MOBILE: COMMON MISTAKES & SOLUTIONS

AGENDA

» iOS: No No 😑

» iOS: Solutions 😋

» Android: No No 😑

» Android: Solutions 😋

» Android Architecture Components

» Worth Reading

» Homework

iOS: NO NO 😑

» MVC aka Massive View Controller?

» Base Bean anti-pattern (BaseViewController)

» Heavy storyboards

» ...

IOS: SOLUTIONS 😋

» Use MVVM to avoid Massive View Controllers

» Extract logic from ViewController to Util/Helper classes (if not UI related)

» Use extensions instead of Inheritance

» Use Coordinators

» 1 VC per storyboard

» Protocol Oriented Programming WWDC, Realm

ANDROID: NO NO 😑

» BaseActivity

» getActivity() with class cast directly in fragments :)

» EventBus for sending data between Activities/ Fragments

» Sending data to newInstance() without saving it to Fragment's arguments

» Updating UI from success/failure callbacks when app is not active

ANDROID: SOLUTIONS 😋

» Use Lifecycle callbacks solution

» Use annotations, e.g. @ActivityLayout

» Use Listeners and implementaions, class cast in onAttach

» Always save Parcelable objects to fragment's arguments

» EventBus can be used, but NOT overused

» Use MVVM and LiveData updates

ANDROID ARCHITECTURE COMPONENTS

COMPONENTS

» Room

» ViewModel

» LiveData

» Lifecycle

» ...

» Source


Android App Architecture

IDEAS:

» MessageHelper should be standalone class

» Logger should be standalone class

» Network layer structure? Normally we should support switching between platforms

» Extensions instead of subclasses (Swift, Kotlin)

» Localization handling (iOS)?

» Struct for Model classes (Swift)

WORTH READING:

» Fantastic iOS Architecture

» Fantastic Android Architecture

» iOS Architecture Patterns

» VIPER

» SOLID principles

HOMEWORK

» Task: show github public repos for the user

» ⬇️

» iOS: Implement MVVM in basic UIViewController with UITableView + Network layer

» Android: Implement MVVM based on Android Architecture Components