

**Pick Pic**

**Design**

**2017 - 1 Human ICT Software Engineering | 2017.04.26**

|  |  |
| --- | --- |
| 20155652 | KangSoYeon |
| 20154019 | LeeYeWon |
| 20151722 | MyungSeKyo |
| 20152164 | LeeWonJun |
| 20151535 | ParkGunHoo |

Table of Contents

[1.0 System architecture 3](#_Toc481000274)

[1.1 Modules and interfaces 3](#_Toc481000275)

[1.2 Data description 3](#_Toc481000276)

[1.3 Design alternatives 4](#_Toc481000277)

[1.4 Assumption 4](#_Toc481000278)

[2.0 Diagrams 5](#_Toc481000279)

[2.1 Class diagrams 5](#_Toc481000280)

[2.2 Sequence diagram 5](#_Toc481000281)

[3.0 Process 6](#_Toc481000282)

[3.1 Risk assessment 6](#_Toc481000283)

[3.2 Project schedule 7](#_Toc481000284)

[3.3 Team structure 7](#_Toc481000285)

[3.4 Test plan 7](#_Toc481000286)

[3.5 Documentation plan 8](#_Toc481000287)

[3.6 Coding style guidelines 9](#_Toc481000288)

[Use Javadoc Standard Comments 9](#_Toc481000289)

[Write Short Methods 9](#_Toc481000290)

[Define Fields in Standard Places 10](#_Toc481000291)

[Limit Variable Scope 10](#_Toc481000292)

[Order Import Statements 10](#_Toc481000293)

[Follow Field Naming Conventions 10](#_Toc481000294)

[Use Standard Brace Style 10](#_Toc481000295)

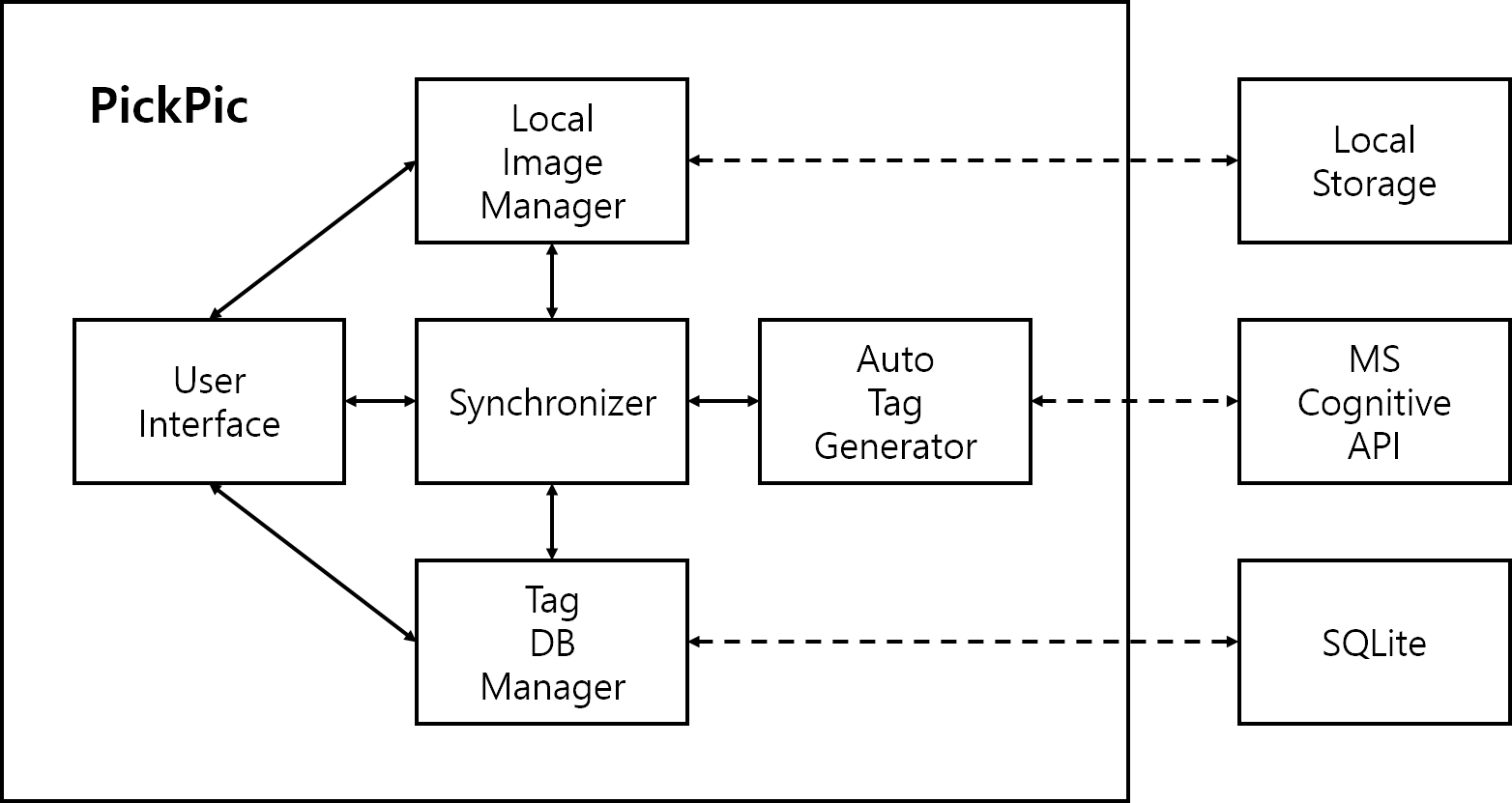
[Limit Line Length 11](#_Toc481000296)

[Use Standard Java Annotations 11](#_Toc481000297)

[Use TODO Comments 11](#_Toc481000298)

# System architecture

## 1.1 Modules and interfaces



Local Image Manager : this module provide easy way to access local storage images.

Synchronizer : this module synchronize between application DB and local storage each time it is run.

Tag DB Manager : this module manage the application DB that include tags.

Auto Tag Generator : this module is responsible for tag generation using API.

Local Storage : Internal storage space in local device.

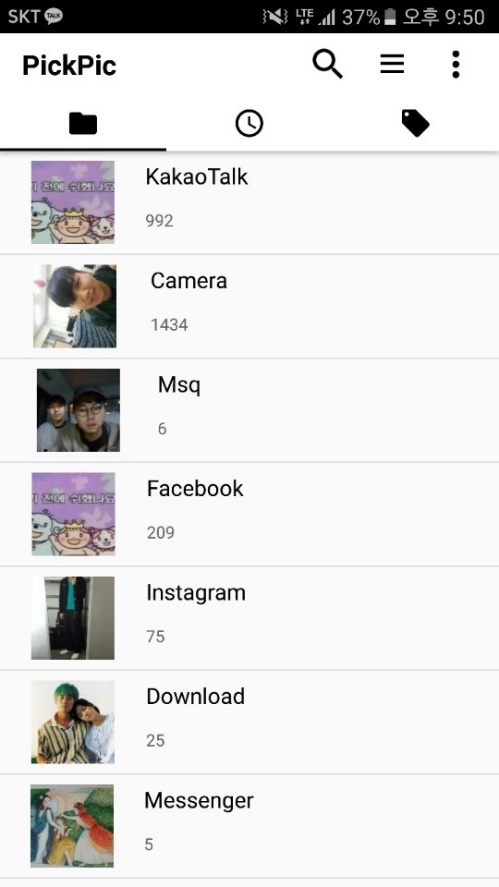
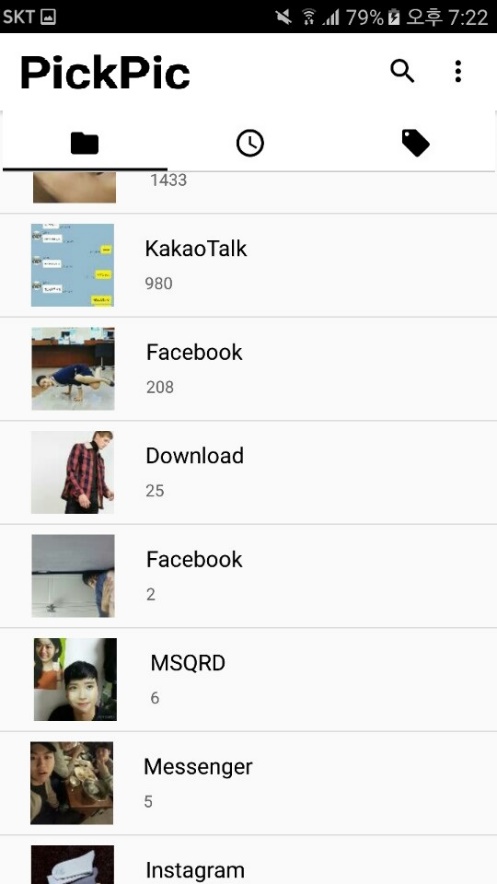
MS Cognitive API : this API return tags when the image is transferred.

SQLite : compact DBMS in Android system.

## 1.2 Data description

|  |  |  |
| --- | --- | --- |
| class | variable | note |
| DirectoryTabListViewItem | Bitmap thumbnail; | thumbnail of this directory |
| String directoryName; | name of this directory |
| String path; | absolute path of this directory |
| Int numOfImage; | number of images |
| DateTabListViewItem | String date | date |
| ArrayList<String> imagePaths | full paths of all images |
| TagTabListViewItem | ArrayList<String> tags | all tags about this image |
| ArrayList<int> numOfImage | number of images |

## 1.3 Design alternatives



We modified a bit in the early version of MainActivity. In the toolbar, we deleted a menu icon. Instead, we add additional functions in the search icon. When users click the search, then go to SearchActivity. And then performs the combined functions of two icons because of 2 reasons.

First is compaction. Having three icons can be complicated for viewing. The functions is not lost, but it combined into one icon.

Second is easy to use. From the user’s point of view, two icons can be easier to use than three. And users mainly use search icon in the MainActivity, so menu icon is not likely to be needed for viewing.

## 1.4 Assumption

Target SDK = 22

Minimum SDK = 15

Suppose MS cognitive API is working.

1. This application run on the Android OS. So, Android device environment must be supported.

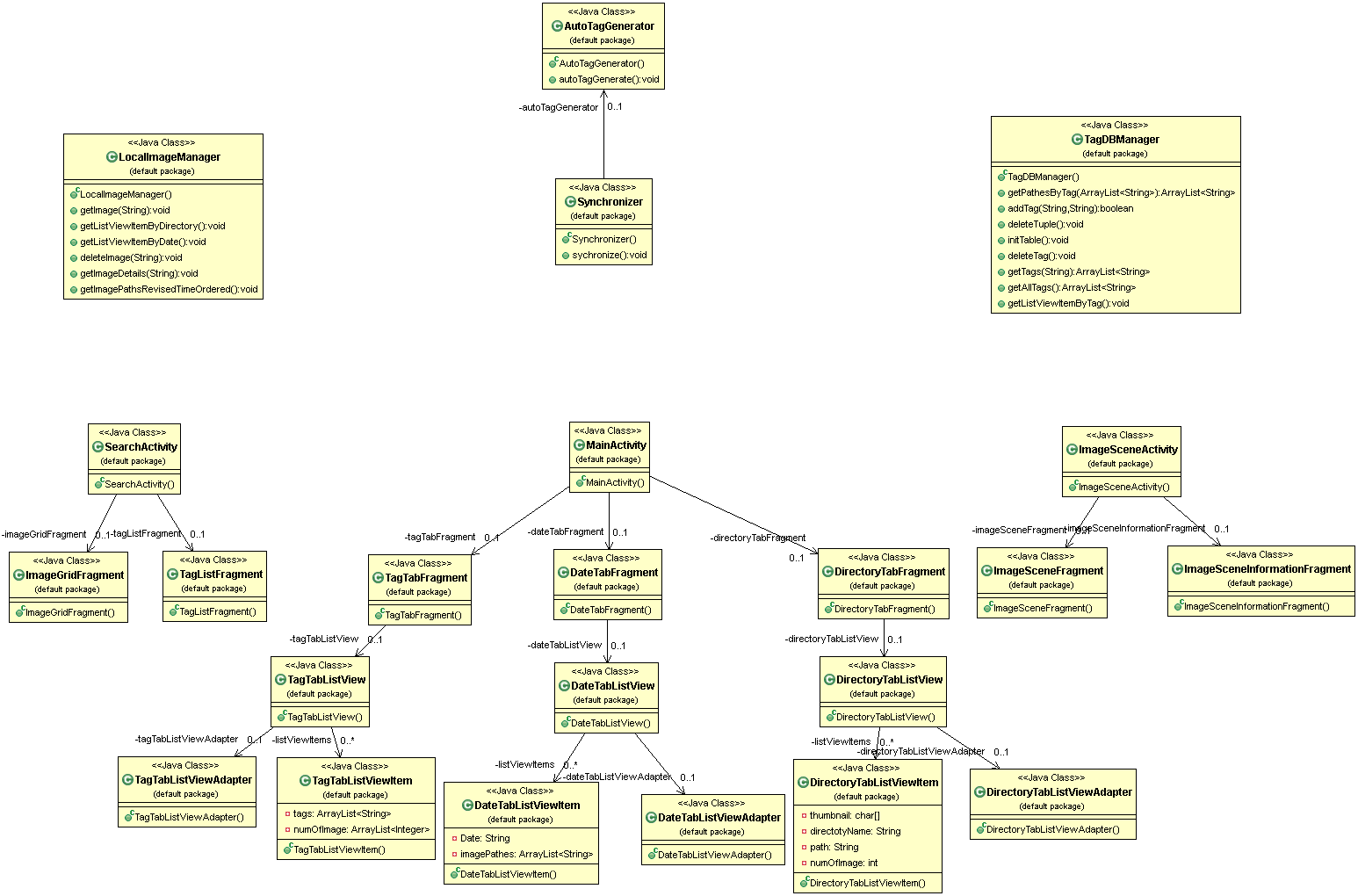
2. This application follow permission policy to Android SDK 22 that is target SDK version.

3. This Application at least require minimum Android SDK version 15.

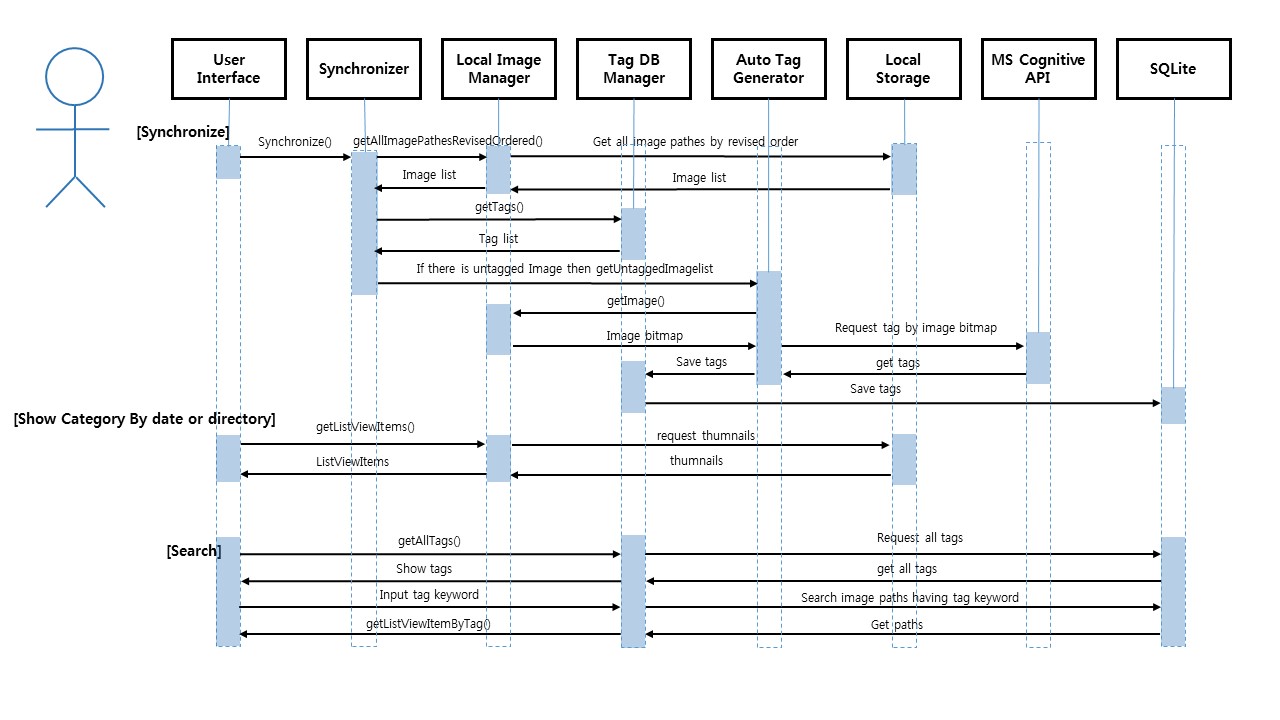
4. Because this application use MS Cognitive API, So the API MUST BE IN NORMAL OPERATION.

# Diagrams

## 2.1 Class diagrams



## 2.2 Sequence diagram

****

# Process

## 3.1 Risk assessment

Likelihood : frequent / infrequent , Severity : low / medium / high

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| No | Activity | Hazard | Risk for whom | Likelihood | Severity | Possible measures |
| 1 | User takes lots of photo. | The application requires synchronizing which needs some time, so users could feel uncomfortable. | user | frequent | low |  |
| 2 | The application needs synchronizing. | Data fee will cost too much unless user uses Wi-Fi. | user | frequent | medium | Recommend using Wi-Fi. |
| 3 | User has too many photos, so synchronize more than 5000 times. | MS cognitive API supply 5000 times free tagging. If user has more than 5000 photos, extra fee will cost. | developer | infrequent | high |  |
| 4 | User want to add more tags to a photo. | Only 30 tags can be tagged because we limited it. So if there exist 30 tags, no more tags can be added. | user | infrequent | low | Recommend deleting unnecessary tags. |
| 5 | User fail to find expected photo. | The tag user searched cannot tagged in a photo. So user cannot find the photo expected. | user | infrequent | low | Recommend search different tag. |
| 6 | The application fail to tagging a photo. | The application is very dependent on MS cognitive API server, so stability of MS server and connection is important. | developer | infrequent | medium | Refresh application or check network |
| 7 | User cannot satisfied with the tag that automatically tagged. | We use MS cognitive API, so we must depend on the accuracy of API. | developer | infrequent | high |  |

## 3.2 Project schedule

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Functions | | | 1st  (03.21~04.01) | | 2nd  (04.02~04.17) | | 3rd  (04.17~04.26) | | 4th  (04.27~05.05) | |
| Design System Architecture | | Idea Develop |  |  |  |  |  |  |  |  |
| Service Scenario |  |  |  |  |  |  |  |  |
| UI Design |  |  |  |  |  |  |  |  |
| Use-case |  |  |  |  |  |  |  |  |
| Scrum | Select | Scrum Meeting |  |  |  |  |  |  |  |  |
| Develop | Coding |  |  |  |  |  |  |  |  |
| Refactoring |  |  |  |  |  |  |  |  |
| Review | Client Meeting |  |  |  |  |  |  |  |  |
| Project closure | | Documentation |  |  |  |  |  |  |  |  |
| All members  Front-End Developer  Back-End Developer | | | | | | | | | | |

Priory, complete the basic functions

1. Load images and automatically tagging
2. Search a tag and show thumbnail image

After complete prior function

1. Sort and show images at main
2. Functions that related search and tag
3. Zoom out / in images in own image view, Show tags and tools in own image view

5/12 zero feature, 6/2 beta

## 3.3 Team structure

Backend : 이원준, 명세교 (Pair programming)

Frontend : 강소연(Search Activity), 이예원(ImageSceneActivity) ,박건후(MainActivity)

## 3.4 Test plan

|  |  |  |
| --- | --- | --- |
| Activity | Frame | Function Specification |
| Main | DirectoryTab | When users click a directory tab icon or swipe the tab, then show DirectoryTab frame. This frame will sort your photos by directory on your device. This frame also shows thumbnail of the directory, directory name, path and a number of images. |
| DateTab | When users click a date tab icon or swipe the tab, then show DateTab frame. This frame will sort your photos by date on your device. This frame also shows day and path. |
| TagTab | When users click a tag tab icon or swipe the tab, then show TagTab frame. This frame will sort your photos by tag on your device. This frame also shows tag and a number of images. |
| SearchIcon | When users click a search icon on the top of tool bar, then go to SearchActivity. |
| SettingIcon | When users click a setting icon on the top of tool bar, then show menu frame. |
| UISettingFrame | This frame appears when SettingIcon is clicked. This frame also shows the setting functions such as manual, synchronization and service center. |
| SearchActivity | ImageGrid | This frame appears when SearchIcon is clicked on MainActivity. When user clicks one of the images, then go to ImageScene. |
| TagList | This frame appears when SearchIcon is clicked on ImageGrid Frame. This frame will show sorted tags in order of appearance. |
| ImageScene | ImageScene | This frame shows closure images. When user clicks the image scene, then display will jump to ImageSceneInformation frame of ImageScene Activity. |
| ImageSceneInformation | This frame appears when touch event is occurs on ImageScene frame of ImageScene Activity. All tags will appear below the image. When these tags are clicked, display will jump to the ImageGrid frame of SearchActivity and show the photos that have those tags. |
| ImageSettingFrame | This frame appears when SettingIcon is clicked. This frame also shows the setting functions such as share, delete photo, add tags and specific information. |

.

## 3.5 Documentation plan

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Documents | Purpose | Audience | Output media | Writer | Start | Completion |
| Project proposal | Propose the project that would be a term project | All students in class and the professor | ppt | SoYeon, Yewon | 170313 | 170314 |
| Software requirement specification | Establish a solid definition of the project | the Professor | docx | SoYeon, Yewon, GunHoo, SeKyo, WonJun | 170328 | 170405 |
| Software design specification | Describe all the details about the project | the Professor | docx | SoYeon, Yewon, GunHoo, SeKyo, WonJun | 170414 | 170426 |
| Interim presentation | Present the summary of requirements and planning | All students in class and the professor | ppt | SoYeon, Yewon | 170414 | 170426 |
| Zero feature | Describe the detail about the initial version of the application | the Professor | docx | SoYeon, Yewon, GunHoo, SeKyo, WonJun | 170427 | 170512 |
| Beta | Describe the detail about the revised version of the application | the Professor | docx | SoYeon, Yewon, GunHoo, SeKyo, WonJun | 170513 | 170602 |
| Final presentation | Present final version of the application | All students in class and the professor | ppt | SoYeon, Yewon | 170513 | 170602 |
| Individual report  (#1 ~ #6) | Show individual progress in the project | the Professor | docx | individual team member | every week | |

## 3.6 Coding style guidelines

**Use Javadoc Standard Comments**

Every file should have a copyright statement at the top, followed by package and import statements (each block separated by a blank line) and finally the class or interface declaration. In the Javadoc comments, describe what the class or interface does.

Every class and nontrivial public method you write must contain a Javadoc comment with at least one sentence describing what the class or method does. This sentence should start with a third person descriptive verb.

### Write Short Methods

When feasible, keep methods small and focused. We recognize that long methods are sometimes appropriate, so no hard limit is placed on method length. If a method exceeds 40 lines or so, think about whether it can be broken up without harming the structure of the program.

### Define Fields in Standard Places

Define fields either at the top of the file or immediately before the methods that use them.

### Limit Variable Scope

Keep the scope of local variables to a minimum. By doing so, you increase the readability and maintainability of your code and reduce the likelihood of error. Each variable should be declared in the innermost block that encloses all uses of the variable.

Local variables should be declared at the point they are first used. Nearly every local variable declaration should contain an initializer. If you don't yet have enough information to initialize a variable sensibly, postpone the declaration until you do.

### Order Import Statements

The ordering of import statements is:

1. Android imports
2. Imports from third parties (com, junit, net, org)
3. java and javax

### Follow Field Naming Conventions

* 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

### Use Standard Brace Style

Braces do not go on their own line; they go on the same line as the code before them

We require braces around the statements for a conditional. Exception: If the entire conditional (the condition and the body) fit on one line, you may (but are not obligated to) put it all on one line.

### Limit Line Length

Each line of text in your code should be at most 100 characters long. While much discussion has surrounded this rule, the decision remains that 100 characters is the maximum with the following exceptions:

* If a comment line contains an example command or a literal URL longer than 100 characters, that line may be longer than 100 characters for ease of cut and paste.
* Import lines can go over the limit because humans rarely see them (this also simplifies tool writing).

### Use Standard Java Annotations

Annotations should precede other modifiers for the same language element. Simple marker annotations (e.g. @Override) can be listed on the same line with the language element. If there are multiple annotations, or parameterized annotations, they should each be listed one-per-line in alphabetical order.

### Use TODO Comments

Use TODO comments for code that is temporary, a short-term solution, or good-enough but not perfect. TODOs should include the string TODO in all caps, followed by a colon:

(reference : https://source.android.com/source/code-style)