Basic Android Chatting Bot

ADA Alpha 2016-11-26
Baker

Contents

1. How this app works
2. How an Android app works
3. How to make this Android app: steps + notes

1. How this app works

- A server is set up on AWS at `http://54.161.23.101/basicRequest`. We can visit it via either curl or browser.

- You input a message

    -> See your message in the list

    -> server responds -> Server’s message is added to the list

- You can also clear the chatting message history

More detailedly...

When you click on “send”:

- The message display area is updated
- A request is sent to server.
  - When the server responds:
  - The message display area is updated again

2. Basic components of an Android App

- AndroidManifest.xml
- View layout xml files
- Java classes
- Resources: drawables, strings, dimen values, etc

What is XML?

- eXtensible Markup Language
- It is a Data Definition Language (DDL)
- It defines and describes data
- XML has a tree structure:
  - A single line describing version and encoding, and
  - A tree-structured cluster of data, having one single root node

2.1 AndroidManifest.xml

- It is an XML
- Activity (including intent-filter)
- Package
- <uses-permission>

manifest

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="me.ziningzhu.basicchattingbot">
    <uses-permission android:name="android.permission.INTERNET" />
    <application
        android:allowBackup="true"
        android:icon="@mipmap/ic_launcher"
        android:label="BasicChattingBot"
        android:supportsRtl="true"
        android:theme="@style/AppTheme">
        <activity android:name=".MainActivity">
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />

                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>
    </application>

</manifest>
```

2.2 Layout XML files

- Specify how widgets are arranged
- Each node has attributes
- Two must-have attributes: android:layout_width and android:layout_height
- Good to have: android:id

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:id="@+id/activity_main"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical"
    android:paddingBottom="16dp"
    android:paddingLeft="64dp"
    android:paddingRight="64dp"
    android:paddingTop="16dp"
    tools:context="me.ziningzhu.basicchattingbot.MainActivity">


    <ListView
        android:id="@+id/my_list_view"
        android:scrollbars="vertical"
        android:layout_width="match_parent"
        android:layout_height="@dimen/listView_max_height"
        />

    <LinearLayout...>

    <Button
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:id="@+id/clear_messages_button"
        android:text="Clear Message History"/>
</LinearLayout>
```

XML: two must-have attributes

android:layout_width

android:layout_height

```
<Button
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:id="@+id/clear_messages_button"
    android:text="@string/clear_string"/>
```

Each can be either of:

- WRAP_CONTENT
- MATCH_PARENT
- Fixed value (link to a dimension. See next slide for linking)

```
<ListView
    android:id="@+id/my_list_view"
    android:scrollbars="vertical"
    android:layout_width="match_parent"
    android:layout_height="@dimen/listView_max_height"
    />
```

XML: How to link attribute: dimension

Link a dimension:

- “@dimen/value_name” in activity_main.xml
- Specify this resource in res/values/dimens.xml

```
<ListView
    android:id="@+id/my_list_view"
    android:scrollbars="vertical"
    android:layout_width="match_parent"
    android:layout_height="@dimen/listView_max_height"
    />
```

dimens.xml

resources dimen

```
<resources>
    <!-- Default screen margins, per the Android Design guidelines. -->
    <dimen name="activity_horizontal_margin">16dp</dimen>
    <dimen name="activity_vertical_margin">16dp</dimen>
    <dimen name="listView_max_height">230dp</dimen>
</resources>
```

XML: How to link attribute: string

Link a resource:

- “@string/clear_string” in activity_main.xml, and
- specify string resource in res/values/string.xml

```
<Button
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:id="@+id/clear_messages_button"
    android:text="Clear Message History"/>
```

```
<Button
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:id="@+id/clear_messages_button"
    android:text="@string/clear_string"/>
```

```
<resources>
    <string name="app_name">BasicChattingBot</string>
    <string name="send_string">Send</string>
    <string name="clear_string">Clear Message History</string>
</resources>
```

XML: How to link attribute: id

Link an id:

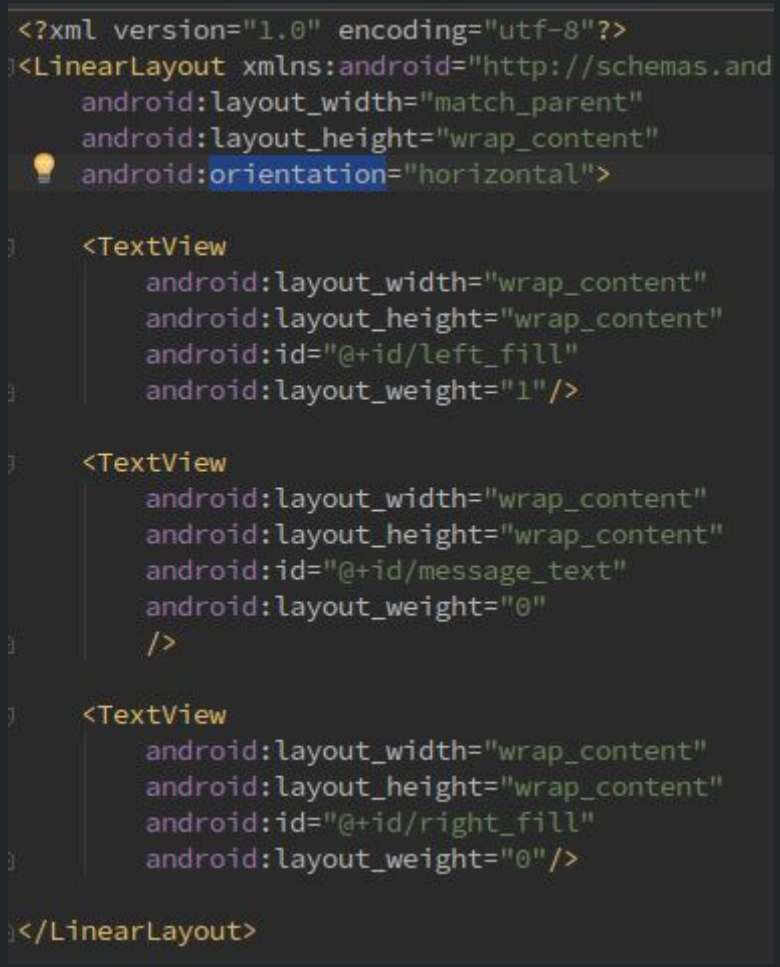
- “@+id/id_name” in activity_main.xml
- Plus sign means create new
- findViewById(R.id.id_name) in MainActivity.java

```
<Button
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:id="@+id/clear_messages_button"
    android:text="@string/clear_string"/>
```

```
mClearButton = (Button)findViewById(R.id.clear_messages_button);
mClearButton.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {
        cleanUpPreviousMessages();
    }
});
```

Layout XML - overall

- Root node of a Layout XML must be a Layout
- LinearLayout, RelativeLayout, GridLayout
- LinearLayout: two more attributes:
  - android:orientation
  - android:layout_weight in children
  - Additional layout spaces distributed proportional to layout_weight



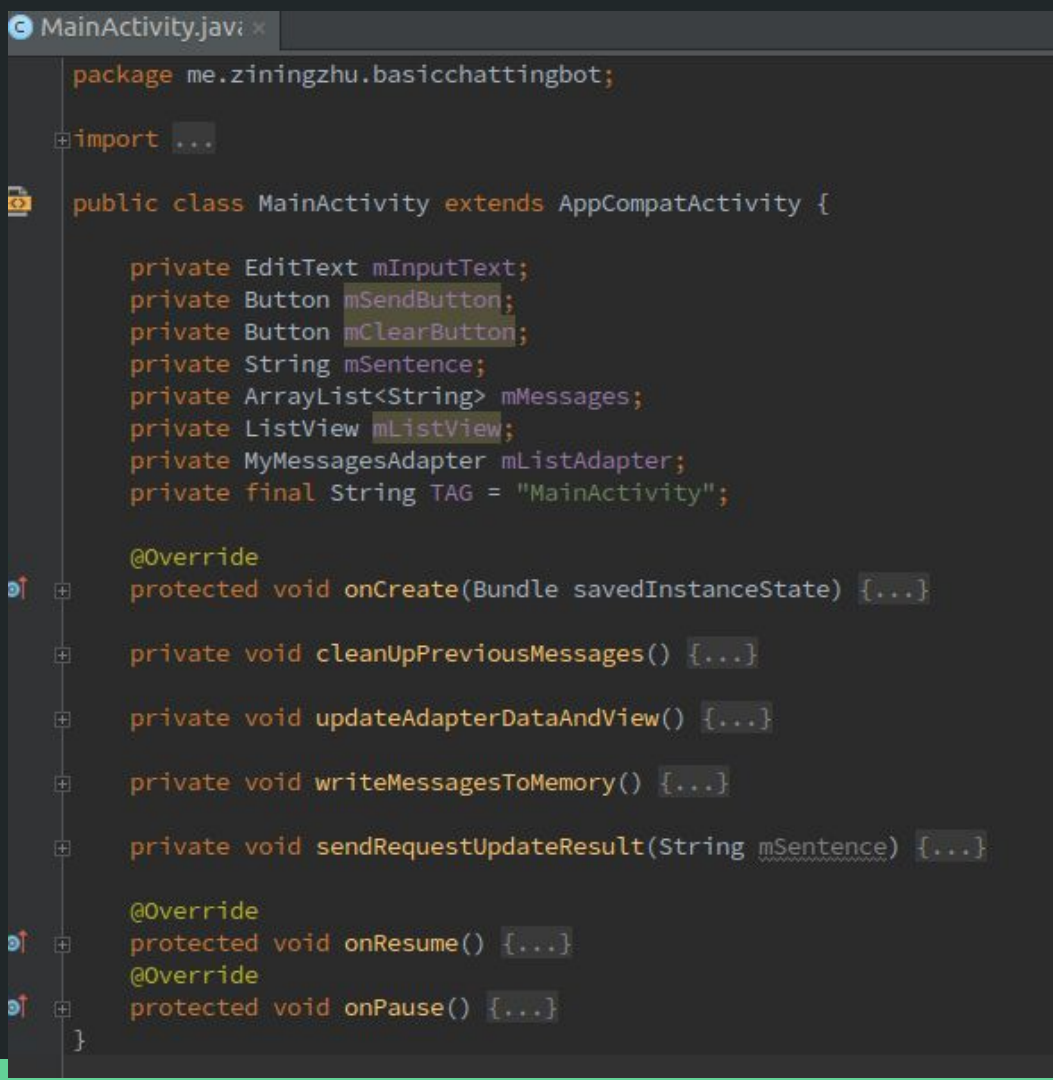
2.3 Java classes

MainActivity.java

- Each Activity must have a class
- Each phase in the Activity's Lifecycle must have a method.

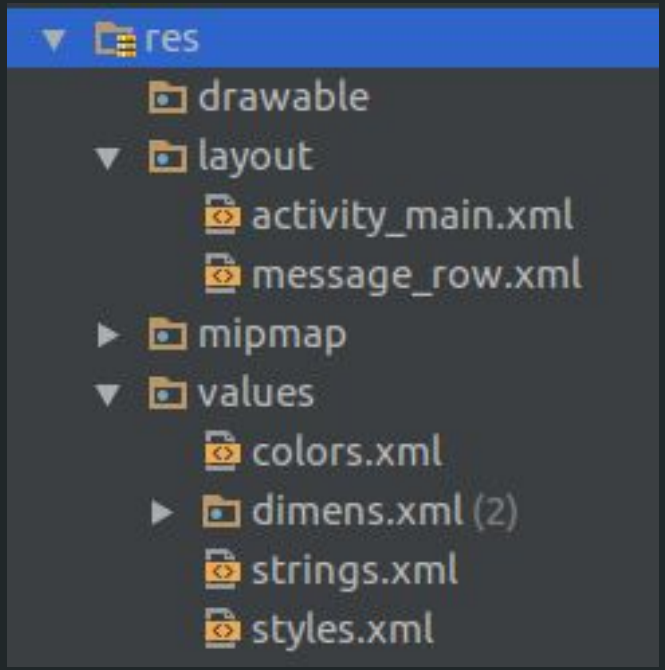
MyMessagesAdapter.java

- Sometimes we define our own classes



2.4 Resources

- “res” folder



3. Steps to do this app

3.1 View Layout

3.2 Wire Up View Widgets

3.3 ListView and Adapter

3.4 Storage

3.5 Internet request (+Internet permission)

3.1 View Layout Setup

- activity_main.xml
- message_row.xml

3.2 Wire Up View Widgets

MainActivity.java

- Declare as class-wide variables
- FindViewById() in onCreate()
- Add Listeners
  - addTextChangedListener() for EditText
  - setOnClickListener() for Button

3.3 Set up ListView and Adapter

1. Define our Adapter class (extending ArrayAdapter):
   - `public class MyMessagesAdapter extends ArrayAdapter<String>`
2. Define Constructor:
3. Override several methods:
   - `public int getCount()`
   - `public String getItem(int i)`
   - `public View getView(int position, View convertView, ViewGroup parent)`
4. More on getView():
   - First check if convertView is null; yes then inflate one from template (message_row.xml)
   - Then set text contents, colors, and positions in convertView
   - Finally return the convertView
5. Throughout the Activity, maintain a single reference to the Adapter.
   - Update its data? Pass them in and call `notifyDataSetChanged();` from inside the Adapter.

3.4 Storage

- The message history will be held in an ArrayList<String>, mMessages, and stored to an internal memory file “message_history.ser” whenever mMessages is updated
- When user clicks on “send”, update mMessages immediately
- When server message arrives, update mMessages (and the UI appearance)
- When user clicks on “clear_memory”, clear mMessages and delete the file

3.5 Internet Request and Response

- Use OkHttp to do internet Request
- Other ways include: Volley, Java Request + AsyncTask, etc
- Needs OkHttp and Okio package
  - How to import package in Android Studio?
  - place OkHttp and Okio in `app/libs/` then

    File-project structure-dependencies-Add file dependency-select
- Set up the Callback object
  - “Call me back when you have time”.
  - More on next slide

```
private void sendRequestUpdateResult(String mSentence) {
    OkHttpClient client = new OkHttpClient();
    Request request = new Request.Builder()
            .url("http://54.147.200.46/basicRequest")
            .build();

    client.newCall(request).enqueue(new Callback() {...});
}
```

Internet Response in Callback

- Results in response.body().string()
- runOnUiThread() since we want to update the UI appearance
- More explanation on Thread in next slide

```
private void sendRequestUpdateResult(String mSentence) {
    OkHttpClient client = new OkHttpClient();
    Request request = new Request.Builder()
            .url("http://54.147.200.46/basicRequest")
            .build();

    client.newCall(request).enqueue(new Callback() {
        @Override
        public void onFailure(Call call, IOException e) { e.printStackTrace(); }

        @Override
        public void onResponse(Call call, Response response) throws IOException {
            if (!response.isSuccessful()) {
                throw new IOException("Unexpected code "+response);
            } else {
                final String responseText = "server:"+response.body().string();

                Log.d(TAG, responseText);


                runOnUiThread(new Runnable() {
                    @Override
                    public void run() {
                        mMessages.add(responseText);
                        updateAdapterDataAndView();
                    }
                });

            }
        }
    });
}
```

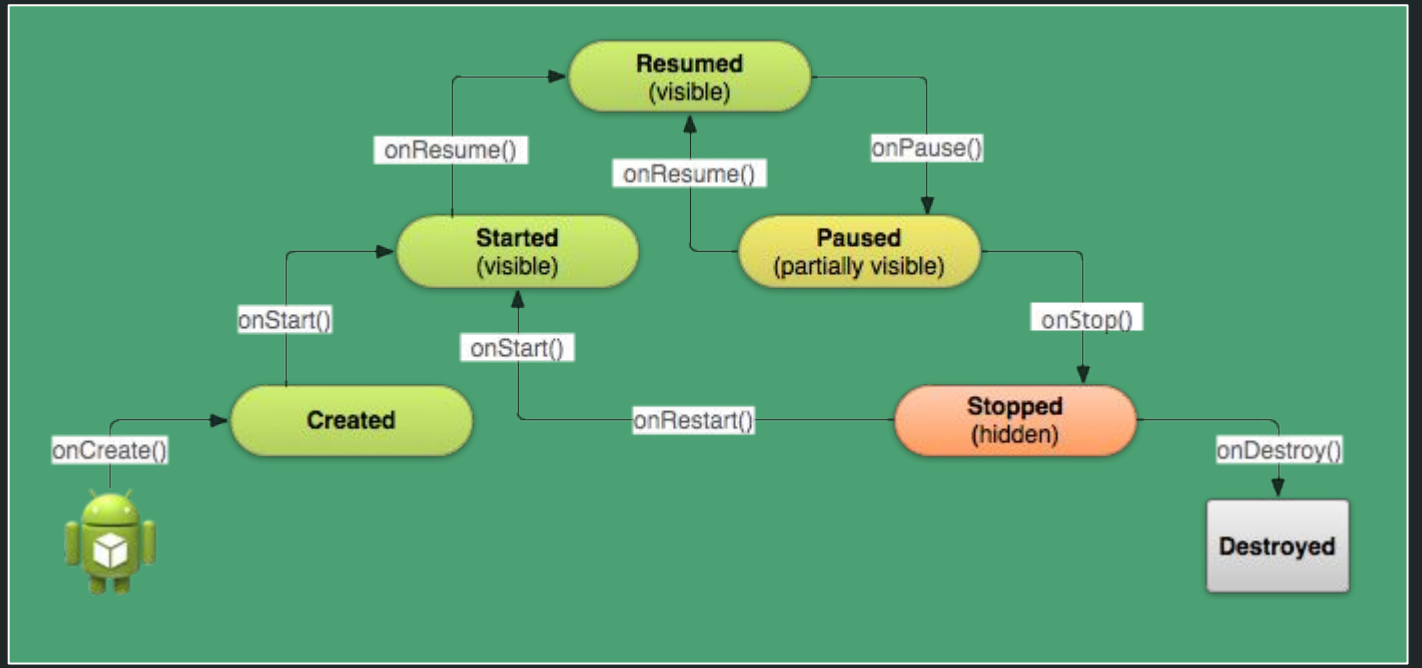
Thread in Android

- What is Thread?
  - Similar to “worker that does work for you simultaneously”
  - Difference from Process? Multiple Threads can run in the same Process, and shares the Process space. An application normally takes up one Process.
- UI Thread and Background Threads
- Internetworking must not be in UI Thread
  - So AsyncTask and OkHttp gives result in another Thread
- UI Update must be in UI Thread
  - That’s why we runOnUiThread() to jump back

3.6 Clean up and preserve data

- Android Activity Lifecycle
- Put init of Adapter

to onResume()



Thank you

- Questions and Feedbacks are welcome:
- Wechat: zzn_Baker