Mobile Apps Trends and How to Implement with Android Studio

Hasna Nur Karimah - 13514106
Informatics/Computer Science
School of Electrical Engineering and Informatics
Bandung Institute of Technology, Jl. Ganesha 10 Bandung 40132, Indonesia
hasnank@s.itb.ac.id

***Abstract*—Today, mobile device is something that people won't forget to bring anywhere. Along with the device, there must be some applications contained in it. Just like fashion, mobile apps also has trends. In this paper, we will discuss about mobile apps trends and how to implement it with Android Studio using Java programming language. This list of trends with guide to Android Studio surely will help us in making mobile apps that suitable with market demands.**

Keywords—android, Android Studio, cloud storage, Java, location based, mobile apps, push notification.

I. Introduction

Information Technology has now grown so big, yet still growing much. People nowadays prefer bringing their own mobile devices anywhere, which is portable and personal. Though mobile devices' capability is limited, there is still huge scope for its development. So as developers, we need to concern to some points that is really needed (or wanted) by users. That is why we have to know much about market situation to fulfill and satisfy our users.

This paper consists of literature study, containing mobile apps trends in 2017, and then continued with how some of those trends implemented on Android OS. And then we came with conclusion at the end of this paper.

II. Literature Study

As cited from Multidots, 2017, there are 32 mobile apps trends in 2017 that will change the way we do business. Here is a list of them:

1. IoT + wearable apps
2. Mobile payments
3. On-demand apps
4. Augmented reality/virtual reality apps
5. Enterprise apps and BYOD
6. Accessibility/adaptability
7. **Cloud based apps**
8. Big data
9. Focus on UX failure mapping
10. Rise of citizen developers
11. App security
12. Prototyping tools
13. The official farewell of the hamburger menu
14. Data driven UI/UX design
15. Focus on building habit forming apps
16. Progressive apps
17. Machine learning, AI, and chat bots
18. **Push notifications**
19. Material design
20. **Android first**
21. App integrations
22. **Beacon/location based services**
23. Lazy loading
24. Blur backgrounds
25. Color + typography
26. Navigation
27. Movements – animations + videos
28. Designing for bigger phones
29. Gestures
30. Micro-mini interactions
31. Design automation
32. Card layouts and swipes

The bold ones are what we are going to implement in the next chapter.

III. Implementation

Pointing to mobile apps trends 2017 by Multidots, 2017, on number 20 we got "Android first". So now, we will try to implement some of the trends above on Android OS, using Android Studio and Java programming language.

A. *Cloud Based Apps*

With cloud support, we can make mobile apps that work across multiple devices. We can also keep a lot of data inside cloud storage. Now, let's learn how to log in to Google Firebase so we can use cloud services.

```
public class LoginActivity extends AppCompatActivity {
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_login);

if(((MyApplication)getApplication()).getmAuth() == null) {
            mAuth = FirebaseAuth.getInstance();

((MyApplication)getApplication()).setmAuth(mAuth);
        }
        else
        {
            mAuth =
((MyApplication)getApplication()).getmAuth();
        }

        mAuthListener = new
FirebaseAuth.AuthStateListener() {
            @Override
            public void onAuthStateChanged(@NonNull
FirebaseAuth firebaseAuth) {
                FirebaseUser user =
firebaseAuth.getCurrentUser();
                if (user != null) {
                    t_userview.setText(user.getEmail());
                    signedin = true;
                    b_signin.setText("Sign out");
                } else {
                    signedin = false;
                    t_userview.setText("");
                    b_signin.setText("Sign in");
                }           }
        };

        b_signin.setOnClickListener(new
View.OnClickListener() {
            @Override
            public void onClick(View view) {
                String username =
te_username.getText().toString();
                String password =
te_password.getText().toString();
                if(!signedin) {
                    if (isValidAuth(username, password)) {
                        SignInUser(username, password);
                    }
                }
                else {
                    mAuth.signOut();
                }
            }
        });
    }

    public void SignInUser(String email,String password)
    {
        mAuth.signInWithEmailAndPassword(email, password)
            .addOnCompleteListener(this, new
OnCompleteListener<AuthResult>() {
            @Override
            public void onComplete(@NonNull
Task<AuthResult> task) {
                if (!task.isSuccessful()) {
                    Toast.makeText(LoginActivity.this,
"Authentication failed.", Toast.LENGTH_SHORT).show();
                }
                else
                {
                    Intent intent = new
Intent(LoginActivity.this, MainActivity.class);

LoginActivity.this.startActivity(intent);
                }
            }
        });
    }

    public FirebaseUser GetUserInfo()
    {
        FirebaseUser user =
FirebaseAuth.getInstance().getCurrentUser();
        if (user != null) {
            String name = user.getDisplayName();
            String email = user.getEmail();
            Uri photoUrl = user.getPhotoUrl();
            String uid = user.getUid();
        }

        return user;
    }

    private FirebaseAuth mAuth;
    private FirebaseAuth.AuthStateListener mAuthListener;
    private String name = "";
    boolean signedin = false;
}
```

B. *Push Notifications*

Google Firebase not only provide database storing, but also messaging service. With Firebase Cloud Messaging, we can receive push notifications. Here's how.

```
public class MyFirebaseMessagingService extends
FirebaseMessagingService {


    @Override
    public void onMessageReceived(RemoteMessage
remoteMessage) {
        Intent intent = new
Intent(this,MainActivity.class);
        intent.setFlags(Intent.FLAG_ACTIVITY_CLEAR_TOP);
        PendingIntent pendingIntent =
PendingIntent.getActivity(this,0,intent,PendingIntent.FLAG_
ONE_SHOT);
        NotificationCompat.Builder notificationBuilder =
new NotificationCompat.Builder(this);
        notificationBuilder.setContentTitle("FCM
NOTIFICATION");

notificationBuilder.setContentText(remoteMessage.getNotific
ation().getBody());
        notificationBuilder.setAutoCancel(true);

notificationBuilder.setSmallIcon(R.mipmap.ic_launcher);

notificationBuilder.setContentIntent(pendingIntent);

        NotificationManager notificationManager =
(NotificationManager)
getSystemService(Context.NOTIFICATION_SERVICE);

notificationManager.notify(0,notificationBuilder.build());

        Bundle bundle = new Bundle();
        bundle.putString("msgBody",
remoteMessage.getNotification().getBody());

        Intent new_intent = new Intent();
        new_intent.setAction("ACTION_STRING_ACTIVITY");
        new_intent.putExtra("msg", bundle);

        sendBroadcast(new_intent);
    }
}
```

C. *Location Based Services*

We can give services from our mobile apps based on user's location. Here is how we can get our user's location using Google Location Services.

```
public class LocationActivity extends AppCompatActivity
implements GoogleApiClient.OnConnectionFailedListener,
GoogleApiClient.ConnectionCallbacks, LocationListener {

    GoogleApiClient mGoogleApiClient;

    Location mLastLocation;

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

        mGoogleApiClient = new
GoogleApiClient.Builder(this)
                .addConnectionCallbacks(this)
                .addOnConnectionFailedListener(this)
                .addApi(LocationServices.API).build();

        if (mGoogleApiClient == null) {
            mGoogleApiClient = new
GoogleApiClient.Builder(this)
.addConnectionCallbacks((GoogleApiClient.ConnectionCallback
s) this)
                    .addOnConnectionFailedListener(this)
                    .addApi(LocationServices.API)
                    .build();
        }
        final Context thisObject = this;
    }


    @Override
    public void onConnected(Bundle connectionHint) {
        if (ActivityCompat.checkSelfPermission(this,
android.Manifest.permission.ACCESS_FINE_LOCATION) !=
PackageManager.PERMISSION_GRANTED) {
            ActivityCompat.requestPermissions(
                    this,
                    new
String[]{android.Manifest.permission.ACCESS_FINE_LOCATION},
                    200);
        }
        mLastLocation =
LocationServices.FusedLocationApi.getLastLocation(mGoogleAp
iClient);
        double lat=0;
        double lng=0;
        if (mLastLocation != null) {
            lat = mLastLocation.getLatitude();
            lng = mLastLocation.getLongitude();
        } else{

            LocationRequest mLocationRequest = new
LocationRequest();
            mLocationRequest.setInterval(10000);
            mLocationRequest.setFastestInterval(5000);

mLocationRequest.setPriority(LocationRequest.PRIORITY_BALAN
CED_POWER_ACCURACY);
            mLocationRequest.setSmallestDisplacement(10);

LocationServices.FusedLocationApi.requestLocationUpdates(mG
oogleApiClient, mLocationRequest, this);
        }
        Geocoder gcd = new Geocoder(this,
Locale.getDefault());
        List<Address> addresses = null;
        try {
            addresses = gcd.getFromLocation(lat, lng, 1);
        } catch (IOException e) {
            e.printStackTrace();
        }
        if (addresses!= null && addresses.size() > 0)
        {
            TextView textview = (TextView)
findViewById(R.id.location);
            location = addresses.get(0).getLocality();
            textview.setText((CharSequence)
addresses.get(0).getLocality());
        }
    }
}
```

IV. Conclusion

Mobile devices is now growing so fast and demanding. As a developer, we have to know what trends there are to make a useful and profitable apps. We also have to have a long life learning skill in this tech industry.

Acknowledgment

First of all, I want to thank Allah SWT., for without His will, this paper won't have finished this way. And then, thanks to my Socio-Informatics and Professionalism course lecturers, Mr. Rinaldi Munir, Mrs. Ayu Purwarianti, and Mrs. Dessi Puji Lestari who taught me lessons and guided me for a well-written paper. Also I want to appreciate my friends who has been inspiring and supporting me through the semester, especially in the making of this paper.

References

[1] Multidots. *32 latest mobile app trends 2017 that will change the way you do business.* Retrieved from https://medium.com/@multidots/the-latest-in-mobile-app-trends-2017-wheres-the-app-ecosystem-heading-cadd1e7c8709, 2017.

Bandung, May 5th 2017

[illegible]

Hasna Nur Karimah - 13514106