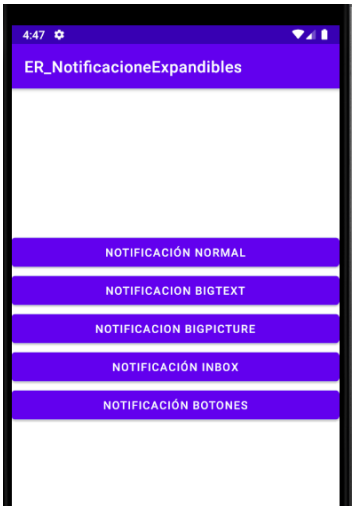
Bloque 12 . Ejercicio Resuelto Notificaciones Expandibles

Descargar este ejercicio

Vamos a realizar una aplicación Android en la que podrá pulsar diferentes botones, que mostraran las notificaciones vistas en la teoría. Esta aplicación tendrá dos categorías, llamadas General y Publicidad, las dos primeras pertenecerán a la categoría Publicidad, mientras que las tres últimas pertenecerán a General. La notificación normal mostrará la notificación sin ningún estilo y avisará con un sonido, también deberá abrir la propia aplicación cuando se elimine la notificación.



Primero crearemos los canales de notificación con un procedimiento de forma parecida a como vimos en los apuntes:

```
private fun crearCanal(
        idCanal: String,
        nombreCanal: String,
        descripcion: String,
        importancia: Int
    ): NotificationChannel? {
        if (Build.VERSION.SDK_INT >= Build.VERSION_CODES.O) {
            val canal = NotificationChannel(idCanal, nombreCanal, importancia)
            canal.description = descripcion
            return canal
        }
        return null
}
```

Antes de crear la notificación deberemos llamar al procedimiento tantas veces como necesitesmos y con los datos necesarios, en este caso se ha decidio crear el General con importancia por

defecto y el Publicidad con importancia alta.

```
private fun crearCananlesNotificacion() {
        if (Build.VERSION.SDK_INT >= Build.VERSION_CODES.O) {
            val notificationManager = getSystemService(
                NotificationManager::class.java
            )
             var canal =  crearCanal(
                CHANNELPUBLICIDAD_ID,
                "Publicidad",
                "Canal de Publicidad",
                NotificationManager.IMPORTANCE_HIGH
            )
            notificationManager.createNotificationChannel(canal!!)
            canal =  crearCanal(
                CHANNELGENERAL_ID,
                "General",
                "Canal General",
                NotificationManager.IMPORTANCE_DEFAULT
            )
            notificationManager.createNotificationChannel(canal!!)
        }
}
```

Creamos una notificación base como hicimos en los apuntes a partir de la cuál iremos añadiendo los estilos:

```
fun notificacionBasica(canal:String): NotificationCompat.Builder {
        val notificacion: NotificationCompat.Builder
        notificacion = NotificationCompat.Builder(applicationContext, canal)
        notificacion.setContentText("Este es el texto de mi notificación")
        notificacion.setContentTitle("Mi notificación")
        notificacion.setSmallIcon(R.mipmap.ic_launcher)
        notificacion.setLargeIcon(
            (ContextCompat.getDrawable(
                this,
                android.R.drawable.sym_action_email
            ) as BitmapDrawable?)!!.bitmap
        )
        notificacion.setTicker("Optional ticker")
        notificacion.setWhen(System.currentTimeMillis())
        notificacion.setAutoCancel(true)
        val intent = Intent(this@MainActivity, MainActivity::class.java)
        notificacion.setDeleteIntent(PendingIntent.getActivity(this@MainActivity, 0,
                                    intent, 0))
        return notificacion
}
```

En el método escuchador de los `Click` interceptaremos la pulsación y crearemos las diferentes notificaciones con sus estilos propios, cada una la identificaremos con un id numérico según la posición que ocupe el botón.
El código para la notificación normal y la de texto largo podría ser como el siguiente:

```
@RequiresApi(Build.VERSION_CODES.M)
    override fun onClick(p0: View?) {
        val notificationManager = NotificationManagerCompat.from(this)
        when(p0?.id)
        {
            binding.button1.id ->
            {
                var notificacion=notificacionBasica(CHANNELPUBLICIDAD_ID)
                val defaultSound = RingtoneManager.getDefaultUri(
                                        RingtoneManager.TYPE_NOTIFICATION)
                notificacion.setSound(defaultSound)
                notificationManager.notify(0, notificacion.build())
            }
            binding.button2.id -> {
                var notificacion=notificacionBasica(CHANNELPUBLICIDAD_ID)
                val pattern = longArrayOf(1000, 500, 1000)
                notificacion.setContentTitle("BigText")
                notificacion.setLargeIcon(
                    (ContextCompat.getDrawable(
                        this,
                        android.R.drawable.ic_btn_speak_now
                    ) as BitmapDrawable?)!!.bitmap
                )
                notificacion.setVibrate(pattern)
                //Vibravión por defecto
                notificacion.setDefaults(Notification.DEFAULT_VIBRATE)
                NotificationCompat.BigTextStyle(notificacion)
                    .bigText("Lorem ipsum dolor sit amet, consectetur
                    adipiscing elit. Donec id ex hendrerit, ullamcorper lacus
                    quis, auctor odio. Curabitur in feugiat elit, ut laoreet
                     ex. Nunc in tristique elit, at posuere magna.
                     Curabitur augue lectus, tristique a consectetur vitae,
                     luctus a nisi. Quisque nec nulla neque. Proin gravida
                     lacinia lorem in tempus. Interdum et malesuada fames ac
                     ante ipsum primis in faucibus.")
                    .setBigContentTitle("Expanded BigText title")
                    .setSummaryText("Summary text")
                notificationManager.notify(2, notificacion.build())
             }
             //...
        }
}
```

La notificación de imagen expandible se lanzará también como hemos visto en los apuntes:

```
binding.button3.id -> {
                var notificacion=notificacionBasica(CHANNELGENERAL_ID)
                notificacion.setContentTitle("BigPicture")
                notificacion.setLargeIcon(
                    (ContextCompat.getDrawable(
                        this,
                        android.R.drawable.ic_dialog_map
                    ) as BitmapDrawable?)!!.bitmap
                )
                NotificationCompat.BigPictureStyle(notificacion)
                    .bigPicture(
                        ContextCompat.getDrawable(
                            this,
                            R.drawable.ic_launcher_foreground
                        )?.toBitmap()
                    )
                    .bigLargeIcon(
                        ContextCompat.getDrawable(
                            this,
                            R.drawable.ic_launcher_foreground
                        )?.toBitmap()
                    )
                    .setBigContentTitle("Expanded BigPicture title")
                    .setSummaryText("Summary text")
                   notificationManager.notify(7, notificacion.build())
            }
```

En cuanto para la notificación Multilínea el código será el siguiente: Añade las líneas cogiéndolas de un array, siempre que la notificación ya exista, en caso de que sea la primera vez que se crea la notificación lo que se hace es crearla y añadir la primera línea.

```
        binding.button4.id -> {
            val nManager = getSystemService(
                NotificationManager::class.java
            )
            //Limpiar la notificación en el caso que se haya leído o eliminado
            var existe = false
            var notification = Notification()
            //Comprobar si existen notificaciones
            // de ese tipo y guardarla en ese caso
            for (x in nManager.getActiveNotifications()) {
                if (x.id == 3) {
                    notification = x.notification
                    existe = true
                }
            }
            if (!existe) {
                lin = 0
                notificacionInboxStyle=null
            } else if (notificacionInboxStyle != null) {
                lin++
                //lines es una array de strings
                notificacionInboxStyle?.addLine(lines.get(lin))
            }
            if (notificacionInboxStyle == null) {
                var notificacion=notificacionBasica(CHANNELGENERAL_ID)
                notificacion.setContentTitle("MultiLineas")
                notificacion.setLargeIcon(
                    (ContextCompat
                        .getDrawable(
                            this,
                            android.R.drawable.ic_dialog_email
                        ) as BitmapDrawable?)!!.bitmap
                )
               val notificacionInboxStyle = NotificationCompat
                    .InboxStyle(notificacion)
                    .setBigContentTitle("Expanded Inbox title")
                    .setSummaryText("Summary text")
                if (existe) {
                    //extraer la clave que nos interesa de
                    //los extras de las notificaciones
                    val claveLineas = notification.extras
                                      .keySet()
                                      .toTypedArray()[2]
                                      .toString()
                    //guardar el número de notificaciones de ese tipo
                    //que se han lanzado
                    val mensajes = notification.extras
                                   .getCharSequenceArray(claveLineas)
                    //añadir los mensajes a la notificación
                    lin = mensajes!!.size
                    for (l in mensajes) notificacionInboxStyle.addLine(l)
```

```
                }
                notificacionInboxStyle.addLine(lines.get(lin))
                this.notificacionInboxStyle=notificacionInboxStyle
            }
            //notify requiere inmutable, por lo tanto hay que asegurarlo con let
            notificationManager.notify(3, notificacionInboxStyle.let
                                        { it?.build()}!!)
        }
```

Un código a destacar, sería el siguiente, en el que extraemos la información sobre las notificaciones que están activas. En caso que no exista la notificación de multilineas (3) porque se ha eliminado o se ha mostrado, dejaremos una variable a false. Y en caso que no exista iniciamos a null y a 0 la notificacion y las líneas respectivamente, para que se inicien los mensajes desde 0. También nos guardaremos la notificación:

```
var existe = false
var notification = Notification()
//Comprobar si existen notificaciones de ese tipo y guardarla en ese caso
for (x in nManager.getActiveNotifications()) {
        if (x.id == 3) {
            notification = x.notification
            existe = true
            }
}
if (!existe){
                lin = 0
                notificacionInboxStyle=null
            }
```

La otra parte interesante, es en la que recuperamos los mensajes de una notificación que esta activa para poder agregarlos a la nueva notificación que lanzará la aplicación una vez haya sido abierta de nuevo. Esto se haría para el caso en que exista notificaciones activas pero la aplicación ha sido cerrada, en ese caso se deberá recuperar la información de la notificación para poder construirla correctamente una vez se vuelva a pulsar el botón. La recuperamos a través de un bundle que aplicamos a la propiedad extras, para ello utilizamos KeySet que nos devuelve la clave que necesitamos:

```
if (existe)
{
    //extraer la clave que nos interesa de los extras de las notificaciones
    val claveLineas = notification.extras.keySet().toTypedArray()[2].toString()
    //guardar el número de notificaciones de ese tipo, que se han lanzado
    val mensajes = notification.extras.getCharSequenceArray(claveLineas)
    //añadir los mensajes a la notificación
    lin = mensajes!!.size
    for (l in mensajes) notificacionInboxStyle.addLine(l)
}
notificacionInboxStyle.addLine(lines.get(lin))
```

Para acabar crearemos la notificación de botones, de igual manera que hemos comentado en los apuntes:

```
binding.button5.id -> {
                var notificacion=notificacionBasica(CHANNELGENERAL_ID)
                notificacion.setContentTitle("Botones")
                notificacion.setLargeIcon(
                    (ContextCompat.getDrawable(
                        this,
                        android.R.drawable.ic_dialog_alert
                    ) as BitmapDrawable?)!!.bitmap
                )
                val i = Intent(Intent.ACTION_VIEW)
                i.data = Uri.parse("http://www.iesdoctorbalmis.com")
                val action = NotificationCompat.Action.Builder(
                    android.R.drawable.ic_menu_delete,
                    "Delete", PendingIntent.getActivity(this@MainActivity, 0, i, 0)
                ).build()
                notificacion.addAction(action)
                notificacion.addAction(
                    android.R.drawable.ic_menu_share,
                    "Share",
                    PendingIntent.getActivity(this@MainActivity, 0, i, 0)
                )
                notificationManager.notify(4, notificacion.build())
            }
```