app/build.gradle

```
apply plugin: 'com.android.application'

repositories {
    mavenCentral()
}

android {
    compileSdkVersion 27
    defaultConfig {
        applicationId "com.example.android.movies"
        minSdkVersion 19
```

AWESOME

You did a great job here! You have your `minSdkVersion` value set to 19, this can make your app be able to run on almost every Android devices. Also, you can gain most of the latest powerful features of Android 4.X.

Read more here:

https://developer.android.com/about/dashboards/index.html

https://medium.com/google-developers/picking-your-compilesdkversion-minsdkversion-targetsdkversion-a098a0341ebd#.8cwc0kvjr

```
        targetSdkVersion 27
        versionCode 1
        versionName "1.0"
        testInstrumentationRunner "android.support.test.runner.AndroidJUnitRunner"
    }
    buildTypes {
        release {
            minifyEnabled false
            proguardFiles getDefaultProguardFile('proguard-android.txt'), 'proguard-rules.pro'
        }
    }
}

dependencies {
    implementation fileTree(dir: 'libs', include: ['*.jar'])
    implementation 'com.android.support:appcompat-v7:27.1.1'
    implementation 'com.android.support.constraint:constraint-layout:1.1.0'
    testImplementation 'junit:junit:4.12'
    androidTestImplementation 'com.android.support.test:runner:1.0.2'
    androidTestImplementation 'com.android.support.test.espresso:espresso-core:3.0.2'
    implementation 'com.android.support:recyclerview-v7:27.1.1'
    implementation 'com.squareup.retrofit2:retrofit:2.4.0'
```

AWESOME

It's great that you are already using this awesome third-party library!

You can also take a look here and here, you will find some great libraries that will help you create wonderful apps, making your life easier and your apps better!

https://github.com/codepath/android_guides/wiki/Must-Have-Libraries
https://github.com/wasabeef/awesome-android-libraries

MainActivity.java:

```
62
63          mMovieAdapter = new MoviesAdapter(mMovieList, this);
64
65          mMovieRecyclerView.setAdapter(mMovieAdapter);
66          mMovieRecyclerView.setLayoutManager(new GridLayoutManager(this, 2));
```

SUGGESTION

Here you can dynamically calculate the number of columns and the layout would adapt to the screen size and orientation

```
... new GridLayoutManager(this, numberOfColumns());
```

the method implementation:

```
    private int numberOfColumns() {
        DisplayMetrics displayMetrics = new DisplayMetrics();
        getWindowManager().getDefaultDisplay().getMetrics(displayMetrics);
        // You can change this divider to adjust the size of the poster
        int widthDivider = 400;
        int width = displayMetrics.widthPixels;
        int nColumns = width / widthDivider;
        if (nColumns < 2) return 2;
        return nColumns;
    }
```

```
67
68          fetchMovies();
69      }
70
71
72      private void fetchMovies() {
73          // clean current values:
74          mMovieAdapter.notifyItemRangeRemoved(0, mMovieList.size());
75          mMovieList.clear();
76
77          //check connectivity:
78          if (Utilities.isNetworkAvailable(this) && Utilities.isOnline()) {
```

AWESOME

Checking connectivity before requesting data is a good practice. Nice catch! 👍

app/src/main/res/layout/activity_detail.xml

```
1 <?xml version="1.0" encoding="utf-8"?>
2
3 <ScrollView xmlns:android="http://schemas.android.com/apk/res/android"
```

AWESOME

Good job setting up your ScrollView. This mechanism allows the users to scroll up and down and reach all content in the screen, well done! 👏

app/src/main/java/com/example/android/movies/models/Movie.java

```
1 package com.example.android.movies.models;
2
3 import android.os.Parcel;
4 import android.os.Parcelable;
5
6 import com.example.android.movies.utils.Utilities;
7 import com.google.gson.annotations.SerializedName;
8
9 public class Movie implements Parcelable {
```

AWESOME

Good job implementing Parcelable instead of Serializable. Learn more about Parcelable vs Serializable

http://www.developerphil.com/parcelable-vs-serializable/

OfflineActivity.java:

```
31
32          mRetryButton.setOnClickListener(new View.OnClickListener() {
33              @Override
34              public void onClick(View v) {
```

SUGGESTION

Butterknife also provides an onClick method, take a look in the documentation

https://jakewharton.github.io/butterknife/

DetailActivity.java:

```
34
35      @Override
36      protected void onCreate(Bundle savedInstanceState) {
37          super.onCreate(savedInstanceState);
38
39          setContentView(R.layout.activity_detail);
40          ButterKnife.bind(this);
```

AWESOME

Nice idea using Butterkife :)

app/src/main/java/com/example/android/movies/ServiceGenerator.java

```java
package com.example.android.movies;

import com.example.android.movies.models.Movie;
import com.example.android.movies.utils.MovieAPIdeserializer;
import com.google.gson.Gson;
import com.google.gson.GsonBuilder;
import com.google.gson.reflect.TypeToken;

import java.util.List;

import retrofit2.Retrofit;
import retrofit2.converter.gson.GsonConverterFactory;

public class ServiceGenerator {
    private final static String BASE_API_URL = "https://api.themoviedb.org/3/movie/";

    private static Gson gson = new GsonBuilder()
                    .registerTypeAdapter(new TypeToken<List<Movie>>(){}.getType(), new MovieAPIdeserial
                    .create();

    private static Retrofit retrofit = null;
```

AWESOME

Nice choice and implementation of Retrofit as REST client!