ProgressBar, DialogFragment & WebView

IN721: Mobile Application Development

Kaiako: Grayson Orr

Today’s Content

- ProgressBar
- Fragment
- DialogFragment
- WebView

Preparation

- Use your practical from last time
- Open your practical in Android Studio
- In strings.xml, change the app name & add the following string resource:

```
<string name="please_wait">Please Wait&#8230;</string>
```

- Create a class called ProgressBar
- Create a class called RateUsDialogFragment which extends DialogFragment
- Copy IDataReceived.kt into interfaces directory
- Copy ic_star_white_24.xml into the drawables res directory
- Copy menu_main.xml into the menu res directory. Override the current file
- Copy progress_bar.xml & fragment_rate_us.xml into the layout res directory
    - Make sure you look at both layout files in Design view

strings.xml

- Add the following to strings.xml

```
<string name="please_wait">Please Wait&#8230;</string>
<string name="rate_us_title">Rate this app</string>
<string name="rate_us_message">If you enjoy using this app, would \n you take a minute to rate this app?</string>
<string name="rate_us_now">Rate Now</string>
<string name="rate_us_no_thanks">No Thanks</string>
```

ProgressBar

styles.xml

- New style called App.ProgressBar
- Sets the window background to transparent

```
<style name="AppTheme.ProgressBar">
    <item name="android:windowBackground">@android:color/transparent</item>
</style>
```

ProgressBar

- UI elements that indicates the progress of an operation
- Progress bar supports two modes:
  - Indeterminate progress - you do not know how long the operation will take
  - Determinate progress - you want to show that a specific quantity of progress has occurred
- Resource: ProgressBar

ProgressBar.kt

- Two functions - show() & dismiss()

```
class ProgressBar(context: Context) {
    private val inflater: LayoutInflater =
        context.getSystemService(LAYOUT_INFLATER_SERVICE) as LayoutInflater
    private val view: View = inflater.inflate(R.layout.progress_bar, null)
    private val dialog = Dialog(context, R.style.AppTheme_ProgressBar)

    fun show() {
        dialog.setContentView(view)
        dialog.show()
    }

    fun dismiss() {
        dialog.dismiss()
    }
}
```

RawDataAsyncTask.kt

- Pass in a reference to Context
- Create a new instance of ProgressBar & pass in the Context
- Create an override function called onPreExecute()
  - Remember, this is the fist
  - Call the ProgressBar show() function
- In onPostExecute(), call the ProgressBar dismiss() function

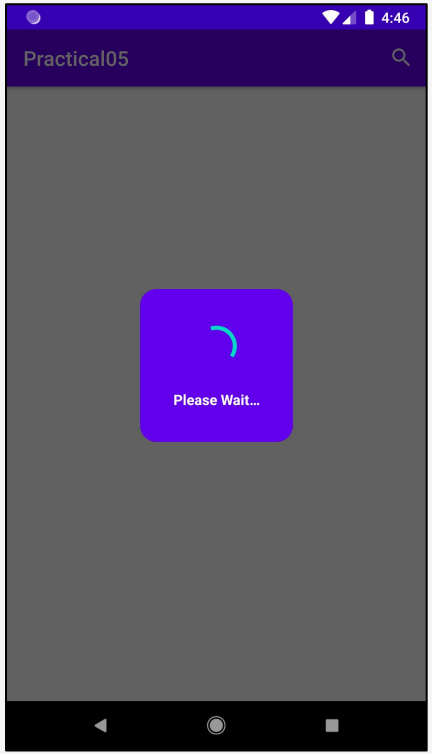
MainActivity.kt

- RawDataAsyncTask now requires two arguments - Listener & Context

```
rawDataAsyncTask = RawDataAsyncTask(this, this@MainActivity)
```

Emulator

- Run app
- Window background is transparent
- You should see a ProgressBar animation



DialogFragment

DialogFragment

- A fragment that displays a dialog window
- Contains a Dialog object - displays based on the fragment's state
- Ensures the Fragment & Dialog states remain consistent
- Watches for events & handles the removal of its own state
- Resource: DialogFragment

RateUsDialogFragment.kt

- Extends DialogFragment
- Reference to IDataReceived.kt
- onCreateView - instantiates it UI view
- onViewCreated - called immediately after onCreateView() has returned
- Resources: onCreateView & onViewCreated

```
class RateUsDialogFragment(private val listener: IDataReceived) : DialogFragment() {
    private lateinit var rateUsNowBtn: Button
    private lateinit var rateUsNoBtn: Button
    private lateinit var rateUsRatingBar: RatingBar

    // override fun onCreateView

    // override fun onViewCreated
}
```

onCreateView

- Set the DialogFragment to be not cancelable
- Return an inflated fragment_rate_us.xml

```
override fun onCreateView(
    inflater: LayoutInflater,
    container: ViewGroup?,
    savedInstanceState: Bundle?
): View? {
    isCancelable = false
    return inflater.inflate(R.layout.fragment_rate_us, container, false)
}
```

onViewCreated

- Describe the following code:

```
override fun onViewCreated(view: View, savedInstanceState: Bundle?) {
    super.onViewCreated(view, savedInstanceState)
    rateUsNowBtn = view.findViewById(R.id.rate_us_now_btn)
    rateUsNoBtn = view.findViewById(R.id.rate_us_no_btn)
    rateUsRatingBar = view.findViewById(R.id.rate_us_rating_bar)

    rateUsNowBtn.isEnabled = false
    rateUsRatingBar.setOnRatingBarChangeListener { _, rating, _ ->
        rateUsNowBtn.isEnabled = rating > 0.0f
    }
    rateUsNoBtn.setOnClickListener { dismiss() }
    rateUsNowBtn.setOnClickListener {
        listener.onDataReceived("Thank you! Your feedback is very helpful for us.")
        dismiss()
    }
}
```

MainActivity.kt

- Implement IDataRecived.kt and its members
- In onDataReceived, create a Toast
  - Pass in data as the CharSequence argument
  - In the practical, you will research how to create a custom Toast
- Create a private function called showDialog
  - This function creates an instance of RateUsDialogFragment
- supportFragmentManager - return the FragmentManager for interacting with fragments

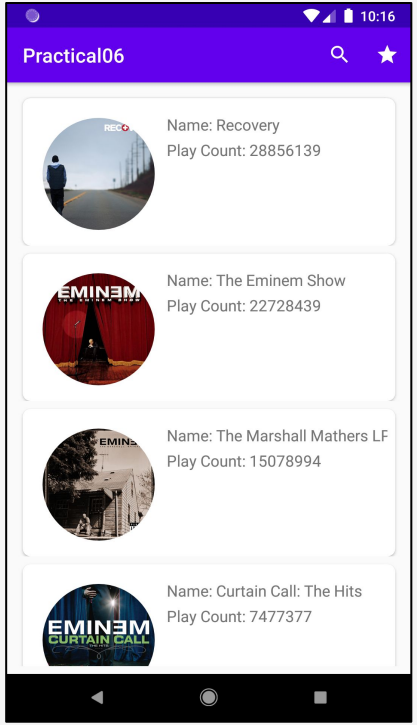
```
private fun showDialog() {
    val dialogFragment = RateUsDialogFragment(this)
    dialogFragment.show(supportFragmentManager, null)
}
```

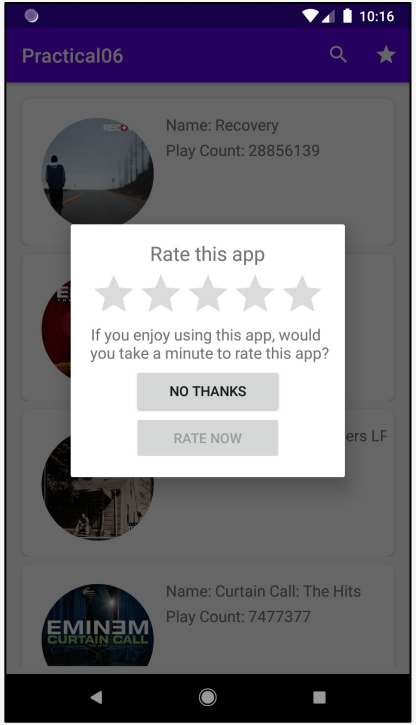
MainActivity.kt

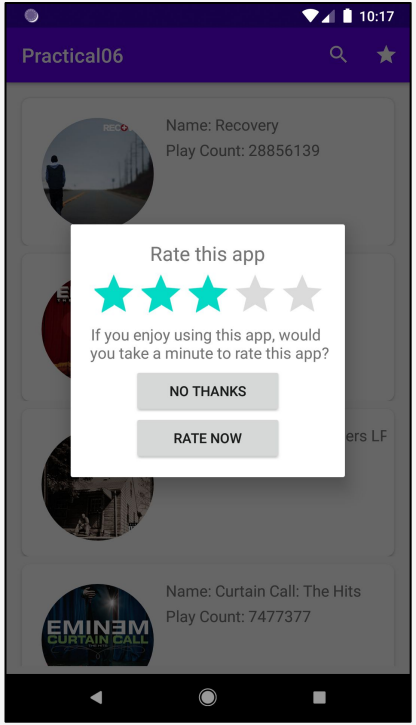
- Add a new menu item id to onOptionsItemSelected
- When the user clicks on the menu item, the DialogFragment will show

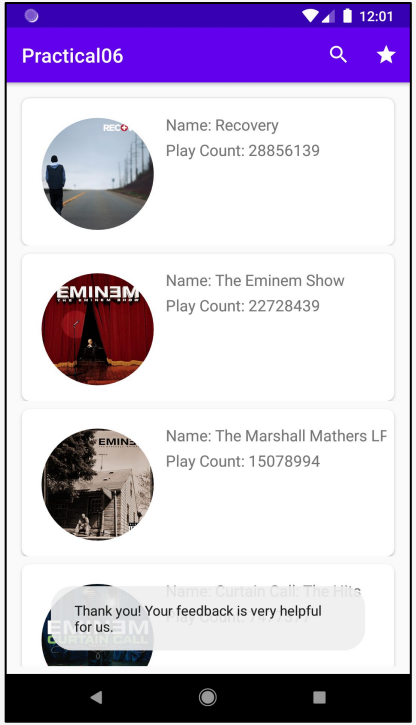
```
override fun onOptionsItemSelected(item: MenuItem): Boolean {
    return when (item.itemId) {
        R.id.action_main -> {
            startActivity(Intent(this@MainActivity, SearchActivity::class.java))
            true
        }
        R.id.action_rate_us -> {
            showDialog()
            true
        }
        else -> super.onOptionsItemSelected(item)
    }
}
```

Emulator


10:16
Practical06
Name: Recovery
Play Count: 28856139
Name: The Eminem Show
Play Count: 22728439
Name: The Marshall Mathers LF
Play Count: 15078994
Name: Curtain Call: The Hits
Play Count: 7477377


10:16
Practical06
Name: Recovery
Play Count: 28856139
Rate this app
If you enjoy using this app, would you take a minute to rate this app?
NO THANKS
RATE NOW
Name: Curtain Call: The Hits
Play Count: 7477377


10:17
Practical06
Name: Recovery
Play Count: 28856139
Rate this app
If you enjoy using this app, would you take a minute to rate this app?
NO THANKS
RATE NOW
Name: Curtain Call: The Hits
Play Count: 7477377


12:01
Practical06
Name: Recovery
Play Count: 28856139
Name: The Eminem Show
Play Count: 22728439
Name: The Marshall Mathers LF
Play Count: 15078994
Thank you! Your feedback is very helpful for us.

WebView

WebView

- Allows you to display web contents as part of your activity layout
- Does not have some features like fully-developed browsers
- Useful when you want to embed webpages, for example, privacy policy

activity_details.xml

- In activity_details.xml, add a WebView under the CardView
- There is no Material Design equivalent

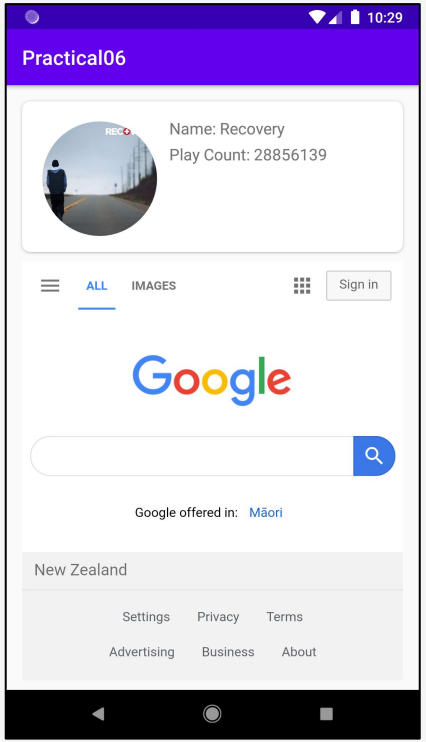
DetailsActivity.kt

- Declare a lateinit var for the WebView
- Find the View by its id in the onCreate()
- Enable JavaScript. What happens if this is not enabled?
- Load the URL. At the moment, it is https://www.google.com

```
albumWebView.settings.javaScriptEnabled = true
albumWebView.loadUrl("https://www.google.com")
```

Emulator

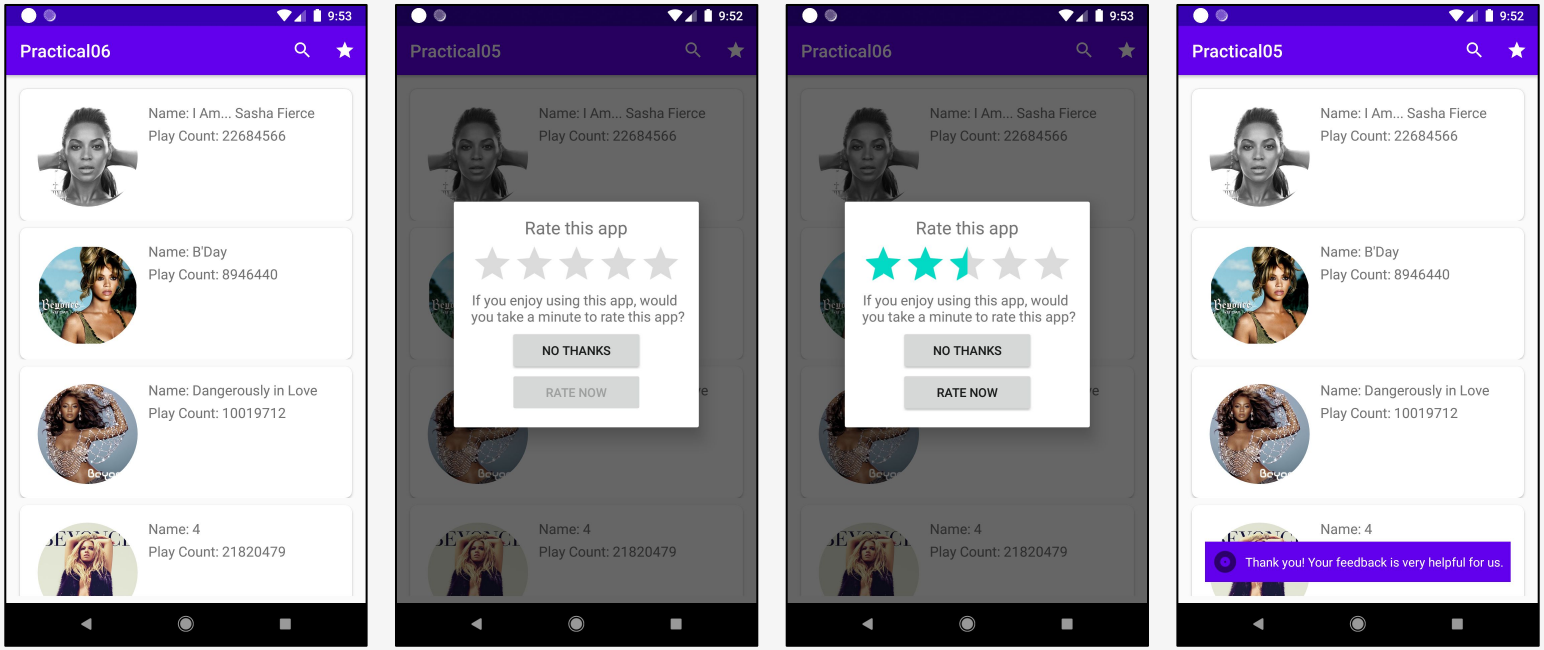
- Run app
- In the practical, you will display the album's URL



Practical

- Please use the current app
- Independent tasks:
  - Implement the code as specified in the previous lecture slides
  - Replace the current Toast in onDataReceived with a custom Toast. This will require you to create a custom layout file. The custom Toast must have an ImageView & TextView. If you are confused about this, refer to the expected output
  - Instead of https://www.google.com, output the album's URL. This will require you to fetch additional data from the API & add a property to Album.kt
  - Once you have completed the practical, create a branch named 04-submission, push the app to the branch, make a pull request & set Grayson-Orr as the reviewer
  - If you do not set Grayson-Orr as a reviewer, I will not mark off your practical
  - DO NOT MERGE YOUR OWN PULL REQUEST!

Expected Output



Expected Output