## **Android Code Style And Guidelines**

## **1 . Coding Style For Contributors**

From here you can learn the coding style that you should follow while contributing to android source code.

### 2 . Project Guidelines

#### 2 .1. Project Structure

New projects should follow the Android Gradle project structure that is defined on the [Android Gradle plugin user guide.](http://tools.android.com/tech-docs/new-build-system/user-guide#TOC-Project-Structure)

#### 2 .2. File naming

##### 2 .2 .1. Class files

Class names are written in [UpperCamelCase](https://en.wikipedia.org/wiki/Camel_case). For classes that extend an Android component, the name of the class should end with the name of the component; for example:

* SignInActivity,
* SignInFragment,
* ImageUploaderService,
* ChangePasswordDialog.

##### 2. 2. 2. Resources files

Resources file names are written in **lowercase\_underscore**.

###### 2. 2. 2. 1. Drawable files

Naming conventions for **drawables**:

|  |  |  |
| --- | --- | --- |
| **Asset Type** | **Prefix** | **Example** |
| Action bar | ab\_ | ab\_stacked.9.png |
| Button | btn\_ | btn\_send\_pressed.9.png |
| Dialog | dialog\_ | dialog\_top.9.png |
| Divider | divider\_ | divider\_horizontal.9.png |
| Icon | ic\_ | ic\_star.png |
| Menu | menu\_ | menu\_submenu\_bg.9.png |
| Notification | notification\_ | notification\_bg.9.png |
| Tabs | tab\_ | tab\_pressed.9.png |
| Other drawables ( logo, pictures .. ) | res\_ | res\_logo.svg |
| background | bg\_ | bg\_input\_email |

Naming conventions for **icons** (taken from [Android iconography guidelines](http://developer.android.com/design/style/iconography.html)):

|  |  |  |
| --- | --- | --- |
| **Asset Type** | **Prefix** | **Example** |
| Icons | ic\_ | ic\_star.png |
| Launcher icons | ic\_launcher | ic\_launcher\_calendar.png |
| Menu icons and Action Bar icons | ic\_menu | ic\_menu\_archive.png |
| Status bar icons | ic\_stat\_notify | ic\_stat\_notify\_msg.png |
| Tab icons | ic\_tab | ic\_tab\_recent.png |
| Dialog icons | ic\_dialog | ic\_dialog\_info.png |

Naming conventions for **selector** states:

|  |  |  |
| --- | --- | --- |
| **State** | **Suffix** | **Example** |
| Normal | \_normal | btn\_order\_normal.9.png |
| Pressed | \_pressed | btn\_order\_pressed.9.png |
| Focused | \_focused | btn\_order\_focused.9.png |
| Disabled | \_disabled | btn\_order\_disabled.9.png |
| Selected | \_selected | btn\_order\_selected.9.png |

###### 2.2.2.2. Layout files

Layout files should match the name of the Android components that they are intended for but moving the top level component name to the beginning. For example, if we are creating a layout for the SignInActivity, the name of the layout file should be activity\_sign\_in.xml.

|  |  |  |
| --- | --- | --- |
| **Component** | **Class Name** | **Layout Name** |
| Activity | UserProfileActivity | activity\_user\_profile.xml |
| Fragment | SignUpFragment | fragment\_sign\_up.xml |
| Dialog | ChangePasswordDialog | dialog\_change\_password.xml |
| AdapterView item | --- | item\_person.xml |
| Partial layout | --- | partial\_stats\_bar.xml |

A slightly different case is when we are creating a layout that is going to be inflated by an **Adapter**, e.g to populate a **ListView**. In this case, the name of the layout should start with item\_.  
  
Note that there are cases where these rules will not be possible to apply. For example, when creating layout files that are intended to be part of other layouts. In this case you should use the prefix partial\_.

###### 2.2.2.3. Menu files

Similar to layout files, menu files should match the name of the component. For example, if we are defining a menu file that is going to be used in the UserActivity, then the name of the file should be activity\_user.xml.

A good practice is to not include the word menu as part of the name because these files are already located in the menu directory.

###### 2.2.2.4. Values files

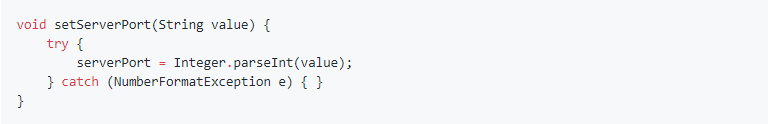
Resource files in the values folder should be plural, e.g. strings.xml, styles.xml, colors.xml, dimens.xml, attrs.xml.

### 3. Code Guidelines

#### 3.1. Java Language Rules

##### 3.1.1. Don’t ignore exceptions

You must never do the following:

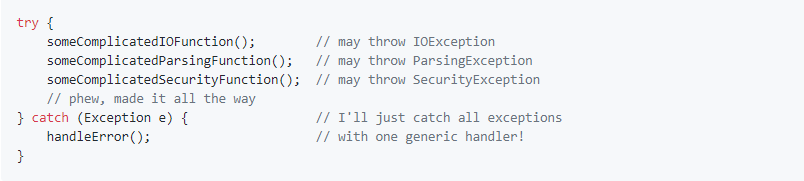


*While you may think that your code will never encounter this error condition or that it is not important to handle it, ignoring exceptions like above creates mines in your code for someone else to trip over some day. You must handle every Exception in your code in some principled way. The specific handling varies depending on the case.* - ([Android code style guidelines](https://source.android.com/source/code-style.html))

See alternatives [here](https://source.android.com/source/code-style.html#dont-ignore-exceptions).

##### 3.1.2. Don’t catch generic exceptions

You should not do this:



See the reason why and some alternatives [here](https://source.android.com/source/code-style.html#dont-catch-generic-exception)

##### 3.1.3. Don’t use finalizers

*We don't use finalizers. There are no guarantees as to when a finalizer will be called, or even that it will be called at all. In most cases, you can do what you need from a finalizer with good exception handling. If you absolutely need it, define a close()method (or the like) and document exactly when that method needs to be called. See InputStream for an example. In this case it is appropriate but not required to print a short log message from the finalizer, as long as it is not expected to flood the logs.* - ([Android code style guidelines](https://source.android.com/source/code-style.html#dont-use-finalizers))

##### 3.1.4. Fully qualify imports

#### 

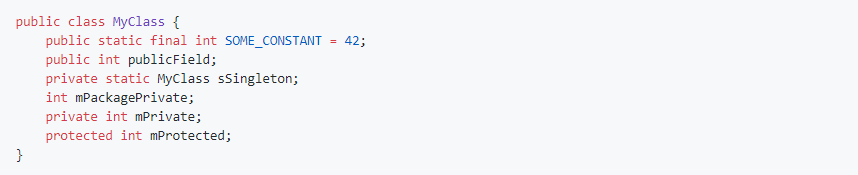
#### 3.2. Java Style Rules

##### 3.2.1. Fields Definition and Naming

Fields should be defined at the top of the file and they should follow the naming rules listed below.

Example:

* Private, non-static field names start with m.
* Private, static field names start with s.
* Other fields start with a lowercase letter.
* Static final fields (constants) are ALL\_CAPS\_WITH\_UNDERSCORES.



**3.2.2. Treat acronyms as Words**

|  |  |
| --- | --- |
| **Good** | **Bad** |
| XmlHttpRequest | XMLHTTPRequest |
| getCustomerId | getCustomerID |
| String url | String URL |
| long id | long ID |

**3.2.3. Use spaces for indentation**

Use 4 space indents for blocks:



Use 8 space indents for line wraps:



##### 3.2.3. Use spaces for indentation

##### 3.2.4. Use standard brace style

##### 3.2.5. Annotations

##### 3.2.6. Limit Variable Scope

##### 3.2.7. Order Import Statements

##### 3.2.8. Logging Guidelines

##### 3.2.9. Class Member Ordering

##### 3.2.10. Parameter Ordering in Methods

##### 3.2.11. String Constants, Naming and Values

##### 3.2.12. Arguments in Fragments and Activities

##### 3.2.13. Line Length Limit

##### 3.2.14. RxJava Chains Styling

#### 3.3. XML Style Rules

##### 3.3.1. Use Self Closing Tags

##### 3.3.2. Resources Naming

##### 3.3.3. Attributes Ordering

#### 3.4. Tests Style Rules

##### 3.4.1. Unit Tests

##### 3.3.1. Espresso Tests

* Networking : through Retrofit library by square
* Database: via ObjectBox by GreenRobot
* CodeStructure:

**Unit testing**

-->JUnit

**Commenting standard**

**Automated UI testing**

-->Instrumentation Testing (Espreso)

-->Robolectric Testing

**Continuous integration and firebase test lab**

-->CircleCI (free but supports only linux and unix)

-->TravisCI (paid)

* <https://github.com/jshiell/checkstyle-idea>
* <https://source.android.com/setup/contribute/code-style>
* <https://google.github.io/styleguide/javaguide.html>

**Assigning Id to components**

|  |  |  |
| --- | --- | --- |
| **Component** | **prefix** | **Example** |
| Button | btn\_ | btn\_login |
| Textview | lbl\_ | lbl\_first\_name |
| Edittext | input\_ | input\_first\_name |
|  |  |  |
|  |  |  |