Software Practice 3

- Today’s lecture
- Today’s Task

Prof. Hwansoo Han
T.A. Jeonghwan Park  43
NOTIFICATION
Notification

- A message that Android displays outside your app’s UI to provide the user with reminders, communication from other people, or other timely information from your app.
Notification Anatomy

1. Small icon: This is required and set with `setSmallIcon()`
2. App name: This is provided by the system
3. Time stamp: This is provided by the system but you can override with `setWhen()` or hide it with `setShowWhen(false)`
4. Largeicon: This is optional (usually used only for contact photos; do not use it for your app icon) and set with `setLargeIcon()`
5. Title: This is optional and set with `setContentTitle()`
6. Text: This is optional and set with `setContentText()`
Notification Channel

- All notifications must be assigned to a channel or it will not appear.
- By categorizing notifications into channels for your app, and users can control the visual and auditory options for each channel.
Importance of Notification

- **Urgent**: Makes a sound and appears as a heads-up notification
- **High**: Makes a sound
- **Medium**: No sound
- **Low**: No sound and does not appear in the status bar
Toast

- One of notification types
- Only used when showing string message without any additional action
- Example of toast message
  
  ```java
  Toast.makeText(getApplicationContext(),  // application context
                  "sending message...",  // toast message to be shown
  Toast.LENGTH_LONG  // life time of toast message
  ).show()
  ```
EXAMPLE
Service & Notification

- Most notifications are generated when users are outside your app’s UI
- For sending notification from this situation, notification generator should be implemented by the combination with Service
PendingIntent

- Common Intent is used when starting Activity or Service from currently running ApplicationContext
- PendingIntent enables an Activity or Service to be executed without application currently running
- Example of usage

```java
Intent intent = new Intent(this, AlertDetails.class);
PendingIntent pendingIntent = PendingIntent.getActivity(this, 0, intent, 0);
```
Three Examples of PendingIntent

- When users make some actions through Notifications
  - NotificationManager starts Intent

- When users make some actions with AppWidget
  - Home screen, which is managed by Android system, starts Intent

- At certain point in the future
  - AlarmManager of Android system starts Intent
Results of Example
# Results of Example

<table>
<thead>
<tr>
<th>Android</th>
<th>100%</th>
<th>4:45</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Thu, May 24</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

- NotificationExample
  - Notification counter
    - # of requested messages: 0

- Android System
  - USB debugging connected
    - Tap to disable USB debugging.

- Android System
  - Virtual SD card
    - For transferring photos and media

- Android System • USB charging this device
  - CLEAR ALL

---

<table>
<thead>
<tr>
<th>Android</th>
<th>100%</th>
<th>4:45</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Thu, May 24</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

- NotificationExample
  - Notification counter
    - # of requested messages: 1

- Android System
  - USB debugging connected
    - Tap to disable USB debugging.

- Android System
  - Virtual SD card
    - For transferring photos and media

- Android System • USB charging this device
  - CLEAR ALL

---

<table>
<thead>
<tr>
<th>Android</th>
<th>100%</th>
<th>4:45</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Thu, May 24</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

- NotificationExample
  - Notification counter
    - # of requested messages: 2

- Android System
  - USB debugging connected
    - Tap to disable USB debugging.

- Android System
  - Virtual SD card
    - For transferring photos and media

- Android System • USB charging this device
  - CLEAR ALL
Notice the content of a file every 10 seconds

- File path: “wow.txt” in cache directory (getCacheDir)
- 1 EditText Component for getting the content of the file
- 1 Button Component for submitting data to write the content of EditText to the file
  - Submit button is not responsible to preserve the previous content of the file, which means that you can refresh all content in every click
Lab – Practice #12

[Images of three screens showing UI elements related to starting and ending services, with text on the screens indicating interaction with the UI.]
Hey Jude, nice to meet you!
[Lab – Practice #12]
Appendix

- There are several types of notifications
  - tap, direct reply, button, etc
- You can adapt one of them to your application with the examples represented in the following post
  - https://developer.android.com/training/notify-user/build-notification