Android Code Review Checklist

IMPLEMENTATION

- Does the code follow the project’s architectural patterns?
- Does the code follow the project’s coding style guide?
- Is there any code that becomes obsolete after the code changes and therefore can be deleted?

ERROR HANDLING AND LOGGING

- Are the error messages being displayed user-friendly?
- Are remote non-fatal errors being logged?
- Should any code be separated into multiple lines for easier stack trace analysis?
- What happens when there’s no network connection?
- What happens when an API call fails?
- What happens when there’s a slow network connection?
- What happens when the device is rotated?
- What happens if some permissions that the feature requires are denied?
- What happens if a user inputs something unexpected?
- What happens when there are edge cases in the data?

USABILITY AND ACCESSIBILITY

Check if the code changes involve UI changes.

- If you have designs provided by a designer, does the UI match them?
- Are the correct view classes being used?
- Do the views and styles follow the company’s design guidelines?
- Are you using fonts and colors from the company’s branding?
- How does the feature look on a small device?
- How does the feature look on a tablet, if supported?
- How does it look in dark mode, if supported?
- How does the feature look with non-default font and display sizes?
- Does the XML code support usability and accessibility?

TESTING AND TESTABILITY

- Is the code testable?
- Can classes that were added or modified be easily mocked?
- If the project uses UI testing, should any new UI tests be added and existing ones updated?
- Are there any tests that become irrelevant or inaccurate after the code changes and should be deleted or updated?

PERFORMANCE

- Are threads used correctly?
- Is there object allocation or other code in frequently-called functions (onDraw(), onBindViewHolder(), etc.) that can be moved somewhere else?
- Are there any API responses that can be cached, in order to avoid high data usage and battery drain?
- Are there any API requests that can be done less frequently?

DEPENDENCIES

Check if the code changes involve adding new dependencies on external libraries or bumping an existing dependency by a major version.

- How necessary is it? If the project only needs a small piece of the library’s functionality, would it be better to reimplement it yourself instead?
- How much does it add to the APK size and is the size increase worth it?
- How widely is the library used in the Android community and is it actively maintained? Who’s maintaining it?
- Does Google recommend using the library or an alternative?
- If there’s a version bump, does the new version introduce any breaking changes?
- If the project uses Proguard or some other code obfuscator, does an obfuscated build of the app still work as expected or do the obfuscation rules need to be updated?

READABILITY

- Are there any places where lambda parameters should be declared explicitly?.
- Are resource annotations (@StringRes, @ColorInt, etc.) used where appropriate?
- Do the code changes follow the project’s naming conventions?
- Are there any places where using named function arguments would be helpful?
- Are there any places where explicit types would be helpful even if the compiler doesn’t require them?
- Are there any places where explicit types are used unnecessarily?
- Are there comments that should follow documentation engine formats like Javadoc or Dokka?
- Are there any comments that are no longer accurate after the code changes and should be updated?

See a deep dive into the checklist at
medium.com/@sherryyuan

See also Michaela Greiler's code review checklist