Fragments, ViewModel, Notifications

Android Lecture 3

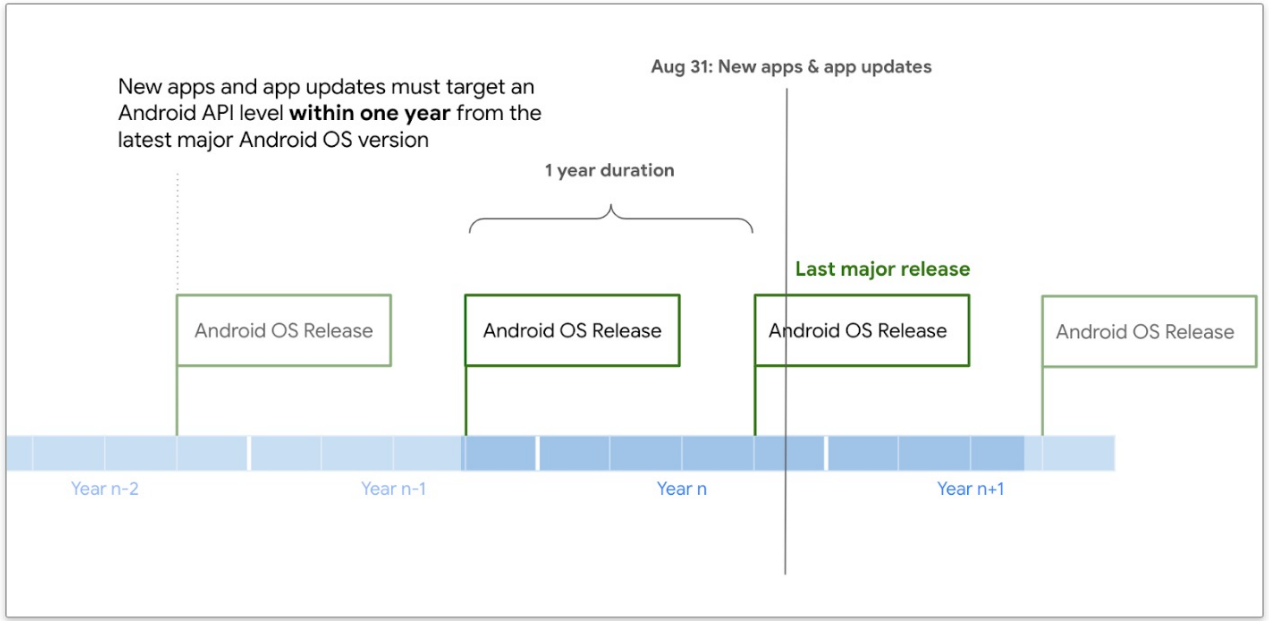
2023

Lukáš Prokop
Simona Kurňavová

Agenda

- **Fragments**
  - Inflating
  - Lifecycle
- **ViewModel**
- **Notifications**

App update requirements


New apps and app updates must target an Android API level **within one year** from the latest major Android OS version
Aug 31: New apps & app updates
1 year duration
Last major release
Android OS Release
Android OS Release
Android OS Release
Android OS Release
Year n-2
Year n-1
Year n
Year n+1

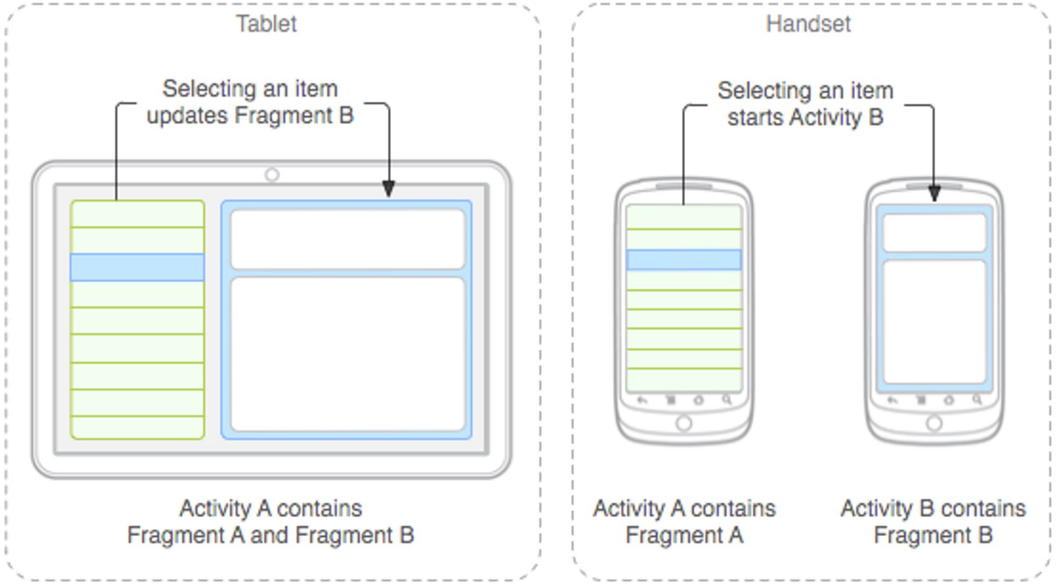
Android OS version (API level)	When are new app and app update submissions required to target this API level?	
	New apps	App updates
Android 13 (API level 33)*	August 31, 2023	August 31, 2023
Android 12 (API level 31)	August 31, 2022	November 1, 2022

https://support.googl

Fragment

Fragment

- Simplify create UI for phones and tablets
- ~~android.app.Fragment~~ - Added API 11, deprecated API 28
- ~~android.support.v4.app.Fragment~~ - replaced by jetpack
- androidx.fragment.app.Fragment - jetpack


Tablet
Selecting an item updates Fragment B
Activity A contains Fragment A and Fragment B
Handset
Selecting an item starts Activity B
Activity A contains Fragment A
Activity B contains Fragment B

Fragment - add statically

activity_main.xml

```xml
<?xml version="1.0" encoding="utf-8"?>
<FrameLayout xmlns:android="http://schemas.android.com/apk/res/android"
   xmlns:tools="http://schemas.android.com/tools"
   android:layout_width="match_parent"
   android:layout_height="match_parent">

   <androidx.fragment.app.FragmentContainerView
       android:id="@+id/fragment_id"
       android:tag="some_string"
       android:name="packagename.class"
       android:layout_width="match_parent"
       android:layout_height="match_parent" />

</FrameLayout>
```

Fragment - Activity.kt

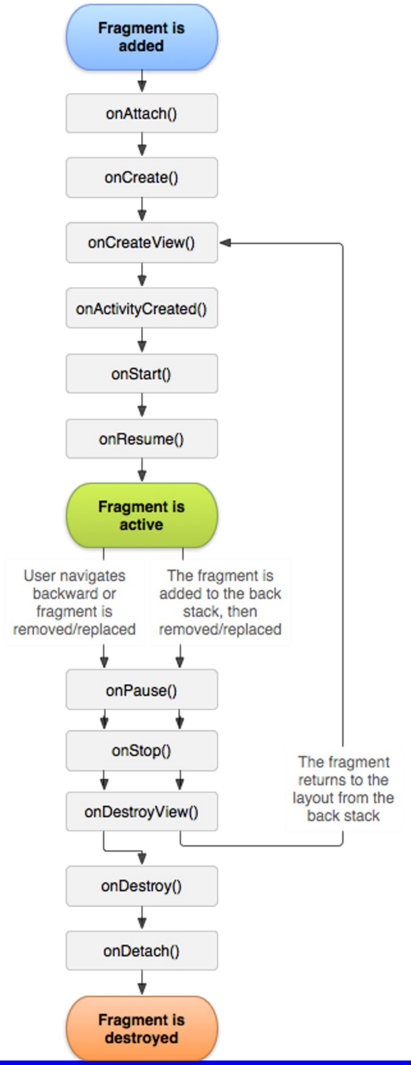
```kotlin
override fun onCreate(savedInstanceState: Bundle?) {
   super.onCreate(savedInstanceState)
   setContentView(R.layout.activity_user)

   supportFragmentManager.beginTransaction()
            .add(R.id.fragment_container, MyFragment.newInstance())
            .addToBackStack(null)
            .commit()
}
```

Gen

Fragment - states

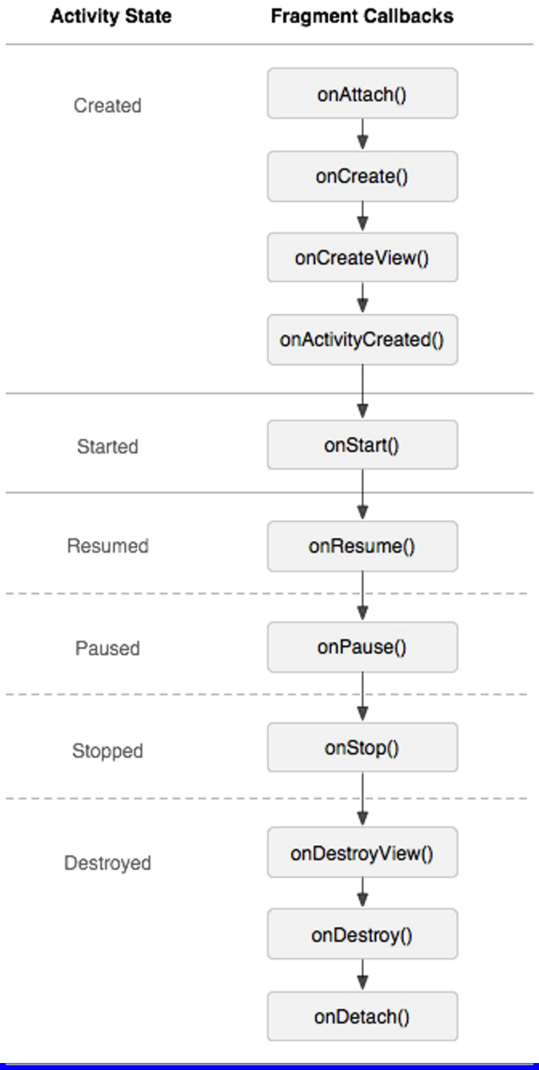
- **Is in same state as host activity**
- **Resumed**
  - Fragment is visible in the running activity
- **Paused**
  - Another activity in foreground, but hosting activity is still visible
- **Stopped**
  - Fragment is not visible
  - Hosting activity is stopped or fragment is removed from the activity, but added to backstack.
  - Alive, can be killed by hosting activity

Fragment - lifecycle



Gen

Fragment inside Activity lifecycle



Fragment - lifecycle

- **onAttach(Activity)**
  - Fragment is associated with the activity
  - Set activity as a listener
- **onCreate(Bundle)**
  - Initial creation of fragment
  - Process fragment extras
  - Not called when fragment is retained across Activity re-creation
- **onCreateView(LayoutInflater, ViewGroup, Bundle)**
  - Called when view hierarchy needs to be created

Fragment - lifecycle

- **onViewCreated(View, Bundle?)**
  - Activity onCreate() completed and view associated with fragment
  - Get references to views
- **onViewStateRestored(Bundle)**
  - All saved state of the view hierarchy was restored
- **onStart()**
  - Fragment visible to user (same as Activity.onStart())
- **onResume()**
  - Fragment interact with user (based on hosting container)
  - Same as Activity.onResume()

Fragment - lifecycle

- **onPause()**
  - Not interact with user anymore
  - Paused activity or fragment manipulation
- **onStop()**
  - No longer visible
  - Stopped activity or fragment manipulation
- **onDestroyView()**
  - Disconnect fragment from view hierarchy created in onCreateView()
- **onDestroy**
  - Fragment going to be destroyed
  - Cleanup all resources
  - Not called for retained fragments
- **onDetach**
  - Detach fragment from activity
  - Remove activity listeners

Headless fragment

- Fragment without UI
- Often retained fragment
- `Fragment#onCreateView()` returns null

Fragment and Activity

- Fragment is not working without activity
- Activity can call fragment methods directly
- Fragment defines interface to be implemented by Activity to handle fragment requirements

Fragment - passing data

```
class DemoFragment: Fragment() {

    companion object {
        @JvmStatic
        fun newInstance(username: String): DemoFragment {
            val fragment = DemoFragment()
            fragment.arguments = bundleOf(
                "Username" to username,
                "id" to 1001
            )

            return fragment
        }
    }
}
```

- Android calls non-params constructor when restoring fragments
- Constructor with parameters will not be called

Gen

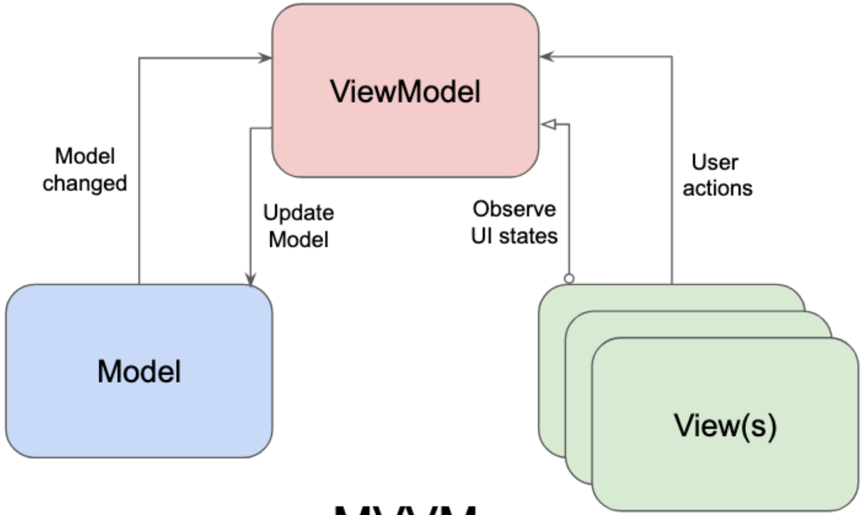
Exercise

1. Inflate LoginFragment in LoginActivity statically
2. Inflate UserFragment in UserActivity dynamicall

ViewModel

ViewModel

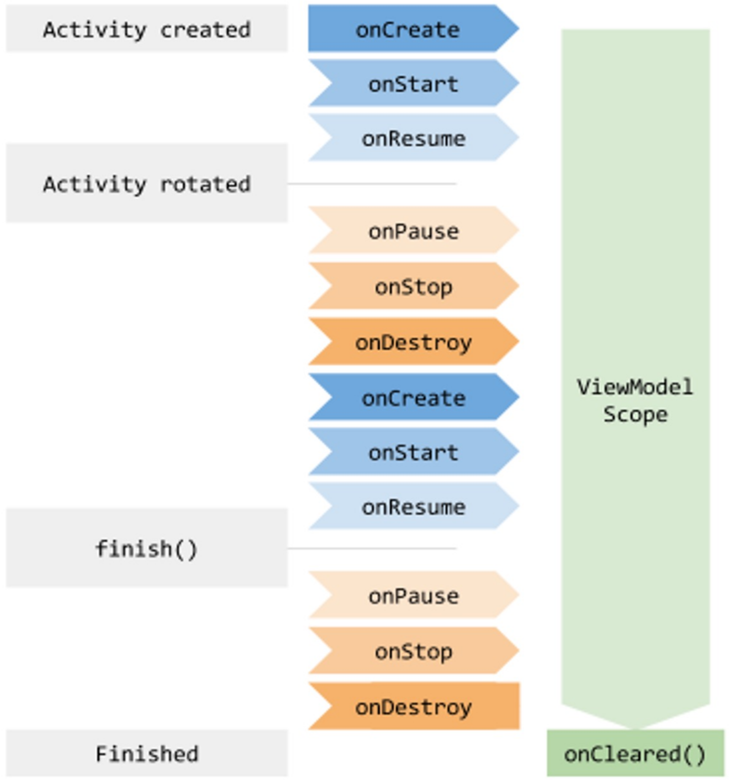
- Encapsulates and provides business logic for UI (activity, fragment)
- Lifecycle-aware
- Automatically retained during configuration change
- Part of Android Jetpack
- Documentation
- **Should never hold reference to View, Lifecycle or activity context (even transitively)**



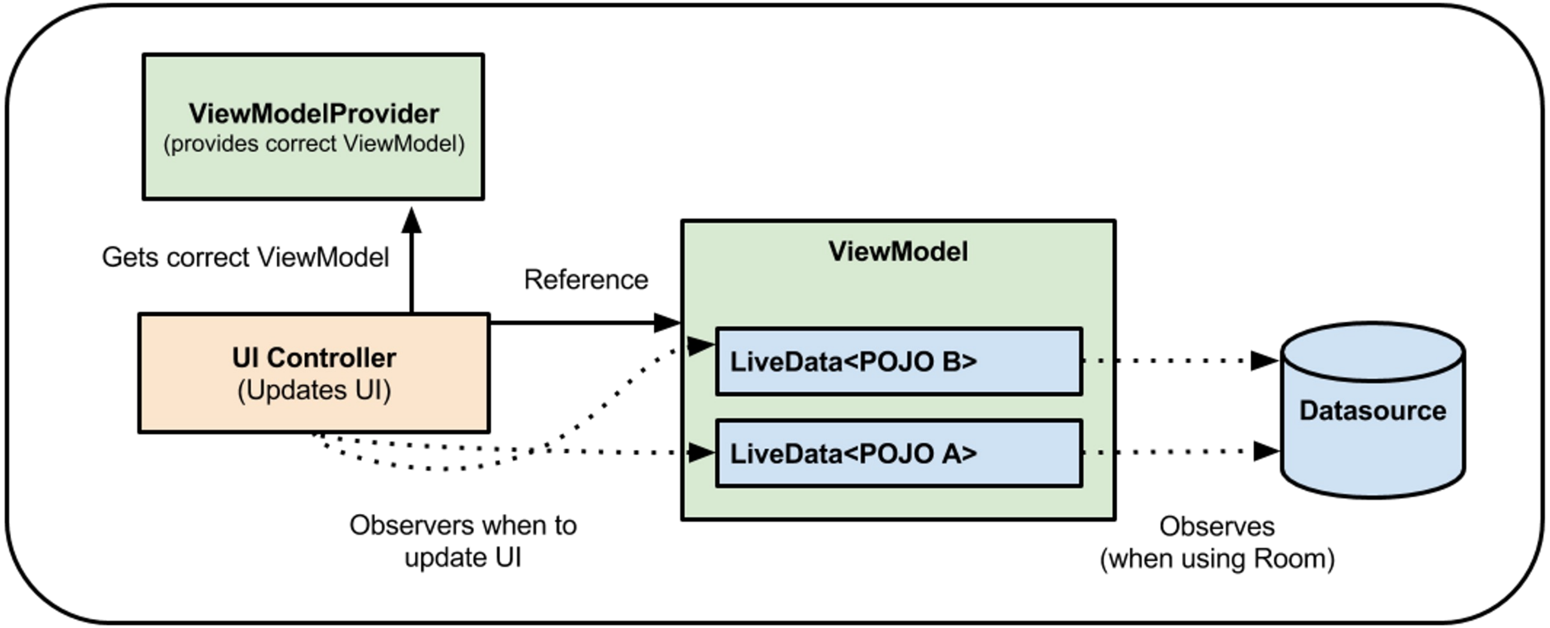
MVVM

ViewModel - Lifecycle

- **Illustration of lifecycle when activity is rotated**
- **onCleared():** Called when ViewModel is going to be destroyed
  - *For Activity*: on finish
  - *For Fragment*: on detach



ViewModel - Architecture



ViewModel - obtain

```kotlin
class DemoViewModel() : ViewModel() {
   private val _data: MutableLiveData<String> = MutableLiveData()
   fun getData(): LiveData<String> = _data
}


class DemoFragment() : Fragment() {

   val demoViewModel: DemoViewModel by viewModels()
}
```

ViewModelProvider.AndroidViewModelFactory

- **Creates instances of ViewModel**
- **Default implementation uses**
  - non-param constructor
  - Constructor which accept Application as the only parameter
- **Often required with dependency injection framework**

LiveData

- Observable data holder class
- Lifecycle aware component
- Provides information about data changes to the observer

```
class DemoFragment() : Fragment() {

    val demoViewModel: DemoViewModel by viewModels()

    override fun onStart() {
        super.onStart()
        demoViewModel.getData().observe(viewLifecycleOwner) {data: String ->

        }
    }
}
```

Gen

Demo time

- Fetching user data using view model

Strict mode

Strict mode

- **Developer tool**
- **Detects application bad behaviour**

```
if (DEVELOPER_MODE) {
    StrictMode.setThreadPolicy(StrictMode.ThreadPolicy.Builder()
            .detectDiskReads()
            .detectDiskWrites()
            .detectNetwork()   // or .detectAll() for all detectable problems
            .penaltyLog()
            .build())
    StrictMode.setVmPolicy(new StrictMode.VmPolicy.Builder()
            .detectLeakedSqlLiteObjects()
            .detectLeakedClosableObjects()
            .penaltyLog()
            .penaltyDeath()
            .build())
}
```

PendingIntent

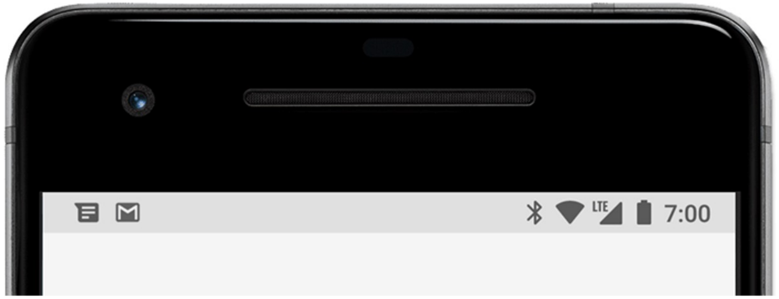
- **Reference to a token describing the original Intent**
- **Possible to use even if the owning application is killed**
- **Used when the orignal Intent is needed later (Notification, Scheduler) for starting Activity, Service or sending Broadcast**

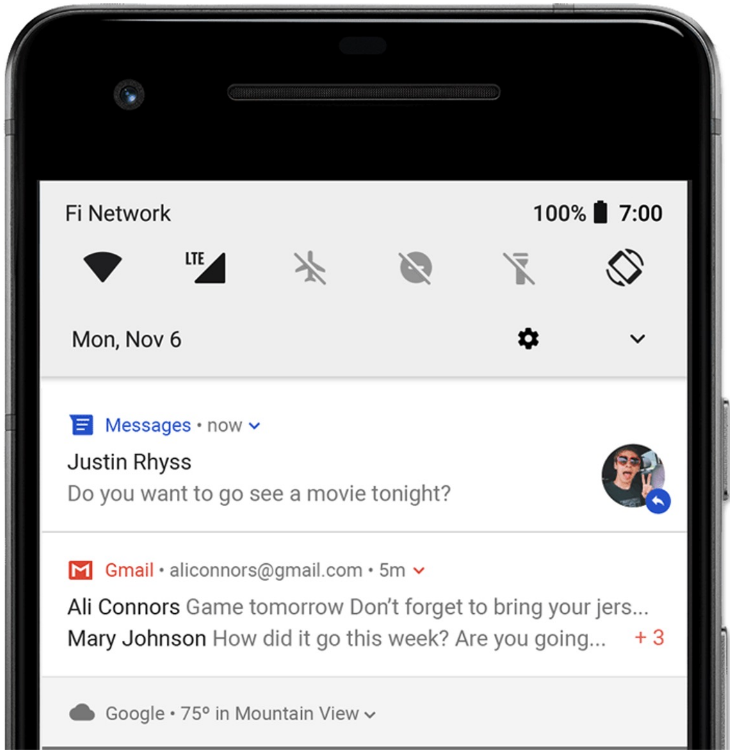
Gen

Notifications

Notifications

- **Notify user, when they are not using your app**
- **Mandatory**
    - Small icon
    - Content title
    - Content text
    - Notification channel - Android 8.0+





Fi Network 100% 7:00

Mon, Nov 6

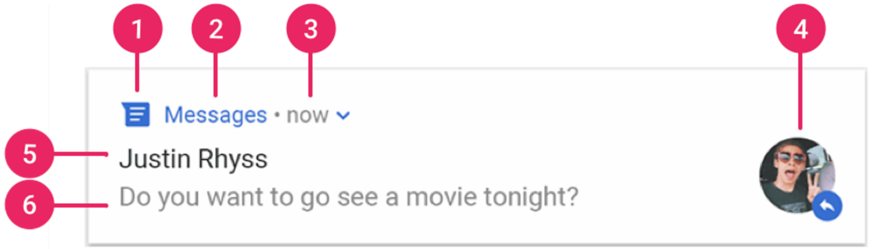
Messages • now
Justin Rhyss
Do you want to go see a movie tonight?

Gmail • aliconnors@gmail.com • 5m
Ali Connors Game tomorrow Don't forget to bring your jers...
Mary Johnson How did it go this week? Are you going... + 3

Google • 75° in Mountain View

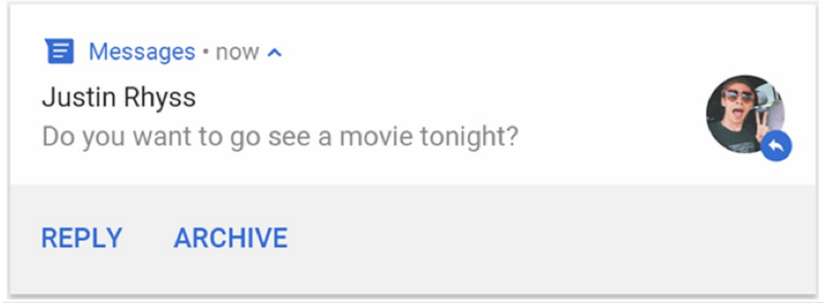
Notification

1. **Small icon - mandatory parameter**
2. **App name - provided by system**
3. **Time stamp: provided by system,**
   - override by setWhen()
   - Hide setShowWhen(false)
4. **Large icon - optional**
5. **Title - optional**
6. **Text - optional**



Notification actions

- Quick actions
- Inline reply action Android 7.0+

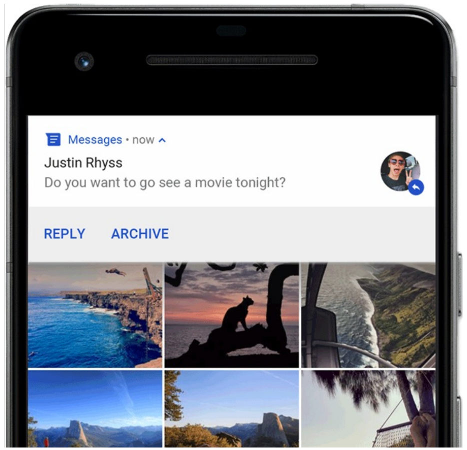

Messages • now

Justin Rhyss

Do you want to go see a movie tonight?

Heads-up notifications

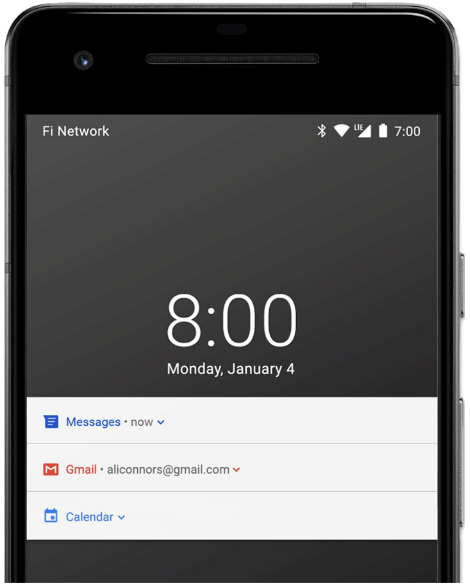
- **Heads-up notifications**
  - Small floating window
  - Shows action buttons to handle action without leaving app
  - Only for high priority notifications
    - If the user’s activity is in full screen mode (app uses fullScreenIntent)
    - Notification has high priority and uses ringtones or vibrations
- **Android 5.0+**



Justin Rhyss
Do you want to go see a movie tonight?

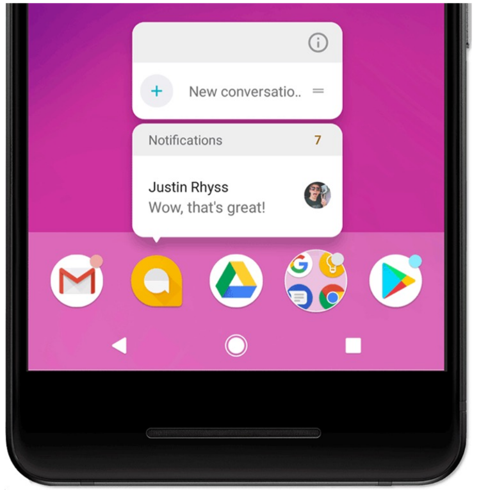
Lock screen notifications

- **Possibility to set which informations when device is locked**
  - VISIBILITY_PUBLIC - Shows full content
  - VISIBILITY_SECRET - Do not show at all
  - VISIBILITY_PRIVATE - Show icon and title, hide content
- **Android 5.0+**



Notifications dots

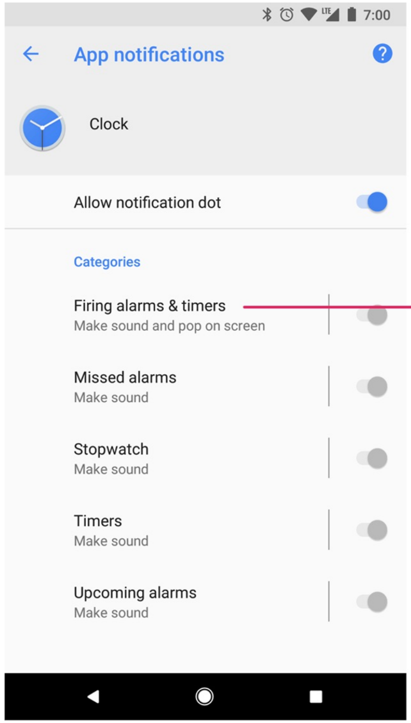
- Dots on launcher app icons
- Shows notification content on long click
- Android 8.0+



Gen

Notification channels

- Notification categories
- User can manage notifications in single category
- Override DnD
- System show number of deleted channels
- Android 8.0+



7:00

App notifications

Clock

Allow notification dot

Categories

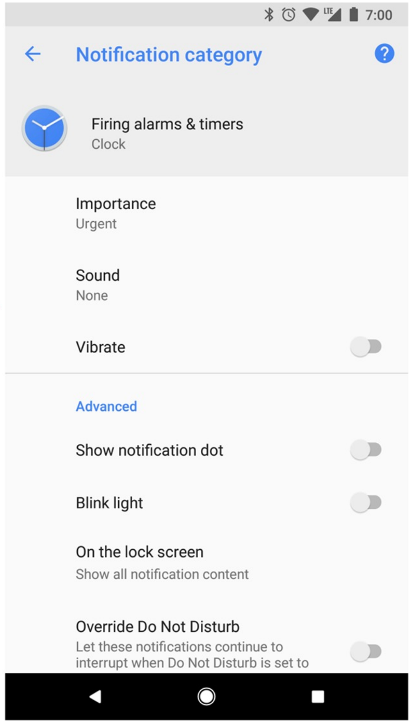
Firing alarms & timers
Make sound and pop on screen

Missed alarms
Make sound

Stopwatch
Make sound

Timers
Make sound

Upcoming alarms
Make sound



7:00

Notification category

Firing alarms & timers
Clock

Importance
Urgent

Sound
None

Vibrate

Advanced

Show notification dot

Blink light

On the lock screen
Show all notification content

Override Do Not Disturb
Let these notifications continue to interrupt when Do Not Disturb is set to

Gen

Key classes

- android.app.Notification
- ~~android.support.v4.app.NotificationCompat~~
  - ~~Action~~
  - ~~Builder~~
  - ~~*Style~~
- androidx.core.app.NotificationCompat
  - Action
  - Builder
  - *Style
- android.app.NotificationManager
- ~~android.support.v4.app.NotificationManagerCompat~~
- androidx.core.app.NotificationManagerCompat
- android.app.NotificationChannel

Notification

- Use `NotificationCompat.Builder`
  - Handles compatibility
- `NotificationManagerCompat.notify(int id, Notification)`
- Possible to set pending intent for actions
- Priority affects position in drawer
- Developer responsibility to handle navigation when user opens application from notification

Create notification

```
var builder = NotificationCompat.Builder(this, CHANNEL_ID)
        .setSmallIcon(R.drawable.notification_icon)
        .setContentTitle("My notification")
        .setContentText("Much longer text that cannot fit one line...")
        .setStyle(NotificationCompat.BigTextStyle()
                .bigText("Much longer text that cannot fit one line..."))
        .setPriority(NotificationCompat.PRIORITY_DEFAULT)
```

Create channel

```kotlin
private fun createNotificationChannel() {
    // Create the NotificationChannel, but only on API 26+ because
    // the NotificationChannel class is new and not in the support library
    if (Build.VERSION.SDK_INT >= Build.VERSION_CODES.O) {
        val name = getString(R.string.channel_name)
        val descriptionText = getString(R.string.channel_description)
        val importance = NotificationManager.IMPORTANCE_DEFAULT
        val channel = NotificationChannel(CHANNEL_ID, name, importance).apply {
            description = descriptionText
        }
        // Register the channel within the system
        val notificationManager: NotificationManager =
            getSystemService(Context.NOTIFICATION_SERVICE) as NotificationManager
        notificationManager.createNotificationChannel(channel)
    }
}
```

Setup notification action

```
// Create an explicit intent for an Activity in your app
val intent = Intent(this, AlertDetails::class.java).apply {
    flags = Intent.FLAG_ACTIVITY_NEW_TASK or Intent.FLAG_ACTIVITY_CLEAR_TASK
}
// Create the TaskStackBuilder
val resultPendingIntent: PendingIntent? = TaskStackBuilder.create(this).run {
    // Add the intent, which inflates the back stack
    addNextIntentWithParentStack(intent)
    // Get the PendingIntent containing the entire back stack
    getPendingIntent(0, PendingIntent.FLAG_UPDATE_CURRENT)
}
val builder = NotificationCompat.Builder(this, CHANNEL_ID)
        ....
        // Set the intent that will fire when the user taps the notification
        .setContentIntent(pendingIntent)
        .setAutoCancel(true)
```

Fire notification

```
with(NotificationManagerCompat.from(this)) {
    // notificationId is a unique int for each notification that you must define
    notify(notificationId, mBuilder.build())
}
```

Notifications - Android 12

- **No longer possible to control full content of notification**
- https://developer.android.com/about/versions/12/behavior-changes-12#custom-notifications

Notifications - Android 13

- **Requires runtime permission - POST_NOTIFICATIONS**
- **Foreground services still available in Foreground service task manager, but not visible in notification drawer**
- **Exemptions**
  - Media sessions
  - App for managing phone calls

Thank you

Lukáš Prokop
Simona Kurňavová

Lukas.Prokop@gendigital.com
Simona.Kurnavova@gendigital.com

Gen™