1. Android 이란?

1.1 안드로이드 개념

모바일 기기의 운영체제로, 리눅스(Linux)를 기반으로한 운영체제이며, 미들웨어, 사용자 인터페이스, 어플리케이션, MMS 서비스 등을 하나로 묶어 제공합니다. 또 다른 모바일 운영체제로는 애플의 iOS, 노키아의 심비안 등이 있습니다.

1.2 안드로이드의 특징

- 완전 개방형 플랫폼으로 모든 소스코드가 공개되어있습니다. 덕분에 개발자들이 SW를 제작하거나 다양하게 응용하여 사용할 수 있게 되었습니다.
- 개발자들에게 친숙한 java 및 kotlin을 사용하여 앱 개발을 할 수 있도록 안드로이드 스튜디오를 지원합니다.
- 안드로이드 버전은 디저트 이름으로 별명을 갖는데 그 순서는 애플파이 - 바나나브레드 - 컵케이크 - 도넛 - 이클레어 - 프로요 - 진저브레드- 허니콤- 아이스크림샌드위치- 젤리빈- 킷캣- 롤리팝- 마시멜로- 누가- 오레오- 파이입니다.

1.3 버전별 특징

- Apple Pie 애플파이

2008년 발표된 안드로이드 최초의 정식 버전으로 안드로이드 마켓과 웹 브라우저, 카메라, 등의 기능을 내포하여 출시하였으나 테스트 성향이 강하였던 버전입니다.

- Banana Bread 바나나 브레드

구글 지도의 상세 정보 확인 및 리뷰, MMS 첨부 파일 저장, 통화 메뉴에서 다이얼 화면 보기 및 숨기기 등의 기능을 추가하였습니다.

- Cupcake 컵케이크

  블루투스 헤드셋 자동 접속, 홈 위젯과 폴더, 애니메이션 화면 효과 등의 업데이트가 진행되었고, 이 때부터 한국어를 지원하기 시작했습니다.

- Donut 도넛

갤러리에서 사용자가 여러 장의 사진을 삭제할 수 있게 되었습니다. 보이스 검색 기능이 추가되었으며 WVGA해상도를 지원하기 시작했습니다.

- Eclair 이클레어

  다양한 제품에 화면 크기와 해상도가 최적화 되었고 라이브 배경화면을 지원하기 시작했습니다. 사용자 인터페이스가 개편되고 하드웨어 속도가 최적화 되었습니다. 카메라의 디지털 줌 기능이 추가되었습니다.

- Froyo 프로요

안드로이드 운영체제 최초로 USB 테더링과 와이파이, 핫스팟 기능이 업데이트 된 버전입니다. 자동 업데이트 기능이 추가되었으며 어도비 플래시 10.1을 지원하게 되었습니다.

- Ginger Bread 진저브레드

  NFC 기능을 지원하기 시작했습니다.

- Honey Comb 허니콤

안드로이드 태블릿을 정식으로 지원하기 위해 개발된 버전으로 태블릿에 최적화되었습니다. 3D 구글 지도 서비스를 제공하고 USB 액세서리를 연결을 지원합니다. 크기 조절이 가능한 위젯과 디지털 카메라로부터 사진을 가져올 수 있는 USBHost 기능이 추가되었습니다.

- Ice cream Sandwich 아이스크림 샌드위치

음성을 인식하여 문자로 변환하는 음성 인식 기능과 보이스 메일 앱이 추가되었습니다. 화면 캡쳐 및 얼굴 인식을 통한 잠금 해제 등의 기능이 추가되었습니다.

- Jellybean 젤리빈

  구글 크롬이 기본 브라우저로 채택되었고 어도비 플래시 플레이어를 지원하지 않게 되었습니다.

- Kitkat 킷캣

  클라우드 프린팅을 지원하고 보안을 향상시켰습니다. 또한 LED 플래쉬 기능이 추가되었습니다.

- Lollipop 롤리팝

2014년 10월 발표한 버전으로 클라우드 프린팅을 강화하고 64비트의 CPU를 정식으로 지원하게 되었습니다. 또 기존 달빅 캐시(Dalvik cache)를 안드로이드 런타임으로 완전히 변경하였습니다.

- Marshmallow 마시멜로

전원 관리 시스템이 탑재되어 사용자가 기기를 손에 들고 있지 않을 경우 자동으로 백그라운드 작업을 정리합니다. 또 지문 인식 등의 기능이 추가되었습니다.

- Nougat 누가

배터리와 데이터 절약 기능이 향상되었는데 스마트폰을 사용하지 않고 이동 중 일 경우 잠자기(Doze)모드가 되어 배터리를 절약하고 일부 앱이 백그라운드에서 데이터를 전송하거나 수신할 수 없게 하여 데이터 사용량을 줄였습니다. 또 3D 랜더링 API인 볼칸(Vulkan)도 지원하게 되었습니다.

- **Oreo 오레오**

누가 버전의 잠자기(Doze) 기능이 큰 폭으로 향상되어 사용자의 불편함을 없앴습니다. 또한 알림창에서 경고, 메세지, 스크린샷 등의 분류를 나누어 알림 기능을 온오프 할 수 있는 업데이트가 진행되었습니다.
백그라운드 실행제한

안드로이드는 그동안 앱 실행여부와 관계없이 백그라운드에서 시스템 자원을 자유롭게 사용할 수 있도록 해왔는데
이는 사용하지 않는 앱 또한 시스템 자원을 과도하게 소모하는 문제로 연결되어 오레오버전부터 백그라운드에서 실행되는 앱의 동작을 제한하는 기능이 추가되었습니다.

제한되는 방식은 두 가지입니다.(참고로 이 제한은 O를 대상으로 하는 앱에만 적용됩니다.)

> 백그라운드 서비스 제한: 앱이 유휴 상태인 경우 백그라운드 서비스의 사용이 제한됩니다. 이 기능은 포그라운드 서비스에는 적용되지 않습니다.

> 브로드캐스트 제한: 제한된 예외의 경우, 앱이 암시적 브로드캐스트에 등록하기 위해 자체 매니페스트를 사용할 수 없습니다. 그렇지만 여전히 앱이 런타임에 브로드캐스트에 등록할 수 있으며, 특정 앱을 대상으로 하는 명시적 브로드캐스트에 등록하기 위해 매니페스트를 사용할 수 있습니다.

대부분의 경우, 앱은 JobScheduler 작업을 사용하여 이러한 제한을 해결할 수 있습니다. 이 접근방식을 통해 앱이 실행되지 않을 때 작업을 수행하면서도, 사용자 환경에 영향을 미치지 않는 방식으로 진행됩니다.
백그라운드 위치 제한
백그라운드 앱이 사용자의 현재 위치를 검색할 수 있는 빈도를 제한합니다.

참고 URL : https://developer.android.com/about/versions/oreo/background

- **Pie 파이**

인공지능의 핵심 기술인 머신러닝(AI)이 최초로 적용된 모바일 운영체제로 사용자의 사용패턴을 분석하는 능동적 사용자 경험 인터페이스가 구축되었습니다. 그리고 앱의 기능 일부를 추출해 안드로이드 운영체제와 일치화시키는 기능인 앱 슬라이스 기능도 도입되었습니다. 스마트폰 이용패턴을 한 눈에 파악한 후 이를 제어할 수 있도록 '계기판'과 '앱 타이머'라는 기능도 추가되었으며, HDR(High Dynamic Range)을 정식 지원하기 시작했습니다.

메세지에 첨부된 사진도 미리 볼 수 있도록 알림기능이 추가되었습니다.
스마트 리플라이 기능(적절한 메세지 답장을 추천해주는 기능)이 추가되었습니다.
눈의 피로를 줄여주는 다크 테마를 적용시킬 수 있게 되었습니다.
노치 디자인을 가진 모바일폰이 나오고 있어 이에 발맞추어 디스플레이 컷아웃 지원을 시작했습니다.

참고 URL : https://developer.android.com/about/versions/pie/android-9.0-changes-all

백그라운드에 있는 앱일 경우 사용자 입력과 센서 데이터에 액세스하는 능력을 제한하는 기능이 추가되었습니다.
백그라운드에 잇는 경우 마이크나 카메라에 액세스할 수 없습니다.
연속 보고 모드를 사용하는 센서(예: 가속도계 및 자이로스코프)는 이벤트를 수신하지 않습니다.
변경 시 또는 원샷 보고 모드를 사용하는 센서는 이벤트를 수신하지 않습니다.

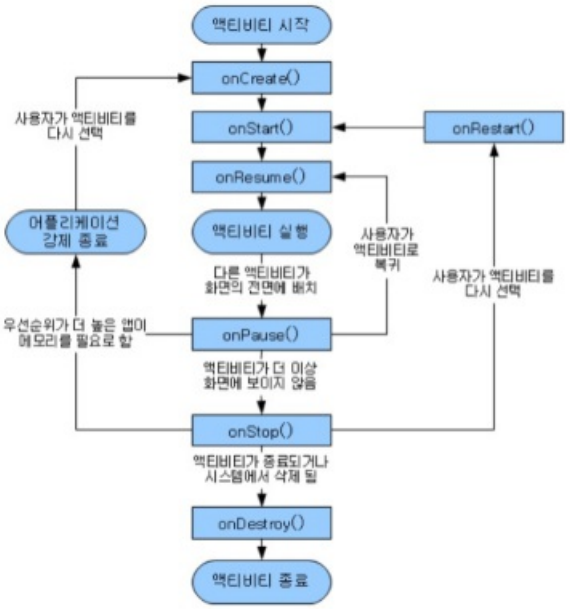
참고 URL : https://developer.android.com/about/versions/oreo/background-location-limits

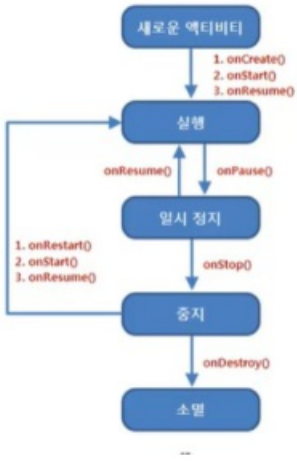
2. Activity

1.1 Activity개념

사용자에게 UI가 있는 화면을 제공하는 앱 컴포넌트입니다. 폰 다이얼러 화면, 카메라 촬영 화면, 이메일 쓰기 화면, 지도 보기 화면 등과 같이 사용자들이 뭔가 하기 위해 상호작용을 할 수 있는 화면을 제공하는 것이 Activity입니다.
각 액티비티는 하나의 윈도우에 UI를 그리며, 그 윈도우가 보통은 화면을 꽉 채우지만, 화면보다 작을수도 있고, 다른 윈도우의 위에 떠 있을 수도 있습니다.
앱은 보통 여러개의 액티비티로 이루어져 있습니다. 기본적으로 하나의 MainActivity를 갖고있으며 그것은 사용자가 앱을 처음 실행했을때 보여집니다. 이 때 이벤트 및 Handler처리를 통해 액티비티에서 다른 액티비티를 실행시킬 수도 있습니다. 다른 액티비티가 실행되면 이전의 액티비티는 정지되지만, 시스템이 백스택이라 불리는 스택에 저장해뒀기 때문에 없어지지는 않습니다.

1.2 Activity 생명 주기





새로운 액티비티가 실행되면서 현재 액티비티가 정지하게 되면, 시스템은 생명주기의 콜백 메소드를 호출함으로써 액티비티의 상태가 변경되었음을 알려줍니다. 이러한 콜백 메소드들은 시스템이 액티비티를 생성하고, 보여주고, 멈추고, 제거하는 등의 상황에 호출되며, 적절히 오버라이드함으로써 각각의 상황에 맞는 동작을 구현할 수 있습니다. 예를 들어, 액티비티는 정지되었을 때 자원이 많이 드는 큰 객체들을 해제하였다가, 액티비티가 다시 화면에 보여질 때 필요한 리소스들을 다시 가져와서 중지되었던 작업들을 다시 실행할 수 있습니다. 이렇게 액티비티가 생성되고 보여지고 멈추고 제거되고 하는 등의 상태변화는 모두 액티비티 생명주기의 일부입니다. 이러한 activity는 앱을 중복해서 열 때 발생하는 것들을 제어하기 위해 사용합니다.

간단하게 MainActivity에서 다음과 같은 코드로 짠 앱을 통해 수명주기를 토스트메세지로 확인해보았습니다.
main.xml에는 버튼(Button button = (Button)findView ById(R.id.button);) 하나만 두고 이에 finish함수를 연결시켜주었습니다.

```
package com.gkskfhdlstmapk.hanpinetree.프로젝트명;

import android.app.Activity;
import android.content.SharedPreferences;
import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.view.View;
import android.widget.Button;
import android.widget.Toast;

public class MainActivity extends AppCompatActivity {

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        //onCreate함수 자체가 이미 수명주기 함수중 하나이다.
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);

        Toast.makeText(this,"onCreate() 함수 호출 됨.",Toast.LENGTH_SHORT).show();

        Button button = (Button)findViewById(R.id.button);
        button.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                finish();
            }
        });
    }
   //어떤 상태들이 호출되는지 확인하기위한 함수들
    @Override
    protected void onStart() {
        super.onStart();
        Toast.makeText(this,"onStart() 함수 호출 됨.",Toast.LENGTH_SHORT).show();
    }

    @Override
    protected void onStop() {
        super.onStop();
        Toast.makeText(this,"onStop() 함수 호출 됨.",Toast.LENGTH_SHORT).show();
    }

    @Override
    protected void onDestroy() {
        super.onDestroy();
        Toast.makeText(this,"onDestroy() 함수 호출 됨.",Toast.LENGTH_SHORT).show();
    }
```

```
    @Override
    protected void onPause() {
        super.onPause();
        Toast.makeText(this,"onPause() 함수 호출 됨.",Toast.LENGTH_SHORT).show();

        //가지고 있던 데이터값을 저장해두기.
        SharedPreferences pref = getSharedPreferences("pref", Activity.MODE_PRIVATE);
        //name은 저장할 때와 복구할 때 같은 이름으로 지정해주어야함

        SharedPreferences.Editor editor = pref.edit();
        editor.putString("name","소녀시대");
        //put으로 넣고 get으로 가져옴
        editor.commit();
        //commit을 꼭 해주어야 저장이 됨.

    }

    @Override
    protected void onResume() {
        super.onResume();
        Toast.makeText(this,"onResume() 함수 호출 됨.",Toast.LENGTH_SHORT).show();

        //onPause일 때 저장해두었던 데이터를 복구해봅시다.
        SharedPreferences pref = getSharedPreferences("pref", Activity.MODE_PRIVATE);

        if(pref != null){
            String name = pref.getString("name","");
            //name값이 없는 경우에는 빈 값을 달라는 뜻

            Toast.makeText(this,"복구 된 이름 : "+name ,Toast.LENGTH_SHORT).show();
        }
    }
}
```