Android naming convention

Ask Question

66

51

I am looking for a thorough Android naming convention suggestion. I found a little bit here:

http://source.android.com/source/code-style.html#follow-field-naming-conventions

which says:

- Non-public, non-static field names start with m.
- Static field names start with s.
- Other fields start with a lower case letter.
- Public static final fields (constants) are ALL_CAPS_WITH_UNDERSCORES.

Yet I am looking for something much more extensive covering all aspects of Android:

- how to name layouts and views within,
- how to name menus

- how to name styles
- how to name database tables (singular, plural) and fields within
- etc

If there is some generally accepted suggestion I would just love to follow that. All SDKs seem to go their own way so I am particular interested in the Android way to do it.

android

naming-conventions

edited May 21 '16 at 14:40

jaysoifer
1,309 5 14 34

asked Oct 13 '12 at 6:19

dorjeduck
4,111 8 41 60

1 Seeing as this is the first hit in Google, I thought I would add that through using "refactor" in both Android-Studio and Eclipse, you can rename something and change all of it's occurrences. This has been useful to me as I'm picky about naming conventions; hence my search. It's super easy to rename that particular instance, and just move on. – EricAnthony Feb 22 '14 at 23:09

Ignore Google coding style, its not explained enough...

and not even a Complete conv. There are not ANY International coding conv., since every company/grp has their own coding conv. Use your own. – Yousha Aleayoub Mar 13 '16 at 14:27

6 Answers

72

ribot's Android Guidelines are a good example of standard naming conventions:

Naming convention for XML files:

```
activity_<ACTIVITY NAMI
dialog_<DIALOG NAME>.xı
row_<LIST_NAME>.xml - ·
fragment_<FRAGMENT_NAMI
```

Naming convention for component/widget in xml files:

All components for *X* activity must start with the activity name all component should have prefix or short name like *btn* for `Button` For example,name for login activity component should be like following.

```
activity_login_btn_log:
activity_login_et_useri
activity_login_et_passı
```

Short name of major components:

```
Button - btn
EditText - et
TextView - tv
ProgressBar - pb
```

```
Checkbox - chk
RadioButton - rb
ToggleButton - tb
Spinner - spn
Menu - mnu
ListView - lv
GalleryView - gv
LinearLayout -ll
RelativeLayout - rl
```

edited Oct 19 '17 at 19:52

bstrauch24
681 4 14

answered Jan 7 '15 at 11:40

Pravin Bhosale
1,551 13 12

5 doesnt realy metter does it.. as long as you (or all you company) adpot 1 style who cares where does it come from. I made like 4 more or less simple android apps so far and I made myself almost identical convention as this one. I think it is all you need. I use 'a_' instead of 'activity' and so on since it is too long lol – Srneczek Apr 28 '15 at 9:55

Does it really necessary to start the name of the components with the name of the activity? I mean you would refer to the names within the respective layout file anyway. – szedjani Nov 19 '15 at 16:08

1 Its not really necessary but when your project is growing that time this will be very helpful – Pravin Bhosale Nov 20 '15 at 7:57

4 MainActivity + activity_main? I know this is the standard, but who the crap invented this? Most un-intutive, especially why the names get longer, when fragments come into place. – brainray Jan 26 '16 at 15:01

Personally I don't find this very consistent – Jethro Feb 9 '16 at 15:52

17

This is an excellent collection of best practices to start with: https://github.com/futurice/android-best-practices

Here's what I use. I'll also copy from that link.

Object naming

- Don't use the `m` or `s` prefix as per Google guidelines. I've stopped for years and I find it easier without them. The IDE will tell you when you're using something private or static; it seems like an obsolete convention.
- CONSTANTS start with caps
- Acronyms should only capitalize the first letter. For example,

`functionUrl` and `unitId`. Not `unitID`.

- Prefix with the type of object. For example a TextView which contains a name would be `tvName`. An EditView with a password would be `etPass`.
- If it's something usually used only once in an activity (e.g. ListView), don't be afraid to just call it `lv`.
- If it's not an object type just name it by it's function. For example, if it's a string that holds the ID, name it as `id`, not stringId. The IDE will tell you when it's a string or a float or a long.
- Keep it legible. Use something like `Pass` instead of `Password`.
- Within the XML, name should be underscore with no capitals, e.g. `tv_name` and `et_pass`
- Put the `android:id` as the first attribute in the XML.

File naming

- Prefix layouts with the type it is. E.g. `fragment_contact_` `details.xml`, `view_primary_button.xml`, `activity_main.xml`.

- For the classes, categorize them into folders, but use suffixes. For example, `/activities/MainActivity.java` or `/fragments/DeleteDialog.java`. My folders are *activities, fragments, adapters, models,* and *utils*.
- Adapters should say how and when they are used. So a ListView adapter for ChatActivity might be called `ChatListAdapter`.

colors.xml and dimens.xml as a pallete

- For color, use names like `gray_light`, not `button_foreground`.
- For dimens, use names like `spacing_large`, not `button_upper_padding`.
- If you want to set something specific for your button color or padding, use a style file.

strings.xml

- Name your strings with keys that resemble namespaces, and don't be afraid of repeating a value for two or more keys.
- Use `error.message.network`, not `network_error`.

Reasoning

The purpose of naming conventions is not to make everything neat and consistent. It's there to flag possible mistakes and improve workflow. Most of these are designed to be convenient for keyboard shortcuts. Try to focus around minimizing bugs and improving workflow rather than looking nice.

Prefixes are great for those, "What's the name of that TextView?" moments.

Suffixes are there for the things which you don't access so often in that manner, but can be confusing. For example, I may not be sure whether I put my code in the Activity, Fragment, or Adapter of that page. They can be dropped if you like.

XML ids are often in lowercase and uses underscores just because everyone seems to do it this way.

answered Dec 23 '15 at 4:47

Muz
3,773 3 34 50

What about the names of classes. e.g: `ActivityMain` or `MainActivity`. Which one would you recommend? I think it makes sense to go by: CLASS: `NameActivity`, LAYOUT: `name_activity`, COMPONENT: `nameactivity_component_name`. An example of this would be MainActivity, main_activity, mainactivity_btn_cancel – Jethro Feb 8 '16 at 14:34

4 I'm using the m and s prefix. I found it very useful, and for sure it doesn't make the code worse. Moreover sometimes I prefer to open some file without the IDE. It is very easy to differentiate fields and simple variables. – Arkadiusz Cieśliński Jun 21 '16 at 13:11

I'm looking at the Camera2 example at the moment, and I really don't care where `mBackgroundHandler` etc. comes from, so naming them `backgroundHandler` puts the important info on the left. If you need it, adding a '' *suffix to parameters and '*_' suffix for local variables lets you visually and mentally skip the underscores unless you need to focus on them. – WillC Jun 27 '17 at 2:45

10

CONSISTENCY
Everyone (unless working in teams) will have their own convention and which one you choose does not matter. Making sure it is **consistent** throughout the whole application does matter.

STRUCTURE
Personally, I use a naming convention like this as it runs from the class name down to component and is consistent throughout the xml:

- **CLASS**: `<ClassName>`
- **ACTIVITY**: `<ClassName>**Activity**`
- **LAYOUT**: `classname_activity`
- **COMPONENT IDS**: `classname_activity_component_name`

An example of this would be `OrderActivity.class` , `order_activity.xml` , `order_activity_bn_cancel` . Notice all the XML is in lowercase.

ABBREVIATING LAYOUTS
If you would like to use shorter names to keep the code tidier; then another method can be to abbreviate **ALL** the names in XML aswell as the layouts.

An example of this would be **OrderActivity**.class: **ord_act**.xml, **ord_act**_bt_can, **ord_act**_ti_nam, **ord_act**_tv_nam. I break down the names into three but this depends how many similar names you have

ABBREVIATING COMPONENT TYPES
When abbreviating component types try to keep these consistent too. I normally use two letters for the component type and three letters for the name. However sometimes the name will not be necessary if that is the only element of that type in the layout. The principle of the ID is to be unique

- **COMPONENT IDS**:

```
nam_act_componen
t_nam
```

COMPONENT TYPE ABBREVIATIONS
(This list shows two letters which is plenty)

Frame Layout:	fl
Linear Layout:	ll
Table Layout:	tl
Table Row:	tr
Grid Layout:	gl
Relative Layout:	rl
Text View:	tv
Button:	bt
Check Box:	cb
Switch:	sw
Toggle Button:	tb
Image Button:	ib
Image View:	iv
Progress Bar:	pb
Seek Bar:	sb
Rating Bar:	rb
Spinner:	sp
WebView:	wv
Edit Text:	et
Radio Group:	rg
List View:	lv
Grid View:	gv
Expandable List View:	el
Scroll View:	sv
Horizontal Scroll View:	hs
Search View:*	se
Tab Host:	th
Video View:	vv
Dialer Filter:	df
Include:	ic
Fragment:	fr
Custom View (other):	cv

edited May 19 '18 at 22:24

answered Feb 8 '16 at 14:50

Jethro

682 1 8 20

2 radio button = rating bar? – Mitch Sep 4 '18 at 9:19

8

I don't think there is a convention for this yet . each company has its own rules and I don't think anyone cares much about it here.

For me , I prefer putting the name to be bound to the context . for example , if there is an activity called "MainActivity" , its layout name would be "main_activity.xml" , and for each resource associated with this activity , I add a prefix "main_activity" so that I know that it uses it . same goes for the ids used for this activity .

The reason I use those naming is that it's easier to find them, delete if needed , and you won't get them replaced with others if you use android libraries since the names are quite unique.

I also try as much as possible to give meaningful names , so you will usually not see "listView" or "imageView2" as ids , but something like "contactsListView" and "contactImageView" .

the same name (or similar) would also match the variables inside the java code, in order to make it easier to find.

So , in short, my tips are:

- try to avoid numbers inside the names . they usually don't mean much , and show that you've only used drag&drop for the UI designer .
- for demos, POCs and for questions here , don't worry yourself about naming .
- try to add a prefix to all of the names of the resources (including ids) to show which context they belong to , and to achieve uniqueness.
- give meaningful names wherever possible .

edited Sep 18 '15 at 10:24
Aditi Parikh
1,402 3 9 32

answered Oct 13 '12 at 8:01
android developer
54.2k 98 468 870

Every body uses his own, The main goal is

1

to avoid mistakes and misinterpretation, specially when others read your code. Though syntax highlighting, and auto code inspection in modern IDE's makes it pretty point less.

But these naming conventions also make it very convenient when code completion is turned on. For example just type `m` and auto complete will show you a list of class fields.

But many times you have to work with other's code, which doesn't use such convention. such protected variables and overridden method parameters just add to the confusion.

Few examples:

- Prefix class variables with m , and make static finals variables all caps, with `_` separating words. Don't prefix any thing to lower scope variables.
- Name layout after the UI parent, for example `act_main.xml` , `frg_detail.xml` , `itm__act_main__list1.xml` ; for an activity `MainActivity` , a

fragment `DetailFragment`, item layout for a `ListView` in `MainActivity` with id `list1`, respectively.

- Name element Id's in xml layouts like: `lsv__act_main__list1` for a ListView and `btn__act_main__submit` for a `Button element. This makes them much easier to find with auto complete.

edited Oct 13 '12 at 8:43

answered Oct 13 '12 at 8:25

S.D.
24.9k 1 62 112

thx - for me coding convention are not really pointless in the age of powerful IDE, just my take on it so I hope to find some more generally accepted ones – dorjeduck Oct 13 '12 at 9:50

That's fine. They have other benefits too: You see ID of a UI view in the LogCat and from the ID you know what it is, and where to look for it in the code. – S.D. Oct 13 '12 at 9:53

I would go with fully qualified names for the prefixes:

`activit_main.xml` , `fragment_main.xml` , `button_activity_main_submit.xml` , etc. – Ramón García-Pérez Nov 12 '13 at 22:36

0

The newest Android Eclipse plugins create some of the files you mention automatically when you create a new project. From that, the naming is something like that:

```
layout/activity_main.xı
menu/activity_main.xml
...
```

I followed this scheme with e.g.

```
layout/fragment_a.xml
layout/fragment_b.xml
...
```

So it's something like with package names, from general to detailed. It also allows for neat sorting.

answered Oct 13 '12 at 8:13

Ridcully
18.6k 7 54 71