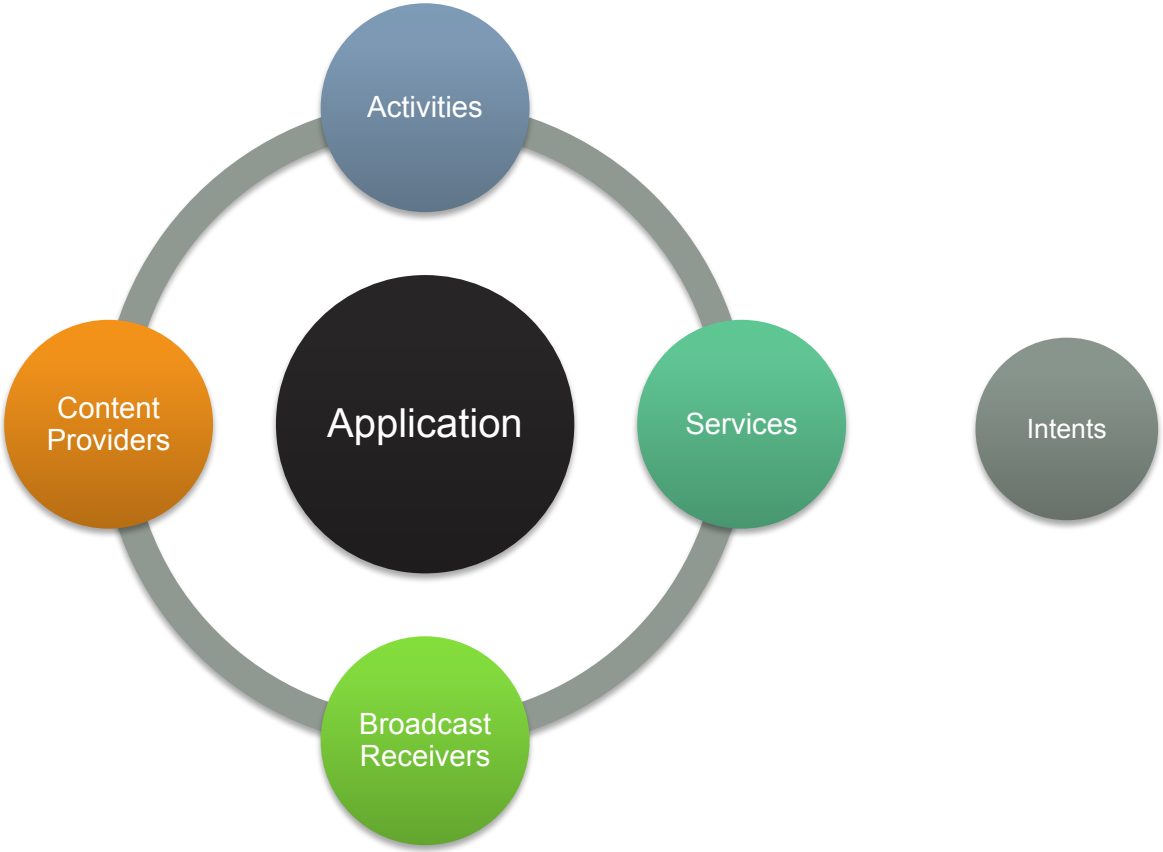
Android

Основные компоненты приложения.
Жизненный цикл приложения.

Компоненты приложения



Activity - одно окно приложения

- Может занимать весь экран или его часть.
- Может быть запущена из других компонент приложения или из другого приложения.
- Activity может возвращать результат

Service – компонент для выполнения длительных фоновых задач

- Не содержит графического интерфейса.
- Может выполняться в том же процессе, что и само приложение, либо в отдельном.
- Повышает значимость процесса с точки зрения Android.

Broadcast Receiver – приемник широковещательных сообщений

- Получает сообщения от Android или других приложений.
- Примеры широковещательных сообщений:
  - BOOT
  - SCREEN_OFF/ON
  - CONNECTIVITY_ACTION
- Должен обрабатывать сообщения быстро, длительные задачи можно делегировать сервису.

Content Provider – компонент для доступа к хранилищу данных

- Используется для доступа к данным, хранимым Android, или другими приложениями.
- Приложение может давать доступ к своим данным для других приложений, реализуя Content Provider.
- Представляет данные в виде таблиц, реализует методы query, insert, update, delete.

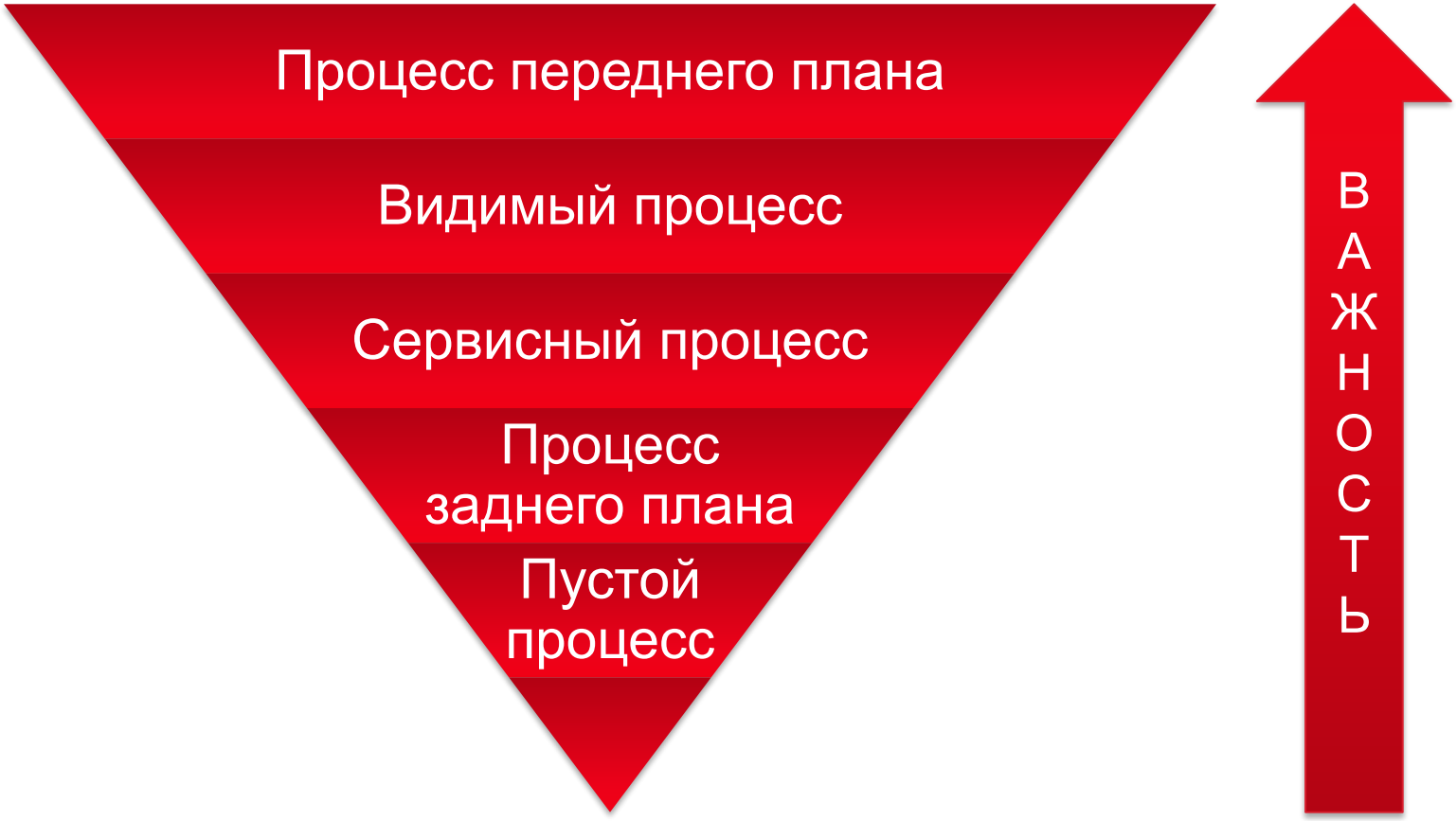
Intent – сущность для описания операции, которую требуется выполнить

Используется для:

- Запуска Activity
- Запуска сервиса
- Отправки широковещательных сообщений
- Выполнения стандартных, предопределенных операций

Жизненный цикл процесса

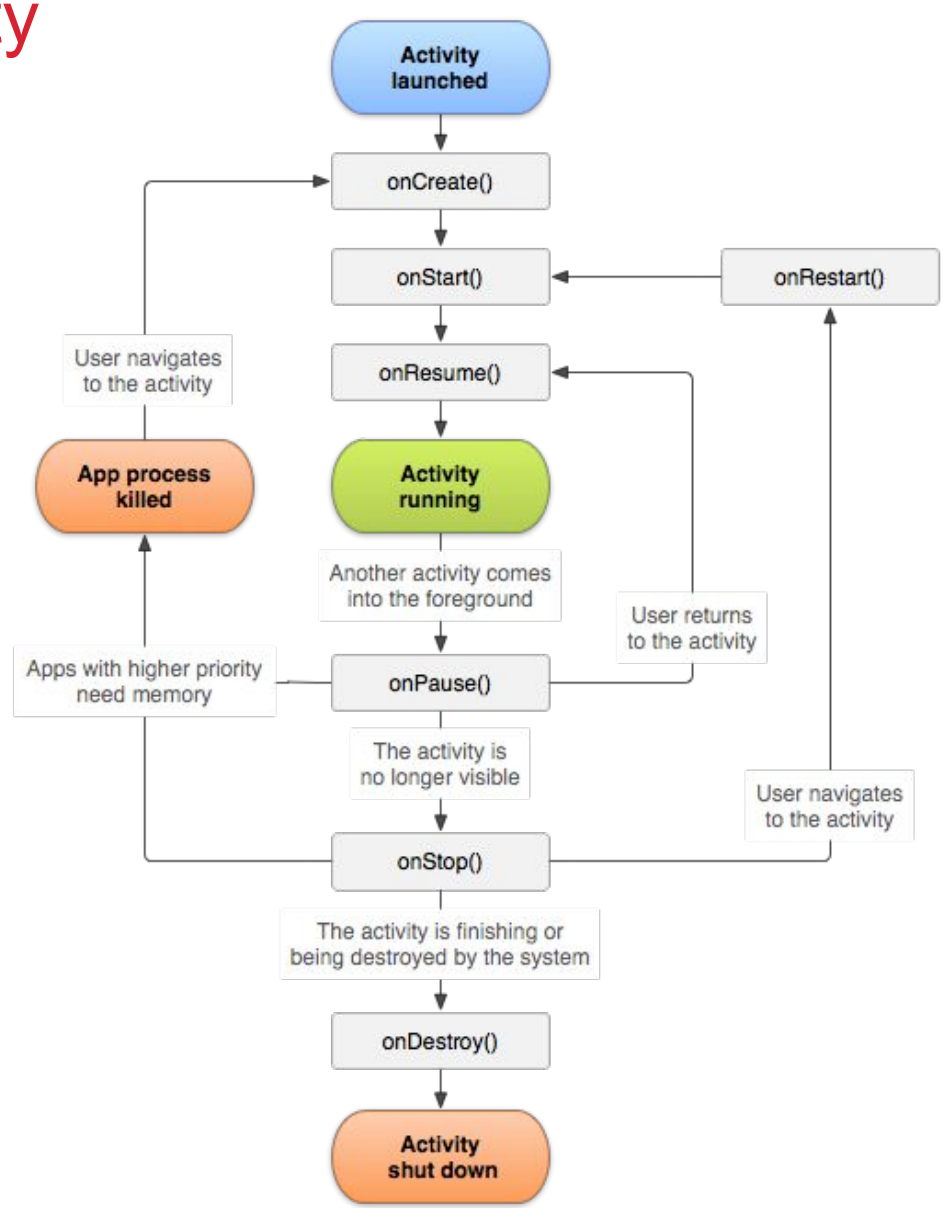
Важность процессов



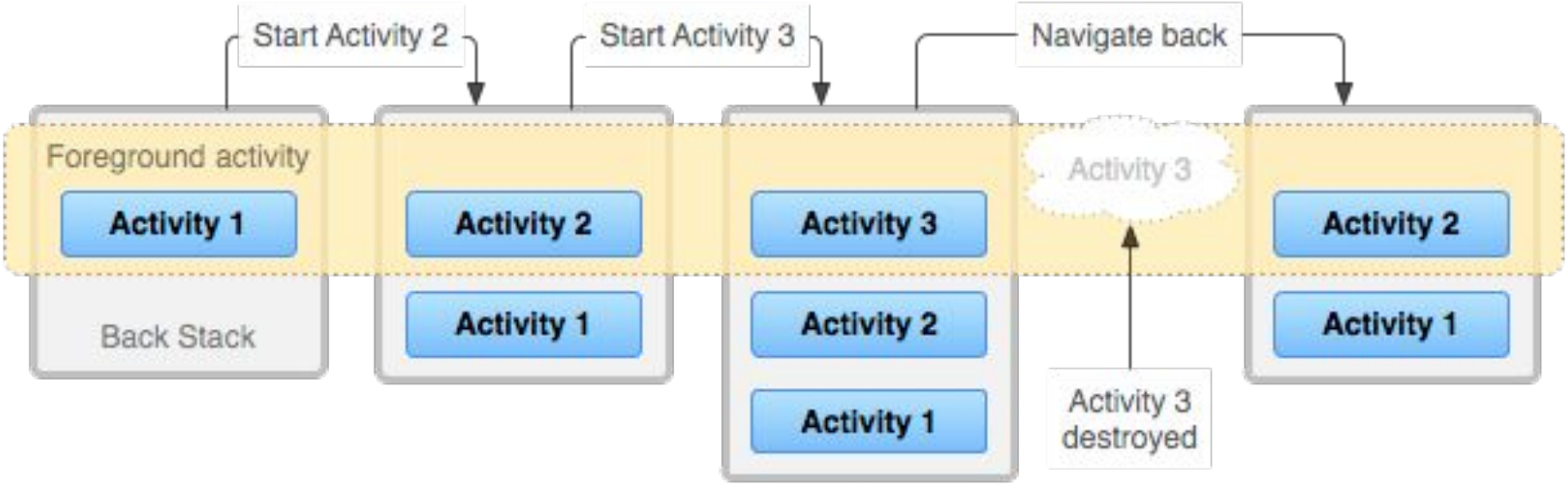
Application

```
public class MyApplication extends Application {
    private String mLocale = null;

    @Override
    public void onCreate() {
        super.onCreate();

        mLocale = getResources().getConfiguration().locale.toString();
        NativeCore.init(mLocale);
    }

    @Override
    public void onConfigurationChanged(final Configuration newConfig) {
        super.onConfigurationChanged(newConfig);

        final String newLocale = newConfig.locale.toString();
        if (!newLocale.equals(mLocale)) {
            mLocale = newLocale;
            NativeCore.setLocale(newLocale);
        }
    }
}
```

Жизненный цикл Activity



Activity back stack



Launch Modes

- **standart(default mode) –** при каждом запуске Activity создается новый экземпляр Activity и помещается на вершину back stack.

- **singleTop** – если в момент запуска экземпляр Activity уже находится на вершине стека, то новый экземпляр не создается, вместо этого вызывается метод onNewIntent() у существующего экземпляра.

- **singleTask** – Activity запускается в своем отдельном Task. Если экземпляр Activity уже существует, то у него вызывается метод onNewIntent(), а все Activity, лежащие в back stack поверх этого экземпляра – уничтожаются

- **singleInstance** – то же, что и **singleTask**, но Activity является в своем таске единственной.

AndroidManifest.xml

```
<activity
        android:label="@string/app_name"
        android:name="com.softmo.smssafe2.views.SmsMainActivity"
        android:windowSoftInputMode="stateAlwaysHidden"
        android:launchMode="singleTask">

    <intent-filter>
        <action android:name="android.intent.action.MAIN"/>
        <category android:name="android.intent.category.LAUNCHER"/>
    </intent-filter>
</activity>
```

Пересоздание Activity

Android пересоздает Activity:

- При изменении конфигурации устройства, например когда
  - изменяется ориентация экрана
  - пользователь меняет язык системы в настройках Android

  и т.п.

- При возврате пользователя к процессу, который был убит

Android для освобождения ресурсов.

Параметр configChanges

android:configChanges="orientation|screenSize"

```
@Override
public void onConfigurationChanged(Configuration newConfig) {
    super.onConfigurationChanged(newConfig);

    // Checks the orientation of the screen
    if (newConfig.orientation == Configuration.ORIENTATION_LANDSCAPE) {
        Toast.makeText(this, "landscape", Toast.LENGTH_SHORT).show();
    } else if (newConfig.orientation == Configuration.ORIENTATION_PORTRAIT){
        Toast.makeText(this, "portrait", Toast.LENGTH_SHORT).show();
    }
}
```

Использование параметра configChanges не избавляет от необходимости корректно обрабатывать пересоздание Activity!
Оно оправдано только в редких, исключительных случаях!

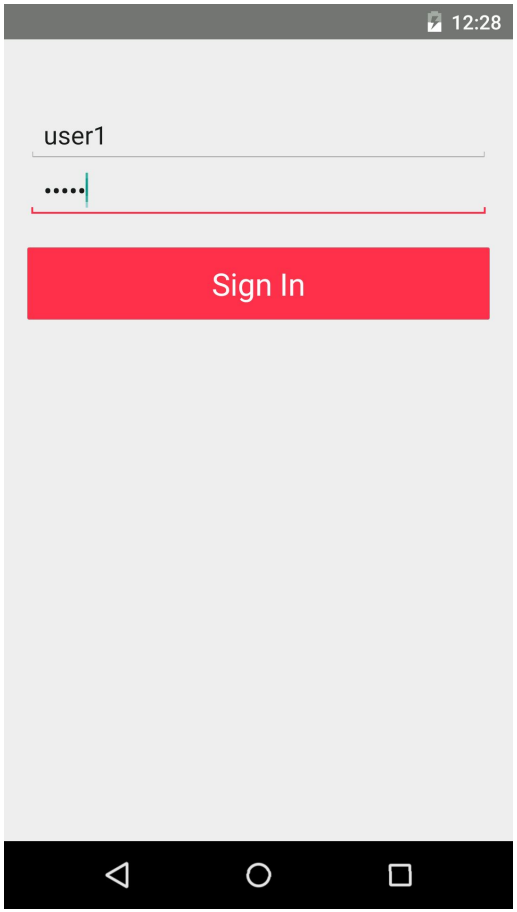
Сохранение состояния при пересоздании Activity

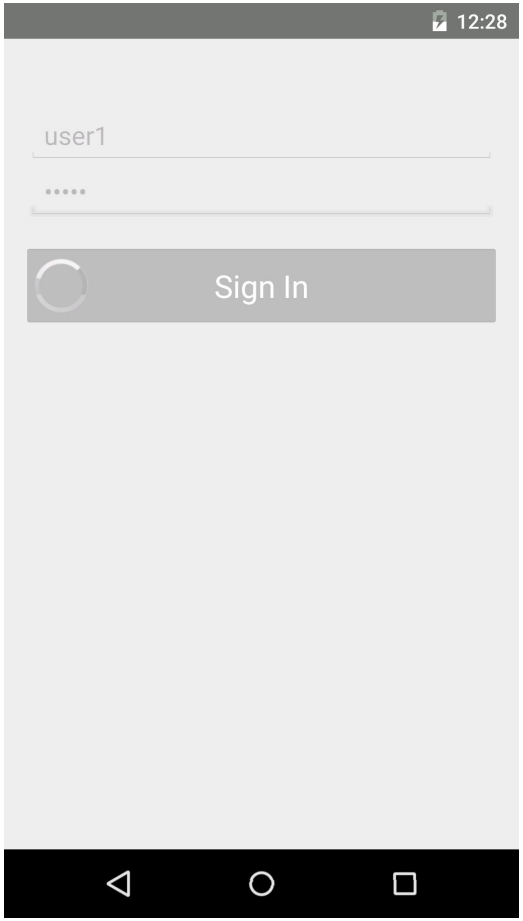
```
public class MyActivity extends Activity {
  public static final String KEY_VISIBLE = "com.parallels.sample.key.VISIBLE";
  private View mView;

  @Override
  public void onSaveInstanceState(Bundle outState) {
    super.onSaveInstanceState(outState);
    outState.putBoolean(KEY_VISIBLE, mView.getVisibility() == VISIBLE);
  }

  @Override
  public void onRestoreInstanceState(Bundle state) {
    super.onRestoreInstanceState(state);
    mView.setVisibility(
          state.getBoolean(KEY_VISIBLE) ? VISIBLE : GONE);
  }

  @Override
  public void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    if (savedInstanceState != null) {
        mView.setVisibility(
              savedInstanceState.getBoolean(KEY_VISIBLE) ? VISIBLE : GONE);
    }
  }
```

Сохранение объекта при пересоздании Activity

1. onRetainNonConfigurationInstance/
   getLastNonConfigurationInstance – **deprecated**
2. Static Field/Singleton/Application object
3. Service
4. Retain Instance Fragment

Sample





https://github.com/rusmonster/signin

Thank you