Android III

JSON, Recycler Views
What we will learn

- JSON Parsing
- Model classes
- What a RecyclerView is and how to build one
  - ViewHolders
  - Adapters
  - LayoutManagers
Simple JSON Parsing

Parsing the JSON on the left

```json
{
    "name": {
        "firstName": "John",
        "lastName": "Doe",
    }
    "title": "Missing Person"
}
```

---

```java
// Initialize JSONObject from JSON string
JSONObject contact = new JSONObject(contact.toJSONString);

// Name into a JSONObject
JSONObject name = contact.getJSONObject("name");

// First and last names
String firstName = name.getString("firstName");
String lastName = name.getString("lastName");

// Title
String title = contact.getString("title");
```
Parsing JSON

- Github Repo Searcher: Our goal is to parse the JSON into an ArrayList of model classes
- Model Class: used to create POJOs containing information
- For us, we will create a Repository model
  - Fields: name, owner, url
- Each Repository interest will contain information to be displayed in an item of our RecyclerView
What is a RecyclerView?

- A flexible view for providing a limited window into a large data set
- It uses smaller views for individual pieces of data (items)
- It mainly manages the recycling of items to avoid wasted time and space
- Replaces ListView
**Using a RecyclerView**

- RecyclerViews need 3 other partners to help:
- ViewHolder: inflates and encapsulates a view item, stores Java references to child views
- Adapter: Creates new Holders with items, populates items with data, returns this to recyclerview
- LayoutManager: Lays out the views in a pattern
<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="wrap_content"
    android:padding="16dp">
    <TextView
        android:id="@+id/tv_item_number"
        style="@style/TextAppearance.AppCompat.Title"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_gravity="center_vertical|start"
        android:fontFamily="monospace"
        android:textSize="42sp"
        tools:text="#42" />
</FrameLayout>

<FrameLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent">
    <TextView
        android:id="@+id/tv_view_holder_instance"
        style="@style/TextAppearance.AppCompat.Title"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_gravity="center_vertical|end"
        android:fontFamily="monospace"
        android:textColor="#000"
        android:textColor="#42"
        tools:text="ViewHolder instance: 7" />
    </FrameLayout>
</FrameLayout>

**The System**

**XML Resources**
- Main
  - `Recyclerview`
- Item
  - Child views
  - `Adapter`
  - `LayoutManager`
    - Actually lays out views
    - We don't code anything here

**Java**
Activity: `onCreate`
- Inflates, get ref. to `RecyclerView`
- Sets adapter to `RecyclerView`
- Sets `LayoutManager` to `RecyclerView`

`Adapter`
- `onCreateViewHolder`
  - Inflate item
  - Put stuff not dependent on data position

`ViewHolder`
- Store item
- Get/store references to children (ViewHolder)
- Other helper methods
- `onBindViewHolder`
  - Bind data (position dependent) to holder
Click Listeners

- These are implementations of an interface that get passed into a view
- The implementations have a “onClick” method that gets called by Android OS when the item is clicked
- The listener pattern is a way to tell an interface what to do by implementing an interface
- A classic example of polymorphism
- We will implement listeners for the items in a Button and in RecyclerView in the live coding demo
Building the System

See in-class live coding demo!