2. Activities && States. AcctState App.

What are Activities?

- Един от четирите базови компонента на Android приложение;
- Основната видима част с която взаимодействат потребителите (click, tap, slide, etc...). Обикновено заемат цял екран, но е възможно и да са по малки (да заемат част от него);
- Едно приложение може да има множество Acitivities; Android разчита на loosely-bound концепция що се отнася до Activities. Тоест Activities са проектират максимално независимо едно от друго; Всяко може да вика друго в ракмтите на приложението; Също така и в рамките на системата; Обратно едно приложение може да „предоставя" Activities на останалите приложения в системата;
- Горната концепция назоваваме, като множетво „входни точки" на Android приложение;
- Обикновено едно Activity определяме за „main". Това е Activity-то, което се показва след като потребителят стартира нашето приложение; Ще се съсредоточим върху него;

Acitivity? Create and/or manage...

- Базов клас – Activity, или производните му;
- Създаваме app Activity като наследим този базов клас;
- Управление – call-back(s);
- Нашето Activity разполага с методи, които биват извиквани от Android докато то е активно; Тоест в следствие действията на протребителя (или системата), биват извиквани различни call-back методи, които отразяват преминаването на Activity-то от едно състояние в друго;
- Тоест състоянията в които се намира нашето Activity, можем да мислим като КДА, чийто върхове са call-back(s), отрачяващи дадено състояние, а ребра – събитията (било потребителски или системни), довеждащи до съответното състояние;

Acitivity? Create and/or manage...

- Базов клас – Activity, или производните му;
- Създаваме app Activity като наследим този базов клас;
- Управление – call-back(s);
- Нашето Activity разполага с методи, които биват извиквани от Android докато то е активно; Тоест в следствие действията на протребителя (или системата), биват извиквани различни call-back методи, които отразяват преминаването на Activity-то от едно състояние в друго;
- Тоест състоянията в които се намира нашето Activity, можем да мислим като КДА, чийто върхове са call-back(s), отрачяващи дадено състояние, а ребра – събитията (било потребителски или системни), довеждащи до съответното състояние;

Acitivity? Create and/or manage...

- Два важни метода:
- onCreate() - Задължително реализираме; Отговорен е (by-design) за инициализирането (създаването) на UI компонентите населяващи (съдържащи се в) дадено Activity; Както и установяването на оформлението (layout?) на това Activity;
- Какво подсказва името му – бива извикван, веднага след като системата инстанциира обекта представляващ нашето Activity;
- onPause() - бива извикван, като индикация че потребителят е на път да напусне нашето Activity /или системата е на път да вземе радикални мерки спрямо него :)/;
- UI компоненти? Кратото определние – „джаджи" (widgets) намиращи се в дадено оформление (layout) и региращи на десйтвия :)
- По същество Java обекти; widgets - наследници на View; layouts – надледници на ViewGroup; Вторите, както предполага и името им, разполагат с механизъм за групиране на първите;
- Android предлага достатъчно „вградени" (базови) UI компоненти, които ще използваме;
- Важното за нас в случая е, че всеки вграден компонент има подходящо представяне във вид на .xml, намиращо се в res папката на нашето Android приложение и освен това;
- AndroidStudio (а и самата архитектура на един Android App проект) ни дава лесен достъп до тези базови .xml представянив в Java код;

Acitivity? Let's go public...

- AndroidManifest.xml – тук обявяваме нашето „main“ Activity и/или още Activities с които нашето приложение разполага;
- Как става това? – разбира се .xml markup :)

```
<manifest…>
    <application…>

<activity android:name="net.yaht.acctstate.MainActivity">
    <intent-filter>
        <action android:name="android.intent.action.MAIN" />
        <category android:name="android.intent.category.LAUNCHER" />
    </intent-filter>
</activity>

    </application>
</manifest>
```

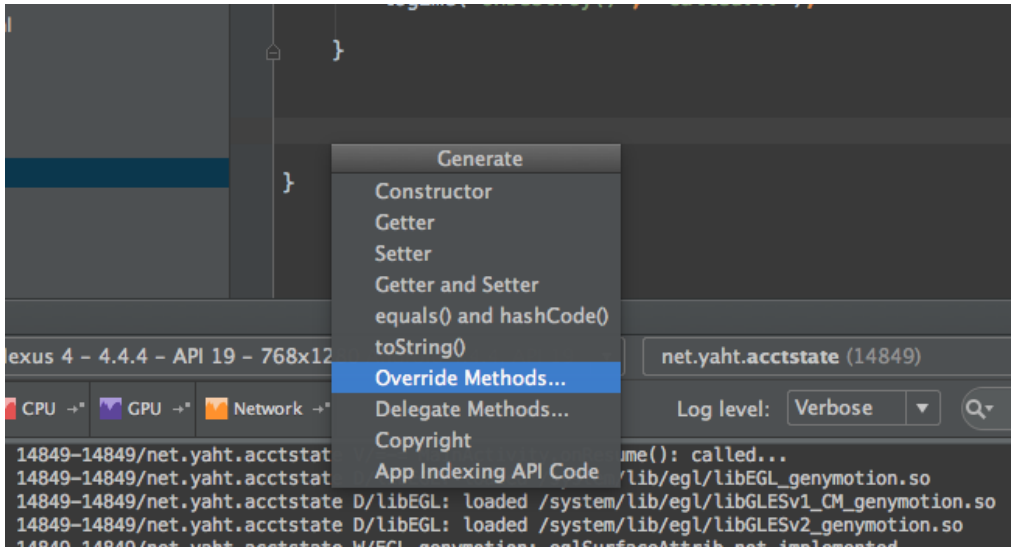
- Тоест задавайки пълнитят път (в границите на пакета на нашето приложение) до нашето Activity, обявихме че то ще е първото, което ще се стратира (**action, android.intent.action.MAIN**);
- Също така уведомихме Android да изведе нашето приложение в системният app-launcher, така че потребителите да могат да го стартират **(category, android.intent.category.LAUNCHER**);

AcctState app. Let's view the „code“.

- week-03, AcctState.zip, download, unpack import;
- MainActivity.java – използвах името предложено ми от AndroidStudio;
- public void log2me(String, String) – по лесен начин да се ориентираме в тоновете съобщения бълвани в „конзолата“ от Android или нашето приложение;
- onSomethingCallback() методи – 6 на брой; Телата им съдържат извкивания към базовата реализацация, т.е. super class реализациите (изискване на Android) и log2me();
- Целта - жизнено Activity и да се запознаем със стоянията през които минава на „живо“ (т.е. в средата на емулатора);

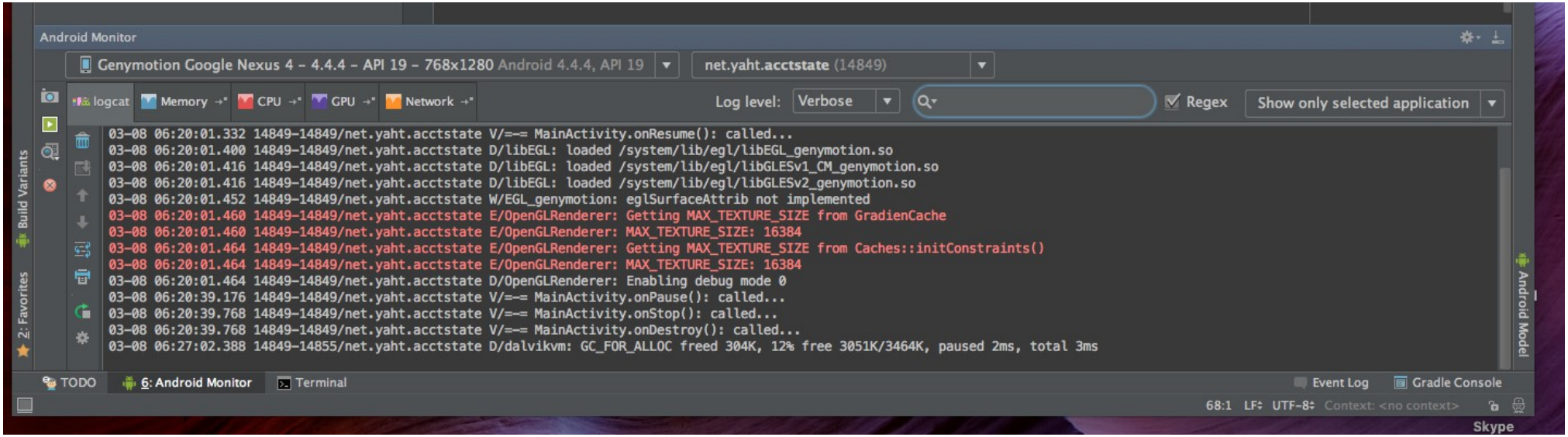
AcctState app. Let's view the „code".

- Как сме реализирали всички call-back методи?
- MainActivity.java – отворен за редактиране;
- Намираме се в кода на MainActivity класа;
- Клавишна комбинация cmd-n (Mac OS), ctrl-n (Win?);
- => Menu Generate => Override Methods … => този който ни трябва ;)



AcctState app. Let's view the „code".

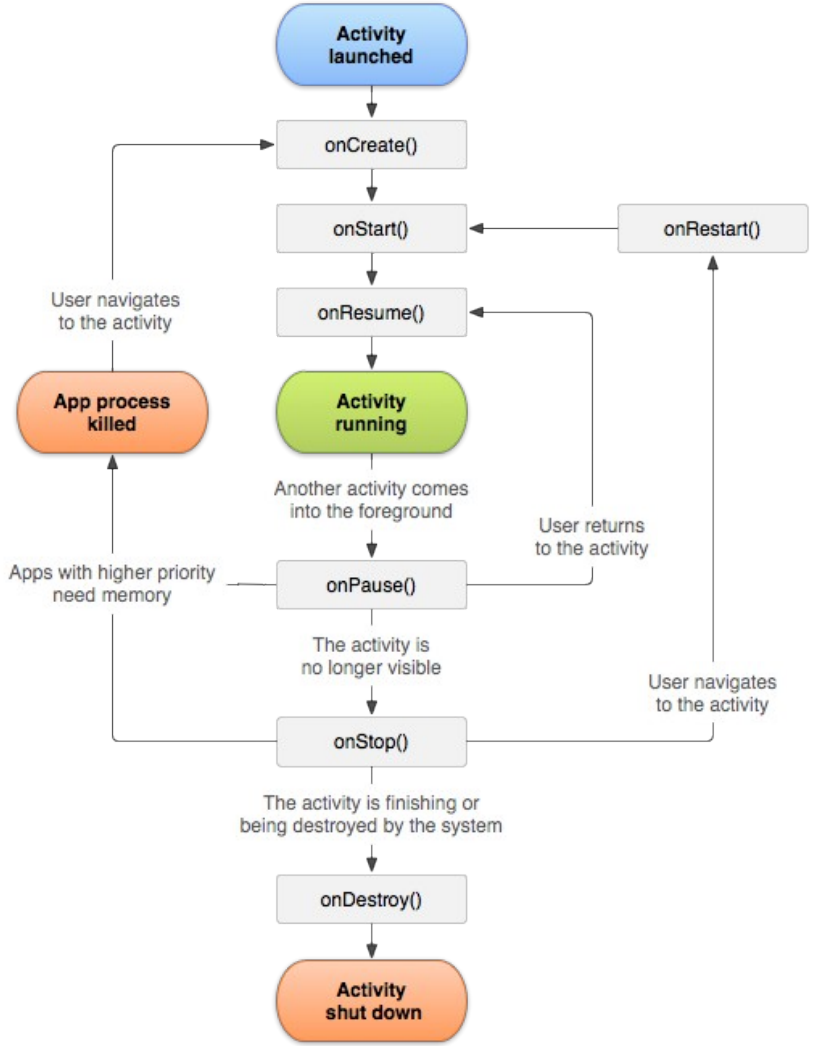
- Можем да стартираме;
- Genymotion => Virtual Image => Running Emulator => Run 'app';
- За какво ни беше log2me(), къде ще видим тези съобщения?
- Android Monitor => logcat => Log level: Verbose => Filter: =-=;



AcctState app. What about those states?

- 3 основни състояния;
- Resumed (Running) – нашето Activity „на-фокус“; Тоест потребителят може да взаимодейства с него; /след onReusme()/
- Paused – друго Activity е на фокус; Нашето е частично видимо (прозрачно) или другото Activity не покрива целият екран; Напълно живо; Activity обекта е в паметта; /след onPause()/
- Stopped – напълно скрито от друго Activity (намира се в backgorund); Activity-то е отново живо; Activity обекта е в паметта; /след onStop()/
- Разлика между Paused и Stopped – при Paused състоянието, нашето Activity е все още „прикачено“ към Android Window Manager;

AcctState app. What about those states – диаграма?



AcctState app. What about those states - даграма?

- И разбира се няколко думи по диаграмата :)
- Целият живот на едно Activity е заключен между извикванията на onCreate() и onDestroy();
- „Видимата" част от живота на едно Activity е заключена между извикванията на onStart() и onStop();
- Foreground живота на Activity е заключена между извикванията на onResume() и onPause();
- През живота си едно Activity може да премине повече от един път през всяко от състоянията /като изключим onDestroy() :)/;
- Основна бележка за onPause(), onStop(), onDestroy() - след тях нашето Activity е killable; Тоест при нужа, системата би унищожила нашето Activity (и процеса на който принадлежи);
- Последното, отговаря и на въпроса защо са ни толкова много състояния?
- От една страна за да си подредим добре Activity логиката по управление на Activity-то - създаваме/инициализираме widgets в onCreate(); управляваме състоянието им в onResume()/onPause(), пускаме Services в onStart(), спираме в onStop();
- От друга страна, самата Android система да е свободна или не да взима решения за нашето Activity/приложение в зависимост от състоянието в което е стъпило;

Enough „Theory“… Let's code a little >:

- on the fly will be ...