1-2. Изначально Google рассчитывала давать версиям Android имена известных роботов, но отказалась из-за проблем с авторскими правами. Каждая версия системы, начиная с версии 1.5, получает собственное кодовое имя на тему сладостей. Кодовые имена присваиваются в алфавитном порядке латинского алфавита. Начиная с версии Android 10 кодовое название ОС получает только во время разработки. Начиная с версии 3.1 обновления будут выходить раз в 6 месяцев.

Также была версия Android 3.0 с кодовым названием Honeycomb, которая была нацелена на использование в устройствах с большим экраном, например планшеты. Данная версия довольно быстро перестала быть актуальной ввиду отсутствия в то время устройств с большим экраном, потому в 2012 году её поддержка закончилась.

Последняя версия ОС Android на данный момент — Android 13, выпущенная 15 августа 2022 года.

Минимальная поддержка — Android KitKat.

Официальное приложение YouTube поддерживается на Android Lollipop и выше.

Сервисы Google Play поддерживаются приблизительно 10 лет с момента выхода Android.

Самая популярная версия — Android 11 (Red Velvet Cake) (27%).

27 сентября 2021 года Google закрыл доступ к аккаунтам пользователей на устаревших версиях ниже Android Honeycomb

3. Файл манифеста AndroidManifest.xml предоставляет основную информацию о программе системе. Каждое приложение должно иметь свой файл AndroidManifest.xml. Редактировать файл манифеста можно вручную, изменяя XML-код или через визуальный редактор Manifest Editor (Редактор файла манифеста), который позволяет осуществлять визуальное и текстовое редактирование файла манифеста приложения.

Назначение файла

-объявляет имя Java-пакета приложения, который служит уникальным идентификатором;

-описывает компоненты приложения — деятельности, службы, приемники широковещательных намерений и контент-провайдеры, что позволяет вызывать классы, которые реализуют каждый из компонентов, и объявляет их намерения;

-содержит список необходимых разрешений для обращения к защищенным частям API и взаимодействия с другими приложениями;

-объявляет разрешения, которые сторонние приложения обязаны иметь для взаимодействия с компонентами данного приложения;

-объявляет минимальный уровень API Android, необходимый для работы приложения;

-перечисляет связанные библиотеки;

4. Базовые компоненты андроид приложения: Activity, Service, BroadcastReceiver, ContentProvider. Каждый из базовых компонент объявляется в андроид манифесте и может являться точкой входа в приложение.

Activity представляет UI и функциональность, видимые пользователю.

Service используется для исполнения долгих операций, которые не требуют взаимодействия с пользователем.

BroadcastReceiver принимает и обрабатывает броадкаст ивенты, отправленные внутри приложения или из других приложений.

ContentProvider используется для обмена данными с другими приложениями.

5. Считайте, что фрагмент - это тот же компонент как Button, TextView или LinearLayout с дополнительными возможностями. Фрагмент, как и кнопку, нужно поместить на экран активности. Но фрагмент является модульным компонентом и один и тот же фрагмент можно встроить в две разные активности. С кнопкой такой номер не пройдёт. Для каждой активности вы должны создать свою отдельную кнопку, даже если их нельзя будет отличить друг от друга.

Фрагмент также немного похож на активность. Но фрагменты - это не замена активности, они не существуют сами по себе, а только в составе активностей. Поэтому в манифесте прописывать их не нужно. Но в отличие от стандартной кнопки, для каждого фрагмента вам придётся создавать отдельный класс, как для активности.

В составе активности есть специальный менеджер фрагментов, который может контролировать все классы фрагментов и управлять ими. Фрагмент может иметь свою разметку, а может обойтись без неё. Также у фрагмента есть свой жизненный цикл, во многом совпадающий с жизненным циклом активности. Пожалуй, это единственное сходство с активностью.

6.

onAttach(Activity)

-Вызывается, когда фрагмент связывается с активностью. С этого момента мы можем получить ссылку на активность через метод getActivity()

onCreate()

-В этом методе можно сделать работу, не связанную с интерфейсом. Например, подготовить адаптер.

onCreateView(LayoutInflater, ViewGroup, Bundle)

-Вызывается для создания компонентов внутри фрагмента

onViewCreated()

-Вызывается сразу после onCreateView()

onActivityCreated(Bundle)

-Вызывается, когда отработает метод активности onCreate(), а значит фрагмент может обратиться к компонентам активности

onStart()

-Вызывается, когда фрагмент видим для пользователя и сразу же срабатывает onStart() активности

onDestroyView()

onResume()

-Вызываем после onResume() активности

-Вызывается, когда набор компонентов удаляется из фрагмента

onDetach()

-Вызывается, когда фрагмент отвязывается от активности

7.

View

-View объекты являются основными строительными элементами элементов пользовательского интерфейса (UI) в Android.

-View - это простой прямоугольник, который реагирует на действия пользователя.

-Примерами являются EditText, Button, CheckBox и т.д.

-View относится к классу android.view.View, который является базовым классом всех классов пользовательского интерфейса.

ViewGroup

-ViewGroup - невидимый контейнер. Он содержит View и ViewGroup

-Например, LinearLayout - это ViewGroup, который содержит кнопку (вид) и другие макеты.

-ViewGroup - базовый класс для макетов.

-ViewGroup - это специальное представление, которое может содержать другие представления (называемые ViewGroup).

8. Эмуляторы, тестирование на самих устройствах, нахождение багов.

9. DialogFragment

-Раньше диалоговое окно вызывали напрямую, сейчас рекомендуется использовать в составе DialogFragment.

-Использование фрагментов для диалоговых окон в силу своей архитектуры является удобным вариантом в приложениях, который лучше справляется с поворотами устройства, нажатием кнопки "Назад", лучше подходит под разные размеры экранов и т.д.

-Для создания диалога следует наследоваться от класса DialogFragment.

-Самый распространённый вариант диалогового окна - это AlertDialog.

В создаваемых диалоговых окнах можно задавать следующие элементы:

-заголовок

-текстовое сообщение

-кнопки: от одной до трёх

-список

-флажки

-переключатели

10. Ресурсы находятся в папке res. Хуй знает что ещё добавить, там уже по ситуации.

11. Для работы с файлами абстрактный класс android.content.Context определяет ряд методов:

-boolean deleteFile (String name): удаляет определенный файл

-String[] fileList (): получает все файлы, которые содержатся в подкаталоге /files в каталоге приложения

-File getCacheDir(): получает ссылку на подкаталог cache в каталоге приложения

-File getDir(String dirName, int mode): получает ссылку на подкаталог в каталоге приложения, если такого подкаталога нет, то он создается

-File getExternalCacheDir(): получает ссылку на папку /cache внешней файловой системы устройства

-File getExternalFilesDir(String type): получает ссылку на каталог /files внешней файловой системы устройства

-File getFileStreamPath(String filename): возвращает абсолютный путь к файлу в файловой системе

-FileInputStream openFileInput(String filename): открывает файл для чтения

-FileOutputStream openFileOutput (String name, int mode): открывает файл для записи

12. Широковещательные сообщения делают приложение более открытым; передавая события, использующие сообщения, вы открываете компоненты своего приложения для сторонних приложений, и сторонние разработчики реагируют на события без необходимости изменять ваше оригинальное приложение. В своём приложении вы можете прослушивать широковещательные сообщения других приложений, заменить или улучшить функциональность собственного (или стороннего) приложения или реагировать на системные изменения и события приложений.

13. Когда мы запускаем приложение на Android, система создает поток, который называется основным потоком приложения или UI-поток. Этот поток обрабатывает все изменения и события пользовательского интерфейса. Однако для вспомогательных операций, таких как отправка или загрузка файла, продолжительные вычисления и т.д., мы можем создавать дополнительные потоки.

Для создания новых потоков нам доcтупен стандартный функционал класса Thread из базовой библиотеки Java из пакета java.util.concurrent, которые особой трудности не представляют. Тем не менее трудности могут возникнуть при обновлении визуального интерфейса из потока.

14. Службы (Сервисы) в Android работают как фоновые процессы и представлены классом android.app.Service. Они не имеют пользовательского интерфейса и нужны в тех случаях, когда не требуется вмешательства пользователя. Сервисы работают в фоновом режиме, выполняя сетевые запросы к веб-серверу, обрабатывая информацию, запуская уведомления и т.д. Служба может быть запущена и будет продолжать работать до тех пор, пока кто-нибудь не остановит её или пока она не остановит себя сама. Сервисы предназначены для длительного существования, в отличие от активностей. Они могут работать, постоянно перезапускаясь, выполняя постоянные задачи или выполняя задачи, требующие много времени.

15. Room имеет три основных компонента: Entity, Dao и Database. Рассмотрим их на небольшом примере, в котором будем создавать базу данных для хранения данных по сотрудникам (англ. - employee).

При работе с Room нам необходимо будет писать SQL запросы.

16. Для получения данных с определенного интернет-ресурса мы можем использовать классы HttpUrlConnection (для протокола HTTP) и HttpsUrlConnection (для протокола HTTPS) из стандартной библиотеки Java.

Вначале создается элемент HttpsURLConnection

После подключение происходит считывание со входного потока

Используя входной поток, мы можем считать его в строку.

Этот метод getContent() затем будет вызываться в обработчике нажатия кнопки.

17. Итак, что нужно сделать PM`y в ходе публикации:

Создать аккаунт в Google Play Developer Console для заказчика, если у заказчика такового нет, или предложить произвести публикацию с нашего аккаунта.

Оформить privacy policy.

Подготовить маркетинговые материалы (иконка, скриншоты, APK, баннер, текст, проморолик).

Обеспечить сборку наличием сертификата цифровой подписи.

Настроить оплату за пользование приложения.

Отправить сборку в Google Play.