1 TODOs in base skeleton Project library

Tags: Checklist for robust library module

Add suggestions for our base project library that you think can improve our project structure and productivity

For July 2019 to

1. **TODO: Photo from response, download, save then call api with multipart - This process needs to be optimized using latest framework instead of relying on Glide**
2. **TODO: For list, use chunk object instead of full object, especially for db**
3. **TODO: For update query, update only specific column data instead of whole object**
4. **TODO:** Consider passing intent/boolean for showDataUi and showNoDataUi in ListScreen Interface for db intent from onEmptyData or onEmptySearchResult
5. **TODO:** Handle cast or parsing exception like expected array but found string in gson parsing
6. **TODO:** Consider dbOpsOnPageOneResponse in AppController
7. **On Progress:** Consider encapsulated initApiResponseListener in AppController
8. **Done:** original list -> filter -> transform in one line! (WeCardsV2 Utils.getFluentIterableList)
9. **Done:** Reduced complex post data like create card, create organization into one line only!
10. **Done:** List transformation! Transform a list of custom object into a list of any other object from available fields with your defined function! (WeCardsV2 Utils.transformList)
11. **TODO:** Type list instead of generic list in appAdapter (getType return to be used in BaseViewHolder?) and baseViewHolder?

1. **TODO:** To implement filter in appAdapter
2. **TODO:** BaseExpandableItemAdapter and BaseExpandableViewHolder
3. **Done:** Android compatible **predicate filter utility**
4. **Done: State list animation for spannable strings** (WeCardsV2 card full detail address, phone, email, web etc… lists with label)
5. **Done:** Now we can have list, in fact multiple types of lists in an entity without @ignore annotation (WeCardsV2) - seamless **Typeconverters**
6. **Done:** solve list selection equal and contain issue to modify selected list or get index of similar element accordingly (WeCardsV2)
7. **Done:** distinct deep copy list - cloneList (WeCardsV2)
8. **ListControlDialog same as ListControlActivity and ListControlFragment for automation**
9. **setDefaultData, setData and setConditionUi in place of onBindViewHolder**
10. **Working on: Improve automation controller using Type (WeCardsV2)**
11. **Done:** Easy parsing of response for object transformation (**Converted several lines into one line**!) for both list and object (Controller)
12. **Working on: Start imageUpload activity**
13. **Done: Common Social media integration** things (WeCardsV2)

14. **Done:** BaseViewHolder (WeCardsV2)
    To ease the single and multiple selection in recyclerView (WeCardsV2)
15. **Done: Appfont font family instead of boiler plate TypefaceUtil** (WeCardsV2)

Below is from June 2018 to June 2019

16. **Done:** Common bottom sheet dialog in oraEtLabora. In order to change style of option list item, you can either extend original adapter and override onCreateViewHolder or you can manage item_layout of viewHolder by api/flag/intent we pass in adapter
17. **Done:** Replace deep link with app link whenever possible (OraEtLabora)
18. **Synchronize update of common lib across multiple projects**
19. **Done:** Reducing boiler-plate codes for list screens (oraEtLabora, ambassador, wecardsv2) by ListControlActivity and ListControlFragment
20. **Done:** IntentForwardActivity to base lib (juzeatz, oraetlabora) as baseIntentForwardActivity
21. **Done:** Common bottomsheetdialog, bottomsheetdialogfragment or dialog with bottomsheet behaviour that gives onShowListener/bottomSheet callbacks
22. **Working on : ViewUtils: onClickListener: Get the rid of monkey clicks**
23. **Interface power with default extension methods**
    https://www.geeksforgeeks.org/java-and-multiple-inheritance/
24. **Working on : Get the rid of OEM Keyboard control :** State always hidden for all base activity, base fragment, base dialogs and all base views (PashuCare)
25. **Take root layout with transparent background to match parent by width and height for dialog, set onClick to dismiss dialog, set contentView by percentage, set no title flag to dim background OR just do not take not touch modal flag, use watch outside flag and in overridden onTouch method, dismiss the dialog**
26. **getOfflineData method name for controller**
27. **setConditionalUi method to include in base, in handleView**
28. **RvScrollback (on load more) to give:** position **and element item**
29. **Done:** An option in AppAdapter methods, addLoadMore and addList method either on top or bottom (to support chat like ui)
30. **Comment strange isLoading = true in loadMoreScroll class to make setLoaded call within loadMoreScroll listener to reduce boilerPlate code**
31. **Done:** Camera Dialog to DialogFragment (Common bottom sheet dialog)
32. **Done** Prepare MapFragment (for all kinds of things related to map)
33. **Done** Suggestion list for each app
34. **Working on suggestion list / feedback list / wish list for the Team**
35. **Working on Changelogs**
36. Encapsulation of ctil for flexibility.
37. **Done :** Compose email log txt common activity like image activity - just call one method and we are done!
38. Disabled and clickable input layout / edit text. - Pressed color, Normal color.
39. **Data binding for dialog**
40. **Make jsonkeys and prefs unite for app and common. Just use separation. To ease the access.**
41. **Done:** Room with base pojo model class. (WeCardsv2) (Room with inheritance benefits)

42. **Started: BasePhotoGestureActivity, fragment, adapters (Interface, abstraction (PashuCare, Ambassador), utility - PashuCare)**

43. **DI**

44. **Common controller having flexibility to handle multiple types of apis with unique pageNumber associated with it**
45. **Done:** Common ViewHolder same as we have made Common AppAdapter, AppController
46. **BaseLogin, BaseSignup and BasePhoneVerification classes to inherit**
47. **Done.** To Remove CustomBtn use from Base lib completely
48. **Started from PashuCare. Base Singleton Pojo model for Registration having common utility methods for typical validation and show error message with return code to perform the operation based on it. - E.g. Request focus on error field or do nothing**
49. **Doing. In line variable to reduce memory being occupied by variables**

50. **Done.** Network profile to faster debugging of network ops

51. **Done:** get user parameter from utils to ease dummy process (id, name, number etc...)
52. **Done** for emailPhone. Common registration, login, forgot pw and change pw error utils
53. **Done:** Base Pojo: id, parentId, title, subTitle, isSelected
54. **Done.** ListView interface to contain common methods related to adapter
55. **Done**: Base dialog
56. **Doing:** Expansion of utils to include maximum common functiionality
57. **Done. BaseApi final strings to be called for ApiInterface and to distinguish api difference.**
58. **Remove unused colors, strings etc... from skeleton**
59. **User hints on create of class**
60. **Done: Library project to AndroidX. Replace butter knife with data binding.**
61. **Use Custom Title Bar from Base Library.**
62. **As soon as we execute callApi of AppController, it should automatic send response via apiListener.onStartApiCall(Intent intent);**
63. **Started: ListActivity and ListFragment that contains all list adapter related methods to implement and override**
64. **BaseUi access modifiers should be: public, protected or private?**
65. **What about maximum offline experience? sync approach?**

66. **Fix names: mController, mRvAdapter, mModelList for convenient**
67. **Started: Instead of for loop of linkedTreeMap, use gson.toJson and convert direct into pojo**
68. **Arrange location class for lat lng data**
69. **Combine: ConvertDateTimeFormat and DateTimeUtil**
70. **Arrange DateTimeFormats class with common prefix and layman variable names**
71. **Can we remove abstract keyword from all message methods in activities and make them encapsulated in baseActivity to hide business logic in all activities?**
72. **Can't we give List<LinkedTreeMap> directly to adapter instead of giving it after pojo conversion?**
73. Preference key reference to json key instead of new memory block
74. **GradientDrawable for CustomTextView to include border and radius for each corner**
75. **What about baseViewStubFragment init after onResume?**
76. Add setCustomTitleBar method in base fragment class too
77. https://github.com/xxv/android-lifecycle
78. Try if only two controllers are enough (LinkedTreeMap and List<LinkedTreeMap>). Two controllers only: one that returns list and one another.
    Both controllers can be made to inherit common properties from super controller class.
    Consider api, adapter, Datalist, page number and load more
    Made AppController.

79. **Prepare all general test criteria check list: signup, login, profile etc...**
80. **Consider Singletones instead of pojo only (<u>that must not being used in list</u>) to eliminate parcel pain**
81. cancelable press again to exit toast
82. ViewUtils NPE Proof for varargs methods...
83. Common No Data TextView and Divider etc... to include
84. Utils class will be final static and will have private close constructor
85. GlideApp instead of Glide for the seamless inline usage
86. **Replace sharedPreference list by Database for convenient edit operations**
87. Resume and Pause animation method in base activity so that changes can be happened at once
88. Java 8. Lambda Expressions. GestureImageView Example.
89. **login, signup, profile and user : should be same, single model.**
90. Note down common methods in login signup forgot password viewmodel to implement through interface *RegistrationInterface.*
91. Abstract methods to Base Activity: (setBinding, getBinding cannot be converted, cannot be used as in that way for now), binding.setViewModel, getViewModel, binding.setLifeCycleOwner, setObservers etc...
92. **Make edit text input fields copy paste proof**
93. Combine CustomBtn and CustomTextView
94. NullProofUtil
95. Revert and split showToastMessage into two methods back to original form **(showErrorMessage(String s), showSuccessMessage(String s)**)
96. Also, have two another method like that as: **showErrorMessage (int stringResId) and showSuccessMessage(int stringResId) to access without context**
97. Need to improve: CustomCameraGalleryDialog, TakeImgCameraScreen and TakeImgGalleryScreen. If possible, eliminate these two activities as well.
98. split images or combine images? which way is better? restaurant app home screen.
99. Sole purpose of adapter is only to communicate with list and not to call api and handle api etc...
100. Implement interface in viewholder, get reference variable through that viewholder in adapter and call methods.
101. Each model class will have field (state): positionTag (for list position) ? No? because **we don't need that** field everywhere? (only require in list-detail flow)
102. KeyboardUtils and LocationUtils from juzeatz and jobsAdBoard
103. All listeners from abstract callbacks class and not individually
104. Remove various bup files and extra drawables if any
105. Remove app xmlns in the middle of any layout
106. Consistent style throughout app: Title, subtitle, margin, size, typeface, color
107. Auto capital first letter of each word in Add Address screen?
108. Style: Size for screen title and Heading by size like: heading_12sp, heading_14sp etc...
109. Size for images
110. Interfaces those are working as "call back listener" should be appended by "Listener" and not by "Interface" like onClickListener!

111. All custom views will have shimmer animation! Loader views will implement all other properties from custom views.
112. About shimmer margin
113. **Done:** Must have: getSourceScreen and setSourceScreen methods in base ui to override in all ui.
114. **Done:** **https://stackoverflow.com/questions/11631408/android-fragment-getactivity-sometimes-returns-null**

     solve getActivity() returns null in fragment by listener way - second answer.

115. **Done:** Always pass savedInstanceState in all helping methods of base class
116. **Done:** Pass activity in interface method: onResumeFragment for fragment lifecycle and get activity by: getHostActivity. set it by: setHostActivity
117. **Done:** One more base method to override in activity: setIntentData(getIntent()) and in fragment: setBundleData(getArguments()) after last abstract method to ensure prevention for view NPE
118. **Done:** Replace implementation by api to expose in app module
119. **Done:** ViewUtils: setClickable, setEnable, setFocusable etc… with view **varargs**
120. **Done.** Consider progressive ripple (setClickListener but setClickable false, handle via parent, ref: juzEatz more section -> profile pic group)
121. **Done.** Common style guide as we have done in uProjects. Just change: textSize, textColor, textTypeface.
122. **Done.** ViewStub for Activity as well in addition to fragments with Data binding : will not work from library module. Make it in app module.
123. **Done.** In lib gradle, unite all util implementation as one like: **utilLibs(configurations)**
124. **Done:** setListeners() instead of setClickListener()
125. **Done.** ViewUtil (show, hide, setClickListeners (yes, can be done by casting the passed activity/fragment into interface. Done in digiDiary -> EditTextUtils.setFocusListener.

2 Common class:_

1. Adapter class
2. User aka pojo, model, bean, raw class (Mostly, parcelable)
3. Session Manager, SharedPreferences to store last session, selection etc...
4. Network
5. All URLs, KEYs etc... of network operation should be static, final and in one common class
6. All local strings should also be managed in one common class
7. Run time permission. Permission management.
8. Custom font, Typeface
9. Validation
10. CustomViews

3 Common methods:_

1. isConnected()
2. CountryCode from sim, Telephony manager
3. ProgressDialog
4. Toast (error, message whatever...)
5. Snackbar
6. Alert Dialog (edit, delete whatever...)
7. New Activity intent
8. Validation
9. APICall

Where is the Impact Analysis Document?
Where is the Design Document?
Have you documented all the assumptions, limitations properly?
Have you done review of all the documents?
Did you get sign off on all the documents from all the stakeholders?

1. Design Approaches
2. Tips and Tricks
3. Special functions, commands and instructions
4. Lessons learnt
5. Peculiar situations
6. Debugging methods
7. Best Practices
8. Anything which can help you in future

1. 4 Before you start coding

Tags: before you start coding, how to improve the speed of coding, before coding

4.1 Design

Are you going to design your app? You can take some inspiration from here:

https://material.io/guidelines/layout/structure.html#

https://material.io/design/

https://www.uplabs.com/android

https://dribbble.com/

4.2 Naming Conventions:

https://source.android.com/source/code-style

Suggested By Udacity Forum Mentor

https://github.com/TreyCai/AndroidNamingConvention
https://github.com/NexMM/android-naming-conventions

Other references:

https://github.com/futurice/android-best-practices
https://github.com/ribot/android-guidelines/blob/master/project_and_code_guidelines.md
https://github.com/ribot/android-boilerplate

Google: clean code for Java, Clean code for android

https://google.github.io/styleguide/javaguide.html

https://developer.android.com/guide/topics/resources/more-resources.html#Dimension

4.3 Project Design (Architecture, Structure)

This is very impressive.

https://academy.realm.io/posts/donn-felker-solid-part-1/

Got from U-Reviewer (Udacity Project Reviewer - ABND)

http://konmik.com/post/introduction_to_model_view_presenter_on_android/

http://web.archive.org/web/20160206225831/https://people.apache.org/~fhanik/kiss.html

https://github.com/florina-muntenescu/MVPvsMVVM

Everything about Android architecture design principles

https://docs.google.com/document/d/1BhCB_KKcpzjjRjtB-6O7pO19t2Q_vXAy3cEplpjjNYs/edit?usp=sharing

Everything about MVVM Data binding Live Data

https://docs.google.com/document/d/19nKnDLYw1i0BtYG1JaRAjWjXWGsircAZZ6x5PRZKim8/edit?usp=sharing

Base UI package:

https://docs.google.com/document/d/1mDhlqPu85pT4SfuEhSo_0tqrTsP-hebz4qjqAxbPvk4/edit?usp=sharing

Android common utils

https://docs.google.com/document/d/1tOKX6hBtKpWgX1hnOpil_hUUEYXJbFqhJGTVqPa5Pqg/edit?usp=sharing

4.4 Method execution time:

Determine whether to use db query or for loop

https://stackoverflow.com/questions/180158/how-do-i-time-a-methods-execution-in-java

https://andbasicsindiascholar.slack.com/archives/C99HKC0UD/p1520153080000020

4.5 Performance tips

1. https://developer.android.com/training/articles/perf-tips.html
2. https://developer.android.com/topic/performance/vitals/render.html
3. https://developer.android.com/topic/performance/vitals/launch-time
4. https://www.youtube.com/playlist?list=PLWz5rJ2EKKc9CBxr3BVjPTPoDPLdPIFCE

4.6 ViewStub

https://developer.android.com/topic/performance/vitals/launch-time

4.7 Lazy Instantiation (Initializing when required)

https://stackoverflow.com/questions/12216327/the-right-way-to-do-lazy-initialization-of-singletons-in-android

4.8 Lazy Instantiation, Singleton, Synchronization, Double check lock, and Reflection protection

https://android.jlelse.eu/how-to-make-the-perfect-singleton-de6b951dfdb0

https://www.javaworld.com/article/2077568/learn-java/java-tip-67--lazy-instantiation.html

4.9 Generic methods with raw or bounded type parameter for throughout usage!

https://docs.oracle.com/javase/tutorial/java/generics/rawTypes.html

https://docs.oracle.com/javase/tutorial/java/generics/boundedTypeParams.html

4.10 Instance Of

https://stackoverflow.com/questions/20589590/why-not-use-instanceof-operator-in-oop-design

https://www.armedia.com/blog/instanceof-avoid-in-code/

4.11 Annotation Process

https://medium.com/@iammert/annotation-processing-dont-repeat-yourself-generate-your-code-8425e60c6657

4.12 Prevent memory leak

https://android.jlelse.eu/memory-leak-patterns-in-android-4741a7fcb570
https://medium.com/square-corner-blog/leakcanary-detect-all-memory-leaks-875ff8360745
https://developer.android.com/studio/profile/am-hprof.html

4.13 Memory monitoring

Source: Udacity Nanodegree Scholarship

1. Enable ADB Integration
2. Open memory monitor
3. Open the app
4. Note down allocated memory before executing the task you want to check memory for
5. Execute the task and see the difference in the allocated memory
6. Notice the change in memory graph constant
7. Execute the task again with the different set-up and business-logics to check the better approach
8. Done!

Side note: Udacity has dedicated course called Android Performance that focuses on Android Memory Management! https://classroom.udacity.com/courses/ud825

https://hsc.com/Blog/Best-Practices-For-Memory-Optimization-on-Android-1

https://developer.android.com/topic/performance/memory.html

4.14 Memory optimization Vs Readability

https://stackoverflow.com/questions/3383465/is-it-better-to-store-value-as-variable-or-call-method-again/3383515#3383515

4.15 Singleton pattern to store data temporary throughout an Application life cycle

https://android.jlelse.eu/how-to-make-the-perfect-singleton-de6b951dfdb0

https://androidresearch.wordpress.com/2012/03/22/defining-global-variables-in-android/

https://tausiq.wordpress.com/2013/01/27/android-make-use-of-android-application-class-as-singleton-object/

4.16 Android Library Project

Tags: Reuse project, modules, classes, packages, Library Module, Library Projects, Android Library

https://developer.android.com/studio/projects/android-library.html https://developer.android.com/studio/projects/index.html#LibraryProjects

https://github.com/codepath/android_guides/wiki/Building-your-own-Android-library

4.17 Gradle arrangement

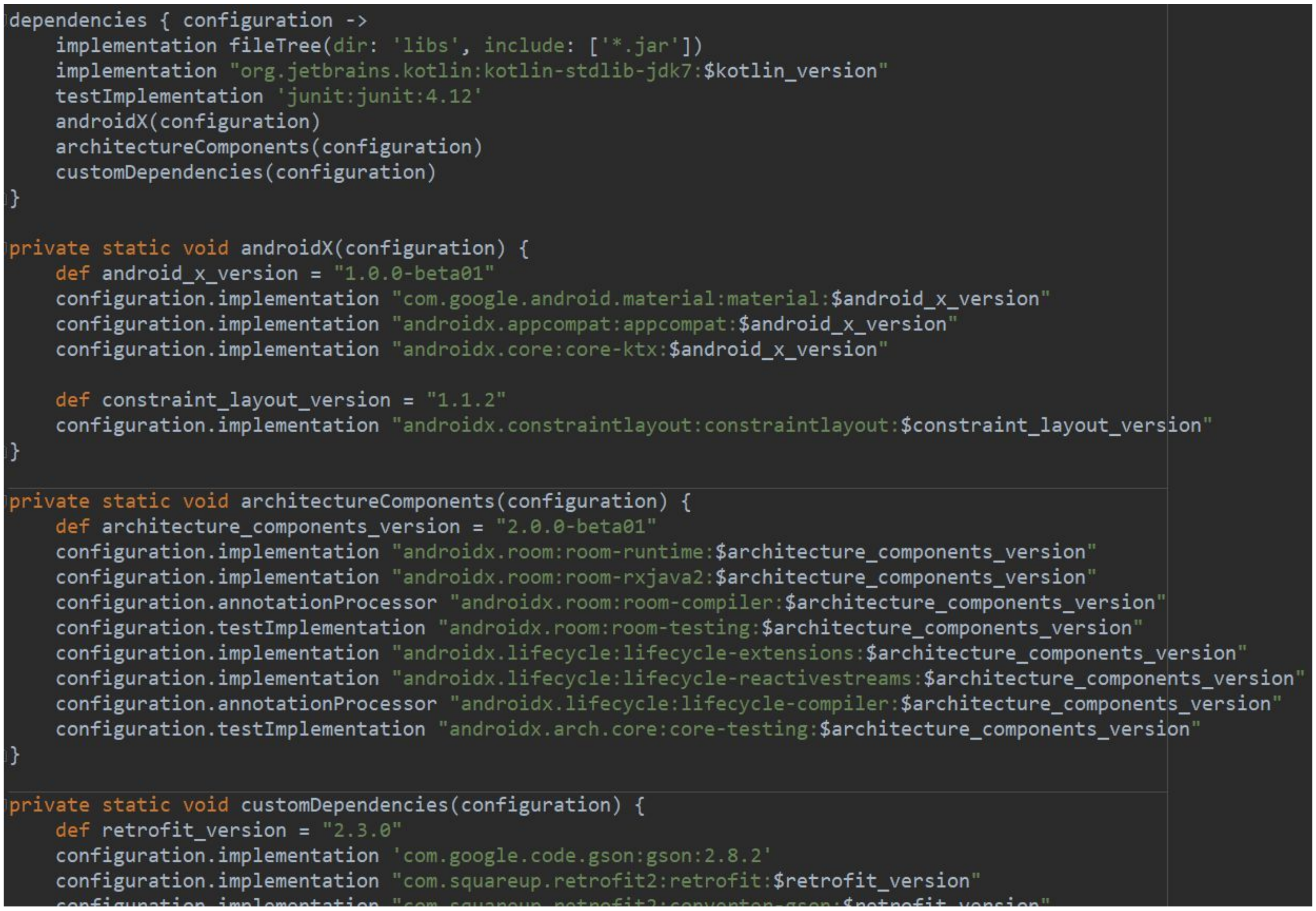
Google: Android studio gradle best practice

Ext block in project level to unify dependency versions in all modules.

Source:
https://github.com/googlesamples/android-architecture/blob/todo-mvvm-live/todoapp/build.gradle

https://github.com/googlesamples/android-architecture/blob/todo-mvvm-live/todoapp/app/build.gradle

Method calls in dependency block to organize a group of dependencies.



```
dependencies { configuration ->
    implementation fileTree(dir: 'libs', include: ['*.jar'])
    implementation "org.jetbrains.kotlin:kotlin-stdlib-jdk7:$kotlin_version"
    testImplementation 'junit:junit:4.12'
    androidX(configuration)
    architectureComponents(configuration)
    customDependencies(configuration)
}

private static void androidX(configuration) {
    def android_x_version = "1.0.0-beta01"
    configuration.implementation "com.google.android.material:material:$android_x_version"
    configuration.implementation "androidx.appcompat:appcompat:$android_x_version"
    configuration.implementation "androidx.core:core-ktx:$android_x_version"

    def constraint_layout_version = "1.1.2"
    configuration.implementation "androidx.constraintlayout:constraintlayout:$constraint_layout_version"
}

private static void architectureComponents(configuration) {
    def architecture_components_version = "2.0.0-beta01"
    configuration.implementation "androidx.room:room-runtime:$architecture_components_version"
    configuration.implementation "androidx.room:room-rxjava2:$architecture_components_version"
    configuration.annotationProcessor "androidx.room:room-compiler:$architecture_components_version"
    configuration.testImplementation "androidx.room:room-testing:$architecture_components_version"
    configuration.implementation "androidx.lifecycle:lifecycle-extensions:$architecture_components_version"
    configuration.implementation "androidx.lifecycle:lifecycle-reactivestreams:$architecture_components_version"
    configuration.annotationProcessor "androidx.lifecycle:lifecycle-compiler:$architecture_components_version"
    configuration.testImplementation "androidx.arch.core:core-testing:$architecture_components_version"
}

private static void customDependencies(configuration) {
    def retrofit_version = "2.3.0"
    configuration.implementation 'com.google.code.gson:gson:2.8.2'
    configuration.implementation "com.squareup.retrofit2:retrofit:$retrofit_version"
    configuration.implementation "com.squareup.retrofit2:converter-gson:$retrofit_version"
```

4.17.1 Compile Vs Implementation

https://stackoverflow.com/questions/44493378/whats-the-difference-between-implementation-and-compile-in-gradle

Read the answer given by humazed and also Read the comments between humazed and suragch.

Also read the answer given by aldok there as it is with example.

xyz is the library that uses abc library. Now, Mr. O once decided to use xyz. So, by the time, just because xyz was using abc and hence, abc was exposed to Mr. O, there were cases where Mr. O used abc. Now, as time changes and change is constant, he found some other library called P better than xyz. But just because he was using abc through xyz, it will be hard for him to remove xyz as it was being used in form of abc!

Implementation solved this problem.

Earlier, just because Mr. O was using abc through xyz, build process used to extend upto usages of abc and hence, it used to compile the whole app for every single change in dependency.

However, as implementation has made it easy and the build process will now only look for usages of xyz and hence build process has become faster.

However, If Mr. O is aware of consequences and he really wants to use the library abc through xyz, he can still use it by replacing the word implementation by the word api.

4.18 Check your Bucket

Based on the requirements.

4.18.1 Device compatibility (Multiple screens and backward OS support)

https://developer.android.com/guide/practices/compatibility.html

https://developer.android.com/training/basics/supporting-devices/index.html

https://developer.android.com/guide/topics/resources/overview.html

https://developer.android.com/guide/topics/resources/providing-resources.html

https://developer.android.com/guide/practices/screens_support.html

http://www.evoketechnologies.com/blog/effective-ui-design-tips-android-devices/

https://stackoverflow.com/questions/32860815/how-to-define-dimens-xml-for-every-different-screen-size-in-android

4.18.2 New Android Libraries

Tags: New native Android libraries, advanced Android native libraries, Android native support libraries, Improvised Android libraries etc... like DataBinding, ViewModel, LiveData, Room, Dagger2, NavigationUtil, Pagination, Downloadable Fonts, Adaptive Icons, bundle apks, instant app support etc…that can ease the development process and can enhance the app efficiency

4.18.2.1 Data Binding

Tags: Library, class, classes to improve code style, classes that help in the improvisation of our coding style, recommended classes, recommended libraries

https://docs.google.com/document/d/19nKnDLYw1i0BtYG1JaRAjWjXWGsircAZZ6x5PRZKim8/edit?usp=sharing

https://hackernoon.com/android-butterknife-vs-data-binding-fffceb77ed88

https://medium.com/google-developers/no-more-findviewbyid-457457644885

https://medium.com/google-developers/android-data-binding-adding-some-variability-1fe001b3abcc

https://medium.com/google-developers/android-data-binding-recyclerview-db7c40d9f0e4

https://medium.com/google-developers/android-data-binding-that-include-thing-1c8791dd6038

https://www.androidhive.info/android-databinding-in-recyclerview-profile-screen/

4.18.2.2 Room

https://docs.google.com/document/d/1dbLdLYPVyuS2ozET9kj-ovGD5y2ClW7OLiilBTMick8/edit?usp=sharing

4.18.2.3 Dagger2

https://docs.google.com/document/d/1IbQxRc5ChsW3IBgQ1xVHjIfdfvi8SJYGEfZMK2YSwak/edit?usp=sharing

4.18.2.4 Downloadable Fonts

1. https://developer.android.com/guide/topics/ui/look-and-feel/downloadable-fonts.html

4.18.2.5 Searchable configuration and not just searchView widget

https://developer.android.com/guide/topics/search/search-dialog.html

4.18.2.6 Multi-window support

https://developer.android.com/guide/index.html

4.18.2.7 Picture-in-Picture support

https://developer.android.com/guide/index.html

4.18.2.8 Adaptive Icons

https://developer.android.com/guide/practices/ui_guidelines/icon_design_adaptive.html

4.18.2.9 Android Bundle Apk/s and Android Instant App (AIA)

https://docs.google.com/document/d/1MHCiJivnzV-a2cBft5vS2w0V3CYtiU-DjLPpstmIYzw/edit?usp=sharing

4.18.2.10 Android Dynamic Feature Delivery

Do not install all codes and libraries such as place apis, location, map, qr codes etc… Let user install the app with minimum features only and then when required additional features, it can be downloaded later...

https://developer.android.com/guide/app-bundle/#playcore

4.18.2.11 Fonts in XML as Resource Just like String Resource

https://developer.android.com/guide/topics/ui/look-and-feel/fonts-in-xml.html

4.18.2.12 Multi-Resume

https://www.xda-developers.com/android-q-splitscreen-multitasking-multi-resume/

4.18.2.13 Android JetPack

4.18.2.14 Pagination

4.18.2.15 Navigation

4.18.2.16 AndroidX and Android Material 2.0

4.18.3 Third-party Useful Libraries

Tags: Library, class, classes to improve code style, classes that help in the improvisation of the development process, our coding style, recommended classes, recommended libraries

Add links of only those libraries that you think can be used in our projects:

For example, TedPermission, ArtOfDevImageCropper, RxJava, Dagger2 etc…

4.18.3.1 TakeOffAndroidRecyclerView

1. Download, extract, place.
2. Go to new -> other -> recyclerview -> done.

RecyclerView with adapter and dummy data.
https://github.com/TakeoffAndroid/RecyclerViewTemplate

4.18.4 UI UX Libraries

Check any of this library can be used in your project to enhance UI/UX or/and application performance or/and development efficiency

https://infinum.co/the-capsized-eight/top-five-android-libraries-every-android-developer-should-know-about-v2015
https://github.com/codepath/android_guides/wiki/Must-Have-Libraries

https://github.com/pmrajani/AndroidLibs/tree/master/AndroidLibs/Custom

https://github.com/wasabeef/awesome-android-ui

4.18.5 Module (Common base)

4.18.5.1 Check-list

1. Make TypefaceUtils, SharedPrefUtils and other Utils class final, a singleton with private constructors.
2.

4.18.5.2 Custom Views with flexible radius and ripple support

Tags: This is something that is recommended to have in common module for all projects that can give more flexibilities, features and ready to use methods, reusability. Checklist - This can be a part of a checklist like: Do you have something like this in your project? Have you implemented this class/library/feature in your project?

1. You can take the inspiration from **shapeOfView**
2. Flexibility. Define radius and get the shape like a rectangle, circular rectangle, capsule, cylindrical, round, oval etc… or have the attributes like shape: options
3. Flexibility to get the border or not, filled or not (default background will be blank) like: filled : true/false, border: true/false, borderColor: color, fillColor: color
4. Consider Flexibility to define default background, pressed background, and ripple color
5. Attributes to set gradient color in xml
6. Also, consider applying color state list alpha effect for text label if any on view

4.18.6 Extended Features and User Experience

https://docs.google.com/document/d/1uUabL8oAiOeJXiLdnjY1Q8_0UDSGFr31xD0AGnRri5M/edit?usp=sharing

4.18.7 Understanding the Work

(This point should be shifted to: before you start coding -> design -> before start implementing design

1. Purpose of application
2. The Flow of application (glimpse, not technical details but UI and UX approach, end-user asset: what the end-user is getting through this app, what are the features for end-user, how end-user will use this app)
3. Layout xml code
    3.1. The purpose of a screen (Focus on the only screen that you are going to work on)
    3.2. User interface and user experience. I.e. Screen behavior such as scrolling part, click effects, animations etc…
    3.3. UI UX Libraries
    3.4. Navigation (From where this screen will appear and where this screen can lead) and animations based on it
4. Common style. Alignment, Find common things and group them as one style: color, size, typeface (font style), behavior
5. Java code
6. Documenting the code
7. Documenting what you have learned

5 During Coding

5.1 Similar UI to compare or replace with

1. UI UX Libraries

5.2 ListView with viewholder pattern Vs RecyclerView

https://stackoverflow.com/questions/28525112/android-recyclerview-vs-listview-with-viewholder#

5.3 Android Performance Patterns Regarding Threads

The most energetic trainer from google I have ever found!

https://www.youtube.com/playlist?list=PLWz5rJ2EKKc9CBxr3BVjPTPoDPLdPIFCE

https://www.youtube.com/watch?v=qk5F6Bxqhr4&index=1&list=PLWz5rJ2EKKc9CBxr3BVjPTPoDPLdPIFCE

http://programmerguru.com/android-tutorial/what-is-asynctask-in-android/

5.4 Java Document

1. http://www.oracle.com/technetwork/articles/java/index-137868.html
2. https://en.wikipedia.org/wiki/Javadoc
3. https://code.tutsplus.com/tutorials/learn-java-for-android-development-javadoc-code-documentation--mobile-3674
4. https://stackoverflow.com/questions/1082050/linking-to-an-external-url-in-javadoc

6 Hide api keys from vcs : Build.config

secure API

https://gist.github.com/VDenis/46c222b16683447bab33

https://medium.com/code-better/hiding-api-keys-from-your-android-repository-b23f5598b906
https://stackoverflow.com/questions/33134031/is-there-a-safe-way-to-manage-api-keys

7 After Coding

7.1 Delete all unused variables, methods and all

Inspection, Analyzer, Clean-Up, Lint, Minify Provided by Android studio

https://stackoverflow.com/questions/33674592/how-can-i-find-all-unused-methods-of-my-project-in-the-android-studio-idea https://stackoverflow.com/questions/40280978/how-to-remove-unused-xml-java-code-from-android-studio

https://stackoverflow.com/questions/22273434/remove-unused-imports-in-android-studio

https://developer.android.com/studio/build/shrink-code.html

http://myhexaville.com/2017/03/08/android-proguard-tips/

https://stackoverflow.com/questions/33589318/error-building-apk-when-minifyenabled-true

https://github.com/krschultz/android-proguard-snippets/blob/master/libraries/proguard-square-okhttp.pro

https://stackoverflow.com/questions/35321742/android-proguard-most-aggressive-optimizations

1. Lint execution once everything is final, tested ok.
   Exclude specific inspection from lint if required.
   http://blog.passos.me/android_lint_ignore_specific_errors/

   https://developer.android.com/studio/write/lint.html#gradle
2. Test: Minify enable true (shrinks the code), make apk, Test.

8 Coding Style, Tips and Tricks:

From Adarsh Sir:

1. Multiple initializations in for loop to save multiple calls on list.size() within the scope of for loop

```
for (int i = 0, j = list.size(); i < j; i++){

}
```

2. Interface. We can access interface method with the help of interface reference variable that can be achieved by the (new) object of the class that implements that interface.

```
Interface I {
void myInterface();
}
class A implements I {
}

class C {
```

```
I myI = new A ();
myI.myInterface();
}
```

3. A Proper way to compare Strings

```
String a = null;
if (a.equals("b")){
}
Vs

if (("b").equals(a)){
}
```

Because we know, “b” will never be null. So, we will not get exception even if String “a” is null.

4. Generic List Trick

```
List<Integer> list = new ArrayList<>();
list.add(1);
playTrick(list);

void playTrick (List list) {
list.add("string value in integer list!");
}
```

9 Android Studio Tips and Tricks

9.1 Intellij Android Studio Java Code Style Configuration

https://www.jetbrains.com/help/idea/code-style-java.html

10 Examples

10.1 Dagger2, Retrofit and RxJava Sample by JakeWharton

https://github.com/JakeWharton/u2020

10.2 A Simple Android App with MVP, Dagger, RxJava, and Retrofit

https://medium.com/@nurrohman/a-simple-android-apps-with-mvp-dagger-rxjava-and-retrofit-4edb214a66d7

11 Extra

11.1 How to read the code

(Pending Google search: How to read code)

https://selftaughtcoders.com/how-to-quickly-and-effectively-read-other-peoples-code/

11.2 Recommended Resources:

https://android-developers.googleblog.com/

http://fragmentedpodcast.com/

https://blog.stylingandroid.com/

https://chris.banes.me/

http://android-developers.blogspot.com/

11.3 What's there inside an app?

https://www.appbrain.com/

12 Kotlin

http://kotlinlang.org/docs/reference/