Popular Movies App (Stages 1 & 2)

Udacity Android Developer Nanodegree Projects 2 & 3
Grow with Google Developer Scholarship Program

@Bob
August 4, 2018

Message from Udacity Staff

Plagiarism is any act claiming or implying another person’s work is your own.

Udacity has zero tolerance for plagiarism. Do not plagiarize any work.

Examples of plagiarism:

- Copying someone’s code exactly, in whole or part (verbatim).
- Combining code copied verbatim from multiple sources.
- Copying someone’s code, in whole or part, and making changes.
- Working on a project together with other students, submitting each other’s code portions.

Not Plagiarism:

- Looking at someone else’s code to get a general idea of implementation.
- Using code that has been approved by Udacity for such use without attribution.
- Using with proper attribution (date and URL) minor open source helper code.
- Sharing ideas on projects with other students before writing code independently.

Source: “What Is Plagiarism?” Udacity Support, 1 Aug. 2018, udacity.zendesk.com/hc/en-us/articles/360001451091-What-is-plagiarism-.

Overview

- Review foundational knowledge
- Understand assignment requirements
- Map knowledge to requirements
- Survey alternatives
- Define structure
- Dive into code
- Test your app

Stage 1

Main Discovery Screen, Detail View, and Settings

Stage 1
Review foundational knowledge

- Data
  - Request (HttpUrlConnection)
- Sync
  - Thread (AsyncTask)
- UI
  - Activity (Explicit Intent, OnClickListener)
  - ViewGroup, View, Menu, Drawable

Stage 1
Understand assignment requirements

- Rubric
    - **Request data** from TMDb Movies 'Popular' and 'Top Rated' API on worker thread
    - **Display movie posters** in the main layout via a grid
    - **Toggle the sort order** of movies by: most popular, highest rated
        - **Change movie posters displayed** in the main layout when selecting new sort criteria
    - **Display the details** for a selected movie (poster, title, date, rating, overview)
        - **Launch the details** in detail layout when selecting movie poster in main layout
    - Write in Java with stable versions of Android Studio and Gradle dependencies
    - Adhere to Nanodegree General Guidelines except for instance state data restoration

Stage 1
Understand assignment requirements

- Guide
    - Request data from the popular and top_rated endpoints
        - Construct URL: http://api.themoviedb.org/3/movie/[endpoint]?api_key=[API_KEY]
        - Submit request and parse response
        - Remove the API key, which is illegal to publish, from public repository
    - Display movie posters with image loading and caching API like **Picasso**
        - Construct URL from base (http://image.tmdb.org/t/p/w185) and ‘poster path’
        - Picasso.get().load(posterURL).into(imageView);

Stage 1
Map knowledge to requirements

Requirement	Knowledge
Request data over the network in the background	Request | Thread
Display movie posters in a grid layout	View | Drawable
Change movie posters based on sort criteria	Menu
Launch details when selecting a movie poster	Activity

Stage 1
Survey alternatives

Knowledge	Alternative
Request, Thread	HttpUrlConnection, RetroFit | AsyncTask
View, Drawable	RecyclerView, GridView | Picasso, Glide
Menu	Settings, Spinner
Activity	Explicit Intent | OnClickListener

Stage 1
Survey alternatives

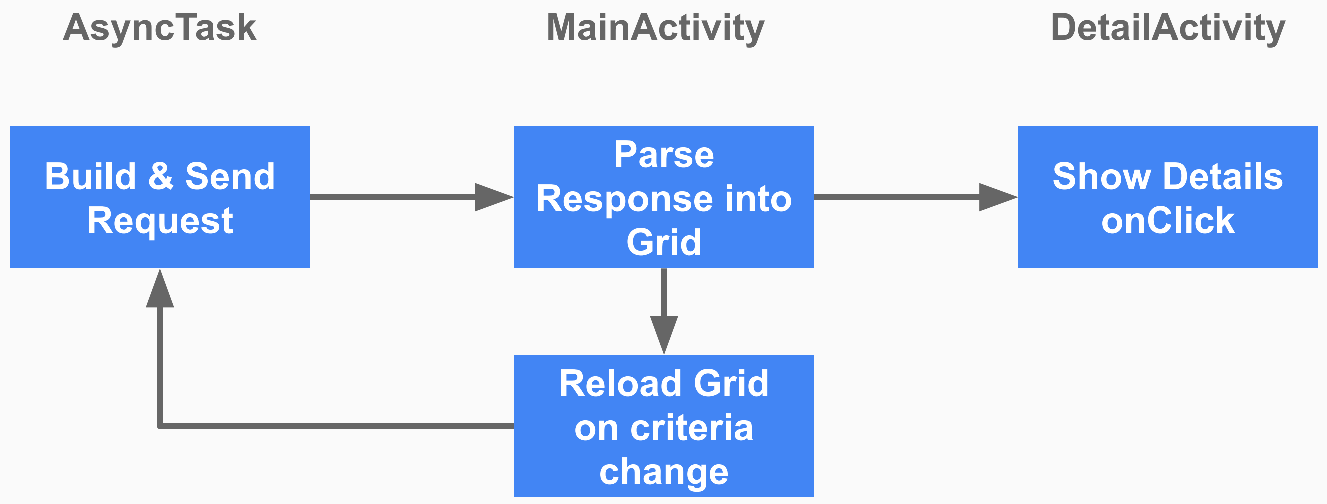
Select tools that suit you or your team’s bandwidth, skill level, and timeline.

- Ideal: Learn and implement RetroFit, Glide and Spinner
- Major constraint: Timeline
- Recommendation: Leverage knowledge with existing codebase (Sunshine)

Stage 1
Define structure

Choice	Plan
HttpUrlConnection | AsyncTask	Run HttpUrlConnection over AsyncTask; Parse response into useable data
RecyclerView| Picasso	Load Picasso images from URL into ViewHolders
Settings	Add sort criteria actions to Menu Settings; Reload RecyclerView on change to sort criteria
Explicit Intent | OnClickListener	Launch DetailActivity from Intent onClick of image

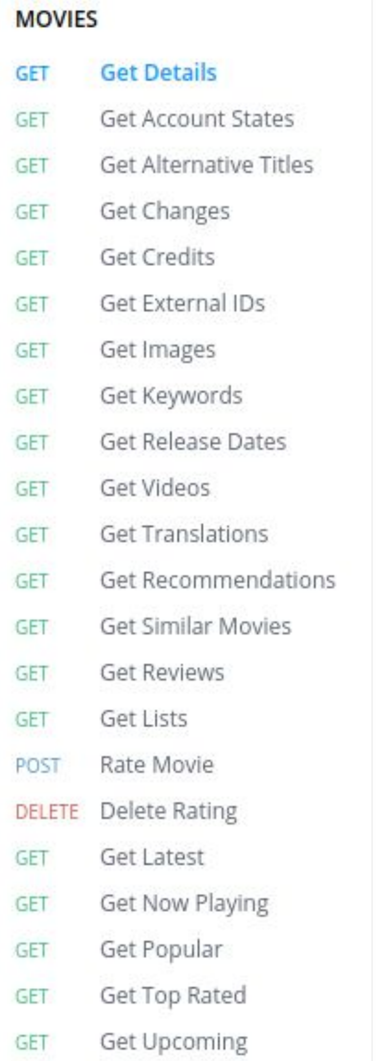
Stage 1
Define structure



Stage 1
Dive into code

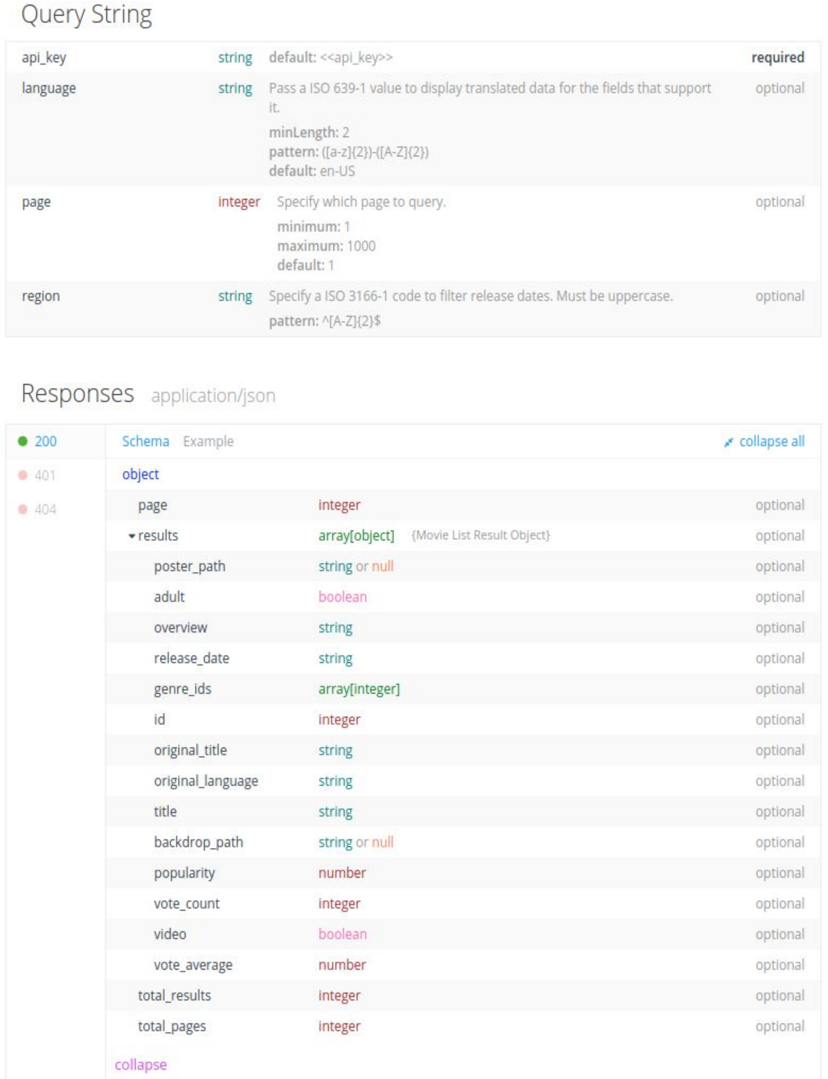
Tech Webinar:
Popular Movies App
(Stages 1 & 2)

@Bob August 2018



MOVIES

GET	Get Details
GET	Get Account States
GET	Get Alternative Titles
GET	Get Changes
GET	Get Credits
GET	Get External IDs
GET	Get Images
GET	Get Keywords
GET	Get Release Dates
GET	Get Videos
GET	Get Translations
GET	Get Recommendations
GET	Get Similar Movies
GET	Get Reviews
GET	Get Lists
POST	Rate Movie
DELETE	Delete Rating
GET	Get Latest
GET	Get Now Playing
GET	Get Popular
GET	Get Top Rated
GET	Get Upcoming



Query String

api_key	string	default: <<api_key>>	required
language	string	Pass a ISO 639-1 value to display translated data for the fields that support it. minLength: 2 pattern: ([a-z]{2})-([A-Z]{2}) default: en-US	optional
page	integer	Specify which page to query. minimum: 1 maximum: 1000 default: 1	optional
region	string	Specify a ISO 3166-1 code to filter release dates. Must be uppercase. pattern: ^[A-Z]{2}$	optional

Responses application/json

● 200 ● 401 ● 404

Schema Example collapse all

object

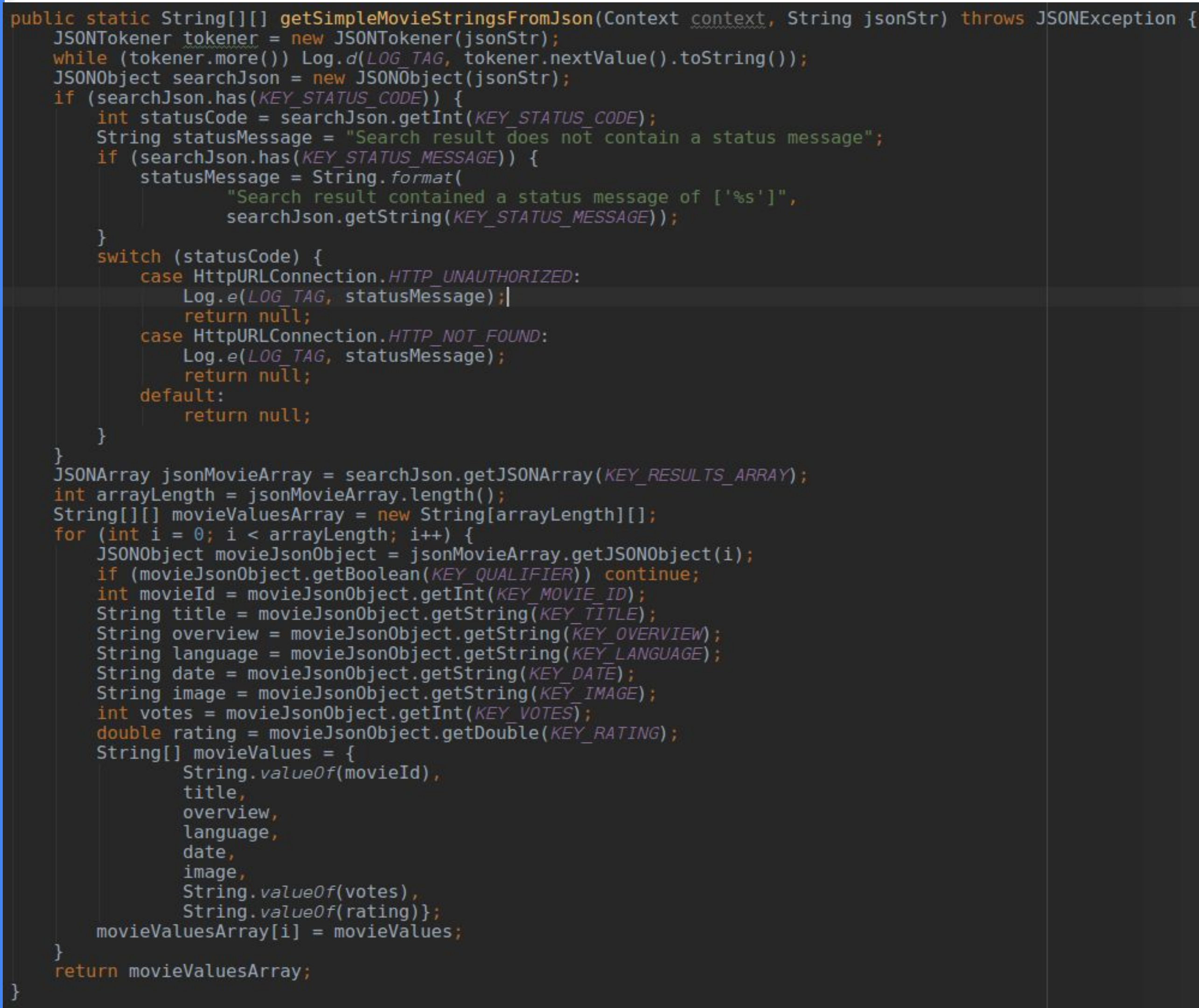
page	integer	optional
▾results	array[object] {Movie List Result Object}	optional
poster_path	string or null	optional
adult	boolean	optional
overview	string	optional
release_date	string	optional
genre_ids	array[integer]	optional
id	integer	optional
original_title	string	optional
original_language	string	optional
title	string	optional
backdrop_path	string or null	optional
popularity	number	optional
vote_count	integer	optional
video	boolean	optional
vote_average	number	optional
total_results	integer	optional
total_pages	integer	optional

collapse

Stage 1
Dive into code

Tech Webinar:
Popular Movies App
(Stages 1 & 2)

@Bob August 2018

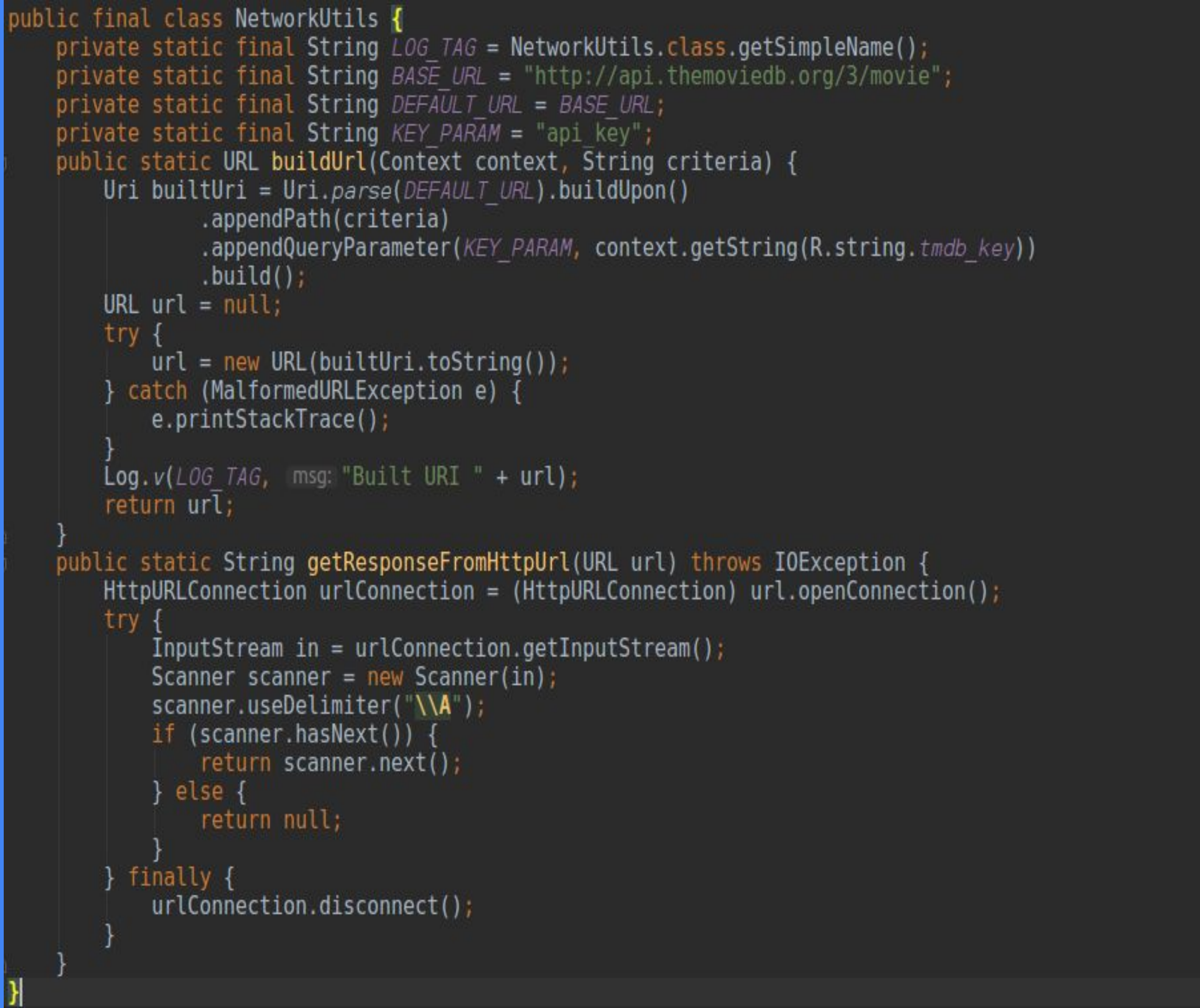


```java
public static String[][] getSimpleMovieStringsFromJson(Context context, String jsonStr) throws JSONException {
    JSONTokener tokener = new JSONTokener(jsonStr);
    while (tokener.more()) Log.d(LOG_TAG, tokener.nextValue().toString());
    JSONObject searchJson = new JSONObject(jsonStr);
    if (searchJson.has(KEY_STATUS_CODE)) {
        int statusCode = searchJson.getInt(KEY_STATUS_CODE);
        String statusMessage = "Search result does not contain a status message";
        if (searchJson.has(KEY_STATUS_MESSAGE)) {
            statusMessage = String.format(
                    "Search result contained a status message of ['%s']",
                    searchJson.getString(KEY_STATUS_MESSAGE));
        }
        switch (statusCode) {
            case HttpURLConnection.HTTP_UNAUTHORIZED:
                Log.e(LOG_TAG, statusMessage);
                return null;
            case HttpURLConnection.HTTP_NOT_FOUND:
                Log.e(LOG_TAG, statusMessage);
                return null;
            default:
                return null;
        }
    }
    JSONArray jsonMovieArray = searchJson.getJSONArray(KEY_RESULTS_ARRAY);
    int arrayLength = jsonMovieArray.length();
    String[][] movieValuesArray = new String[arrayLength][];
    for (int i = 0; i < arrayLength; i++) {
        JSONObject movieJsonObject = jsonMovieArray.getJSONObject(i);
        if (movieJsonObject.getBoolean(KEY_QUALIFIER)) continue;
        int movieId = movieJsonObject.getInt(KEY_MOVIE_ID);
        String title = movieJsonObject.getString(KEY_TITLE);
        String overview = movieJsonObject.getString(KEY_OVERVIEW);
        String language = movieJsonObject.getString(KEY_LANGUAGE);
        String date = movieJsonObject.getString(KEY_DATE);
        String image = movieJsonObject.getString(KEY_IMAGE);
        int votes = movieJsonObject.getInt(KEY_VOTES);
        double rating = movieJsonObject.getDouble(KEY_RATING);
        String[] movieValues = {
                String.valueOf(movieId),
                title,
                overview,
                language,
                date,
                image,
                String.valueOf(votes),
                String.valueOf(rating)};
        movieValuesArray[i] = movieValues;
    }
    return movieValuesArray;
}
```

Stage 1
Dive into code

Tech Webinar:
Popular Movies App
(Stages 1 & 2)

@Bob August 2018

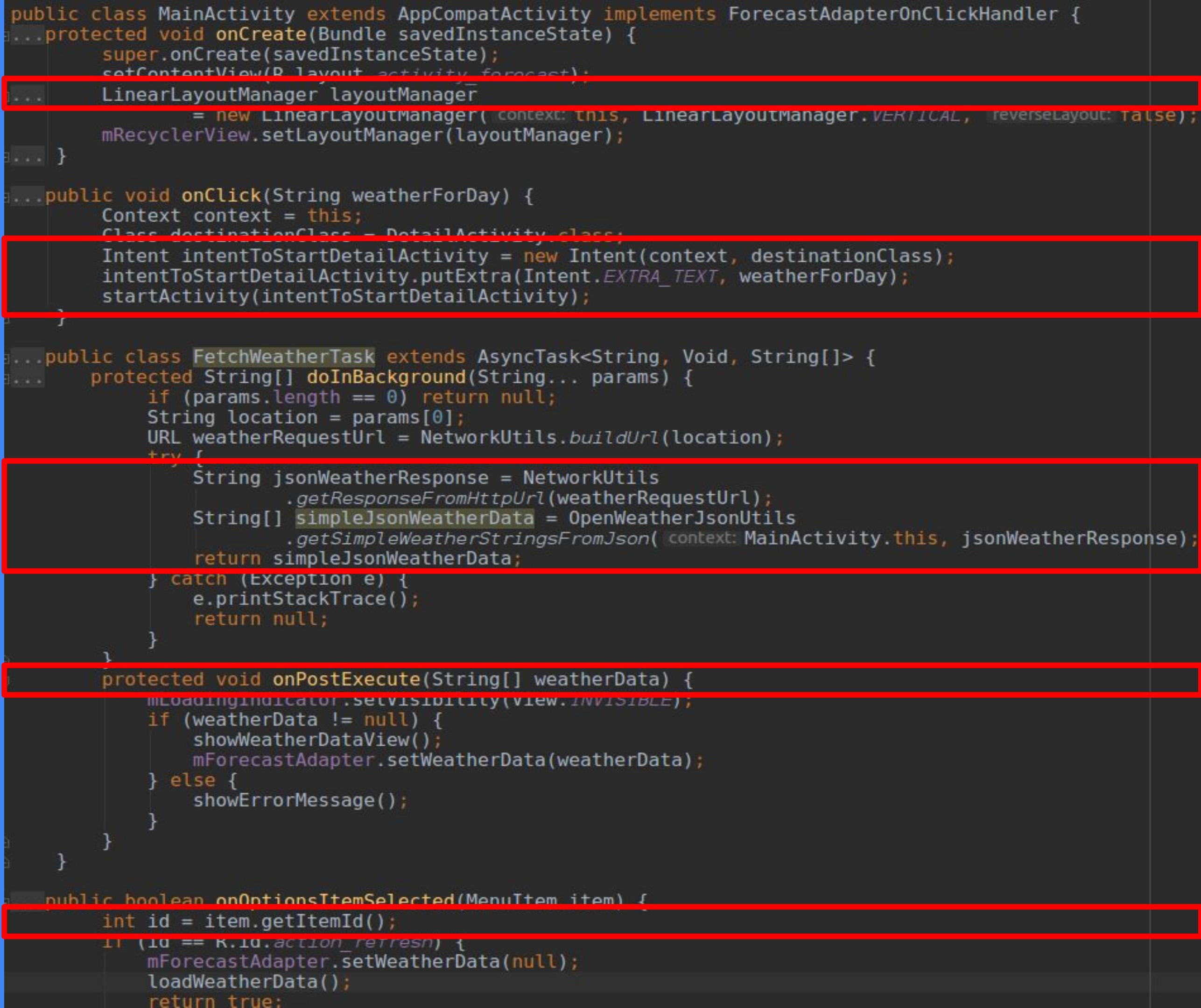


```java
public final class NetworkUtils {
    private static final String LOG_TAG = NetworkUtils.class.getSimpleName();
    private static final String BASE_URL = "http://api.themoviedb.org/3/movie";
    private static final String DEFAULT_URL = BASE_URL;
    private static final String KEY_PARAM = "api_key";
    public static URL buildUrl(Context context, String criteria) {
        Uri builtUri = Uri.parse(DEFAULT_URL).buildUpon()
                .appendPath(criteria)
                .appendQueryParameter(KEY_PARAM, context.getString(R.string.tmdb_key))
                .build();
        URL url = null;
        try {
            url = new URL(builtUri.toString());
        } catch (MalformedURLException e) {
            e.printStackTrace();
        }
        Log.v(LOG_TAG, msg: "Built URI " + url);
        return url;
    }
    public static String getResponseFromHttpUrl(URL url) throws IOException {
        HttpURLConnection urlConnection = (HttpURLConnection) url.openConnection();
        try {
            InputStream in = urlConnection.getInputStream();
            Scanner scanner = new Scanner(in);
            scanner.useDelimiter("\\A");
            if (scanner.hasNext()) {
                return scanner.next();
            } else {
                return null;
            }
        } finally {
            urlConnection.disconnect();
        }
    }
}
```

Stage 1
Dive into code

Tech Webinar:
Popular Movies App
(Stages 1 & 2)

@Bob August 2018

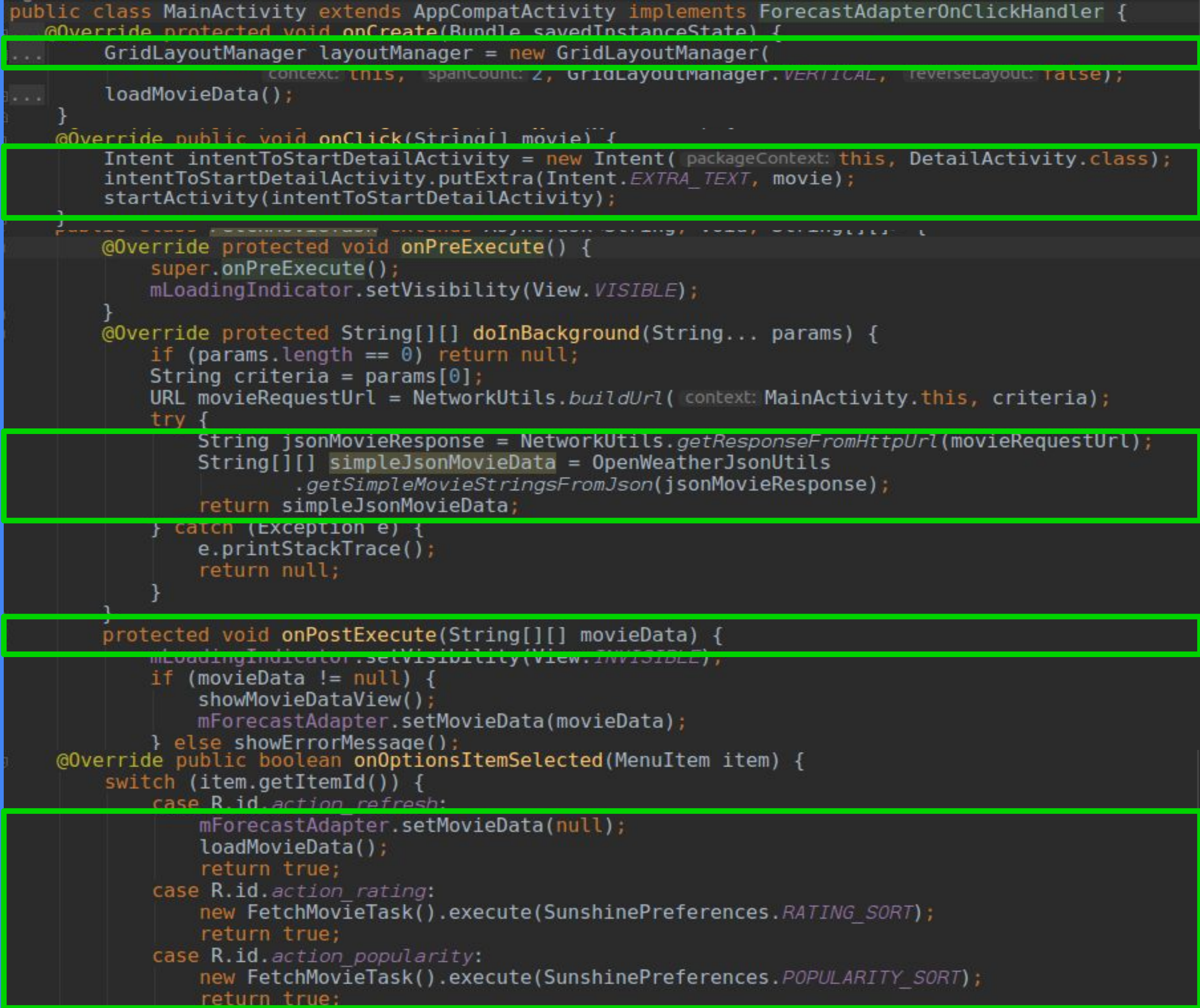


```java
public class MainActivity extends AppCompatActivity implements ForecastAdapterOnClickHandler {
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_forecast);
        LinearLayoutManager layoutManager
                = new LinearLayoutManager( context: this, LinearLayoutManager.VERTICAL,  reverseLayout: false);
        mRecyclerView.setLayoutManager(layoutManager);
    }

    public void onClick(String weatherForDay) {
        Context context = this;
        Class destinationClass = DetailActivity.class;
        Intent intentToStartDetailActivity = new Intent(context, destinationClass);
        intentToStartDetailActivity.putExtra(Intent.EXTRA_TEXT, weatherForDay);
        startActivity(intentToStartDetailActivity);
    }

    public class FetchWeatherTask extends AsyncTask<String, Void, String[]> {
        protected String[] doInBackground(String... params) {
            if (params.length == 0) return null;
            String location = params[0];
            URL weatherRequestUrl = NetworkUtils.buildUrl(location);
            try {
                String jsonWeatherResponse = NetworkUtils
                        .getResponseFromHttpUrl(weatherRequestUrl);
                String[] simpleJsonWeatherData = OpenWeatherJsonUtils
                        .getSimpleWeatherStringsFromJson( context: MainActivity.this, jsonWeatherResponse);
                return simpleJsonWeatherData;
            } catch (Exception e) {
                e.printStackTrace();
                return null;
            }
        }
        protected void onPostExecute(String[] weatherData) {
            mLoadingIndicator.setVisibility(View.INVISIBLE);
            if (weatherData != null) {
                showWeatherDataView();
                mForecastAdapter.setWeatherData(weatherData);
            } else {
                showErrorMessage();
            }
        }
    }

    public boolean onOptionsItemSelected(MenuItem item) {
        int id = item.getItemId();
        if (id == R.id.action_refresh) {
            mForecastAdapter.setWeatherData(null);
            loadWeatherData();
            return true;
```

Stage 1
Dive into code

Tech Webinar:
Popular Movies App
(Stages 1 & 2)

@Bob August 2018

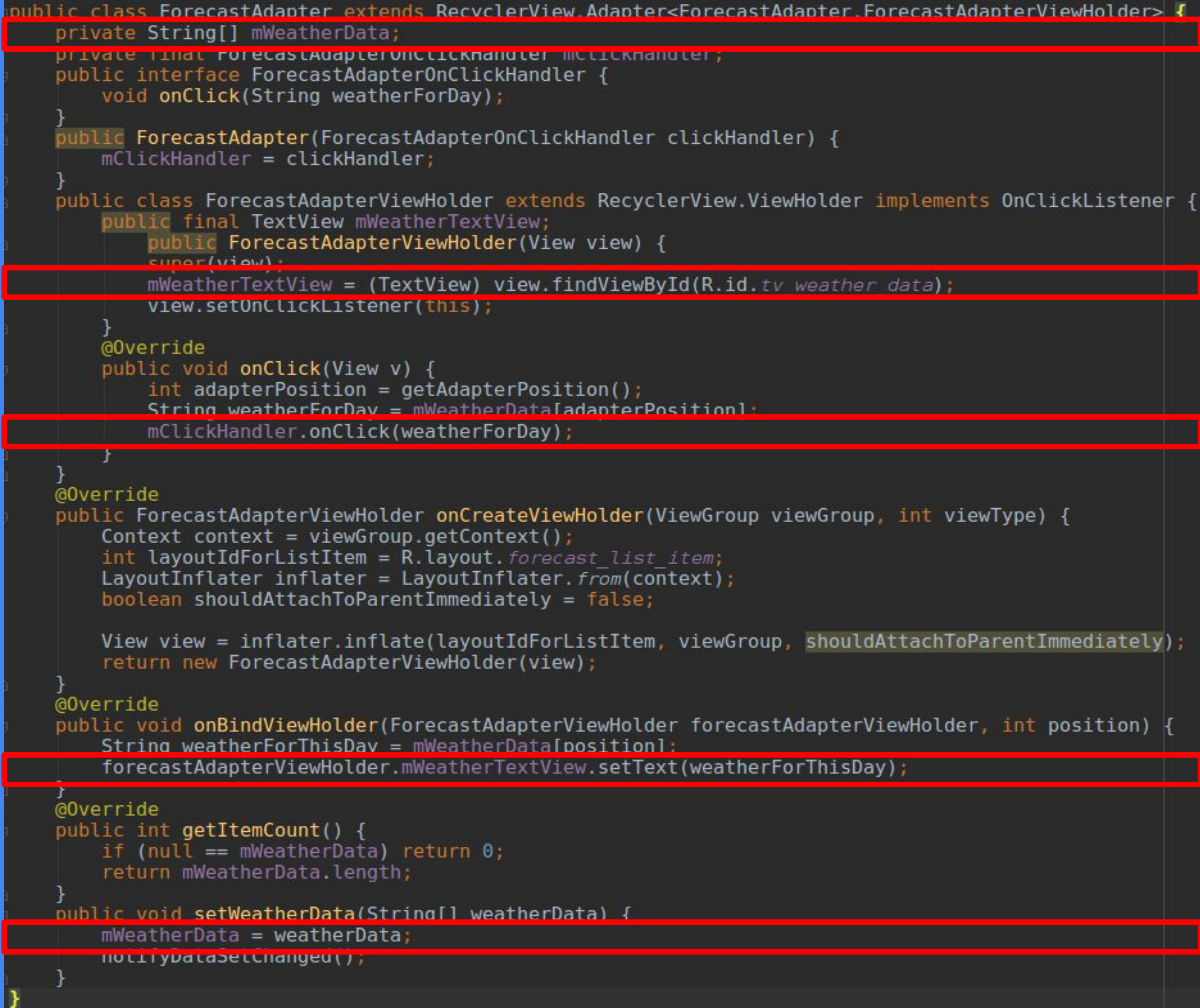


```
public class MainActivity extends AppCompatActivity implements ForecastAdapterOnClickHandler {
  @Override protected void onCreate(Bundle savedInstanceState) {
        GridLayoutManager layoutManager = new GridLayoutManager(
                context: this, spanCount: 2, GridLayoutManager.VERTICAL, reverseLayout: false);
        loadMovieData();
    }
    @Override public void onClick(String[] movie) {
        Intent intentToStartDetailActivity = new Intent( packageContext: this, DetailActivity.class);
        intentToStartDetailActivity.putExtra(Intent.EXTRA_TEXT, movie);
        startActivity(intentToStartDetailActivity);
    }
    public class FetchMovieTask extends AsyncTask<String, Void, String[][]> {
        @Override protected void onPreExecute() {
            super.onPreExecute();
            mLoadingIndicator.setVisibility(View.VISIBLE);
        }
        @Override protected String[][] doInBackground(String... params) {
            if (params.length == 0) return null;
            String criteria = params[0];
            URL movieRequestUrl = NetworkUtils.buildUrl( context: MainActivity.this, criteria);
            try {
                String jsonMovieResponse = NetworkUtils.getResponseFromHttpUrl(movieRequestUrl);
                String[][] simpleJsonMovieData = OpenWeatherJsonUtils
                        .getSimpleMovieStringsFromJson(jsonMovieResponse);
                return simpleJsonMovieData;
            } catch (Exception e) {
                e.printStackTrace();
                return null;
            }
        }
        protected void onPostExecute(String[][] movieData) {
            mLoadingIndicator.setVisibility(View.INVISIBLE);
            if (movieData != null) {
                showMovieDataView();
                mForecastAdapter.setMovieData(movieData);
            } else showErrorMessage();
    @Override public boolean onOptionsItemSelected(MenuItem item) {
        switch (item.getItemId()) {
            case R.id.action_refresh:
                mForecastAdapter.setMovieData(null);
                loadMovieData();
                return true;
            case R.id.action_rating:
                new FetchMovieTask().execute(SunshinePreferences.RATING_SORT);
                return true;
            case R.id.action_popularity:
                new FetchMovieTask().execute(SunshinePreferences.POPULARITY_SORT);
                return true;
```

Stage 1
Dive into code

Tech Webinar:
Popular Movies App
(Stages 1 & 2)

@Bob August 2018

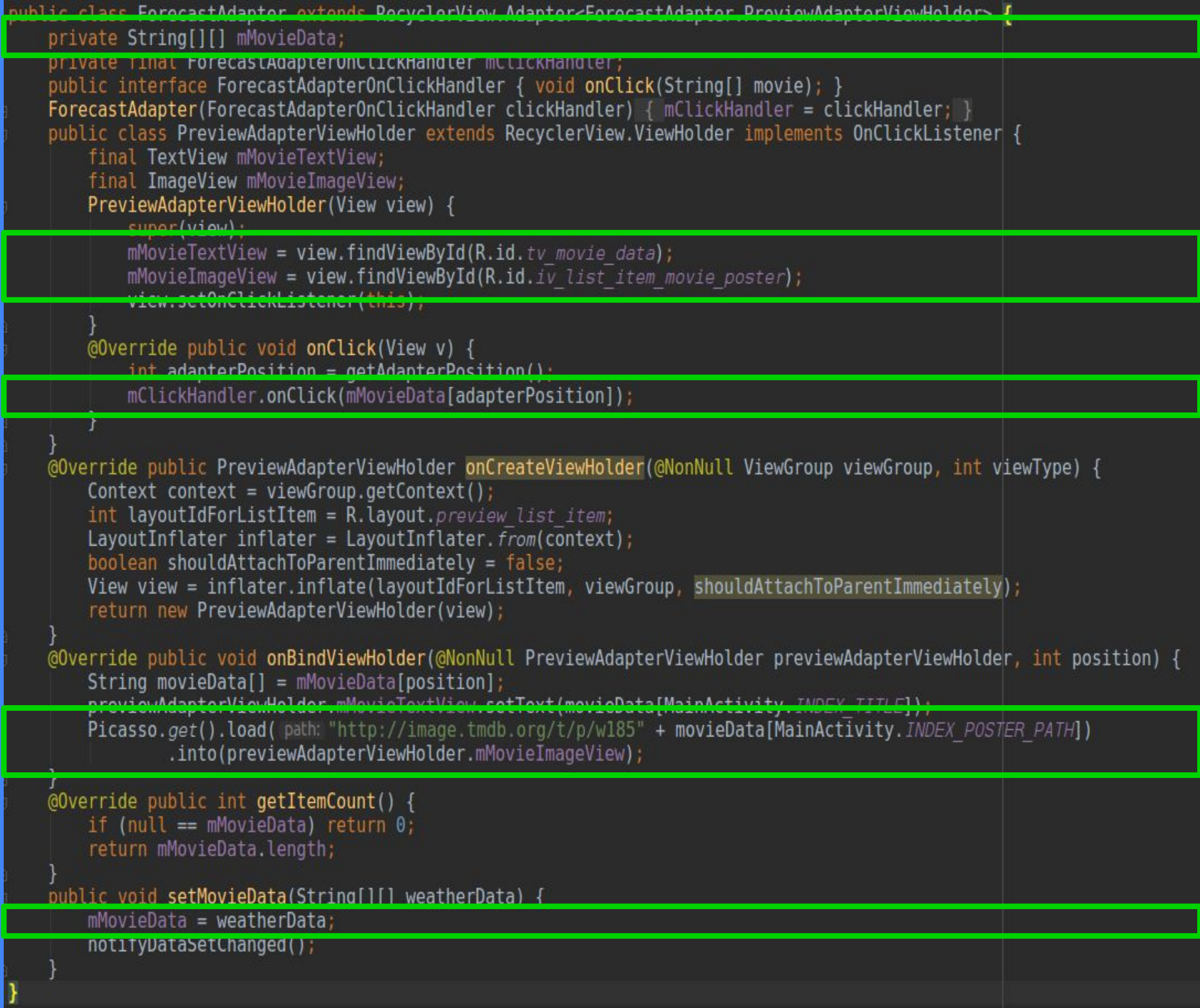


```
public class ForecastAdapter extends RecyclerView.Adapter<ForecastAdapter.ForecastAdapterViewHolder> {
    private String[] mWeatherData;
    private final ForecastAdapterOnClickHandler mClickHandler;
    public interface ForecastAdapterOnClickHandler {
        void onClick(String weatherForDay);
    }
    public ForecastAdapter(ForecastAdapterOnClickHandler clickHandler) {
        mClickHandler = clickHandler;
    }
    public class ForecastAdapterViewHolder extends RecyclerView.ViewHolder implements OnClickListener {
        public final TextView mWeatherTextView;
            public ForecastAdapterViewHolder(View view) {
            super(view);
            mWeatherTextView = (TextView) view.findViewById(R.id.tv_weather_data);
            view.setOnClickListener(this);
        }
        @Override
        public void onClick(View v) {
            int adapterPosition = getAdapterPosition();
            String weatherForDay = mWeatherData[adapterPosition];
            mClickHandler.onClick(weatherForDay);
        }
    }
    @Override
    public ForecastAdapterViewHolder onCreateViewHolder(ViewGroup viewGroup, int viewType) {
        Context context = viewGroup.getContext();
        int layoutIdForListItem = R.layout.forecast_list_item;
        LayoutInflater inflater = LayoutInflater.from(context);
        boolean shouldAttachToParentImmediately = false;

        View view = inflater.inflate(layoutIdForListItem, viewGroup, shouldAttachToParentImmediately);
        return new ForecastAdapterViewHolder(view);
    }
    @Override
    public void onBindViewHolder(ForecastAdapterViewHolder forecastAdapterViewHolder, int position) {
        String weatherForThisDay = mWeatherData[position];
        forecastAdapterViewHolder.mWeatherTextView.setText(weatherForThisDay);
    }
    @Override
    public int getItemCount() {
        if (null == mWeatherData) return 0;
        return mWeatherData.length;
    }
    public void setWeatherData(String[] weatherData) {
        mWeatherData = weatherData;
        notifyDataSetChanged();
    }
}
```

Stage 1
Dive into code

Tech Webinar:
Popular Movies App
(Stages 1 & 2)

@Bob August 2018

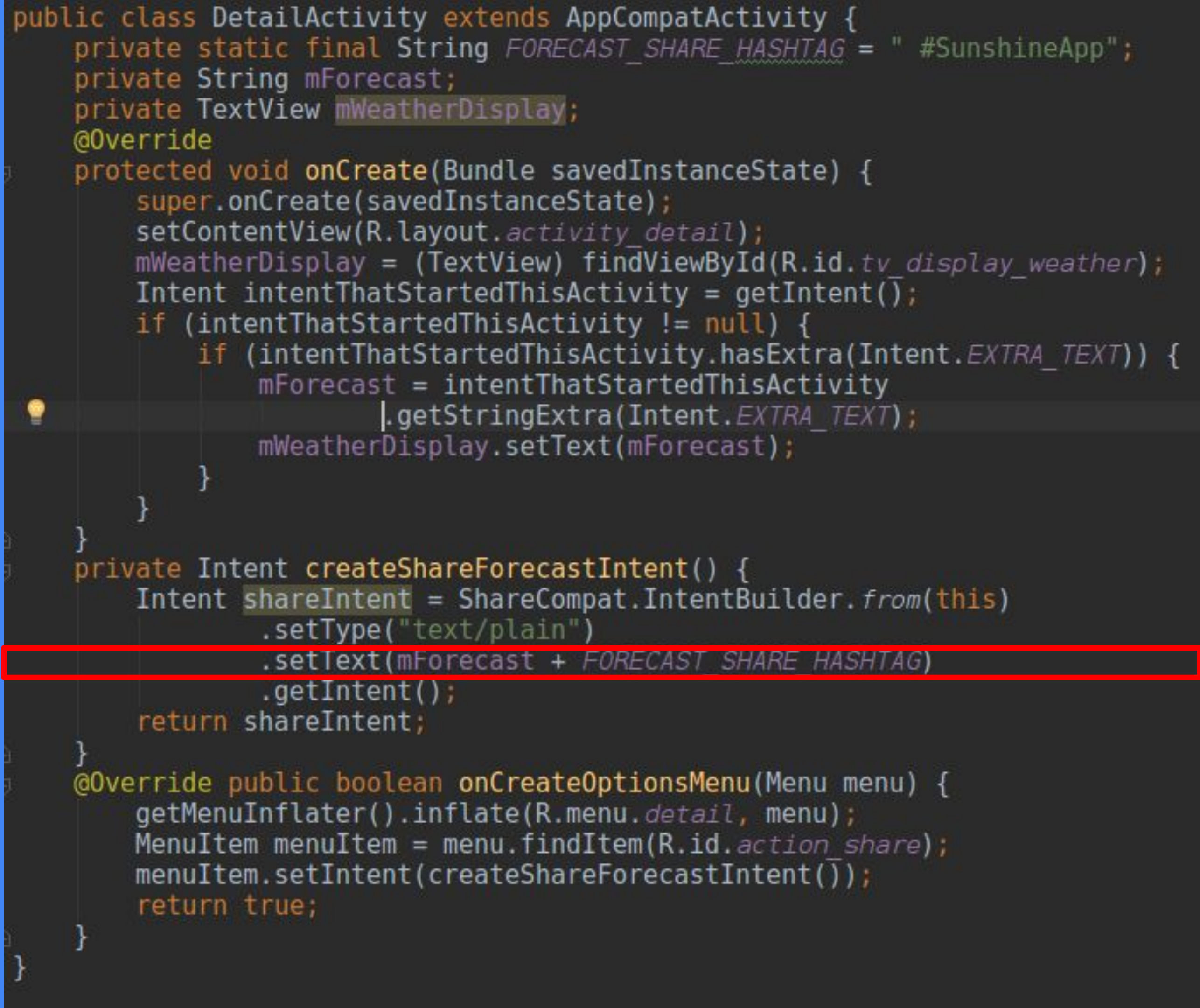


```
public class ForecastAdapter extends RecyclerView.Adapter<ForecastAdapter.PreviewAdapterViewHolder> {
    private String[][] mMovieData;
    private final ForecastAdapterOnClickHandler mClickHandler;
    public interface ForecastAdapterOnClickHandler { void onClick(String[] movie); }
    ForecastAdapter(ForecastAdapterOnClickHandler clickHandler) { mClickHandler = clickHandler; }
    public class PreviewAdapterViewHolder extends RecyclerView.ViewHolder implements OnClickListener {
        final TextView mMovieTextView;
        final ImageView mMovieImageView;
        PreviewAdapterViewHolder(View view) {
            super(view);
            mMovieTextView = view.findViewById(R.id.tv_movie_data);
            mMovieImageView = view.findViewById(R.id.iv_list_item_movie_poster);
            view.setOnClickListener(this);
        }
        @Override public void onClick(View v) {
            int adapterPosition = getAdapterPosition();
            mClickHandler.onClick(mMovieData[adapterPosition]);
        }
    }
    @Override public PreviewAdapterViewHolder onCreateViewHolder(@NonNull ViewGroup viewGroup, int viewType) {
        Context context = viewGroup.getContext();
        int layoutIdForListItem = R.layout.preview_list_item;
        LayoutInflater inflater = LayoutInflater.from(context);
        boolean shouldAttachToParentImmediately = false;
        View view = inflater.inflate(layoutIdForListItem, viewGroup, shouldAttachToParentImmediately);
        return new PreviewAdapterViewHolder(view);
    }
    @Override public void onBindViewHolder(@NonNull PreviewAdapterViewHolder previewAdapterViewHolder, int position) {
        String movieData[] = mMovieData[position];
        previewAdapterViewHolder.mMovieTextView.setText(movieData[MainActivity.INDEX_TITLE]);
        Picasso.get().load( path: "http://image.tmdb.org/t/p/w185" + movieData[MainActivity.INDEX_POSTER_PATH])
                .into(previewAdapterViewHolder.mMovieImageView);
    }
    @Override public int getItemCount() {
        if (null == mMovieData) return 0;
        return mMovieData.length;
    }
    public void setMovieData(String[][] weatherData) {
        mMovieData = weatherData;
        notifyDataSetChanged();
    }
}
```

Stage 1
Dive into code

Tech Webinar:
Popular Movies App
(Stages 1 & 2)

@Bob August 2018

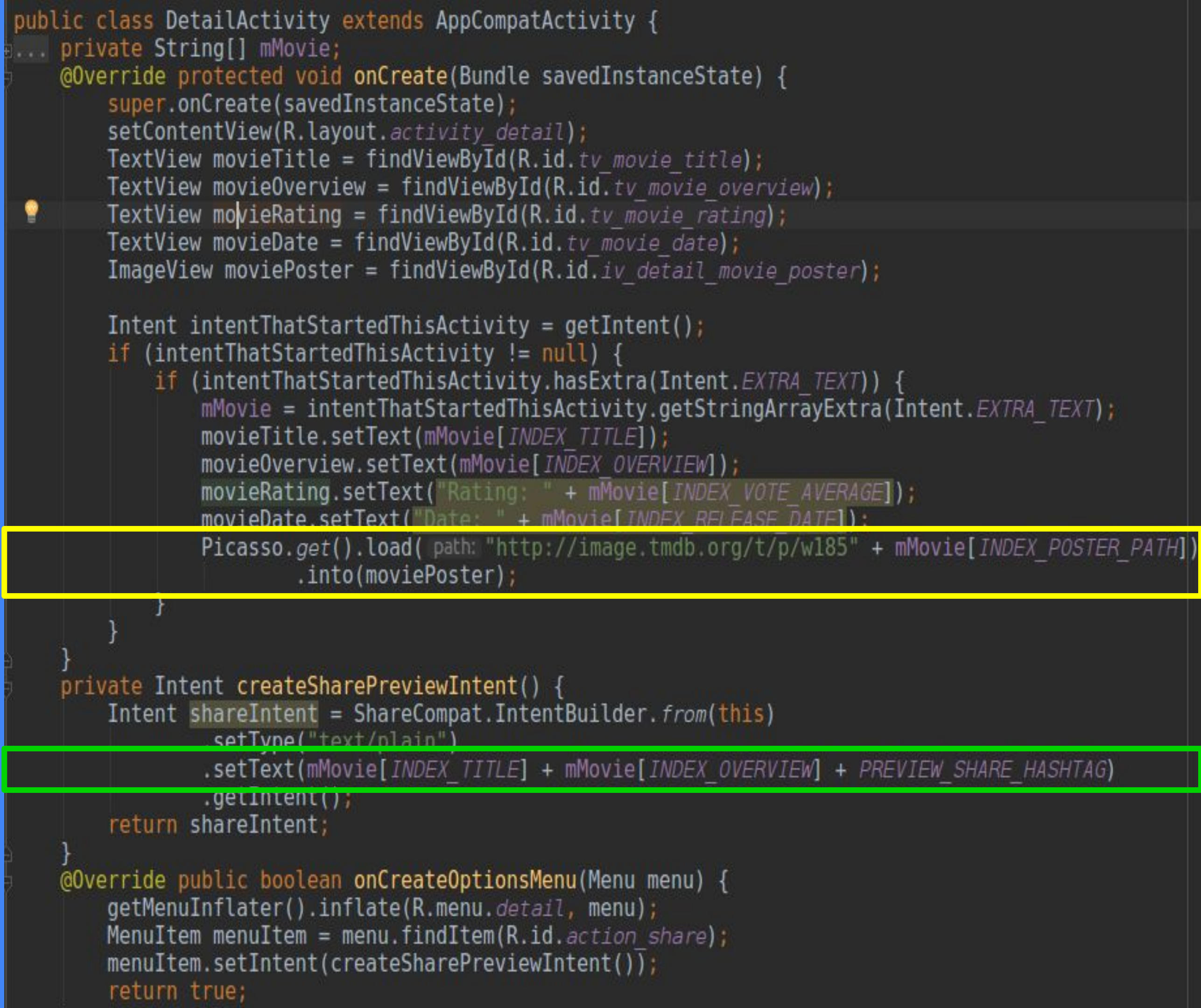


```java
public class DetailActivity extends AppCompatActivity {
    private static final String FORECAST_SHARE_HASHTAG = " #SunshineApp";
    private String mForecast;
    private TextView mWeatherDisplay;
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_detail);
        mWeatherDisplay = (TextView) findViewById(R.id.tv_display_weather);
        Intent intentThatStartedThisActivity = getIntent();
        if (intentThatStartedThisActivity != null) {
            if (intentThatStartedThisActivity.hasExtra(Intent.EXTRA_TEXT)) {
                mForecast = intentThatStartedThisActivity
                        .getStringExtra(Intent.EXTRA_TEXT);
                mWeatherDisplay.setText(mForecast);
            }
        }
    }
    private Intent createShareForecastIntent() {
        Intent shareIntent = ShareCompat.IntentBuilder.from(this)
                .setType("text/plain")
                .setText(mForecast + FORECAST_SHARE_HASHTAG)
                .getIntent();
        return shareIntent;
    }
    @Override public boolean onCreateOptionsMenu(Menu menu) {
        getMenuInflater().inflate(R.menu.detail, menu);
        MenuItem menuItem = menu.findItem(R.id.action_share);
        menuItem.setIntent(createShareForecastIntent());
        return true;
    }
}
```

Stage 1
Dive into code

Tech Webinar:
Popular Movies App
(Stages 1 & 2)

@Bob August 2018

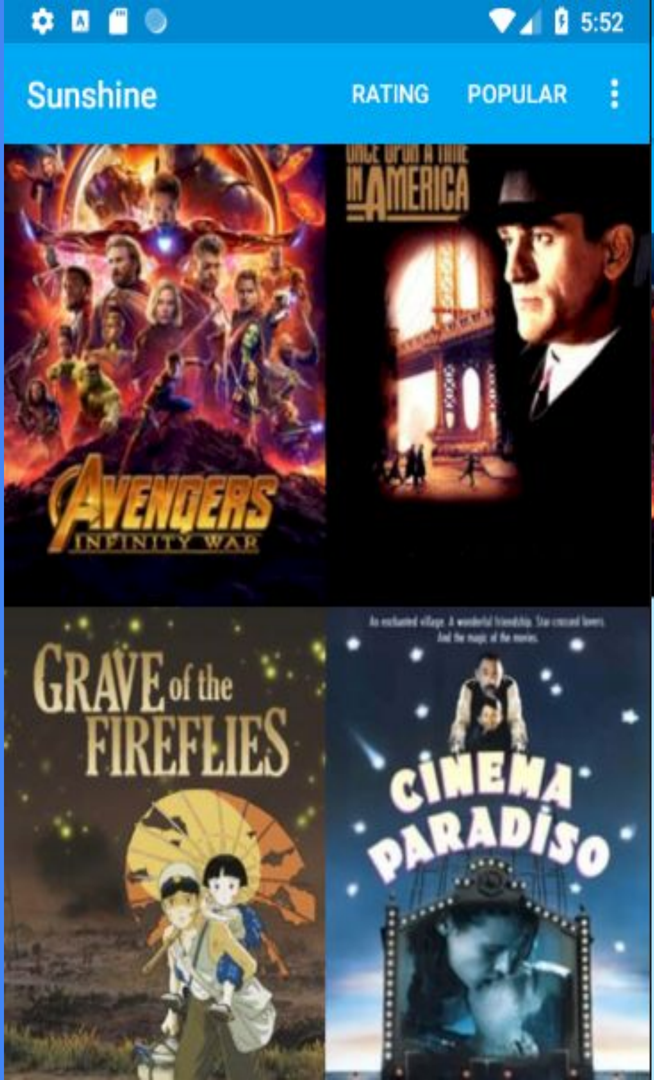


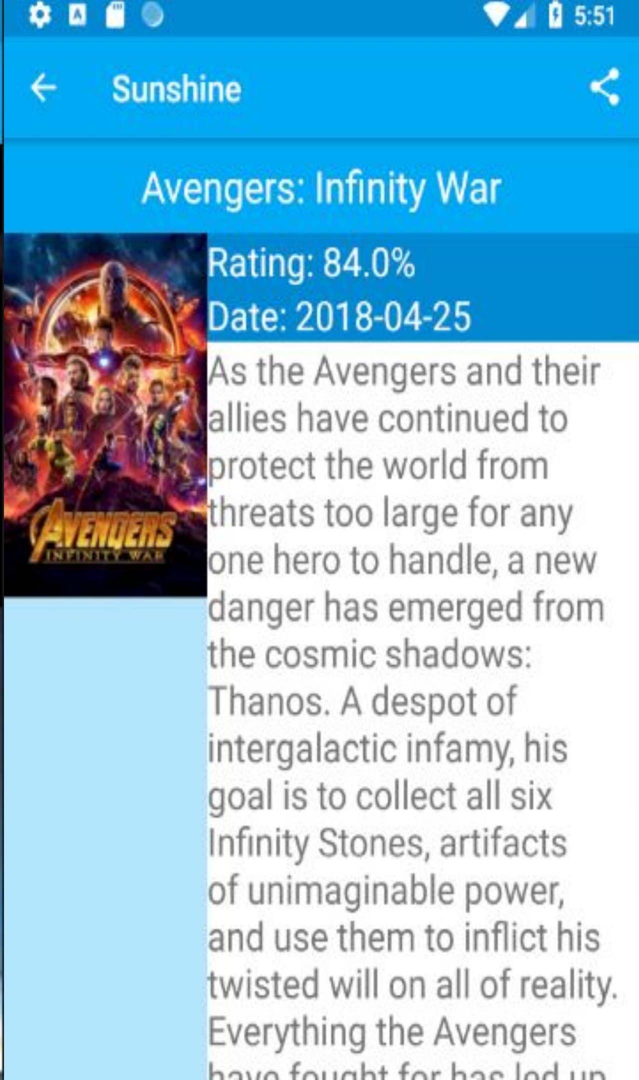
```
public class DetailActivity extends AppCompatActivity {
... private String[] mMovie;
    @Override protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_detail);
        TextView movieTitle = findViewById(R.id.tv_movie_title);
        TextView movieOverview = findViewById(R.id.tv_movie_overview);
        TextView movieRating = findViewById(R.id.tv_movie_rating);
        TextView movieDate = findViewById(R.id.tv_movie_date);
        ImageView moviePoster = findViewById(R.id.iv_detail_movie_poster);

        Intent intentThatStartedThisActivity = getIntent();
        if (intentThatStartedThisActivity != null) {
            if (intentThatStartedThisActivity.hasExtra(Intent.EXTRA_TEXT)) {
                mMovie = intentThatStartedThisActivity.getStringArrayExtra(Intent.EXTRA_TEXT);
                movieTitle.setText(mMovie[INDEX_TITLE]);
                movieOverview.setText(mMovie[INDEX_OVERVIEW]);
                movieRating.setText("Rating: " + mMovie[INDEX_VOTE_AVERAGE]);
                movieDate.setText("Date: " + mMovie[INDEX_RELEASE_DATE]);
                Picasso.get().load( path: "http://image.tmdb.org/t/p/w185" + mMovie[INDEX_POSTER_PATH])
                        .into(moviePoster);
            }
        }
    }
    private Intent createSharePreviewIntent() {
        Intent shareIntent = ShareCompat.IntentBuilder.from(this)
                .setType("text/plain")
                .setText(mMovie[INDEX_TITLE] + mMovie[INDEX_OVERVIEW] + PREVIEW_SHARE_HASHTAG)
                .getIntent();
        return shareIntent;
    }
    @Override public boolean onCreateOptionsMenu(Menu menu) {
        getMenuInflater().inflate(R.menu.detail, menu);
        MenuItem menuItem = menu.findItem(R.id.action_share);
        menuItem.setIntent(createSharePreviewIntent());
        return true;
```

Stage 1
Test your app

Tech Webinar:
Popular Movies App
(Stages 1 & 2)

@Bob August 2018



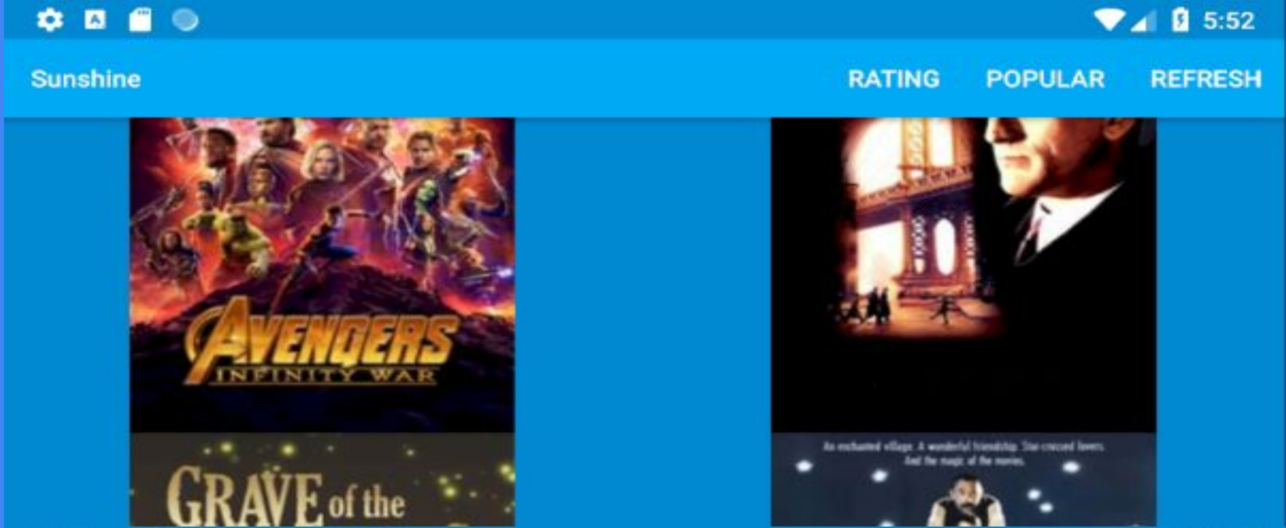


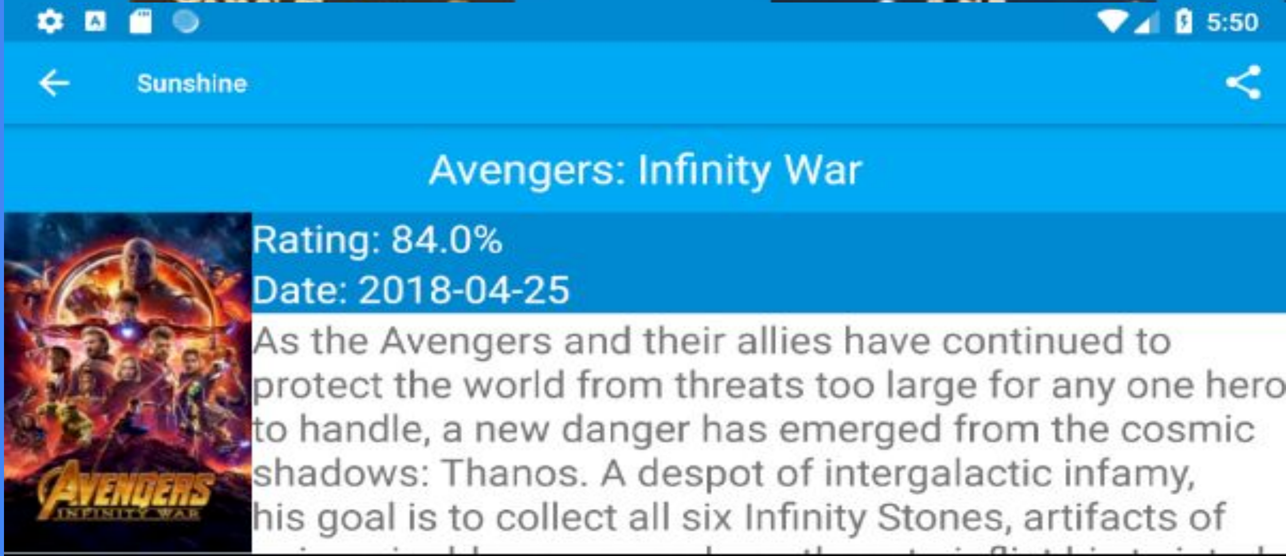
Avengers: Infinity War

Rating: 84.0%
Date: 2018-04-25

As the Avengers and their allies have continued to protect the world from threats too large for any one hero to handle, a new danger has emerged from the cosmic shadows: Thanos. A despot of intergalactic infamy, his goal is to collect all six Infinity Stones, artifacts of unimaginable power, and use them to inflict his twisted will on all of reality. Everything the Avengers

Stage 1
Test your app





Avengers: Infinity War

Rating: 84.0%
Date: 2018-04-25
As the Avengers and their allies have continued to protect the world from threats too large for any one hero to handle, a new danger has emerged from the cosmic shadows: Thanos. A despot of intergalactic infamy, his goal is to collect all six Infinity Stones, artifacts of

Tech Webinar:
Popular Movies App
(Stages 1 & 2)

@Bob August 2018

Stage 2

Trailers, Reviews, and Favorites

Stage 2
Review foundational knowledge

- Data
  - Request (HttpUrlConnection)
  - Preference (SharedPreferences)
  - Database (ContentProvider, Room)
- UI
  - Activity, Fragment (Explicit Intent)
  - Lifecycle (SavedInstanceState)
  - ViewGroup, View, Menu, Drawable
  - Responsive Design
- Sync
  - Thread (AsyncTask)
  - Service (IntentService)
  - Callback (Loader, LiveData)
  - Broadcast (Implicit Intent)

Stage 2
Understand assignment requirements

- Rubric
  - **Meet all Stage 1 requirements**
  - **Request additional data** from TMDb Movies 'Videos' and 'Reviews' API on worker thread
    - **Display trailer video links and user reviews** in detail layout
  - **Launch a video player** with an Intent when a trailer video link is selected
  - **Store data** in SQLiteDatabase and expose via ContentProvider or Room
    - If Room is used, query the database only as often as necessary
  - **Update data** when users save a movie to their collection

Stage 2
Understand assignment requirements

- Guide
    - Request additional data from the reviews and videos endpoints
        - Construct URL: http://api.themoviedb.org/3/movie/[movieId]/[endpoint]?api_key=[K]
    - Update data by implementing a star button that adds/removes movies to/from a collection
        - Store the collection locally such that API requests are unnecessary
        - Add additional sort criteria to display only collected movies in the main layout
    - Recreate Activities with lifecycle methods such that the app preserves its state on rotation

Stage 2
Map knowledge to requirements

Requirement	Knowledge
Request data over the network in the background	Request, Thread, Service, Callback
Launch a native or web-based video player	Broadcast
Locally store and expose data	Database, Thread
Update data when saving to a collection	Menu, Thread

Stage 2
Survey alternatives

Knowledge	Alternative
Request, Thread, Service, Callback	HttpUrlConnection, RetroFit | AsyncTask, IntentService
Broadcast	Implicit Intent
Database, Thread	ContentProvider, Room | Loader, LiveData
Menu, Thread	Settings, Spinner | SharedPreferences | AsyncTask

Stage 2
Survey alternatives

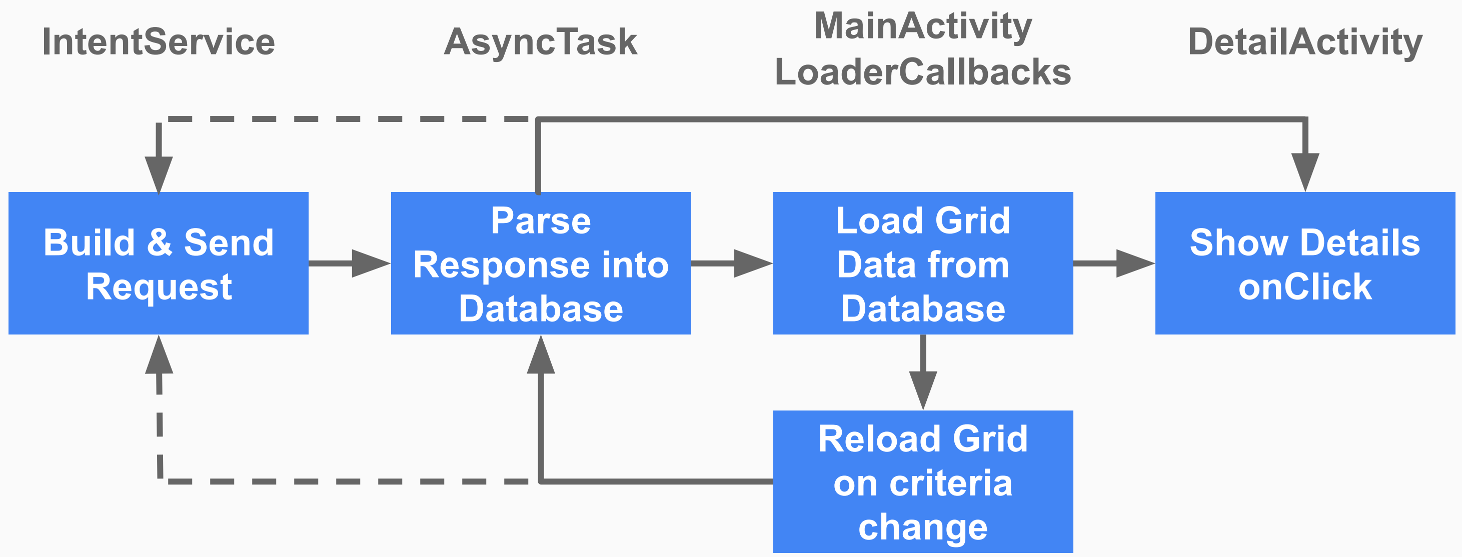
Select tools that suit you or your team’s bandwidth, skill level, and timeline.

- Ideal: Learn and implement RetroFit, Room, LiveData, Glide and Spinner
- Major constraint: Timeline (one month overdue)
- Recommendation: Leverage knowledge with existing codebase (Sunshine)

Stage 2
Decide structure and tools

Choice	Plan
HttpUrlConnection | AsyncTask, IntentService	Run HttpUrlConnection over IntentService; Parse response into ContentValues
Implicit Intent	Activate an implicit Intent with appropriate action
ContentProvider | Loader	Insert ContentValues into ContentProvider; Expose changes via LoaderCallbacks
Settings | SharedPreferences | AsyncTask	Insert and delete ContentProvider row corresponding to saved/unsaved movie

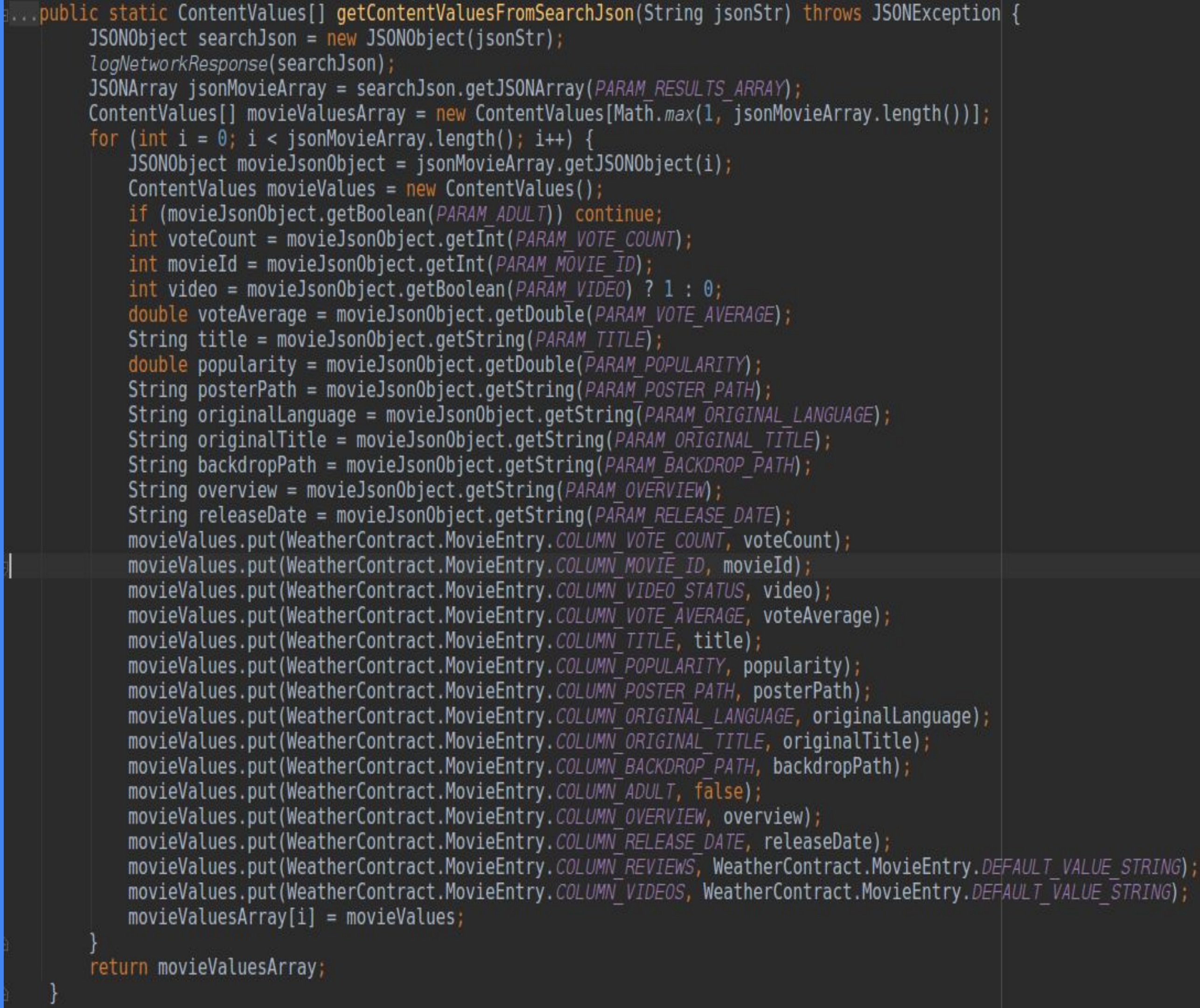
Stage 2
Decide structure and tools


IntentService
AsyncTask
MainActivity LoaderCallbacks
DetailActivity
Build & Send Request
Parse Response into Database
Load Grid Data from Database
Show Details onClick
Reload Grid on criteria change

Stage 2
Dive into code

Tech Webinar:
Popular Movies App
(Stages 1 & 2)

@Bob August 2018

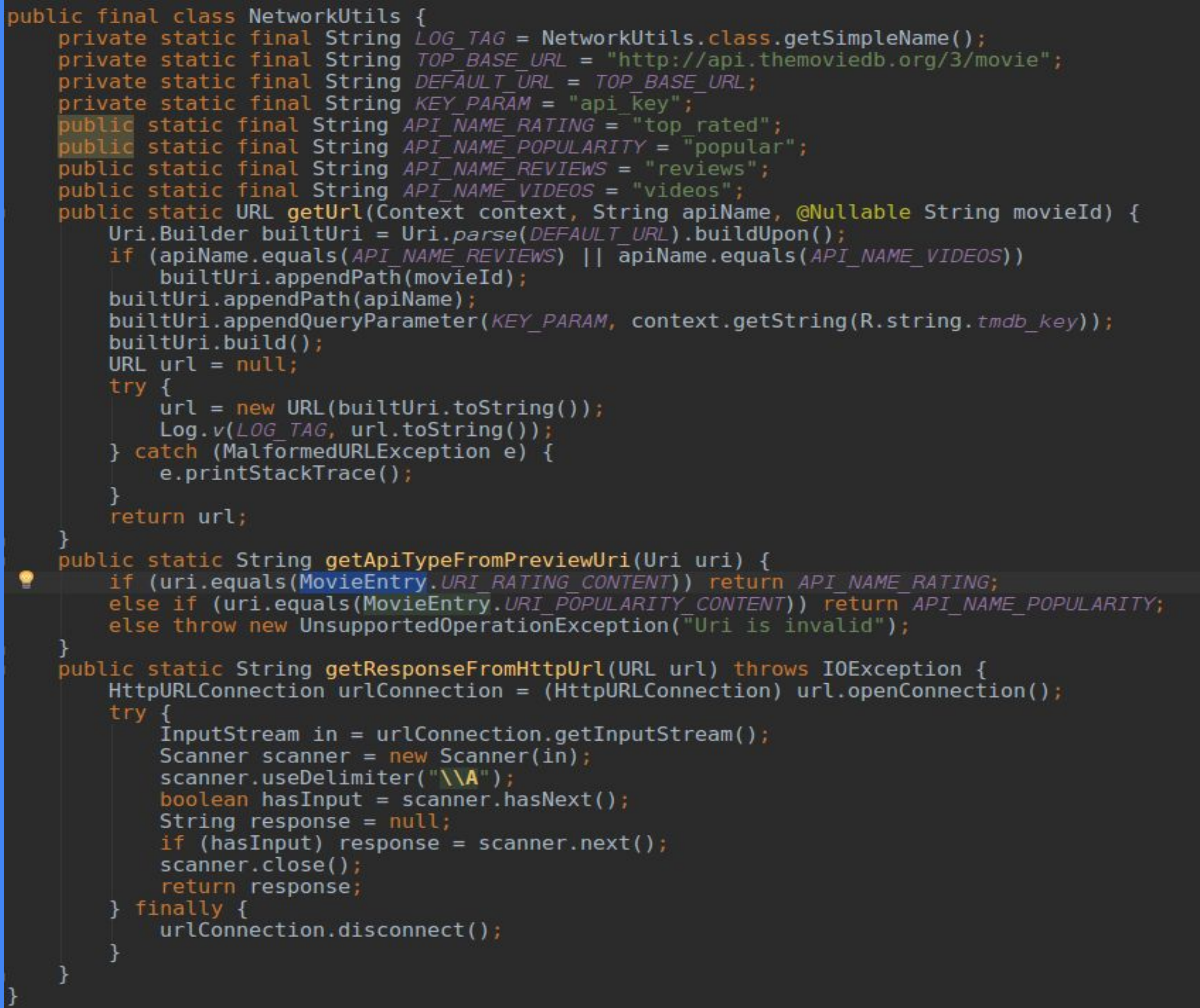


```java
public static ContentValues[] getContentValuesFromSearchJson(String jsonStr) throws JSONException {
    JSONObject searchJson = new JSONObject(jsonStr);
    logNetworkResponse(searchJson);
    JSONArray jsonMovieArray = searchJson.getJSONArray(PARAM_RESULTS_ARRAY);
    ContentValues[] movieValuesArray = new ContentValues[Math.max(1, jsonMovieArray.length())];
    for (int i = 0; i < jsonMovieArray.length(); i++) {
        JSONObject movieJsonObject = jsonMovieArray.getJSONObject(i);
        ContentValues movieValues = new ContentValues();
        if (movieJsonObject.getBoolean(PARAM_ADULT)) continue;
        int voteCount = movieJsonObject.getInt(PARAM_VOTE_COUNT);
        int movieId = movieJsonObject.getInt(PARAM_MOVIE_ID);
        int video = movieJsonObject.getBoolean(PARAM_VIDEO) ? 1 : 0;
        double voteAverage = movieJsonObject.getDouble(PARAM_VOTE_AVERAGE);
        String title = movieJsonObject.getString(PARAM_TITLE);
        double popularity = movieJsonObject.getDouble(PARAM_POPULARITY);
        String posterPath = movieJsonObject.getString(PARAM_POSTER_PATH);
        String originalLanguage = movieJsonObject.getString(PARAM_ORIGINAL_LANGUAGE);
        String originalTitle = movieJsonObject.getString(PARAM_ORIGINAL_TITLE);
        String backdropPath = movieJsonObject.getString(PARAM_BACKDROP_PATH);
        String overview = movieJsonObject.getString(PARAM_OVERVIEW);
        String releaseDate = movieJsonObject.getString(PARAM_RELEASE_DATE);
        movieValues.put(WeatherContract.MovieEntry.COLUMN_VOTE_COUNT, voteCount);
        movieValues.put(WeatherContract.MovieEntry.COLUMN_MOVIE_ID, movieId);
        movieValues.put(WeatherContract.MovieEntry.COLUMN_VIDEO_STATUS, video);
        movieValues.put(WeatherContract.MovieEntry.COLUMN_VOTE_AVERAGE, voteAverage);
        movieValues.put(WeatherContract.MovieEntry.COLUMN_TITLE, title);
        movieValues.put(WeatherContract.MovieEntry.COLUMN_POPULARITY, popularity);
        movieValues.put(WeatherContract.MovieEntry.COLUMN_POSTER_PATH, posterPath);
        movieValues.put(WeatherContract.MovieEntry.COLUMN_ORIGINAL_LANGUAGE, originalLanguage);
        movieValues.put(WeatherContract.MovieEntry.COLUMN_ORIGINAL_TITLE, originalTitle);
        movieValues.put(WeatherContract.MovieEntry.COLUMN_BACKDROP_PATH, backdropPath);
        movieValues.put(WeatherContract.MovieEntry.COLUMN_ADULT, false);
        movieValues.put(WeatherContract.MovieEntry.COLUMN_OVERVIEW, overview);
        movieValues.put(WeatherContract.MovieEntry.COLUMN_RELEASE_DATE, releaseDate);
        movieValues.put(WeatherContract.MovieEntry.COLUMN_REVIEWS, WeatherContract.MovieEntry.DEFAULT_VALUE_STRING);
        movieValues.put(WeatherContract.MovieEntry.COLUMN_VIDEOS, WeatherContract.MovieEntry.DEFAULT_VALUE_STRING);
        movieValuesArray[i] = movieValues;
    }
    return movieValuesArray;
}
```

Stage 2
Dive into code

Tech Webinar:
Popular Movies App
(Stages 1 & 2)

@Bob August 2018

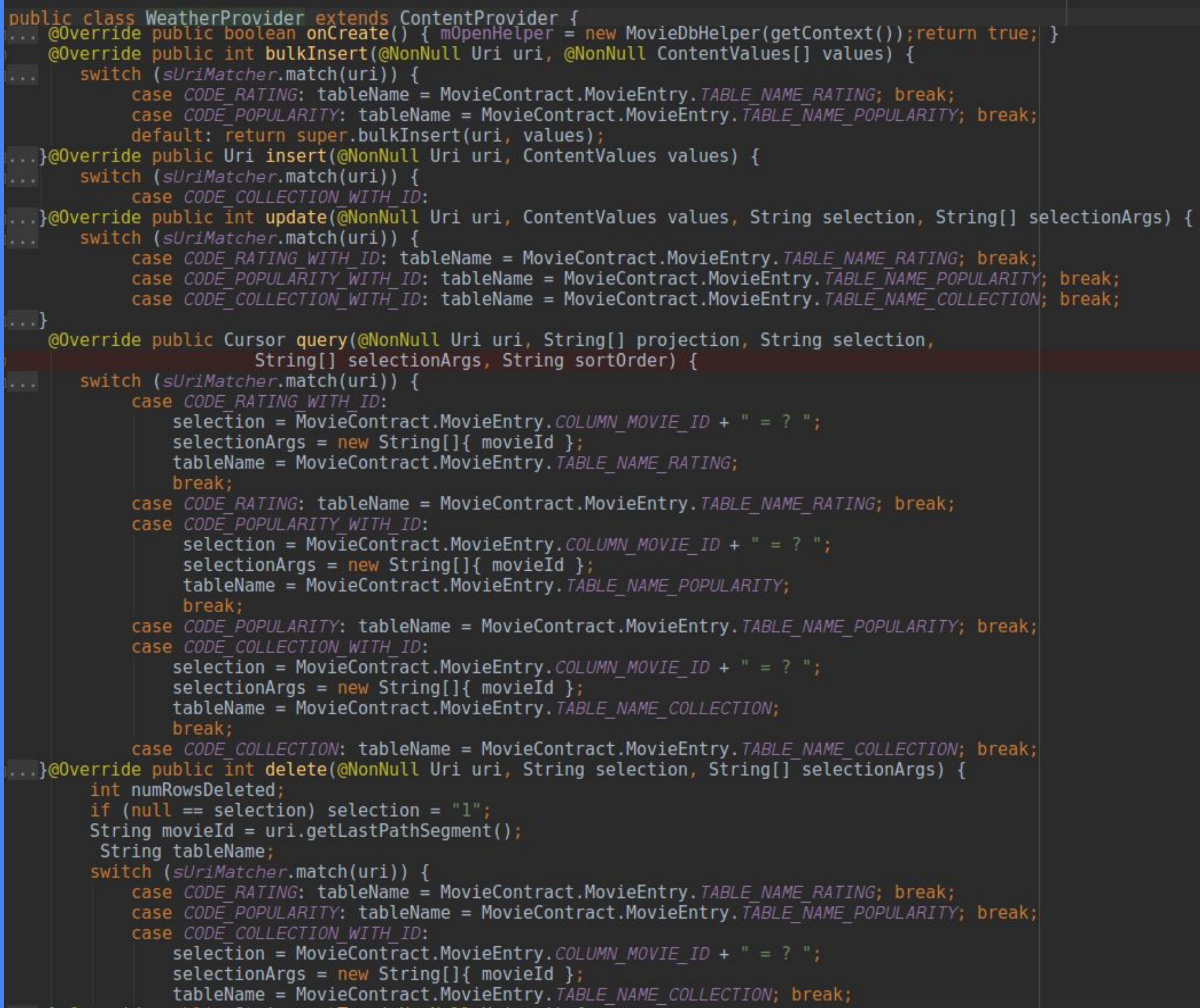


```java
public final class NetworkUtils {
    private static final String LOG_TAG = NetworkUtils.class.getSimpleName();
    private static final String TOP_BASE_URL = "http://api.themoviedb.org/3/movie";
    private static final String DEFAULT_URL = TOP_BASE_URL;
    private static final String KEY_PARAM = "api_key";
    public static final String API_NAME_RATING = "top_rated";
    public static final String API_NAME_POPULARITY = "popular";
    public static final String API_NAME_REVIEWS = "reviews";
    public static final String API_NAME_VIDEOS = "videos";
    public static URL getUrl(Context context, String apiName, @Nullable String movieId) {
        Uri.Builder builtUri = Uri.parse(DEFAULT_URL).buildUpon();
        if (apiName.equals(API_NAME_REVIEWS) || apiName.equals(API_NAME_VIDEOS))
            builtUri.appendPath(movieId);
        builtUri.appendPath(apiName);
        builtUri.appendQueryParameter(KEY_PARAM, context.getString(R.string.tmdb_key));
        builtUri.build();
        URL url = null;
        try {
            url = new URL(builtUri.toString());
            Log.v(LOG_TAG, url.toString());
        } catch (MalformedURLException e) {
            e.printStackTrace();
        }
        return url;
    }
    public static String getApiTypeFromPreviewUri(Uri uri) {
        if (uri.equals(MovieEntry.URI_RATING_CONTENT)) return API_NAME_RATING;
        else if (uri.equals(MovieEntry.URI_POPULARITY_CONTENT)) return API_NAME_POPULARITY;
        else throw new UnsupportedOperationException("Uri is invalid");
    }
    public static String getResponseFromHttpUrl(URL url) throws IOException {
        HttpURLConnection urlConnection = (HttpURLConnection) url.openConnection();
        try {
            InputStream in = urlConnection.getInputStream();
            Scanner scanner = new Scanner(in);
            scanner.useDelimiter("\\A");
            boolean hasInput = scanner.hasNext();
            String response = null;
            if (hasInput) response = scanner.next();
            scanner.close();
            return response;
        } finally {
            urlConnection.disconnect();
        }
    }
}
```

Stage 2
Dive into code

Tech Webinar:
Popular Movies App
(Stages 1 & 2)

@Bob August 2018

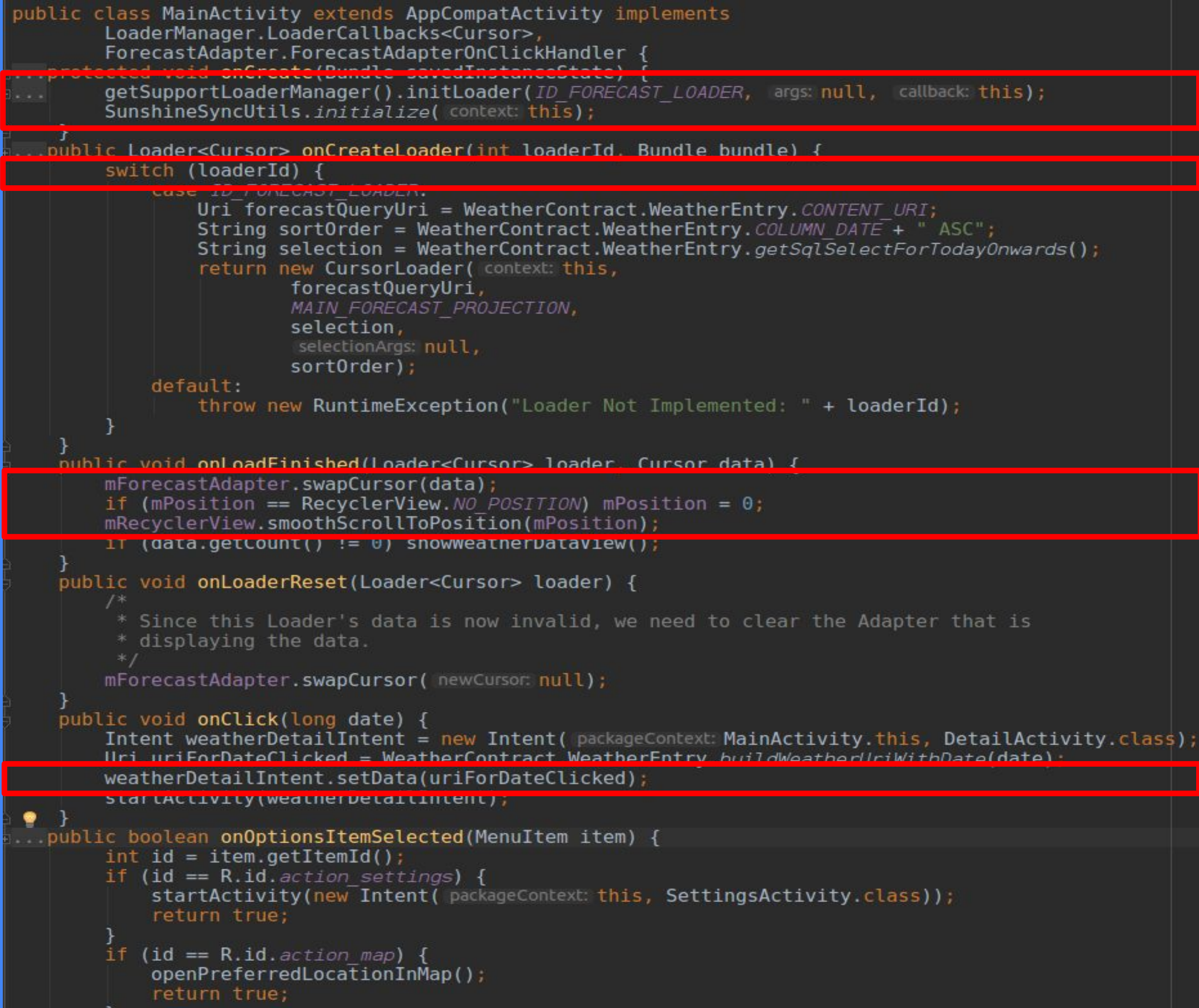


```
public class WeatherProvider extends ContentProvider {
...  @Override public boolean onCreate() { mOpenHelper = new MovieDbHelper(getContext());return true; }
     @Override public int bulkInsert(@NonNull Uri uri, @NonNull ContentValues[] values) {
...      switch (sUriMatcher.match(uri)) {
             case CODE_RATING: tableName = MovieContract.MovieEntry.TABLE_NAME_RATING; break;
             case CODE_POPULARITY: tableName = MovieContract.MovieEntry.TABLE_NAME_POPULARITY; break;
             default: return super.bulkInsert(uri, values);
...}@Override public Uri insert(@NonNull Uri uri, ContentValues values) {
...      switch (sUriMatcher.match(uri)) {
             case CODE_COLLECTION_WITH_ID:
...}@Override public int update(@NonNull Uri uri, ContentValues values, String selection, String[] selectionArgs) {
...      switch (sUriMatcher.match(uri)) {
             case CODE_RATING_WITH_ID: tableName = MovieContract.MovieEntry.TABLE_NAME_RATING; break;
             case CODE_POPULARITY_WITH_ID: tableName = MovieContract.MovieEntry.TABLE_NAME_POPULARITY; break;
             case CODE_COLLECTION_WITH_ID: tableName = MovieContract.MovieEntry.TABLE_NAME_COLLECTION; break;
...}
     @Override public Cursor query(@NonNull Uri uri, String[] projection, String selection,
                                   String[] selectionArgs, String sortOrder) {
...      switch (sUriMatcher.match(uri)) {
             case CODE_RATING_WITH_ID:
                 selection = MovieContract.MovieEntry.COLUMN_MOVIE_ID + " = ? ";
                 selectionArgs = new String[]{ movieId };
                 tableName = MovieContract.MovieEntry.TABLE_NAME_RATING;
                 break;
             case CODE_RATING: tableName = MovieContract.MovieEntry.TABLE_NAME_RATING; break;
             case CODE_POPULARITY_WITH_ID:
                  selection = MovieContract.MovieEntry.COLUMN_MOVIE_ID + " = ? ";
                  selectionArgs = new String[]{ movieId };
                  tableName = MovieContract.MovieEntry.TABLE_NAME_POPULARITY;
                  break;
             case CODE_POPULARITY: tableName = MovieContract.MovieEntry.TABLE_NAME_POPULARITY; break;
             case CODE_COLLECTION_WITH_ID:
                 selection = MovieContract.MovieEntry.COLUMN_MOVIE_ID + " = ? ";
                 selectionArgs = new String[]{ movieId };
                 tableName = MovieContract.MovieEntry.TABLE_NAME_COLLECTION;
                 break;
             case CODE_COLLECTION: tableName = MovieContract.MovieEntry.TABLE_NAME_COLLECTION; break;
...}@Override public int delete(@NonNull Uri uri, String selection, String[] selectionArgs) {
         int numRowsDeleted;
         if (null == selection) selection = "1";
         String movieId = uri.getLastPathSegment();
          String tableName;
         switch (sUriMatcher.match(uri)) {
             case CODE_RATING: tableName = MovieContract.MovieEntry.TABLE_NAME_RATING; break;
             case CODE_POPULARITY: tableName = MovieContract.MovieEntry.TABLE_NAME_POPULARITY; break;
             case CODE_COLLECTION_WITH_ID:
                 selection = MovieContract.MovieEntry.COLUMN_MOVIE_ID + " = ? ";
                 selectionArgs = new String[]{ movieId };
                 tableName = MovieContract.MovieEntry.TABLE_NAME_COLLECTION; break;
```

Stage 2
Dive into code

Tech Webinar:
Popular Movies App
(Stages 1 & 2)

@Bob August 2018

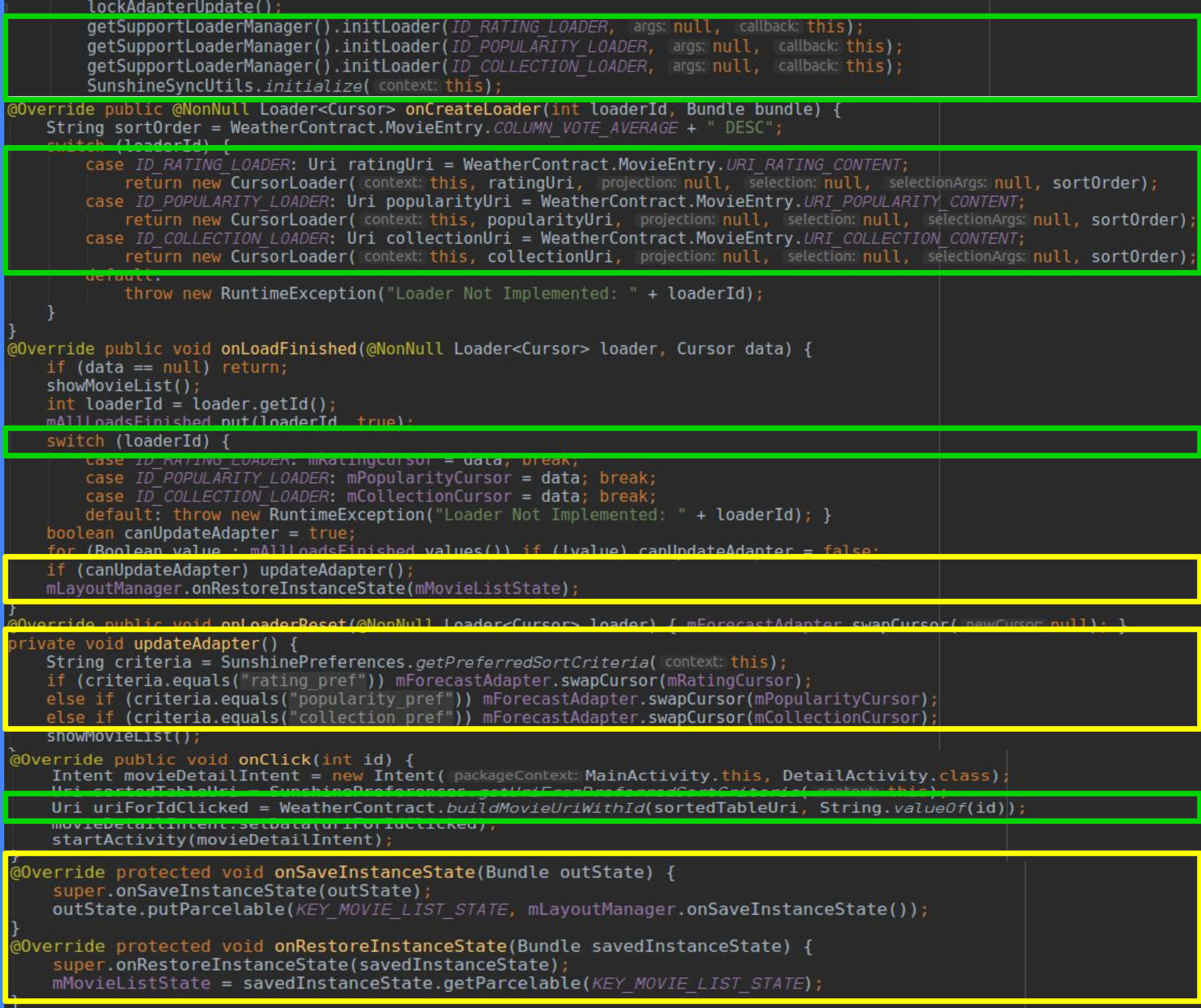


```
public class MainActivity extends AppCompatActivity implements
        LoaderManager.LoaderCallbacks<Cursor>,
        ForecastAdapter.ForecastAdapterOnClickHandler {
...protected void onCreate(Bundle savedInstanceState) {
...     getSupportLoaderManager().initLoader(ID_FORECAST_LOADER, args: null, callback: this);
        SunshineSyncUtils.initialize( context: this);
    }
...public Loader<Cursor> onCreateLoader(int loaderId, Bundle bundle) {
        switch (loaderId) {
            case ID_FORECAST_LOADER:
                Uri forecastQueryUri = WeatherContract.WeatherEntry.CONTENT_URI;
                String sortOrder = WeatherContract.WeatherEntry.COLUMN_DATE + " ASC";
                String selection = WeatherContract.WeatherEntry.getSqlSelectForTodayOnwards();
                return new CursorLoader( context: this,
                        forecastQueryUri,
                        MAIN_FORECAST_PROJECTION,
                        selection,
                        selectionArgs: null,
                        sortOrder);
            default:
                throw new RuntimeException("Loader Not Implemented: " + loaderId);
        }
    }
    public void onLoadFinished(Loader<Cursor> loader, Cursor data) {
        mForecastAdapter.swapCursor(data);
        if (mPosition == RecyclerView.NO_POSITION) mPosition = 0;
        mRecyclerView.smoothScrollToPosition(mPosition);
        if (data.getCount() != 0) showWeatherDataView();
    }
    public void onLoaderReset(Loader<Cursor> loader) {
        /*
         * Since this Loader's data is now invalid, we need to clear the Adapter that is
         * displaying the data.
         */
        mForecastAdapter.swapCursor( newCursor: null);
    }
    public void onClick(long date) {
        Intent weatherDetailIntent = new Intent( packageContext: MainActivity.this, DetailActivity.class);
        Uri uriForDateClicked = WeatherContract.WeatherEntry.buildWeatherUriWithDate(date);
        weatherDetailIntent.setData(uriForDateClicked);
        startActivity(weatherDetailIntent);
    }
...public boolean onOptionsItemSelected(MenuItem item) {
        int id = item.getItemId();
        if (id == R.id.action_settings) {
            startActivity(new Intent( packageContext: this, SettingsActivity.class));
            return true;
        }
        if (id == R.id.action_map) {
            openPreferredLocationInMap();
            return true;
```

Stage 2
Dive into code

Tech Webinar:
Popular Movies App
(Stages 1 & 2)

@Bob August 2018

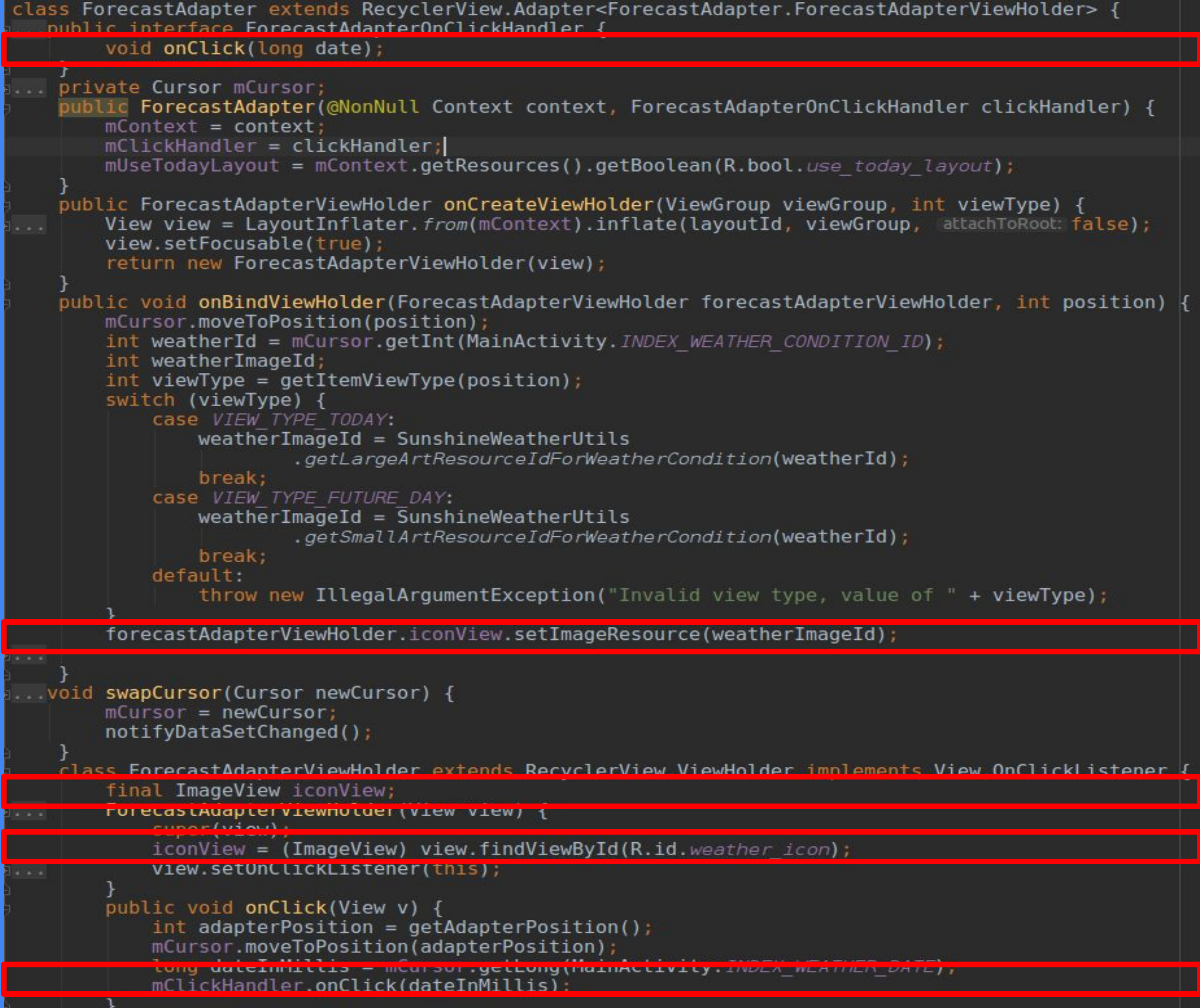


```
        lockAdapterUpdate();
        getSupportLoaderManager().initLoader(ID_RATING_LOADER, args: null, callback: this);
        getSupportLoaderManager().initLoader(ID_POPULARITY_LOADER, args: null, callback: this);
        getSupportLoaderManager().initLoader(ID_COLLECTION_LOADER, args: null, callback: this);
        SunshineSyncUtils.initialize( context: this);
@Override public @NonNull Loader<Cursor> onCreateLoader(int loaderId, Bundle bundle) {
    String sortOrder = WeatherContract.MovieEntry.COLUMN_VOTE_AVERAGE + " DESC";
    switch (loaderId) {
        case ID_RATING_LOADER: Uri ratingUri = WeatherContract.MovieEntry.URI_RATING_CONTENT;
            return new CursorLoader( context: this, ratingUri, projection: null, selection: null, selectionArgs: null, sortOrder);
        case ID_POPULARITY_LOADER: Uri popularityUri = WeatherContract.MovieEntry.URI_POPULARITY_CONTENT;
            return new CursorLoader( context: this, popularityUri, projection: null, selection: null, selectionArgs: null, sortOrder);
        case ID_COLLECTION_LOADER: Uri collectionUri = WeatherContract.MovieEntry.URI_COLLECTION_CONTENT;
            return new CursorLoader( context: this, collectionUri, projection: null, selection: null, selectionArgs: null, sortOrder);
        default:
            throw new RuntimeException("Loader Not Implemented: " + loaderId);
    }
}
@Override public void onLoadFinished(@NonNull Loader<Cursor> loader, Cursor data) {
    if (data == null) return;
    showMovieList();
    int loaderId = loader.getId();
    mAllLoadsFinished.put(loaderId, true);
    switch (loaderId) {
        case ID_RATING_LOADER: mRatingCursor = data; break;
        case ID_POPULARITY_LOADER: mPopularityCursor = data; break;
        case ID_COLLECTION_LOADER: mCollectionCursor = data; break;
        default: throw new RuntimeException("Loader Not Implemented: " + loaderId); }
    boolean canUpdateAdapter = true;
    for (Boolean value : mAllLoadsFinished.values()) if (!value) canUpdateAdapter = false;
    if (canUpdateAdapter) updateAdapter();
    mLayoutManager.onRestoreInstanceState(mMovieListState);
}
@Override public void onLoaderReset(@NonNull Loader<Cursor> loader) { mForecastAdapter.swapCursor( newCursor: null); }
private void updateAdapter() {
    String criteria = SunshinePreferences.getPreferredSortCriteria( context: this);
    if (criteria.equals("rating_pref")) mForecastAdapter.swapCursor(mRatingCursor);
    else if (criteria.equals("popularity_pref")) mForecastAdapter.swapCursor(mPopularityCursor);
    else if (criteria.equals("collection_pref")) mForecastAdapter.swapCursor(mCollectionCursor);
    showMovieList();
}
@Override public void onClick(int id) {
     Intent movieDetailIntent = new Intent( packageContext: MainActivity.this, DetailActivity.class);
     Uri sortedTableUri = SunshinePreferences.getUriFromPreferredSortCriteria( context: this);
     Uri uriForIdClicked = WeatherContract.buildMovieUriWithId(sortedTableUri, String.valueOf(id));
     movieDetailIntent.setData(uriForIdClicked);
     startActivity(movieDetailIntent);
}
@Override protected void onSaveInstanceState(Bundle outState) {
     super.onSaveInstanceState(outState);
     outState.putParcelable(KEY_MOVIE_LIST_STATE, mLayoutManager.onSaveInstanceState());
}
@Override protected void onRestoreInstanceState(Bundle savedInstanceState) {
     super.onRestoreInstanceState(savedInstanceState);
     mMovieListState = savedInstanceState.getParcelable(KEY_MOVIE_LIST_STATE);
}
```

Stage 2
Dive into code

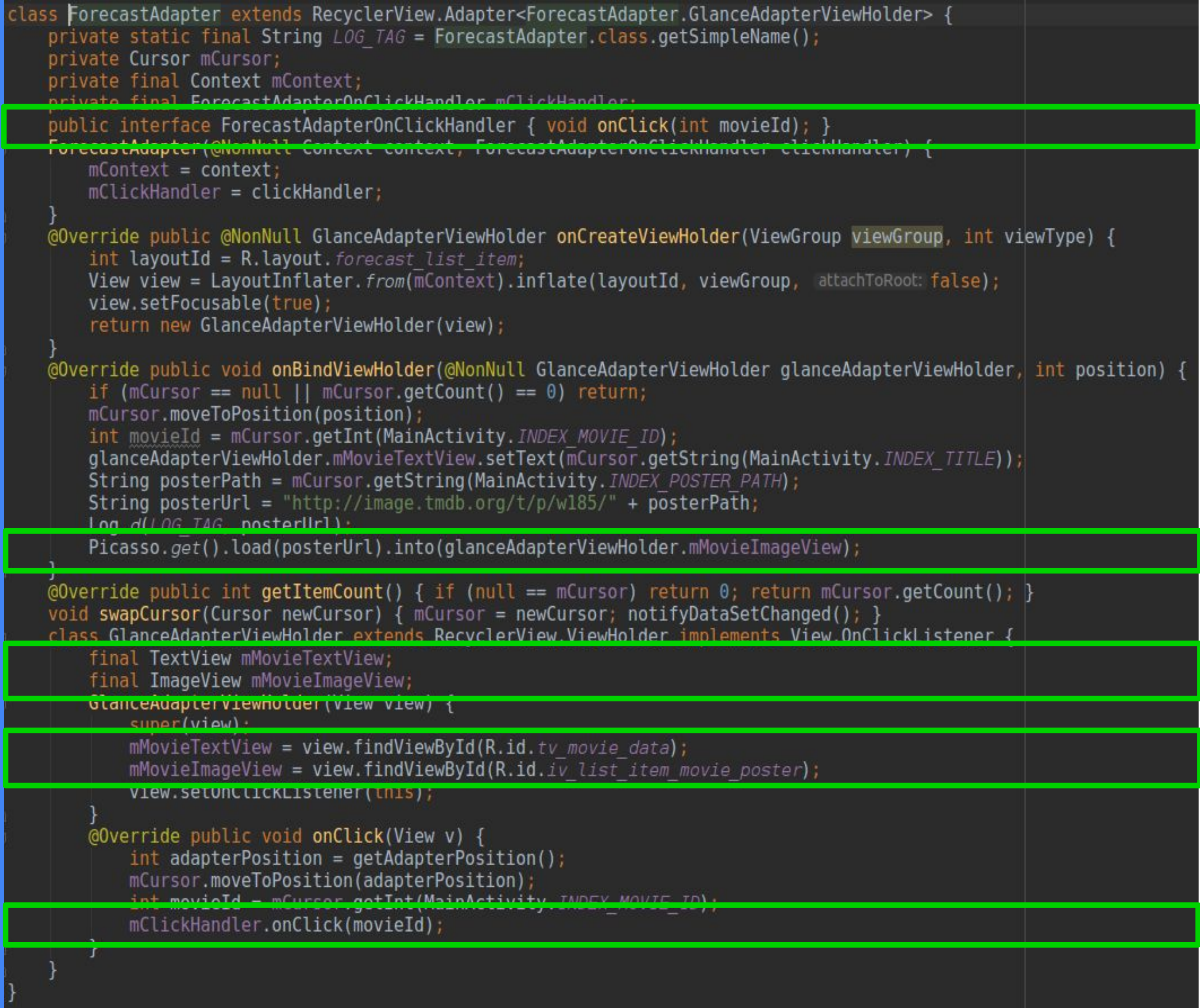
Tech Webinar:
Popular Movies App
(Stages 1 & 2)

@Bob August 2018



```java
class ForecastAdapter extends RecyclerView.Adapter<ForecastAdapter.ForecastAdapterViewHolder> {
    public interface ForecastAdapterOnClickHandler {
        void onClick(long date);
    }
    private Cursor mCursor;
    public ForecastAdapter(@NonNull Context context, ForecastAdapterOnClickHandler clickHandler) {
        mContext = context;
        mClickHandler = clickHandler;
        mUseTodayLayout = mContext.getResources().getBoolean(R.bool.use_today_layout);
    }
    public ForecastAdapterViewHolder onCreateViewHolder(ViewGroup viewGroup, int viewType) {
        View view = LayoutInflater.from(mContext).inflate(layoutId, viewGroup,  attachToRoot: false);
        view.setFocusable(true);
        return new ForecastAdapterViewHolder(view);
    }
    public void onBindViewHolder(ForecastAdapterViewHolder forecastAdapterViewHolder, int position) {
        mCursor.moveToPosition(position);
        int weatherId = mCursor.getInt(MainActivity.INDEX_WEATHER_CONDITION_ID);
        int weatherImageId;
        int viewType = getItemViewType(position);
        switch (viewType) {
            case VIEW_TYPE_TODAY:
                weatherImageId = SunshineWeatherUtils
                        .getLargeArtResourceIdForWeatherCondition(weatherId);
                break;
            case VIEW_TYPE_FUTURE_DAY:
                weatherImageId = SunshineWeatherUtils
                        .getSmallArtResourceIdForWeatherCondition(weatherId);
                break;
            default:
                throw new IllegalArgumentException("Invalid view type, value of " + viewType);
        }
        forecastAdapterViewHolder.iconView.setImageResource(weatherImageId);

    }
    void swapCursor(Cursor newCursor) {
        mCursor = newCursor;
        notifyDataSetChanged();
    }
    class ForecastAdapterViewHolder extends RecyclerView.ViewHolder implements View.OnClickListener {
        final ImageView iconView;
        ForecastAdapterViewHolder(View view) {
            super(view);
            iconView = (ImageView) view.findViewById(R.id.weather_icon);
            view.setOnClickListener(this);
        }
        public void onClick(View v) {
            int adapterPosition = getAdapterPosition();
            mCursor.moveToPosition(adapterPosition);
            long dateInMillis = mCursor.getLong(MainActivity.INDEX_WEATHER_DATE);
            mClickHandler.onClick(dateInMillis);
        }
```

Stage 2
Dive into code



```
class ForecastAdapter extends RecyclerView.Adapter<ForecastAdapter.GlanceAdapterViewHolder> {
    private static final String LOG_TAG = ForecastAdapter.class.getSimpleName();
    private Cursor mCursor;
    private final Context mContext;
    private final ForecastAdapterOnClickHandler mClickHandler;
    public interface ForecastAdapterOnClickHandler { void onClick(int movieId); }
    ForecastAdapter(@NonNull Context context, ForecastAdapterOnClickHandler clickHandler) {
        mContext = context;
        mClickHandler = clickHandler;
    }
    @Override public @NonNull GlanceAdapterViewHolder onCreateViewHolder(ViewGroup viewGroup, int viewType) {
        int layoutId = R.layout.forecast_list_item;
        View view = LayoutInflater.from(mContext).inflate(layoutId, viewGroup,  attachToRoot: false);
        view.setFocusable(true);
        return new GlanceAdapterViewHolder(view);
    }
    @Override public void onBindViewHolder(@NonNull GlanceAdapterViewHolder glanceAdapterViewHolder, int position) {
        if (mCursor == null || mCursor.getCount() == 0) return;
        mCursor.moveToPosition(position);
        int movieId = mCursor.getInt(MainActivity.INDEX_MOVIE_ID);
        glanceAdapterViewHolder.mMovieTextView.setText(mCursor.getString(MainActivity.INDEX_TITLE));
        String posterPath = mCursor.getString(MainActivity.INDEX_POSTER_PATH);
        String posterUrl = "http://image.tmdb.org/t/p/w185/" + posterPath;
        Log.d(LOG_TAG, posterUrl);
        Picasso.get().load(posterUrl).into(glanceAdapterViewHolder.mMovieImageView);
    }
    @Override public int getItemCount() { if (null == mCursor) return 0; return mCursor.getCount(); }
    void swapCursor(Cursor newCursor) { mCursor = newCursor; notifyDataSetChanged(); }
    class GlanceAdapterViewHolder extends RecyclerView.ViewHolder implements View.OnClickListener {
        final TextView mMovieTextView;
        final ImageView mMovieImageView;
        GlanceAdapterViewHolder(View view) {
            super(view);
            mMovieTextView = view.findViewById(R.id.tv_movie_data);
            mMovieImageView = view.findViewById(R.id.iv_list_item_movie_poster);
            view.setOnClickListener(this);
        }
        @Override public void onClick(View v) {
            int adapterPosition = getAdapterPosition();
            mCursor.moveToPosition(adapterPosition);
            int movieId = mCursor.getInt(MainActivity.INDEX_MOVIE_ID);
            mClickHandler.onClick(movieId);
        }
    }
}
```

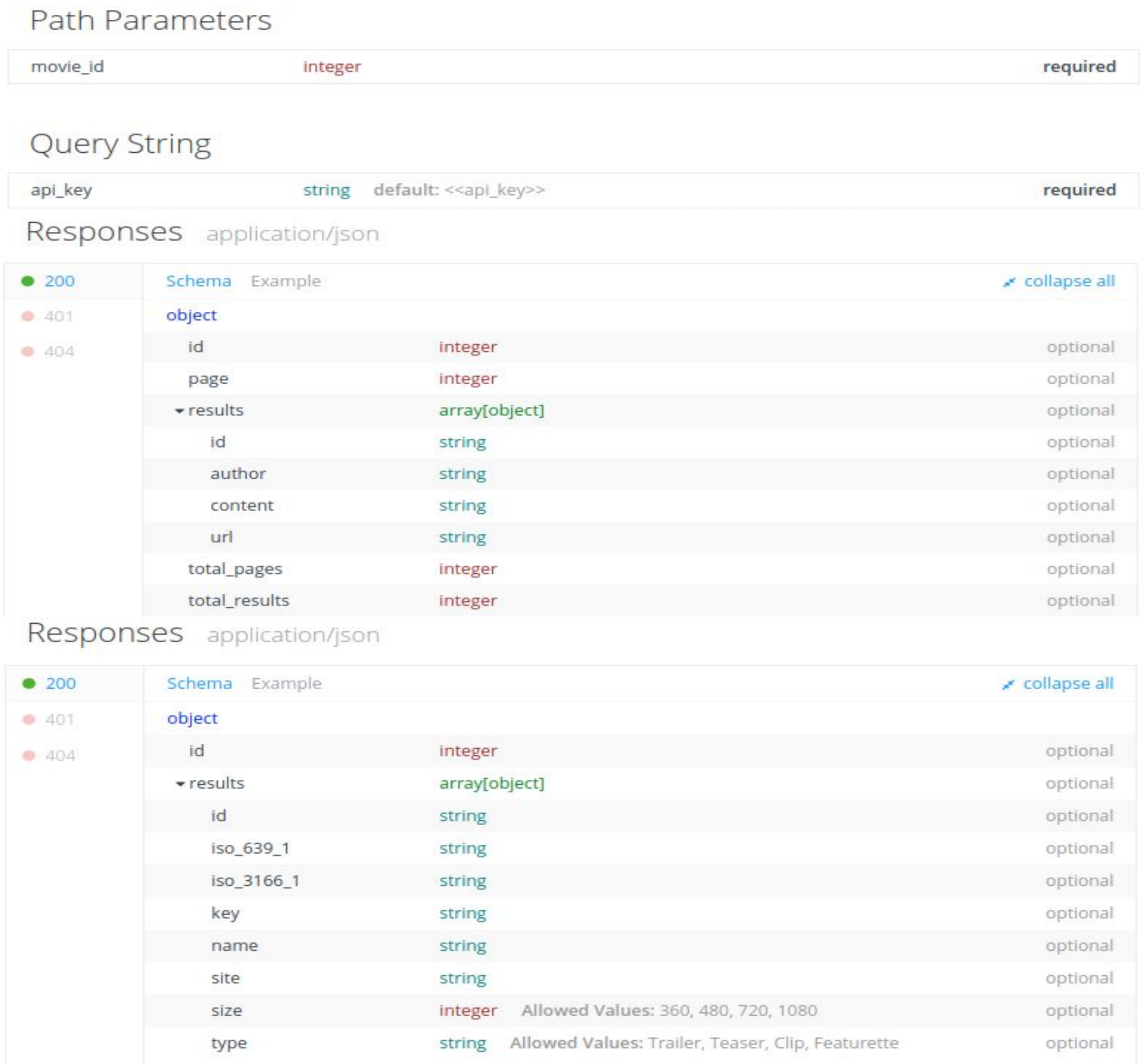
Tech Webinar:
Popular Movies App
(Stages 1 & 2)

@Bob August 2018

Stage 2
Dive into code

Tech Webinar:
Popular Movies App
(Stages 1 & 2)

@Bob August 2018



Path Parameters

movie_id	integer	required

Query String

api_key	string	default: <<api_key>>	required

Responses application/json

● 200 · ● 401 · ● 404 — Schema Example — collapse all

object			
id	integer		optional
page	integer		optional
▾ results	array[object]		optional
id	string		optional
author	string		optional
content	string		optional
url	string		optional
total_pages	integer		optional
total_results	integer		optional

Responses application/json

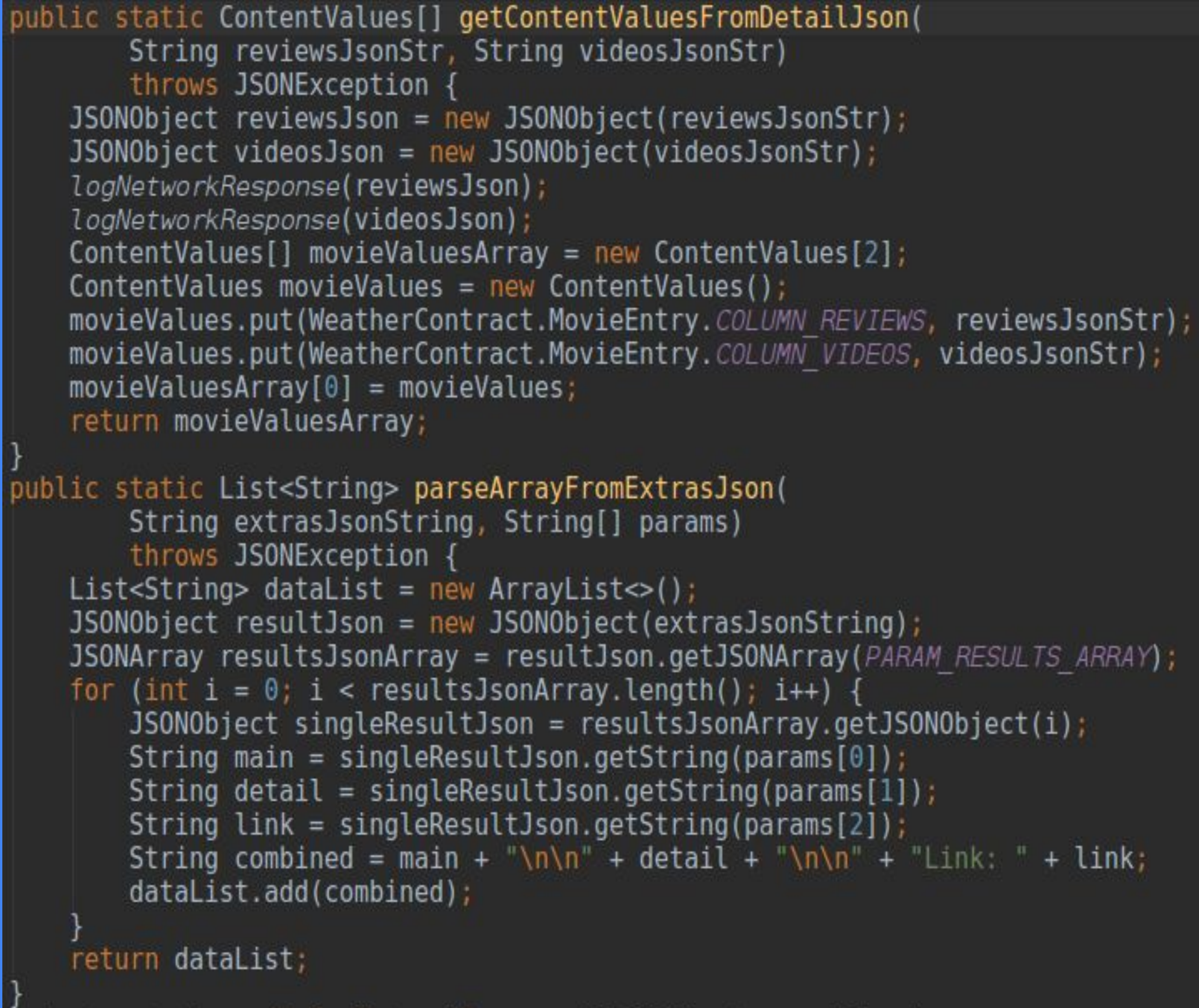
● 200 · ● 401 · ● 404 — Schema Example — collapse all

object			
id	integer		optional
▾ results	array[object]		optional
id	string		optional
iso_639_1	string		optional
iso_3166_1	string		optional
key	string		optional
name	string		optional
site	string		optional
size	integer	Allowed Values: 360, 480, 720, 1080	optional
type	string	Allowed Values: Trailer, Teaser, Clip, Featurette	optional

Stage 2
Dive into code

Tech Webinar:
Popular Movies App
(Stages 1 & 2)

@Bob August 2018

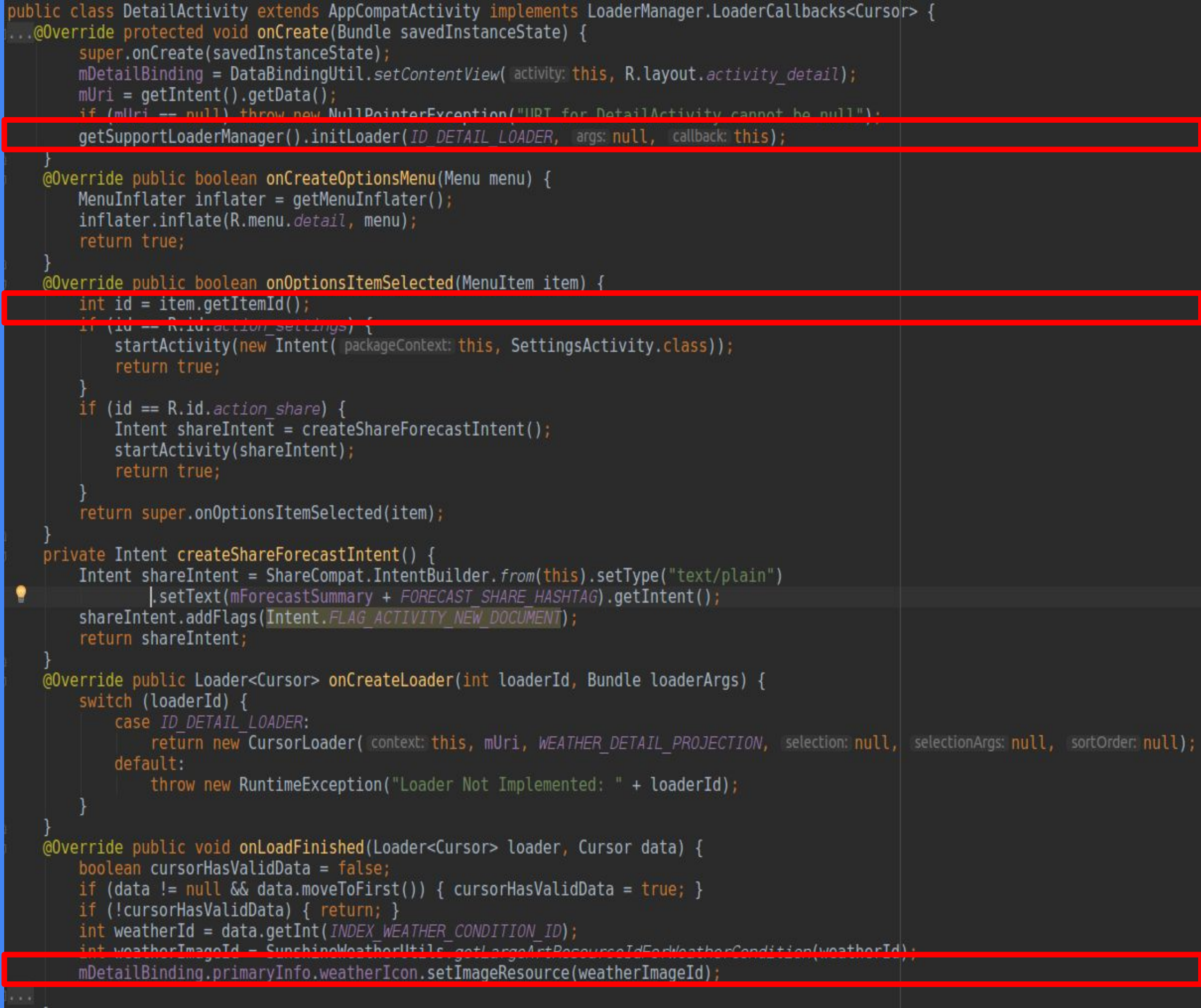


```java
public static ContentValues[] getContentValuesFromDetailJson(
        String reviewsJsonStr, String videosJsonStr)
        throws JSONException {
    JSONObject reviewsJson = new JSONObject(reviewsJsonStr);
    JSONObject videosJson = new JSONObject(videosJsonStr);
    logNetworkResponse(reviewsJson);
    logNetworkResponse(videosJson);
    ContentValues[] movieValuesArray = new ContentValues[2];
    ContentValues movieValues = new ContentValues();
    movieValues.put(WeatherContract.MovieEntry.COLUMN_REVIEWS, reviewsJsonStr);
    movieValues.put(WeatherContract.MovieEntry.COLUMN_VIDEOS, videosJsonStr);
    movieValuesArray[0] = movieValues;
    return movieValuesArray;
}
public static List<String> parseArrayFromExtrasJson(
        String extrasJsonString, String[] params)
        throws JSONException {
    List<String> dataList = new ArrayList<>();
    JSONObject resultJson = new JSONObject(extrasJsonString);
    JSONArray resultsJsonArray = resultJson.getJSONArray(PARAM_RESULTS_ARRAY);
    for (int i = 0; i < resultsJsonArray.length(); i++) {
        JSONObject singleResultJson = resultsJsonArray.getJSONObject(i);
        String main = singleResultJson.getString(params[0]);
        String detail = singleResultJson.getString(params[1]);
        String link = singleResultJson.getString(params[2]);
        String combined = main + "\n\n" + detail + "\n\n" + "Link: " + link;
        dataList.add(combined);
    }
    return dataList;
}
```

Stage 2
Dive into code

Tech Webinar:
Popular Movies App
(Stages 1 & 2)

@Bob August 2018

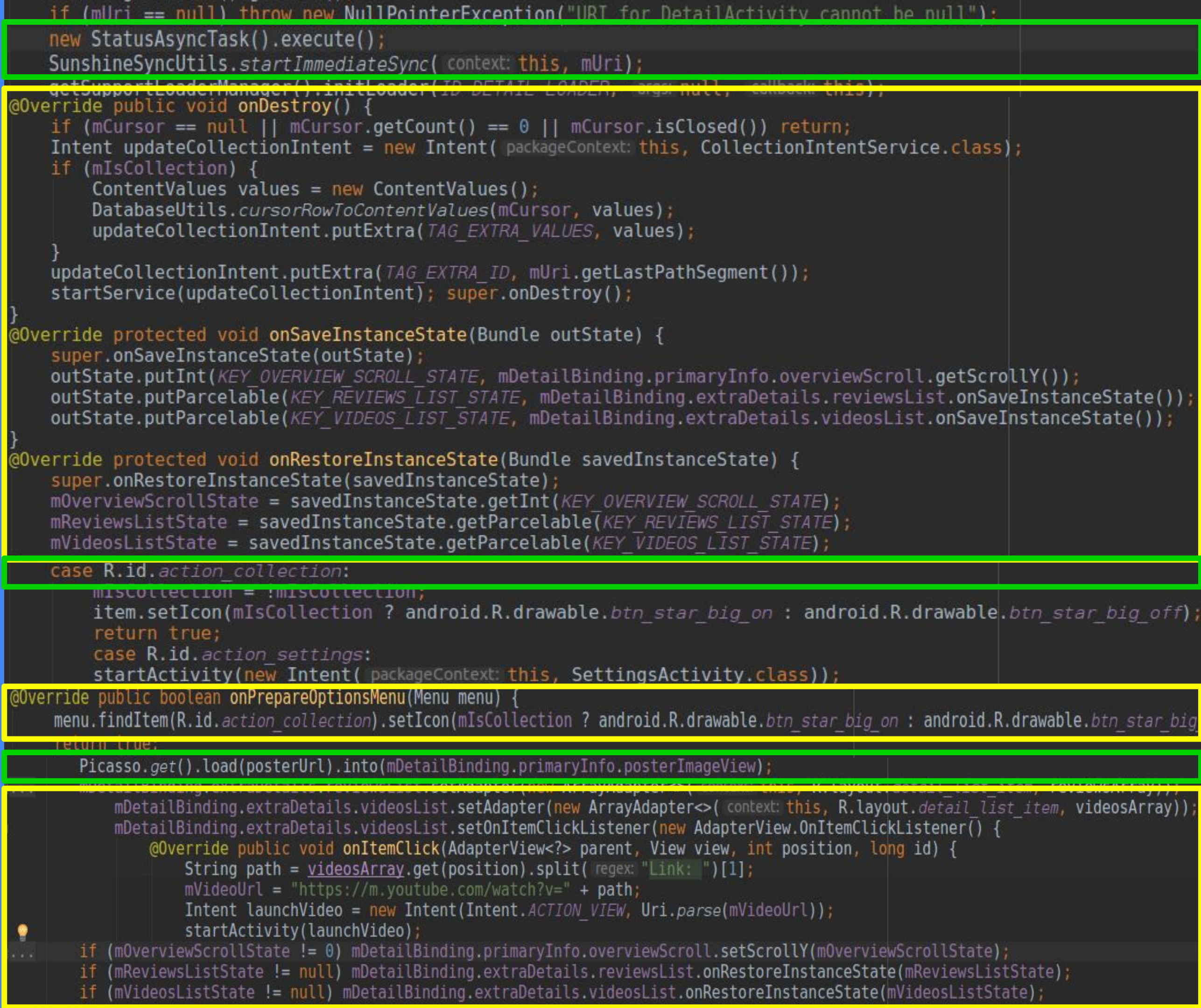


```
public class DetailActivity extends AppCompatActivity implements LoaderManager.LoaderCallbacks<Cursor> {
...@Override protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        mDetailBinding = DataBindingUtil.setContentView( activity: this, R.layout.activity_detail);
        mUri = getIntent().getData();
        if (mUri == null) throw new NullPointerException("URI for DetailActivity cannot be null");
        getSupportLoaderManager().initLoader(ID_DETAIL_LOADER,  args: null,  callback: this);
    }
    @Override public boolean onCreateOptionsMenu(Menu menu) {
        MenuInflater inflater = getMenuInflater();
        inflater.inflate(R.menu.detail, menu);
        return true;
    }
    @Override public boolean onOptionsItemSelected(MenuItem item) {
        int id = item.getItemId();
        if (id == R.id.action_settings) {
            startActivity(new Intent( packageContext: this, SettingsActivity.class));
            return true;
        }
        if (id == R.id.action_share) {
            Intent shareIntent = createShareForecastIntent();
            startActivity(shareIntent);
            return true;
        }
        return super.onOptionsItemSelected(item);
    }
    private Intent createShareForecastIntent() {
        Intent shareIntent = ShareCompat.IntentBuilder.from(this).setType("text/plain")
                .setText(mForecastSummary + FORECAST_SHARE_HASHTAG).getIntent();
        shareIntent.addFlags(Intent.FLAG_ACTIVITY_NEW_DOCUMENT);
        return shareIntent;
    }
    @Override public Loader<Cursor> onCreateLoader(int loaderId, Bundle loaderArgs) {
        switch (loaderId) {
            case ID_DETAIL_LOADER:
                return new CursorLoader( context: this, mUri, WEATHER_DETAIL_PROJECTION,  selection: null,  selectionArgs: null,  sortOrder: null);
            default:
                throw new RuntimeException("Loader Not Implemented: " + loaderId);
        }
    }
    @Override public void onLoadFinished(Loader<Cursor> loader, Cursor data) {
        boolean cursorHasValidData = false;
        if (data != null && data.moveToFirst()) { cursorHasValidData = true; }
        if (!cursorHasValidData) { return; }
        int weatherId = data.getInt(INDEX_WEATHER_CONDITION_ID);
        int weatherImageId = SunshineWeatherUtils.getLargeArtResourceIdForWeatherCondition(weatherId);
        mDetailBinding.primaryInfo.weatherIcon.setImageResource(weatherImageId);
...
```

Stage 2
Dive into code

Tech Webinar:
Popular Movies App
(Stages 1 & 2)

@Bob August 2018

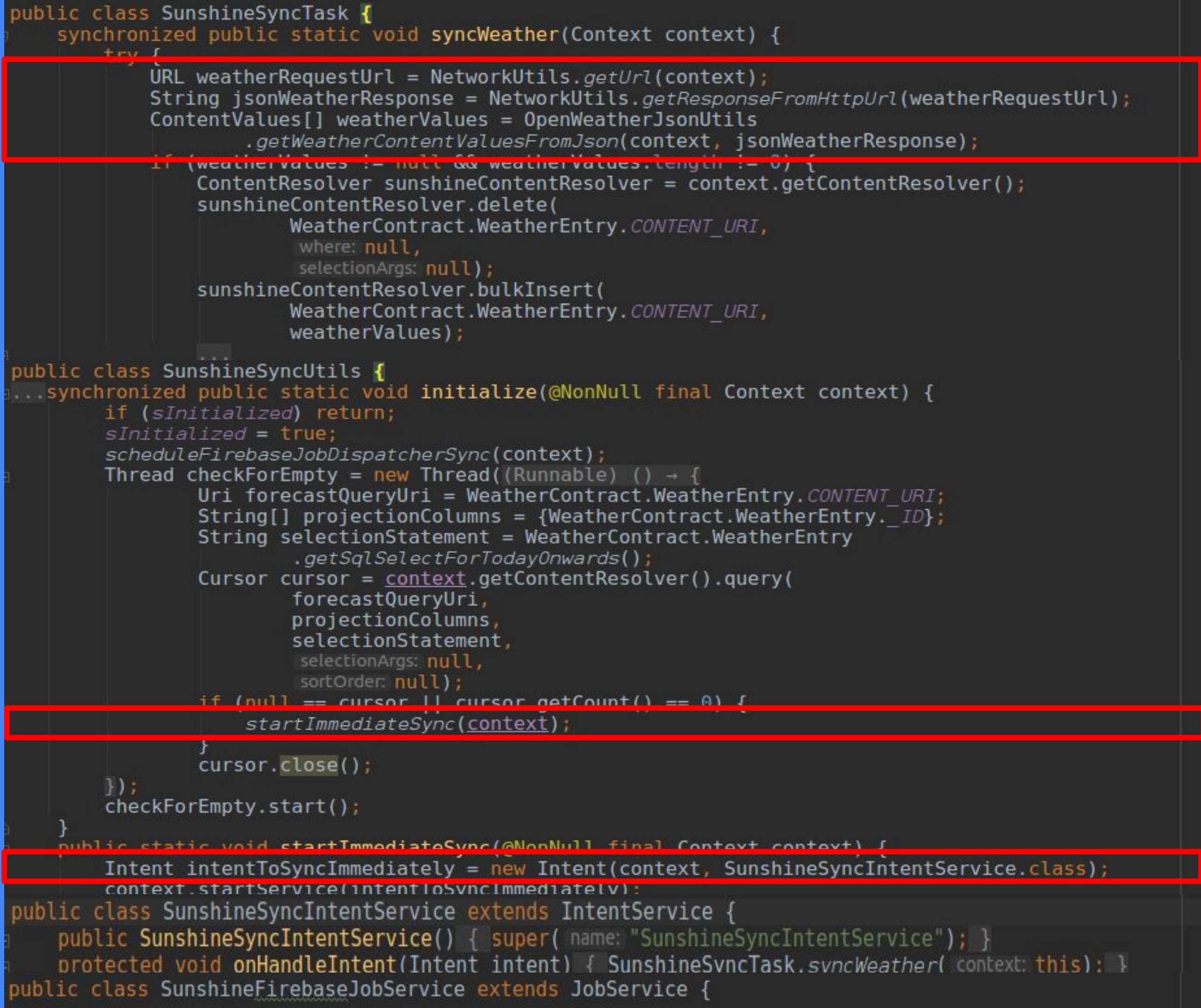


```
    if (mUri == null) throw new NullPointerException("URI for DetailActivity cannot be null");
    new StatusAsyncTask().execute();
    SunshineSyncUtils.startImmediateSync( context: this, mUri);
    getSupportLoaderManager().initLoader(ID_DETAIL_LOADER, args: null, callback: this);
@Override public void onDestroy() {
    if (mCursor == null || mCursor.getCount() == 0 || mCursor.isClosed()) return;
    Intent updateCollectionIntent = new Intent( packageContext: this, CollectionIntentService.class);
    if (mIsCollection) {
        ContentValues values = new ContentValues();
        DatabaseUtils.cursorRowToContentValues(mCursor, values);
        updateCollectionIntent.putExtra(TAG_EXTRA_VALUES, values);
    }
    updateCollectionIntent.putExtra(TAG_EXTRA_ID, mUri.getLastPathSegment());
    startService(updateCollectionIntent); super.onDestroy();
}
@Override protected void onSaveInstanceState(Bundle outState) {
    super.onSaveInstanceState(outState);
    outState.putInt(KEY_OVERVIEW_SCROLL_STATE, mDetailBinding.primaryInfo.overviewScroll.getScrollY());
    outState.putParcelable(KEY_REVIEWS_LIST_STATE, mDetailBinding.extraDetails.reviewsList.onSaveInstanceState());
    outState.putParcelable(KEY_VIDEOS_LIST_STATE, mDetailBinding.extraDetails.videosList.onSaveInstanceState());
}
@Override protected void onRestoreInstanceState(Bundle savedInstanceState) {
    super.onRestoreInstanceState(savedInstanceState);
    mOverviewScrollState = savedInstanceState.getInt(KEY_OVERVIEW_SCROLL_STATE);
    mReviewsListState = savedInstanceState.getParcelable(KEY_REVIEWS_LIST_STATE);
    mVideosListState = savedInstanceState.getParcelable(KEY_VIDEOS_LIST_STATE);
    case R.id.action_collection:
        mIsCollection = !mIsCollection;
        item.setIcon(mIsCollection ? android.R.drawable.btn_star_big_on : android.R.drawable.btn_star_big_off);
        return true;
        case R.id.action_settings:
        startActivity(new Intent( packageContext: this, SettingsActivity.class));
@Override public boolean onPrepareOptionsMenu(Menu menu) {
    menu.findItem(R.id.action_collection).setIcon(mIsCollection ? android.R.drawable.btn_star_big_on : android.R.drawable.btn_star_big
    return true;
      Picasso.get().load(posterUrl).into(mDetailBinding.primaryInfo.posterImageView);
        mDetailBinding.extraDetails.reviewsList.setAdapter(new ArrayAdapter<>( context: this, R.layout.detail_list_item, reviewsArray));
        mDetailBinding.extraDetails.videosList.setAdapter(new ArrayAdapter<>( context: this, R.layout.detail_list_item, videosArray));
        mDetailBinding.extraDetails.videosList.setOnItemClickListener(new AdapterView.OnItemClickListener() {
            @Override public void onItemClick(AdapterView<?> parent, View view, int position, long id) {
                String path = videosArray.get(position).split( regex: "Link: ")[1];
                mVideoUrl = "https://m.youtube.com/watch?v=" + path;
                Intent launchVideo = new Intent(Intent.ACTION_VIEW, Uri.parse(mVideoUrl));
                startActivity(launchVideo);
      if (mOverviewScrollState != 0) mDetailBinding.primaryInfo.overviewScroll.setScrollY(mOverviewScrollState);
      if (mReviewsListState != null) mDetailBinding.extraDetails.reviewsList.onRestoreInstanceState(mReviewsListState);
      if (mVideosListState != null) mDetailBinding.extraDetails.videosList.onRestoreInstanceState(mVideosListState);
```

Stage 2
Dive into code

Tech Webinar:
Popular Movies App
(Stages 1 & 2)

@Bob August 2018

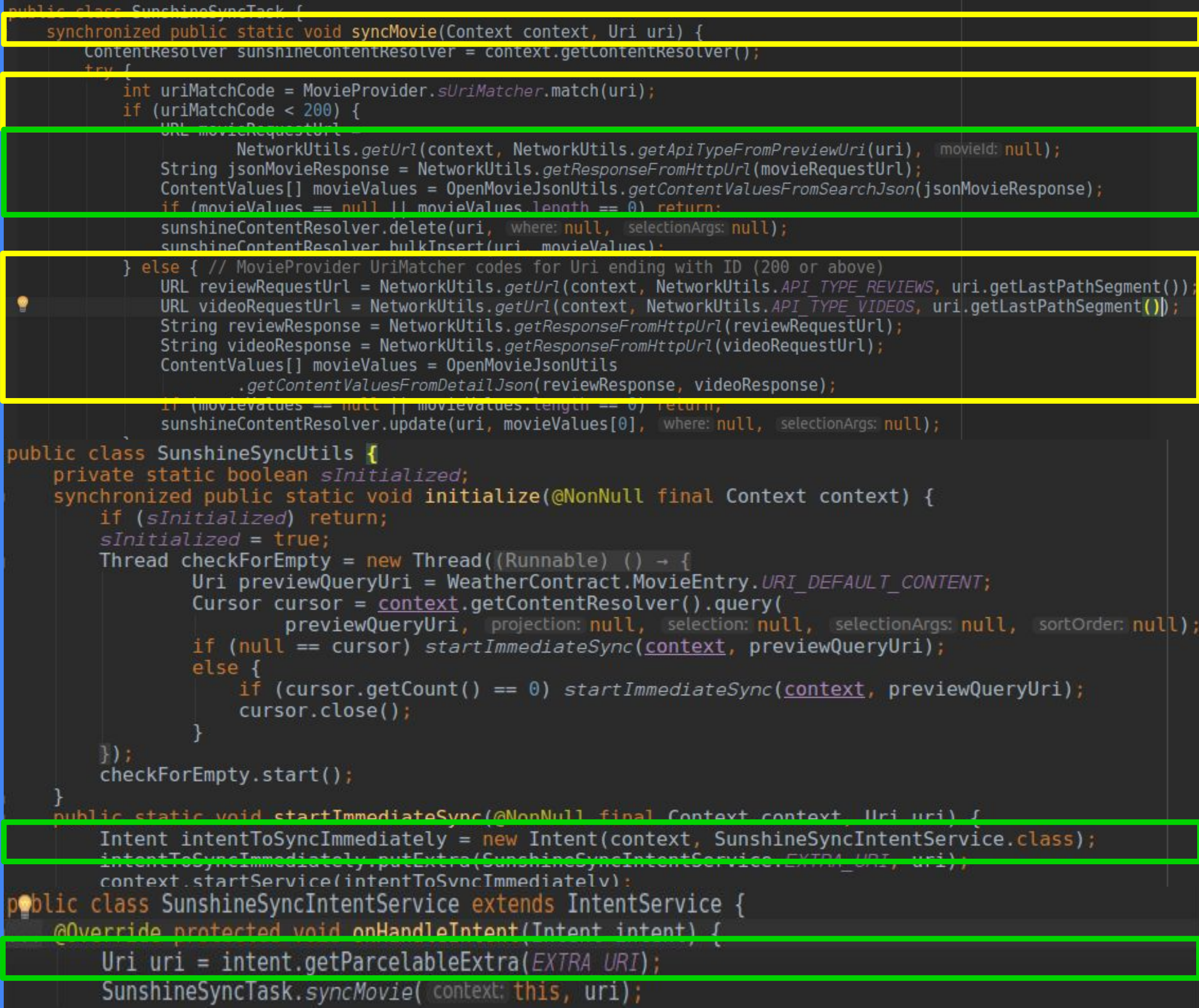


```java
public class SunshineSyncTask {
    synchronized public static void syncWeather(Context context) {
        try {
            URL weatherRequestUrl = NetworkUtils.getUrl(context);
            String jsonWeatherResponse = NetworkUtils.getResponseFromHttpUrl(weatherRequestUrl);
            ContentValues[] weatherValues = OpenWeatherJsonUtils
                    .getWeatherContentValuesFromJson(context, jsonWeatherResponse);
            if (weatherValues != null && weatherValues.length != 0) {
                ContentResolver sunshineContentResolver = context.getContentResolver();
                sunshineContentResolver.delete(
                        WeatherContract.WeatherEntry.CONTENT_URI,
                        where: null,
                        selectionArgs: null);
                sunshineContentResolver.bulkInsert(
                        WeatherContract.WeatherEntry.CONTENT_URI,
                        weatherValues);
                ...
public class SunshineSyncUtils {
...synchronized public static void initialize(@NonNull final Context context) {
        if (sInitialized) return;
        sInitialized = true;
        scheduleFirebaseJobDispatcherSync(context);
        Thread checkForEmpty = new Thread((Runnable) () → {
                Uri forecastQueryUri = WeatherContract.WeatherEntry.CONTENT_URI;
                String[] projectionColumns = {WeatherContract.WeatherEntry._ID};
                String selectionStatement = WeatherContract.WeatherEntry
                        .getSqlSelectForTodayOnwards();
                Cursor cursor = context.getContentResolver().query(
                        forecastQueryUri,
                        projectionColumns,
                        selectionStatement,
                        selectionArgs: null,
                        sortOrder: null);
                if (null == cursor || cursor.getCount() == 0) {
                    startImmediateSync(context);
                }
                cursor.close();
        });
        checkForEmpty.start();
    }
    public static void startImmediateSync(@NonNull final Context context) {
        Intent intentToSyncImmediately = new Intent(context, SunshineSyncIntentService.class);
        context.startService(intentToSyncImmediately);
public class SunshineSyncIntentService extends IntentService {
    public SunshineSyncIntentService() { super( name: "SunshineSyncIntentService"); }
    protected void onHandleIntent(Intent intent) { SunshineSyncTask.syncWeather( context: this); }
public class SunshineFirebaseJobService extends JobService {
```

Stage 2
Dive into code

Tech Webinar:
Popular Movies App
(Stages 1 & 2)

@Bob August 2018



```java
public class SunshineSyncTask {
    synchronized public static void syncMovie(Context context, Uri uri) {
        ContentResolver sunshineContentResolver = context.getContentResolver();
        try {
            int uriMatchCode = MovieProvider.sUriMatcher.match(uri);
            if (uriMatchCode < 200) {
                URL movieRequestUrl =
                        NetworkUtils.getUrl(context, NetworkUtils.getApiTypeFromPreviewUri(uri),  movieId: null);
                String jsonMovieResponse = NetworkUtils.getResponseFromHttpUrl(movieRequestUrl);
                ContentValues[] movieValues = OpenMovieJsonUtils.getContentValuesFromSearchJson(jsonMovieResponse);
                if (movieValues == null || movieValues.length == 0) return;
                sunshineContentResolver.delete(uri,  where: null,  selectionArgs: null);
                sunshineContentResolver.bulkInsert(uri, movieValues);
            } else { // MovieProvider UriMatcher codes for Uri ending with ID (200 or above)
                URL reviewRequestUrl = NetworkUtils.getUrl(context, NetworkUtils.API_TYPE_REVIEWS, uri.getLastPathSegment());
                URL videoRequestUrl = NetworkUtils.getUrl(context, NetworkUtils.API_TYPE_VIDEOS, uri.getLastPathSegment());
                String reviewResponse = NetworkUtils.getResponseFromHttpUrl(reviewRequestUrl);
                String videoResponse = NetworkUtils.getResponseFromHttpUrl(videoRequestUrl);
                ContentValues[] movieValues = OpenMovieJsonUtils
                        .getContentValuesFromDetailJson(reviewResponse, videoResponse);
                if (movieValues == null || movieValues.length == 0) return;
                sunshineContentResolver.update(uri, movieValues[0],  where: null,  selectionArgs: null);
```

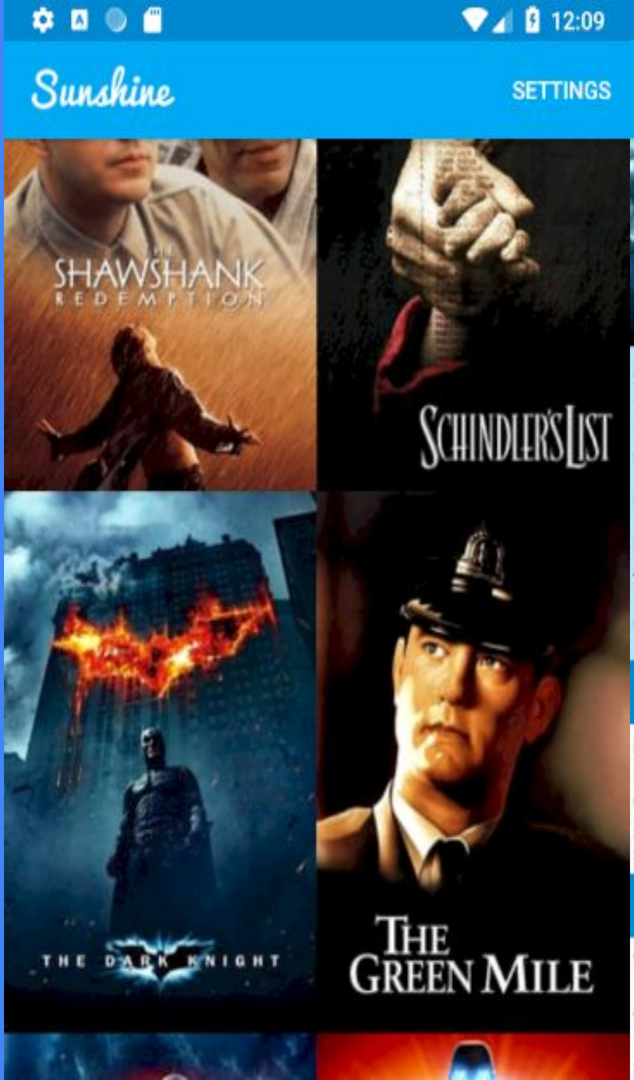
```java
public class SunshineSyncUtils {
    private static boolean sInitialized;
    synchronized public static void initialize(@NonNull final Context context) {
        if (sInitialized) return;
        sInitialized = true;
        Thread checkForEmpty = new Thread((Runnable) () → {
                Uri previewQueryUri = WeatherContract.MovieEntry.URI_DEFAULT_CONTENT;
                Cursor cursor = context.getContentResolver().query(
                        previewQueryUri,  projection: null,  selection: null,  selectionArgs: null,  sortOrder: null);
                if (null == cursor) startImmediateSync(context, previewQueryUri);
                else {
                    if (cursor.getCount() == 0) startImmediateSync(context, previewQueryUri);
                    cursor.close();
                }
        });
        checkForEmpty.start();
    }
    public static void startImmediateSync(@NonNull final Context context, Uri uri) {
        Intent intentToSyncImmediately = new Intent(context, SunshineSyncIntentService.class);
        intentToSyncImmediately.putExtra(SunshineSyncIntentService.EXTRA_URI, uri);
        context.startService(intentToSyncImmediately);
```

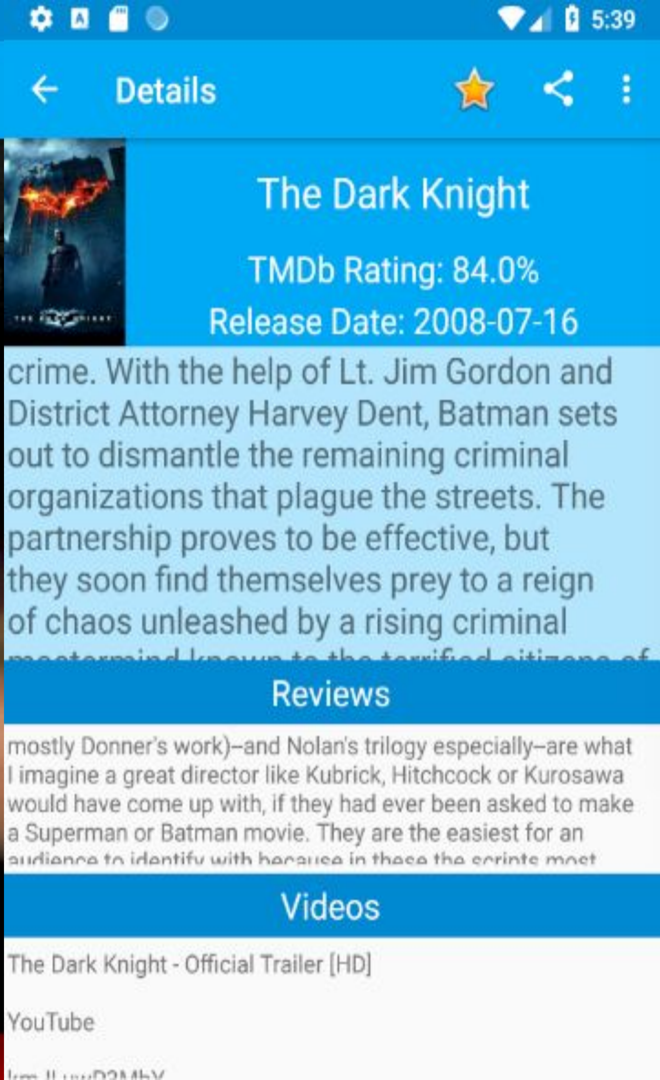
```java
public class SunshineSyncIntentService extends IntentService {
    @Override protected void onHandleIntent(Intent intent) {
        Uri uri = intent.getParcelableExtra(EXTRA_URI);
        SunshineSyncTask.syncMovie( context: this, uri);
```

Stage 2
Test your app

Tech Webinar:
Popular Movies App
(Stages 1 & 2)

@Bob August 2018





The Dark Knight

TMDb Rating: 84.0%

Release Date: 2008-07-16

crime. With the help of Lt. Jim Gordon and District Attorney Harvey Dent, Batman sets out to dismantle the remaining criminal organizations that plague the streets. The partnership proves to be effective, but they soon find themselves prey to a reign of chaos unleashed by a rising criminal mastermind known to the terrified citizens of

Reviews

mostly Donner's work)–and Nolan's trilogy especially–are what I imagine a great director like Kubrick, Hitchcock or Kurosawa would have come up with, if they had ever been asked to make a Superman or Batman movie. They are the easiest for an audience to identify with because in these the scripts most

Videos

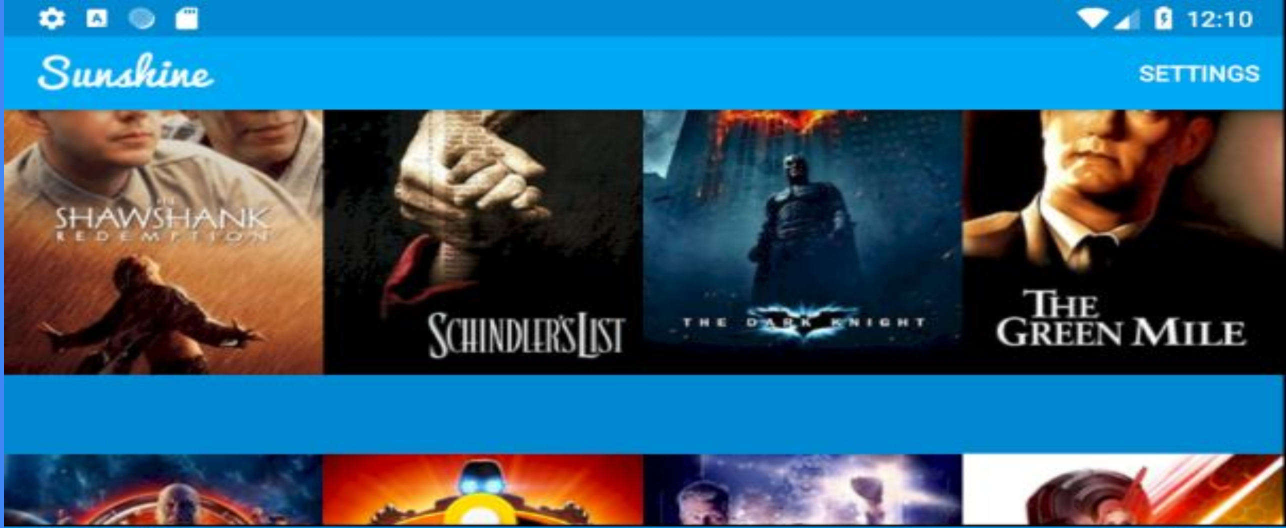
The Dark Knight - Official Trailer [HD]

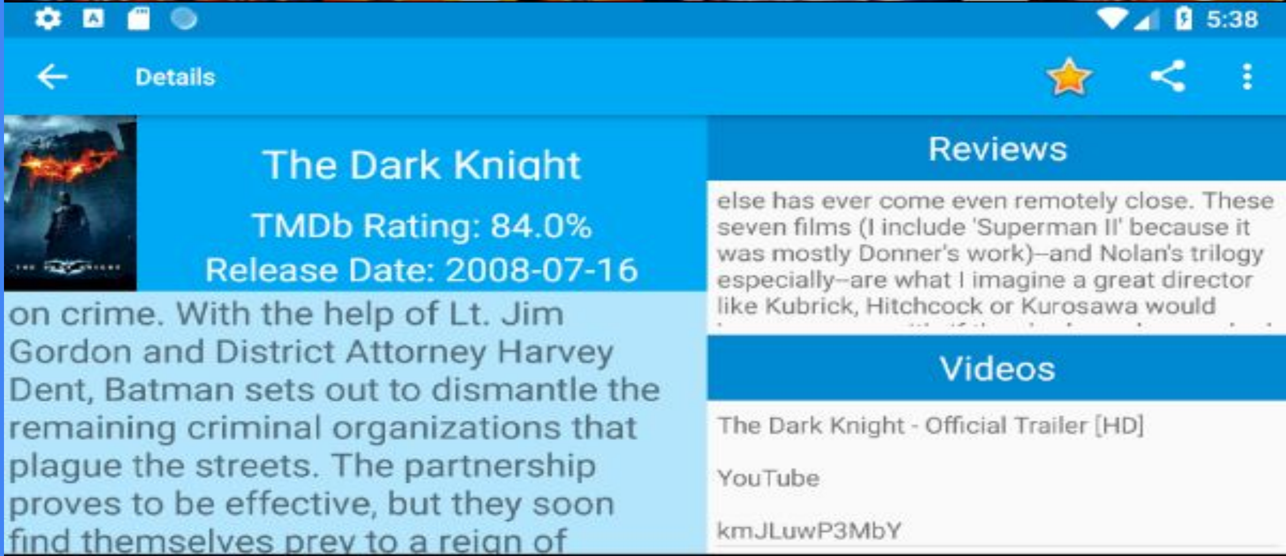
YouTube

Stage 2
Test your app

Tech Webinar:
Popular Movies App
(Stages 1 & 2)

@Bob August 2018





The Dark Knight

TMDb Rating: 84.0%

Release Date: 2008-07-16

on crime. With the help of Lt. Jim Gordon and District Attorney Harvey Dent, Batman sets out to dismantle the remaining criminal organizations that plague the streets. The partnership proves to be effective, but they soon find themselves prey to a reign of

else has ever come even remotely close. These seven films (I include 'Superman II' because it was mostly Donner's work)--and Nolan's trilogy especially--are what I imagine a great director like Kubrick, Hitchcock or Kurosawa would

The Dark Knight - Official Trailer [HD]

YouTube

kmJLuwP3MbY

Conclusion

Guidelines: https://www.udacity.com/legal/community-guidelines

Codebases

- Stage 1: https://github.com/rjbx/PopularSunshine1
- Stage 2: https://github.com/rjbx/PopularSunshine2

Questions?

- DM me on Slack (@Bob)
- Post in #and_live_help M-Th 5-9pm PT to Udacity staff
- Look out for Udacity Project Coach AMA times
- Reach out to Udacity mentors in your classroom