View. Android имеет ряд готовых к использованию компонентов, включая кнопки, текстовые поля, полосы прокрутки, меню, флажки и многое другое.

Далее представлена структура деятельностей проектируемого приложения. MainActivity – главное activity, которое отображается при запуске приложения, стрелками обозначена возможность перехода из одной деятельности в другую. Обратная стрелка обозначает возможность возврата в вызывающе activity при помощи кнопки телефона «Назад» или пли при помощи интерфейсной кнопки.

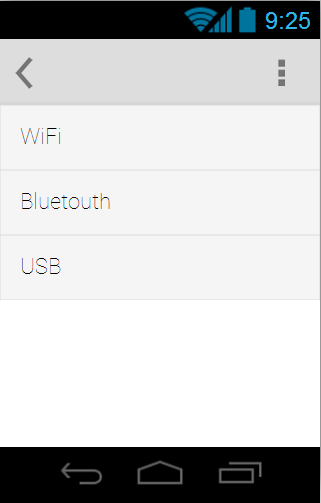


Далее рассмотрим сценарии действий пользователя для каждого activity и предлагаемый интерфейс.

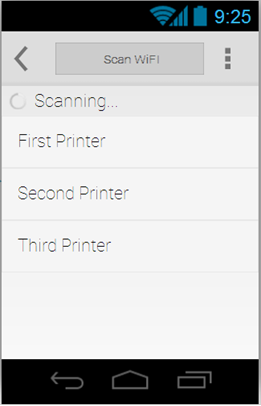
MainActivity – главное activity, которое отображается при первом запуске приложения. В верхней части приложения находится кнопка Manage Printers, открывающая DiscoverActivity. Если принтер уже установлен, надпись кнопки меняется на название текущего принтера. Далее следует список, который позволяет открывать файловый менеджер, галерею, веб-страницы, электронную почту. Впоследствии этот список можно расширять различными облачными сервисами хранения файлов. Предлагаемый интерфейс представлен на следующем рисунке.



DiscoverActivity – окно, позволяющее пользователю выбирать тип поиска принтера: WiFi, Bluetooth, USB. Впоследствии можно расширить список Google Cloud Printers и Windows Shared Printers. Предлагаемый интерфейс представлен на следующем рисунке.

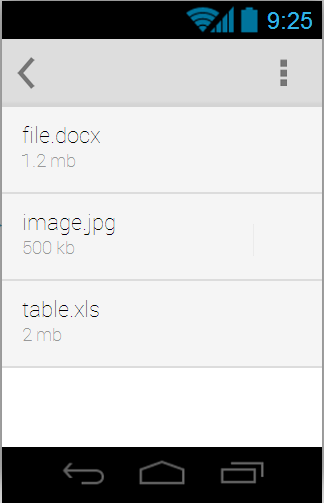


WiFiActivity – В данной activity происходит поиск сетевых принтеров, которые динамически добавляются в список. По нажатию на найденный принтер, начинается процесс установки драйвера принтера, по окончанию которого пользователь попадает в главное окно с сообщением об успешном соединении с принтером. Интерфейс состоит из кнопки Scan, которая позволяет возобновить процесс поиска принтеров, анимированного значка, отображающий процесс поиска, и самого списка найденных принтеров. Предлагаемый интерфейс представлен на следующем рисунке.

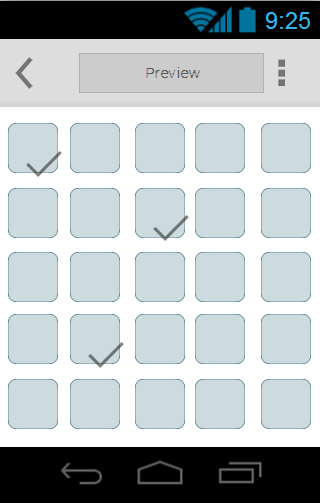


BluetoothDiscoverActivity и USBDiscoverActivity выглядят аналогичным образом и выполняют аналогичные функции.

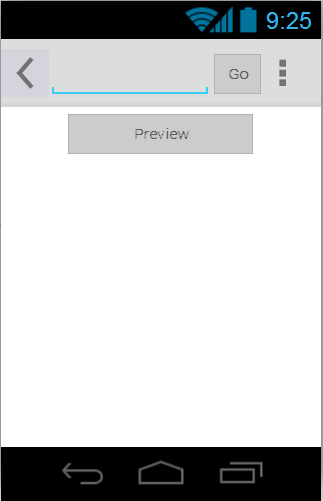
FilesManagerActivity – activity, которое позволяет пользователю выбирать документы или изображения из файловой системы устройства для дальнейшей распечатки. При нажатии на файл из списка, открывается ActivityPreviewFiles или ActivityPreviewImages в зависимости от типа файла. Интерфейс состоит из списка файлов и папок. Изначально список файлов строится из корневой папки, далее пользователь может переходить по папкам, чтоб открыть нужный ему документ или изображение. Предлагаемый интерфейс представлен на следующем рисунке.



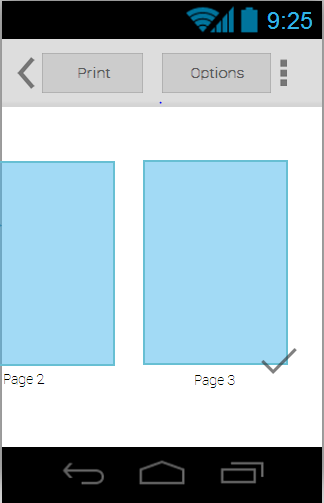
GalleryActivity – activity, которое отображает коллекцию изображения пользователя. Пользователь может выбрать одно или несколько изображений, для последующей подготовки к печати. По нажатию на кнопку Preview, открывается ActivityPreviewImages. Интерфейс представляет из себя плитку с эскизами изображений, каждое изображение можно отметить галочкой, тем самым выбрав его для распечатки. Предлагаемый интерфейс представлен на следующем рисунке.



WebBrowserActivity – представляет собой встроенный браузер, позволяющий отображать интернет-страницы. По нажатию на кнопку Preview, открывается ActivityPreviewWebPages для подготовки страниц к печати. Интерфейс представляет собой пространство для отображения страниц, адресную строку для ввода ссылок, кнопку Preview. Предлагаемый интерфейс представлен на следующем рисунке.



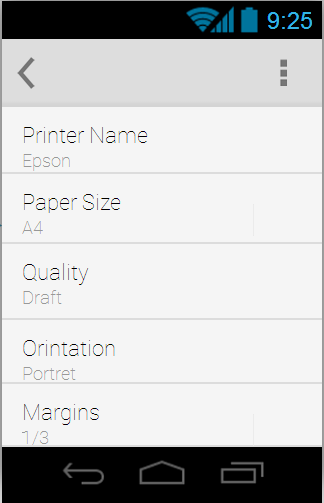
ActivityPreviewFiles, ActivityPreviewImages, ActivityPreviewWebPages по своей сути представляют один и тот же функционал, а именно просмотр страниц в том виде, в котором они распечатаются. Эта возможность представляет пользователю быть уверенным в том, что его материал распечатается так, как он желает. Для того чтобы просмотреть весь диапозон страниц нужно их прокрутить. Если пользователю нужно распечатать только несколько страниц из большого документа, он может сделать это, отметив необходимые страницы. Интерфейс представляет из себя эскизы распечатываемых страниц, кнопка Options, открывающая activity параметров принтера и опций печати, кнопка Print, которая показывает диалоговое окно, в котором пользователь может выбрать количество копий и диапазон распечатываемых страниц. Во время обработки и передачи данных на принтер, показывается диалоговое окно, уведомляющее о прогрессе печати. По окончанию передачи, появляется диалоговое окно об успешном задании печати. Предлагаемый интерфейс представлен на следующем рисунке.



Для изменение настроек принтера, таких как размер бумаги, качество печати, пользователь может перейти в PrinterOptionsActivity из activity предпросмотра страниц. В зависимости от типа контента, который будет распечатан, окно настроек представляет собой набор настроек печати, а именно:

* Для ActivityPreviewFiles: вмещать содержимое в область печати или в размер страницы.
* Для ActivityPreviewImages:
  + Размер изображения
  + Ориентация страницы
  + Размер отступов
  + Расположение на странице
  + Вмещать содержимое в область печати или в размер страницы
* Для ActivityPreviewWebPages:
  + Ориентация страницы
  + Размер отступов
  + Вмещать содержимое в область печати или в размер страницы

Предлагаемый интерфейс представлен на следующем рисунке.

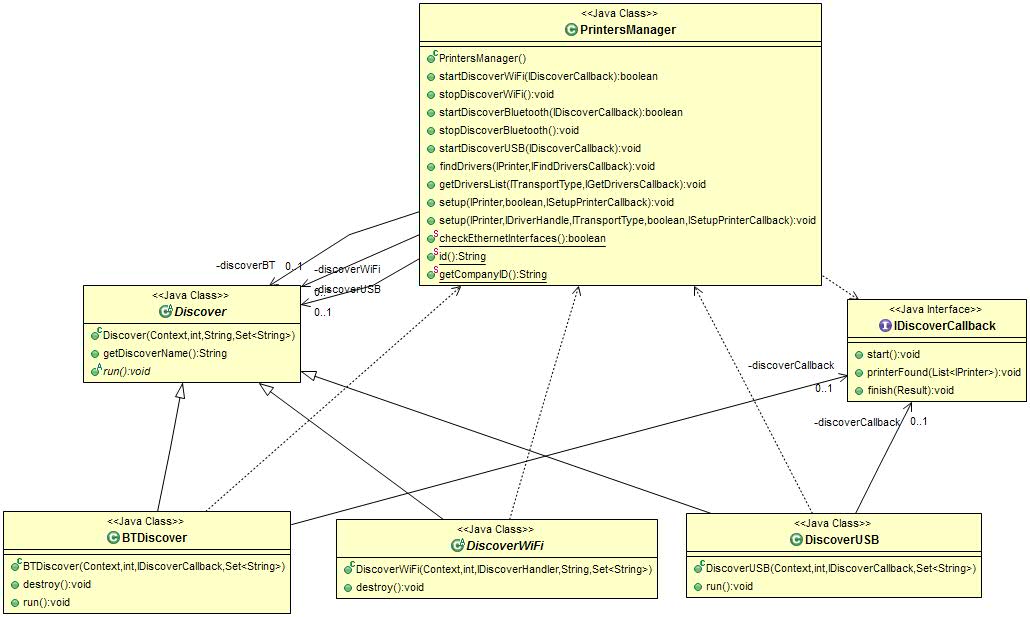


### 2.5.3 Проектирование классов

Проектирование классов, является не менее важным этапом, чем проектирование структуры приложения и проектирование интерфейса. На данном этапе важно спроектировать классы так, чтобы не нарушать главного правила мобильного приложения, а именно не загружать главный поток, то есть поток интерфейса. Нельзя выполнять длительные операции в главном потоке, иначе приложение будет остановлено или приложение не будет откликаться на действия пользователя, что совершенно не приемлемо. Для длительных операций существуют фоновые потоки, однако из этих потоков нельзя обновлять пользовательский интерфейс, поэтому необходимо использовать обработчик сообщений, который будет изменять интерфейс в соответствии с пришедшим сообщением.

Основным классом, который будет управлять процессом поиска, установкой и управление принтерами будет называться PrintersManager. Данный класс будет инициализировать начало поиска принтеров по таким протоколам как WiFi, Bluetooth, USB, будет обрабатывать сообщения о найденных принтерах, добавляя их динамически в список принтеров в интерфейсе приложения. Также этот класс инициализирует начало процесса поиска драйверов и их установки.

Класс Discover является предком таких классов, как DiscoverWiFi, DiscoverBluetooth, DiscoverUSB. Каждый из этих классов является потоком поиска принтеров, реализуемый в зависимости от типа связи. Тут же для найденного принтера определяются возможные транспорты – тип соединения, по которому будут передаваться данные. Так же здесь определяются поддерживаемые принтером возможности, например такие, как поддержка языков описания страниц. Для того, чтобы иметь возможность связи этих потоков с главным потоком необходимо реализовать классы интерфейса DiscoverCallback. Это позволит нам отправлять нужные сообщения в нужный момент времени, для дальнейшей их обработки в главном потоке. Такими сообщениями будут являться сообщения о начале поиска, о конце поиска, и сообщения, которые будут нести в себе найденные принтеры. На следующем рисунке отображена диаграмма этих классов.



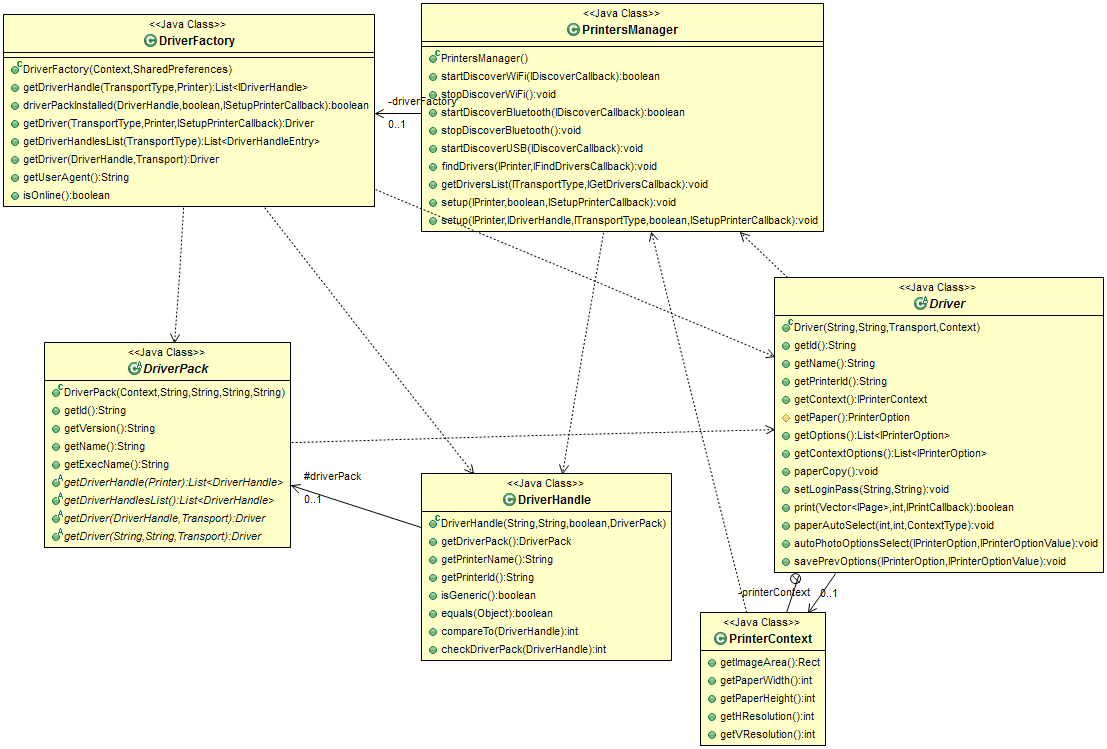
Следующая диаграмма классов отображает логику поиска драйверов и их установку.

После того, как пользователь нажимает в списке на найденный принетер, PrintersManager инициализирует начало процесса поиска и установки драйвера. Все начинается с класса DriverFactory.

DriverFactory – класс, который определяет, какие использовать обработчики драйверов по типу транспорта и по возможностям принтера. Определив обработчик, а именно DriverHandle, начинается поиск принтера в классе DriverPack.

DriverPack – класс, который содержит список принтеров. Если найденный принтер находится в данном списке, DriverPack назначает принтеру соответствующий драйвер.

Сам же драйвер отвечает за назначение принтеру таких настроек, как список размеров бумаг, список типов качества печати, список поддерживаемых лотков. Также при отправке на печать, драйвер обрабатывает данные в поддерживаемом принтером виде, и отправляет сообщения в главной поток о начале передачи, о прогрессе передачи данных, и об успешном или неуспешном окончании печати. Далее представлена диаграмма этих классов.



# Заключение.

В результате проделанной работы цель была достигнута и следующие задачи были выполнены:

* Изучение предметной области.
* Выбор используемых технологий.
* Изучение публичного API.
* Подготовка интерфейса мобильного приложения.
* Разработка технического задания на разработку мобильного приложения.

Таким образом, проделанная работа позволяет поставить новую цель – разработка мобильного приложения печати.

**Приложение №1. Техническое задание на разработку мобильного приложения.**

**Содержание**

1. Введение   
1.1. Наименование приложения   
1.2. Назначение и область применения   
2. Требования к приложению   
2.1. Требования к функциональным характеристикам   
2.2. Требования к надежности   
3. Условия эксплуатации   
3.1. Климатические условия эксплуатации   
3.2. Требования к составу и параметрам технических средств   
3.3. Требования к информационной и программной совместимости   
3.4. Специальные требования   
4. Требования к документации   
4.1. Предварительный состав документации   
5. Технико-экономические показатели   
5.1. Экономические преимущества разработки   
6. Стадии и этапы разработки   
6.1. Стадии разработки   
6.2. Этапы разработки   
6.3. Содержание работ по этапам   
7. Порядок контроля и приемки   
7.1. Виды испытаний   
7.2. Общие требования к приемке работы

**1. Введение**

**1.1. Наименование мобильного приложения**

Наименование приложения: "Мобильная печать"

**1.2. Назначение и область применения**

Мобильное приложение предназначено для распечатывания документов и изображений. Приложение позволяет:

1.2.1. Обнаруживать сетевые принтеры  
1.2.2. Устанавливать с принтерами соединение.  
1.2.3. Отправлять задание на печать.  
1.2.4. Предпросмотр задания на печать

**2. Требования к приложению**

**2.1. Требования к функциональным характеристикам**

Приложение должно обеспечивать возможность выполнения перечисленных ниже функций:

* + 1. Офисные документы, в том числе PDF
    2. Изображения различных форматов
    3. Веб-страницы
    4. Электронные письма
    5. СМС сообщения

**2.2. Требования к надежности**

**2.2.1 Требования к обеспечению надежного функционирования приложения**

Надежное (устойчивое) функционирование мобильного приложения должно быть обеспечено выполнением Заказчиком совокупности организационно-технических мероприятий, перечень которых приведен ниже:   
а) организацией исправной работы мобильного устройства;   
б) использованием подходящей версии операционной системы;   
в) регулярным выполнением требований ГОСТ 51188-98. Защита информации. Испытания программных средств на наличие компьютерных вирусов

**2.2.2. Время восстановления после отказа**

Время восстановления после отказа не должно превышать времени, требуемого на устранение неисправностей технических средств и повторного запуска приложения.

**2.2.3. Отказы из-за некорректных действий пользователей системы**

Отказы приложения вследствие некорректных действий пользователя при взаимодействии с приложением через мобильный интерфейс недопустимы.

**3. Условия эксплуатации**

**3.1. Климатические условия эксплуатации**

Климатические условия эксплуатации, при которых должны обеспечиваться заданные характеристики, должны удовлетворять требованиям, предъявляемым к мобильным устройствам в части условий их эксплуатации

**3.2. Требования к составу и параметрам технических средств**

**3.2.1. В состав технических средств должно входить мобильное, удовлетворяющее следующим свойствам:**

3.1.1.1. процессор, с тактовой частотой не менее 1000МГц;   
3.2.1.2. оперативную память объемом не менее 512МБ;   
3.2.1.3. свободную внутреннюю память объемом не менее 15МБ;   
3.2.1.4. операционную систему Android 2.1 или выше;

**3.3. Требования к информационной и программной совместимости**

**3.3.1. Требования к информационным структурам и методам решения**

Приложение должно поддерживать обработку данных в формате офисных документов

**3.3.2. Требования к исходным кодам и языкам программирования**

Используемый язык программирования – Java 7.0 в рамках Android SDK. Исходный код должен быть задокументирован согласно стандарту Javadoc.

**3.3.3. Требования к программным средствам, используемым приложением**

В качестве операционной системы должен использоваться Android версии 2.1 или выше. На устройстве не должны быть установлены приложения, блокирующие http трафик. Требований к дополнительному программному обеспечению нет.

**3.3.4. Требования к защите информации и программ**

Требования к защите информации и программ не предъявляются.

**3.4. Специальные требования**

Приложение должно обеспечивать одновременную работу пользователей посредством мобильного интерфейса. А так же приложение должно соответствовать подготовленному прототипу.

**4. Требования к документации**

**4.1. Предварительный состав документации**

Состав документации должен включать в себя:   
4.1.1. техническое задание;   
4.1.2. руководство пользователя.

**5. Технико-экономические показатели**

**5.1. Экономические преимущества разработки**

Ориентировочная экономическая эффективность не рассчитываются. Аналогия не проводится ввиду уникальности предъявляемых требований к разработке.

**6. Стадии и этапы разработки**

**6.1. Стадии разработки**

Разработка должна быть проведена в три стадии:   
1. разработка технического задания;   
2. разработка приложения;   
3. публикация в Google Play.

**6.2. Этапы разработки**

На стадии разработки технического задания должен быть выполнен этап разработки, согласования и утверждения настоящего технического задания.   
На стадии разработки должны быть выполнены перечисленные ниже этапы работ:

1. разработка приложения;   
2. разработка документации;   
3. тестирование приложения.

**6.3. Содержание работ по этапам**

На этапе разработки технического задания должны быть выполнены перечисленные ниже работы:   
1. постановка задачи;   
2. определение и уточнение требований к техническим средствам;   
3. определение требований к приложению;  
4. определение стадий, этапов и сроков разработки приложения и документации на неё;   
5. согласование и утверждение технического задания.   
На этапе разработки приложения должна быть выполнена работа по программированию (кодированию) и отладке приложения.   
На этапе разработки документации должна быть выполнена разработка документов в соответствии с требованиями к составу документации.   
На этапе тестирования приложения должны быть выполнены перечисленные ниже виды работ:   
1. разработка, согласование и утверждение и методики тестирование;   
2. корректировка приложения и документации по результатам тестирования.   
На этапе публикации должна быть выполнена работа по подготовке и передаче приложения в систему Google Play.

**7. Порядок контроля и приемки**

**7.1. Виды испытаний**

Приемо-сдаточные испытания должны проводиться на объекте Заказчика в оговоренные сроки.   
Приемо-сдаточные испытания приложения должны проводиться согласно разработанной Исполнителем и согласованной Заказчиком Программы и методик тестирования.   
Ход проведения приемо-сдаточных испытаний Заказчик и Исполнитель документируют в Протоколе проведения испытаний.

**7.2. Общие требования к приемке работы**

На основании Протокола проведения испытаний Исполнитель совместно с Заказчиком подписывает Акт приемки-сдачи приложения в эксплуатацию.