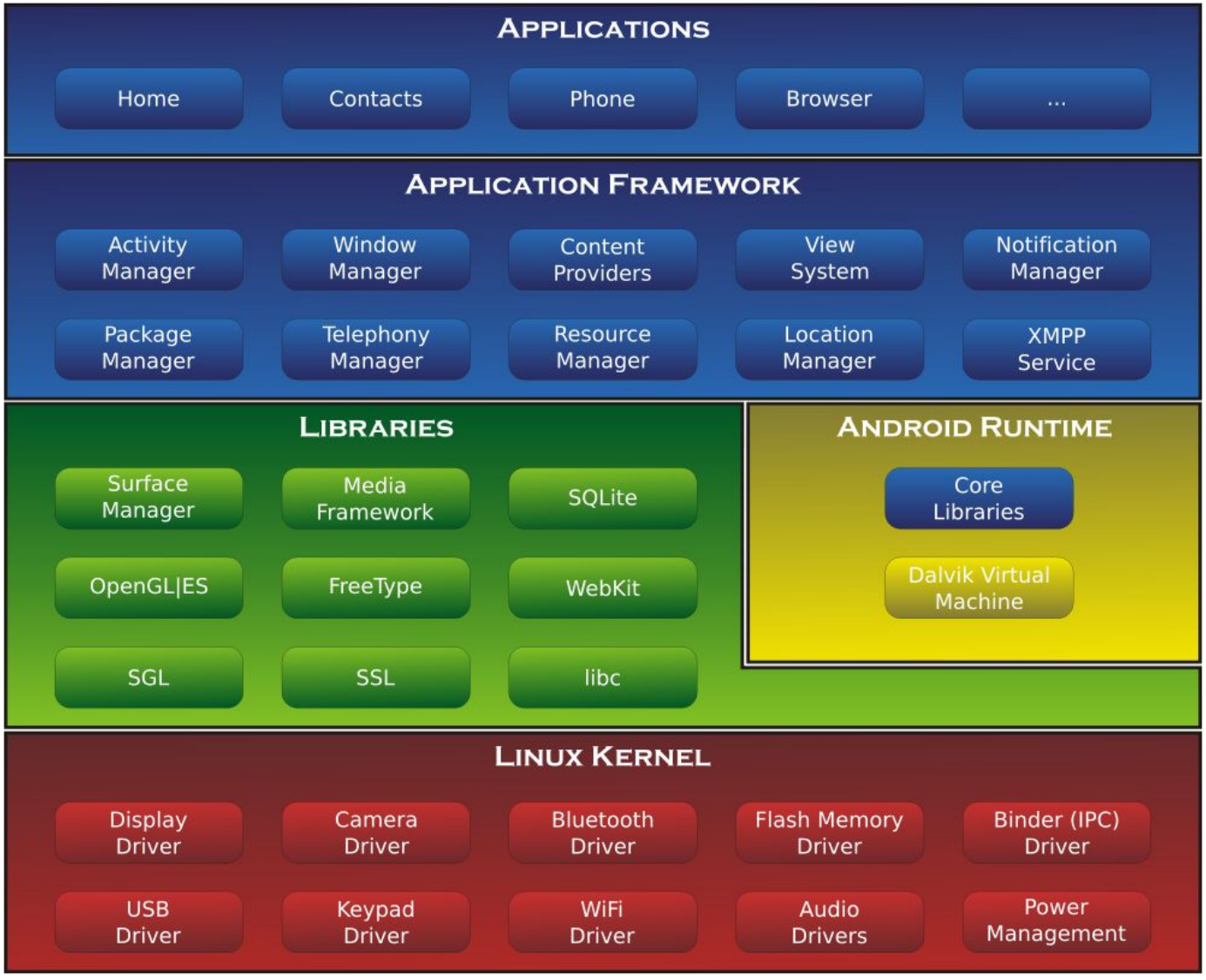
Архитектура Андроид-приложения

Архитектура

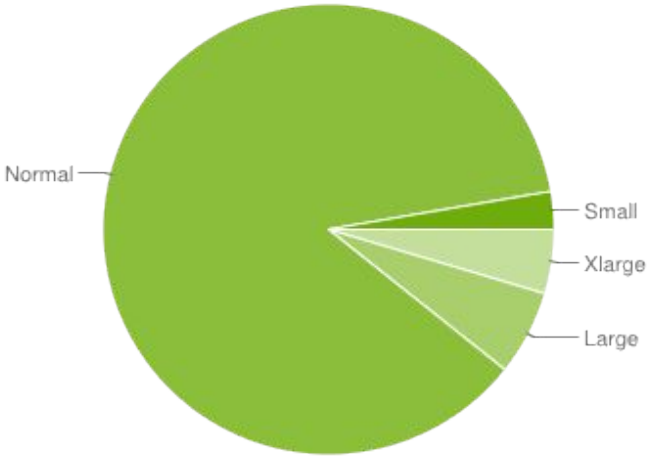


Изолированность приложения Android

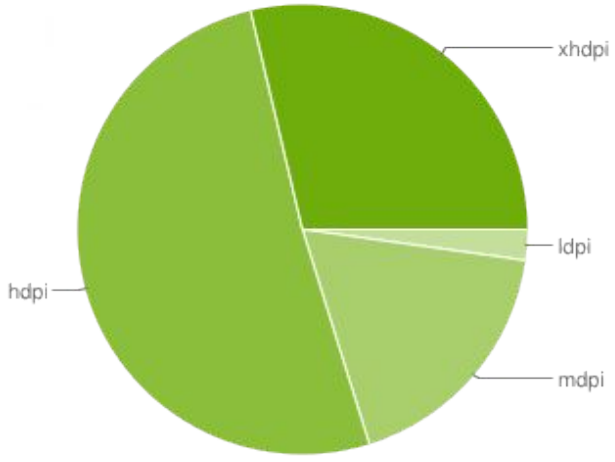
1. Каждое приложение имеет свой Linux user Id
2. Для каждого приложения запускается своя Dalvik VM
3. Любой компонент приложения, будет запускаться в своем Linux процессе

Размеры экранов

Размер



Плотность



	ldpi	mdpi	hdpi	xhdpi
small	1.7%		1.0%	
normal	0.4%	11%	50.1%	25.1%
large	0.1%	2.4%		3.6%
xlarge		4.6%		

Как работать?

ldpi: 1 dip = 0,75px

mdpi: 1 dip = 1px

hdpi: 1 dip = 1,5px

xhdpi: 1 dip = 2px

px - пиксели

in, mm, pt - дюймы, миллиметры, точки (1/72 дюйма)

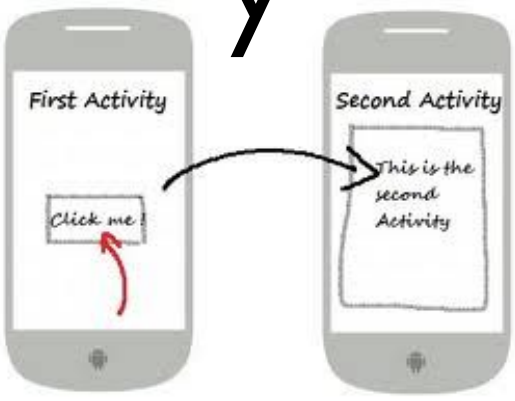
dip (device independent pixel) - абстрактные пиксели, зависящие от плотности экрана. Для экрана плотностью 160 dpi один dp равняется одному пикселю (px).

Строительные блоки приложения

- Activity
- Service
- ContentProvider – ПОСТАВЩИК СОДЕРЖИМОГО
- BroadcastReceiver – ПОЛУЧЕНИЕ ВНЕШНИХ СОБЫТИЙ И РЕАКЦИЯ НА НИХ
- Intent – НАМЕРЕНИЯ

Рабочие лошадки

Activity



малосвязанные

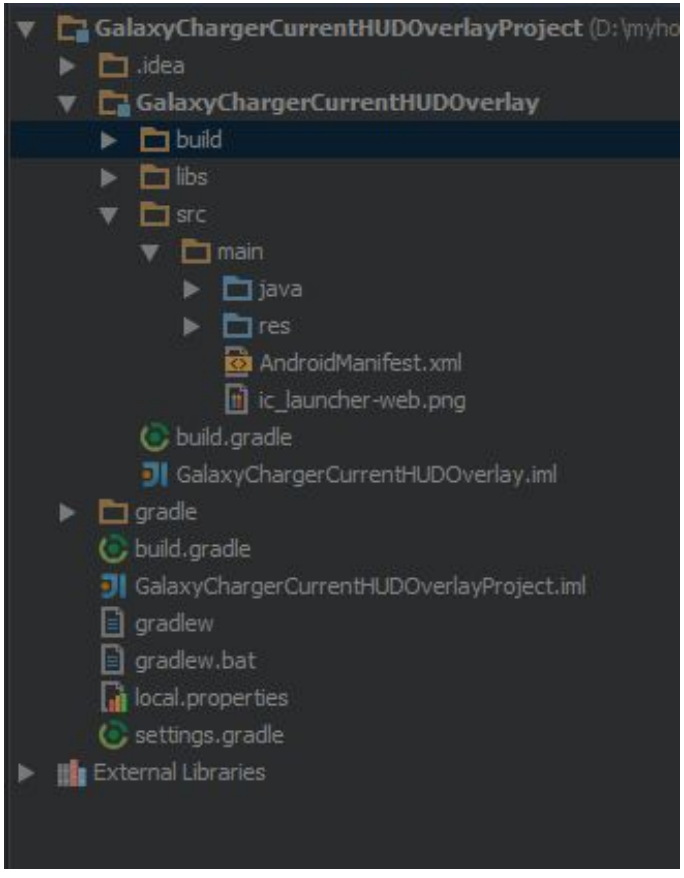
не подходят для длительных операций

Service

работают в фоне (backgroud)

подходят для длительных операций

Структура проекта (Android Studio)



- Вместо bin — build, там всё сгенерированное
- Папка с исходными файлами теперь содержит и ресурсы
- Файлы настройки Gradle
- Многие настройки вынесены в Gradle
- Библиотеки можно как подкладывать руками, так и подгружать автоматически

AndroidManifest.xml

- определяет имя Java-пакета приложения = уникальный идентификатор для приложения.
- описывает компоненты приложения: Activity, Service, BroadcastReceiver, ContentProvider. Определяет имена классов, реализующие каждый из компонентов и оглашает их возможности (например, какие Intent-сообщения они могут обрабатывать). Эти объявления позволяют системе Android знать, какие компоненты и при каких условиях могут быть запущены.
- объявляет разрешения, которые приложение должно иметь для доступа к защищённым частям API и взаимодействия с другими приложениями.
- объявляет минимальный уровень Android API, который требует приложение.
- и другие...

Ресурсы

Типы ресурсов

- drawable
  - .png, .jpg, .gif
  - .9.png
  - .xml
    - shape
    - selector
- layout
- anim
- values
  - strings
  - dimensions
  - colors
  - arrays
- xml

Селекторы ресурсов

- Ориентация
  - land, port
- Плотность точек
  - ldpi, mdpi, hdpi, xhdpi, nodpi
- Размер экрана
  - small, normal, large, xlarge
- Версия Android
  - v3, . . . , v15
- Язык
  - en, fr, ru, . . .

Ресурсы R.java

автоматически генерируется средой разработки

содержит ссылки на все ресурсы проекта

```
package my.favorite;

public final class R {
    public static final class attr {
    }
    public static final class drawable {
        public static final int ic_launcher=0x7f020000;
    }
    public static final class id {
        public static final int message_edit_text=0x7f050000;
        public static final int message_show_text_view=0x7f050003;
        public static final int send_button=0x7f050002;
        public static final int show_button=0x7f050001;
    }
    public static final class layout {
        public static final int message_input_layout=0x7f030000;
        public static final int message_show_layout=0x7f030001;
    }
    public static final class string {
        public static final int app_name=0x7f040003;
        public static final int message_input_message_hint=0x7f040000;
        public static final int message_input_send_button=0x7f040002;
        public static final int message_input_show_button=0x7f040001;
    }
}
```

Строковые ресурсы

- res/values/strings.xml

```
<string name="simple_string">It is a simple string</string>
<string name="args_string">It is a string with integer arg: %1$d</string>
```

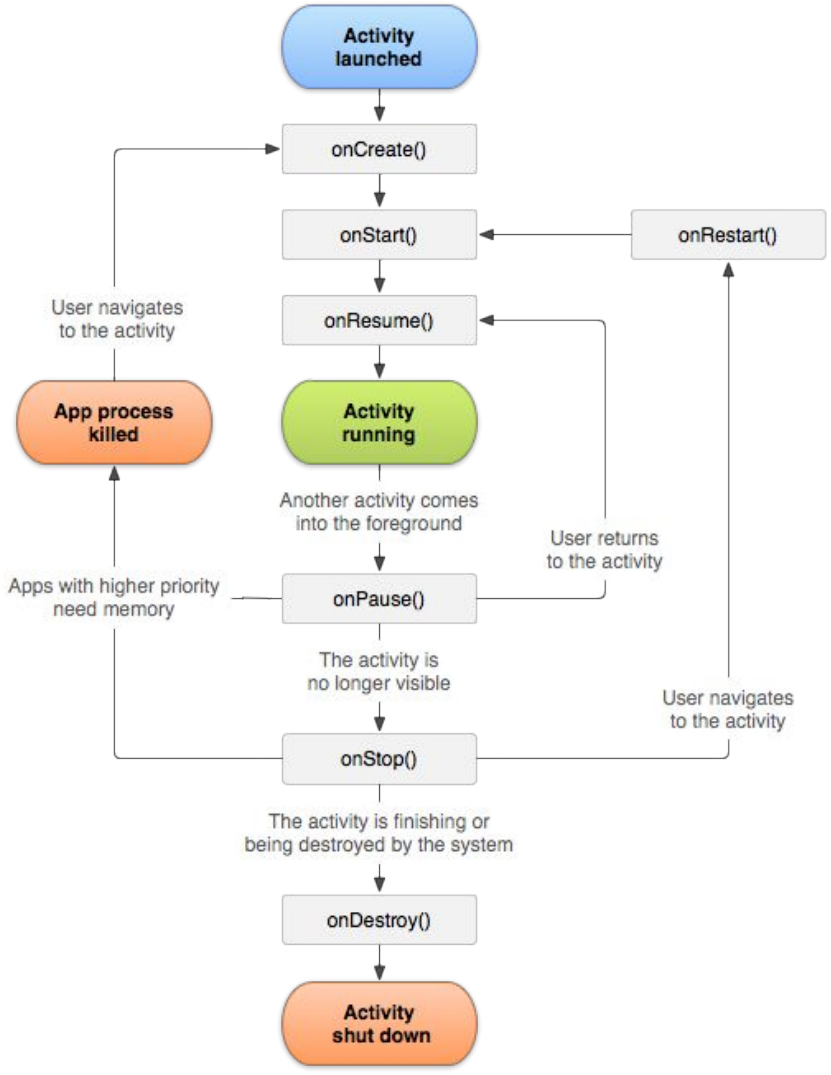
- res/values-ru/strings.xml

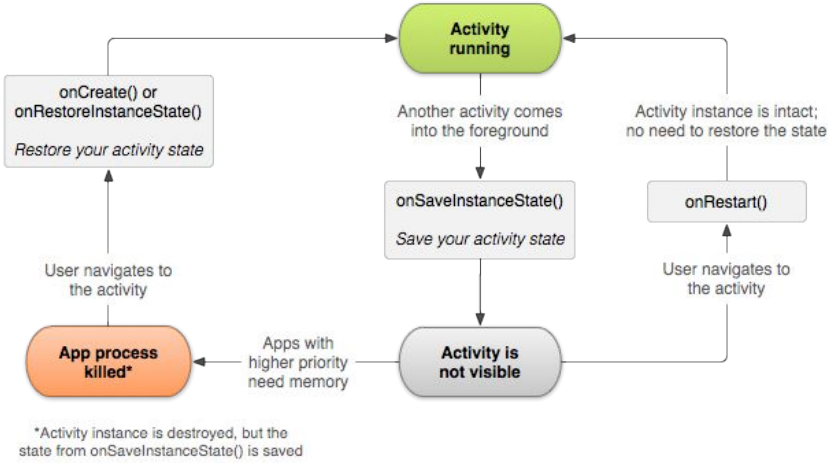
```
<string name="simple_string">Это обычная строка</string>
<string name="args_string">Это строка с целочисленным аргументом:
%1$d</string>
```

- src/my/favorite/TestActivity.java

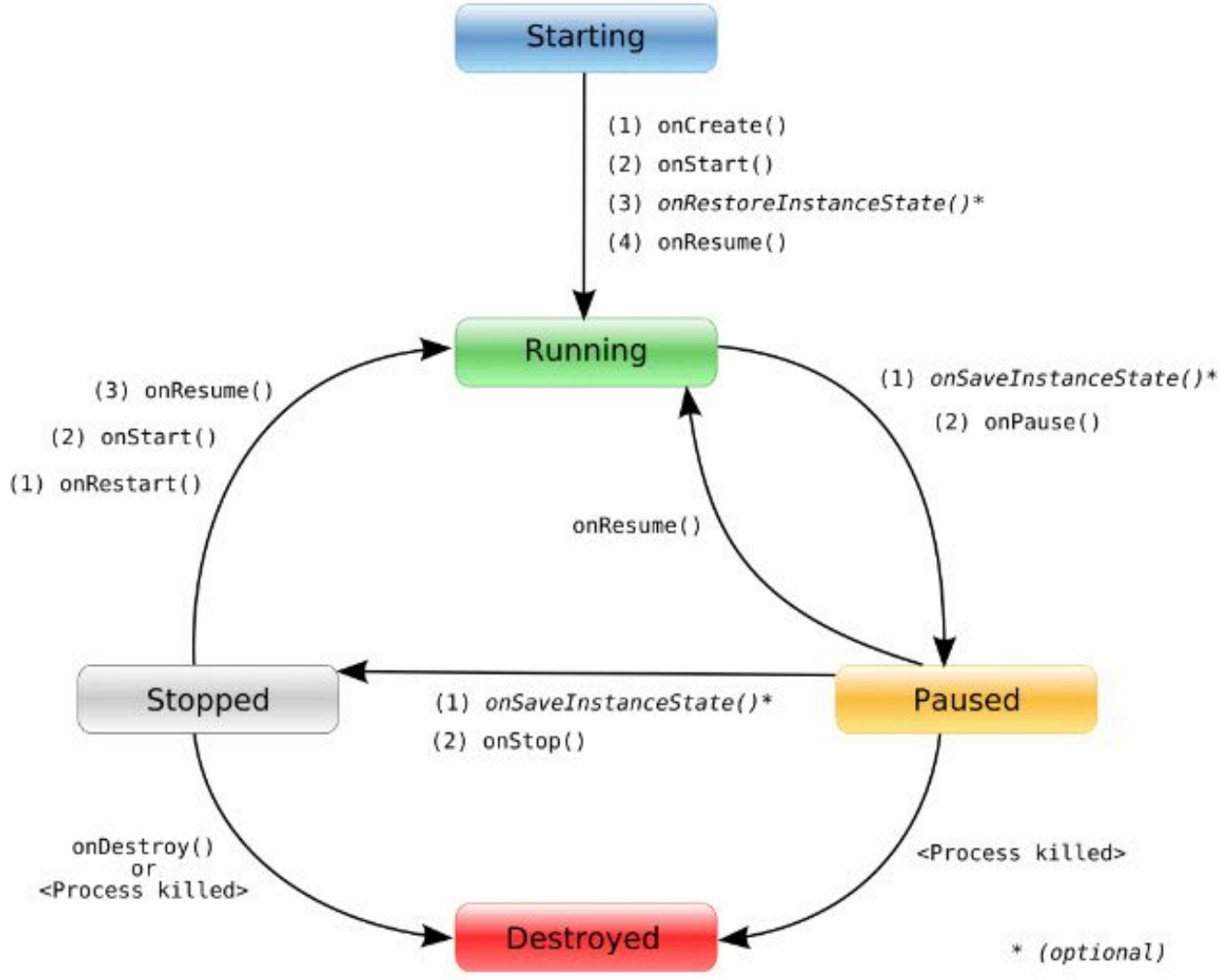
```
TextView simpleText =
(TextView)findViewById(R.id.text_view);
simpleText.setText(getString(R.string.simple_string));

TextView argsText =
(TextView)findViewById(R.id.text_view);
simpleText.setText(getString(R.string.args_string, 999));
```

Жизненный цикл Activity





Жизненный цикл Activity



Жизненный цикл Activity

```
public class TestActivity extends Activity {
    private static final String TAG = TestActivity.class.getSimpleName();

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        Log.d(TAG,"onCreate");  setContentView(R.layout.main);
    }

    @Override
    protected void onRestoreInstanceState(Bundle savedInstanceState) {
        super.onRestoreInstanceState(savedInstanceState);
        Log.d(TAG,"onRestoreInstanceState");
    }

    @Override
    protected void onSaveInstanceState(Bundle outState) {
        super.onSaveInstanceState(outState);
        Log.d(TAG,"onSaveInstanceState");
    }

    @Override
    protected void onDestroy() {
        Log.d(TAG,"onDestroy");
        super.onDestroy();
    }
    //...
}
```

Дополнительные ссылки

http://developer.android.com

http://android-developers.blogspot.com

http://startandroid.ru

http://developer.alexanderklimov.ru/android/theory