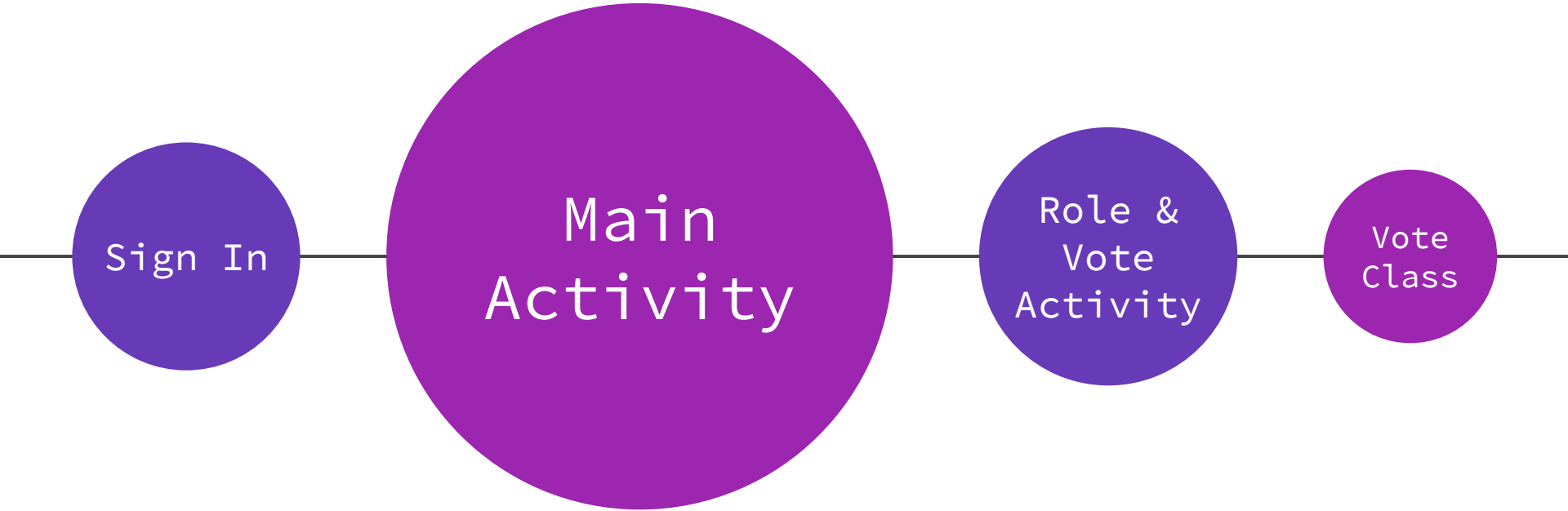
DJ Party Chat

Neo, Michael, Hunter

Look... you can't *not* download this

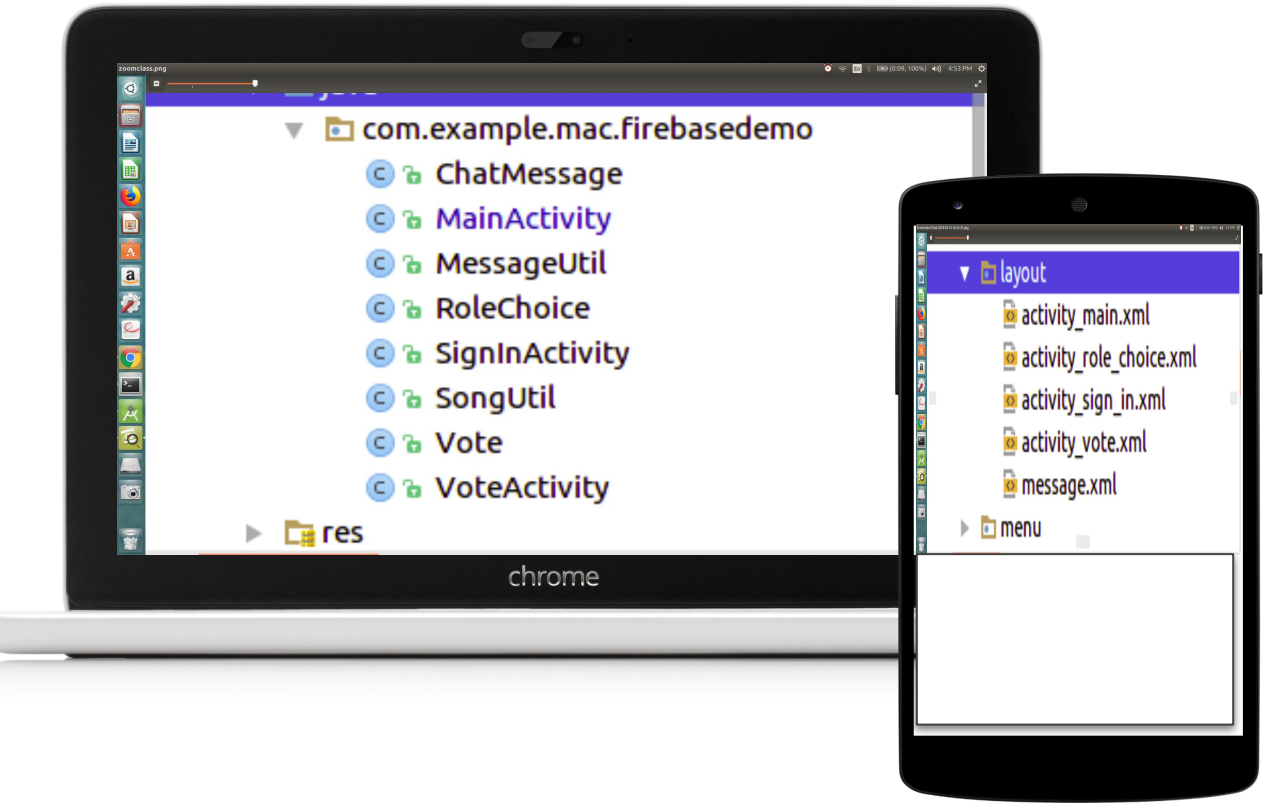
- Connect to Chat
- Choose a DJ
- Choose a Song
- Log out (*forever*)

Flow overview



Easy to understand

Classes and XML



```java
package com.example.mac.firebasedemo;

import ...

public class RoleChoice extends AppCompatActivity {

    private Button djButton, viewerButton;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_role_choice);

        // Assign fields
        djButton = (Button)findViewById(R.id.djButton);
        viewerButton = (Button)findViewById(R.id.viewerButton);

        // set click listeners
        djButton.setOnClickListener((v) -> { assignDj(); });

        viewerButton.setOnClickListener((v) -> { assignViewer(); });
    }
```

RoleChoice onCreate

After we inherit from AppComp, we create the DJ Button, Viewer Button, and onClickListeners for each Button

```java
    private void assignDj() {
        System.out.println("DJ Button");
        startActivity(new Intent(RoleChoice.this, MainActivity.class));
        finish();
    }

    private void assignViewer() {
        System.out.println("Viewer Button");
        startActivity(new Intent(RoleChoice.this, VoteActivity.class));
        finish();
    }
}
```

RoleChoice Functions

We have can Assign a DJ, and Assign a Viewer!

Vote Activity
As a Process

1) A DJ is Chosen

Song text is edited
DJ hits “send” and sends Songs

2) Vote Activity Begins

Users can select DJ’s songs
Each click on a song is tallied

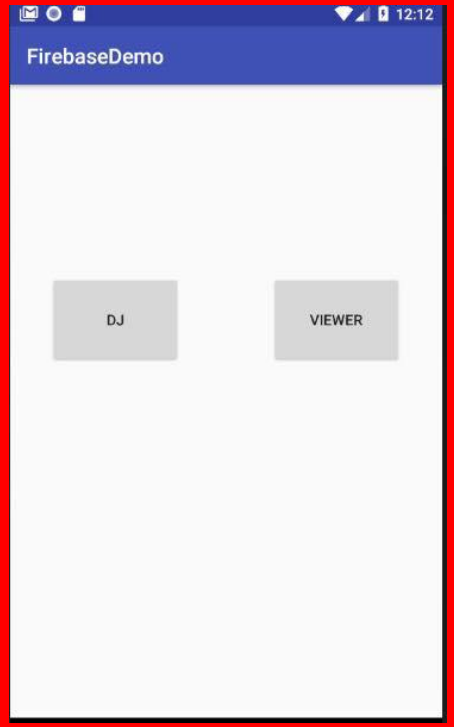
3) Votes are Counted

Contains all songs
Allows update of each song count

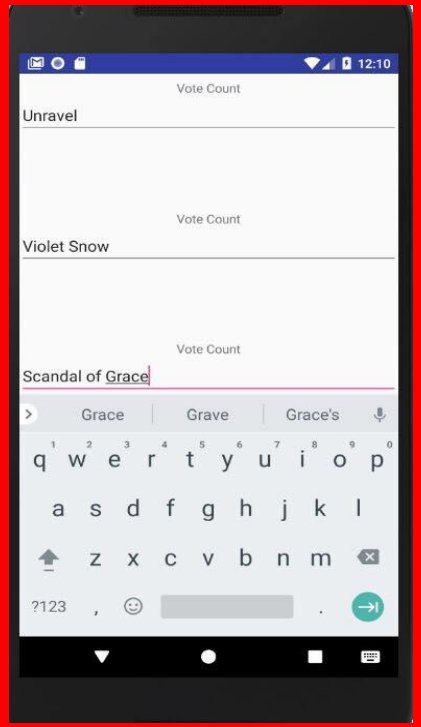
- Gets state from superclass appComp
- Sets layout with XML reference
- Creates buttons using XML elements
- Tells buttons they can be clicked

```
@Override
protected void onCreate(Bundle savedInstanceState) {
    Log.d(voteTAG, "starting voting activity");
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_vote);


    //assign fields
    song1Button = (Button) findViewById(R.id.song1Vote);
    song2Button = (Button) findViewById(R.id.song2Vote);
    song3Button = (Button) findViewById(R.id.song3Vote);

    // set click listeners

    song1Button.setOnClickListener((v) → { vote1(); });

    song2Button.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View v) {

        }
    });

    song3Button.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View v) {

        }
    });
```

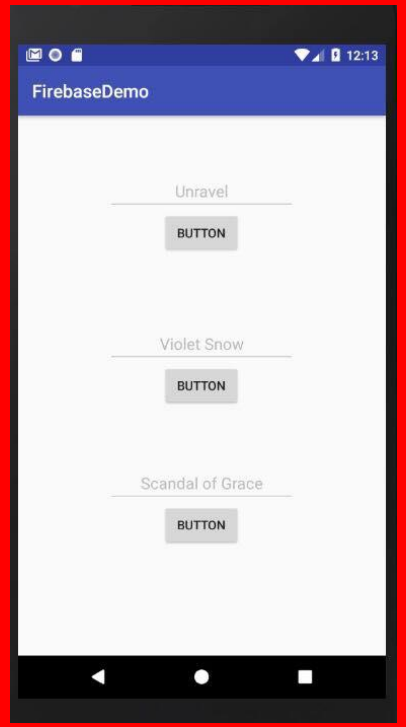
VoteActivity onCreate



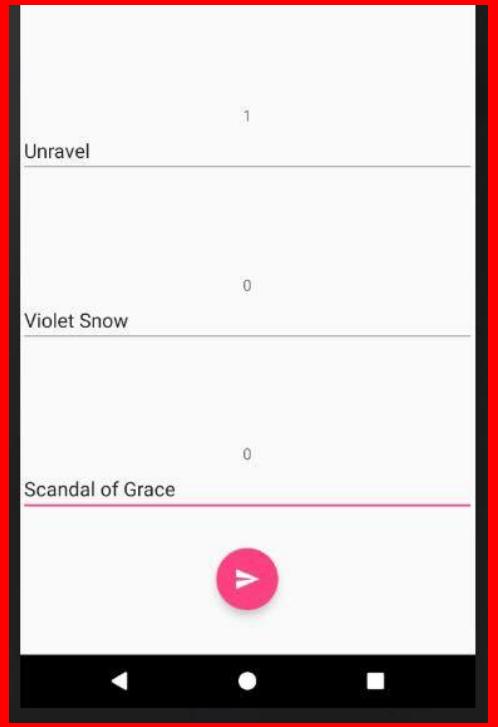
1) Role Choice



2) DJ Submits Songs



3) One Vote per Viewer



4) Votes are Counted

The Plumbing

Parents, Kids and nodes

1. Make a Firebase Database reference
2. Make a DataSnapshot of the reference
3. Create Parents
4. User Input creates Kids of Parents, and nodes of Kids

Example of Parent, Kid, Node relationship