Andorid Programming

Week 11

Mina Jung

EECS, Syracuse University

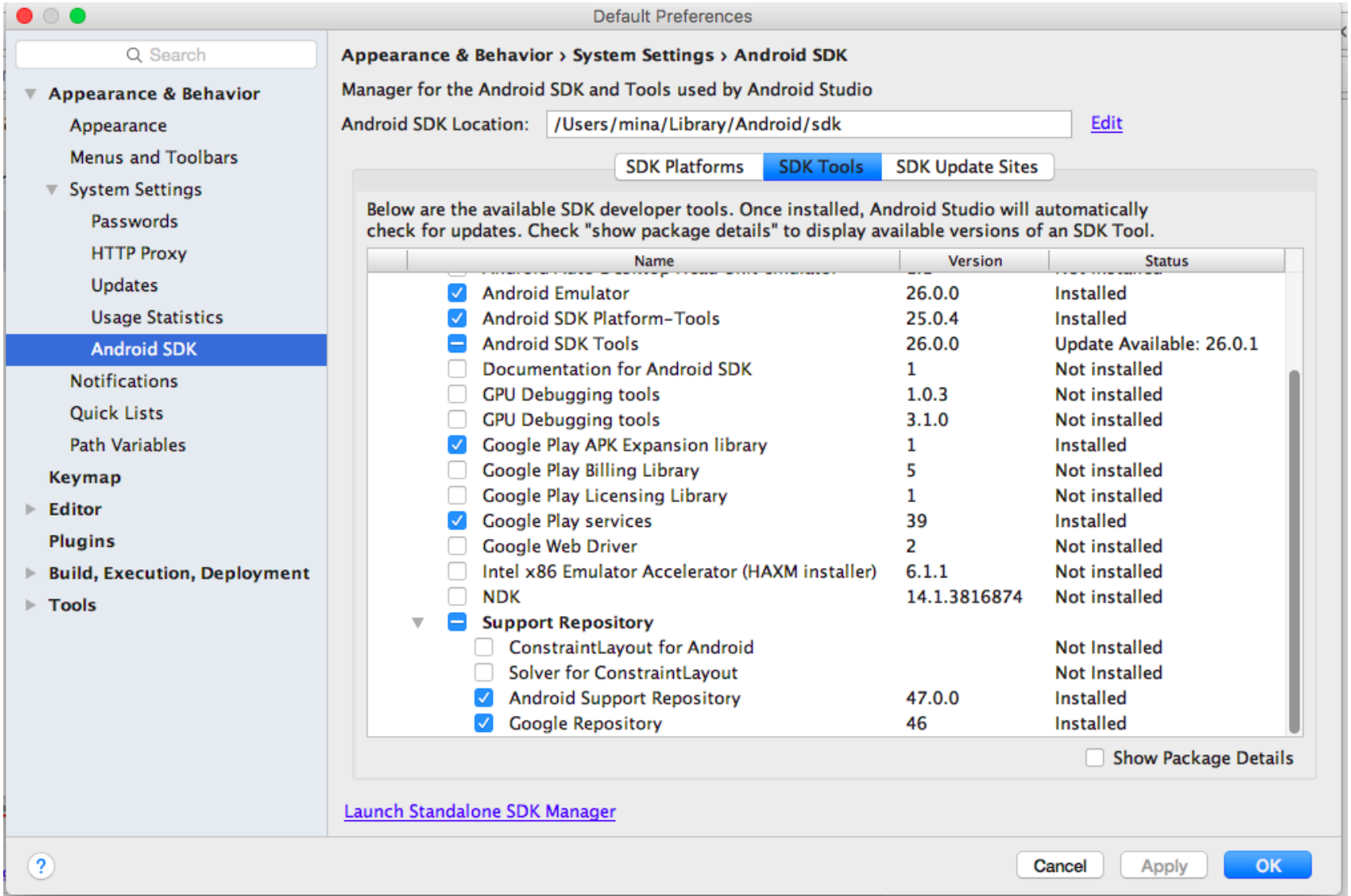
Spring 2017

Part I

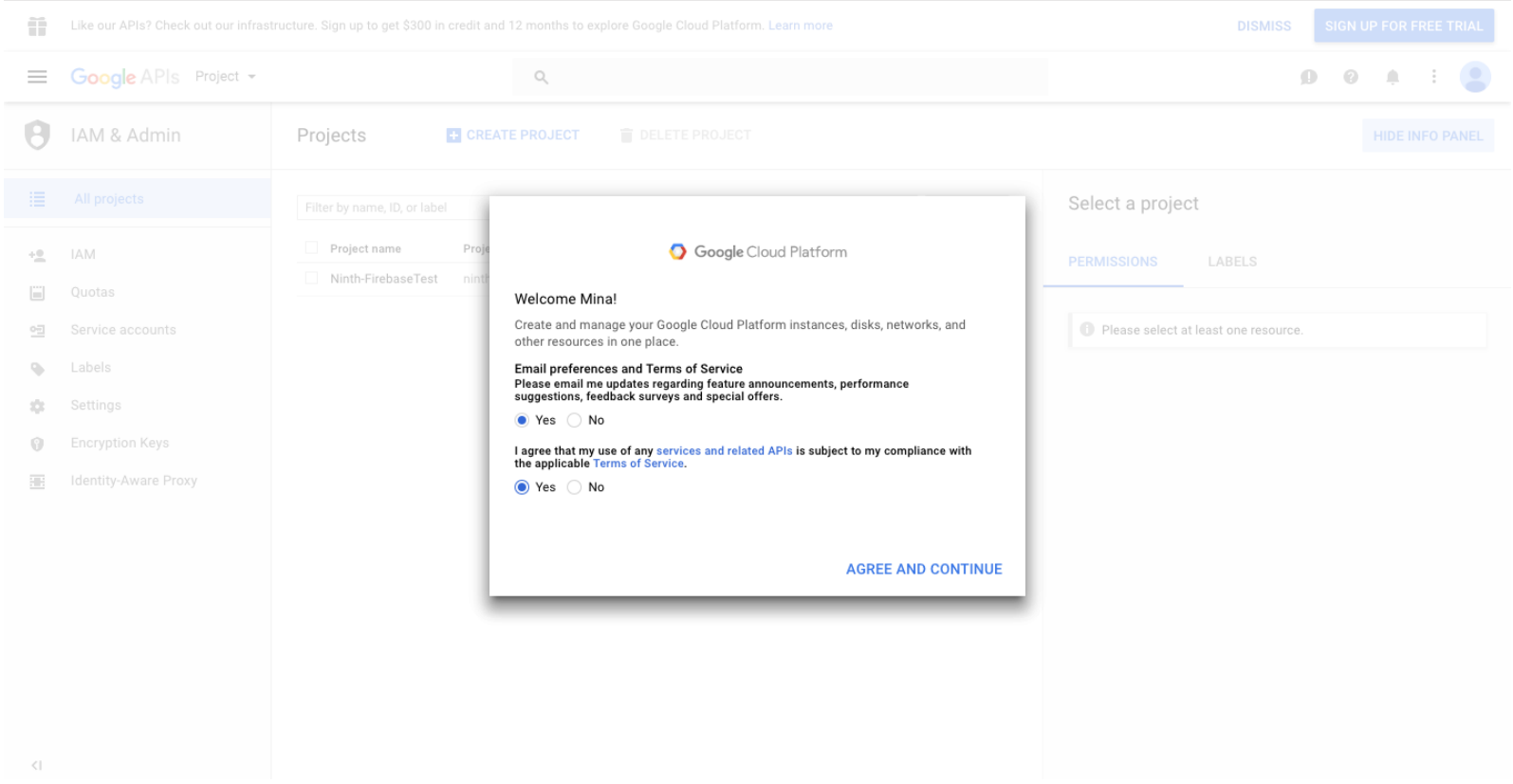
Google Map

Outline I

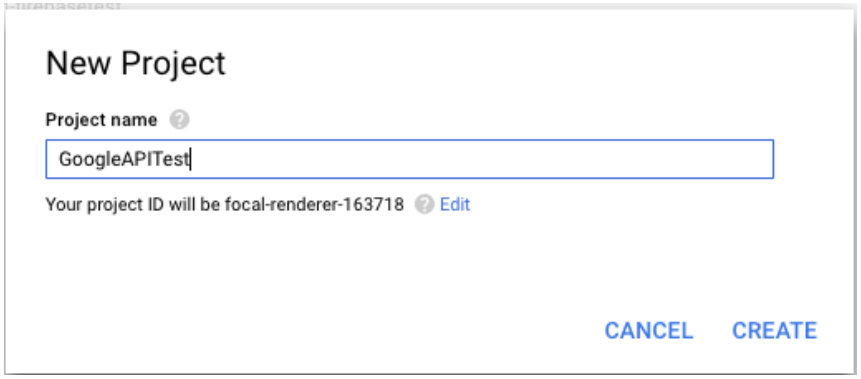
https://guides.codepath.com/android/Genymotion-2.0-Emulators-with-Google-Play-support



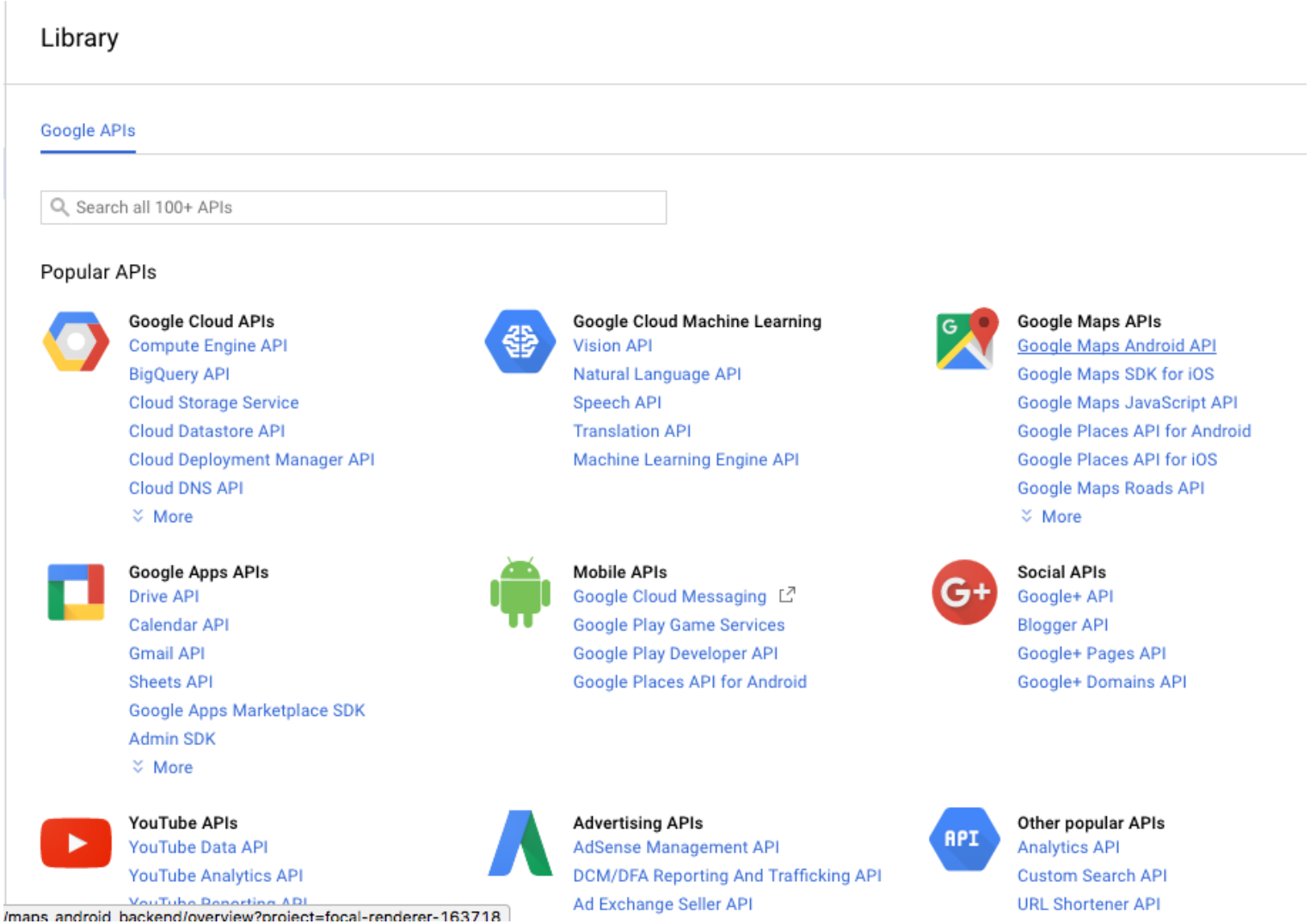
1. go to https://console.developers.google.com/



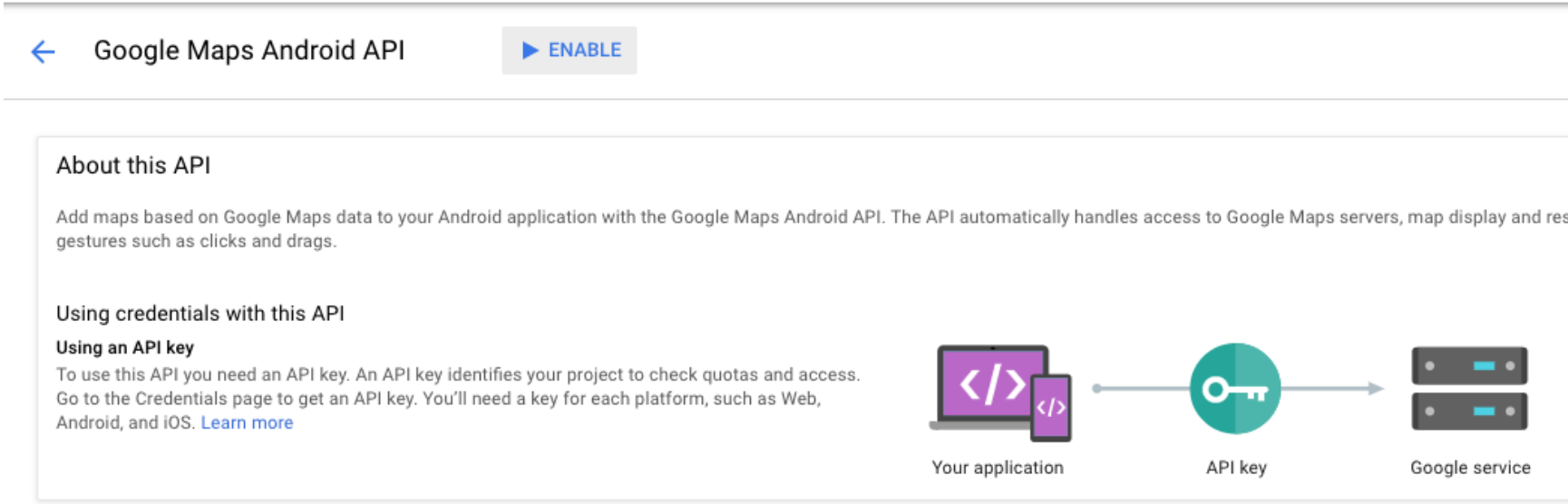
2. create a new project



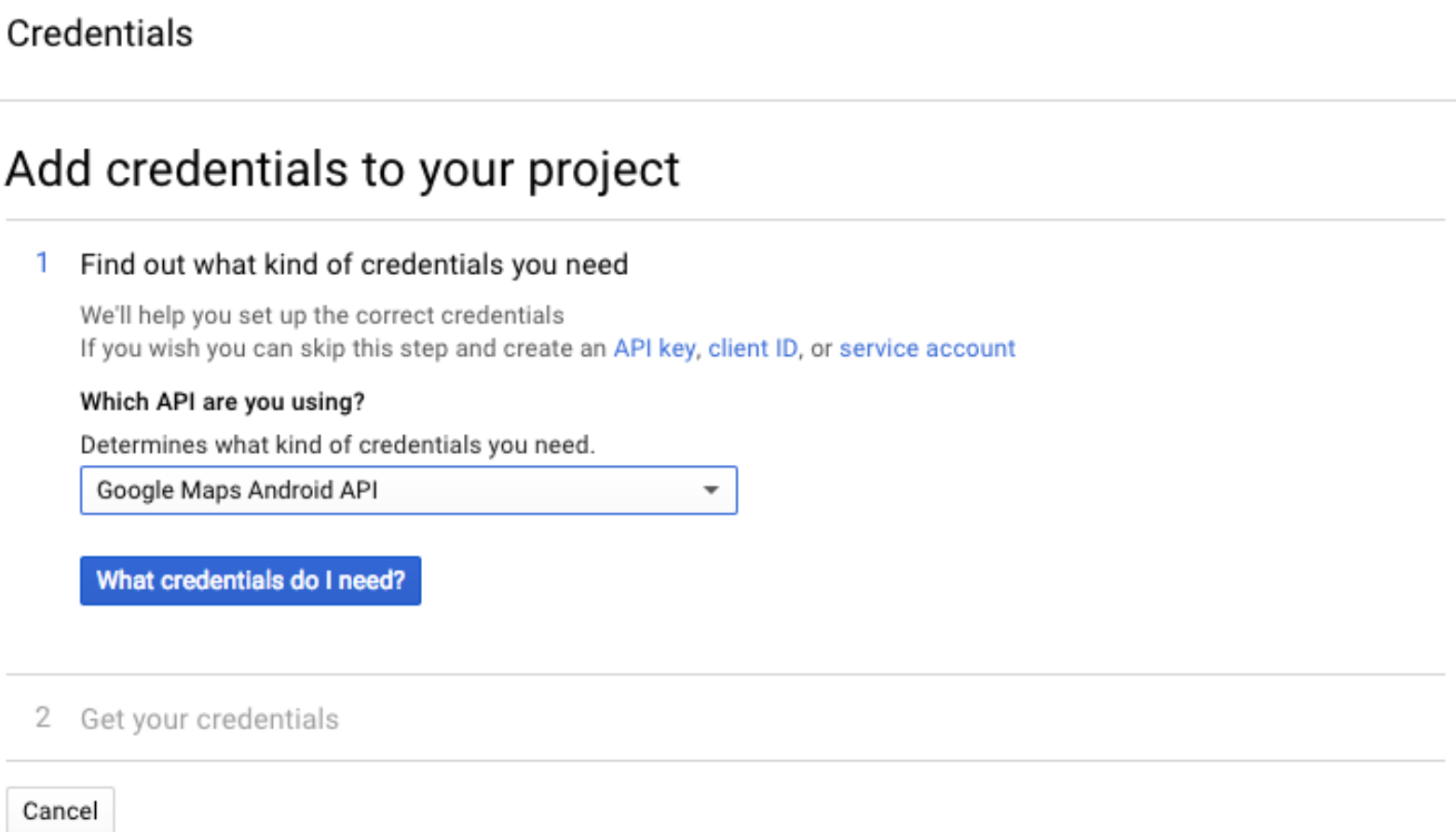
3. select API


Library
Google APIs
Search all 100+ APIs
Popular APIs
Google Cloud APIs
Compute Engine API
BigQuery API
Cloud Storage Service
Cloud Datastore API
Cloud Deployment Manager API
Cloud DNS API
More
Google Cloud Machine Learning
Vision API
Natural Language API
Speech API
Translation API
Machine Learning Engine API
Google Maps APIs
Google Maps Android API
Google Maps SDK for iOS
Google Maps JavaScript API
Google Places API for Android
Google Places API for iOS
Google Maps Roads API
More
Google Apps APIs
Drive API
Calendar API
Gmail API
Sheets API
Google Apps Marketplace SDK
Admin SDK
More
Mobile APIs
Google Cloud Messaging
Google Play Game Services
Google Play Developer API
Google Places API for Android
Social APIs
Google+ API
Blogger API
Google+ Pages API
Google+ Domains API
YouTube APIs
YouTube Data API
YouTube Analytics API
Advertising APIs
AdSense Management API
DCM/DFA Reporting And Trafficking API
Ad Exchange Seller API
Other popular APIs
Analytics API
Custom Search API
URL Shortener API
/maps.android.backend/overview?project=focal-renderer-163718

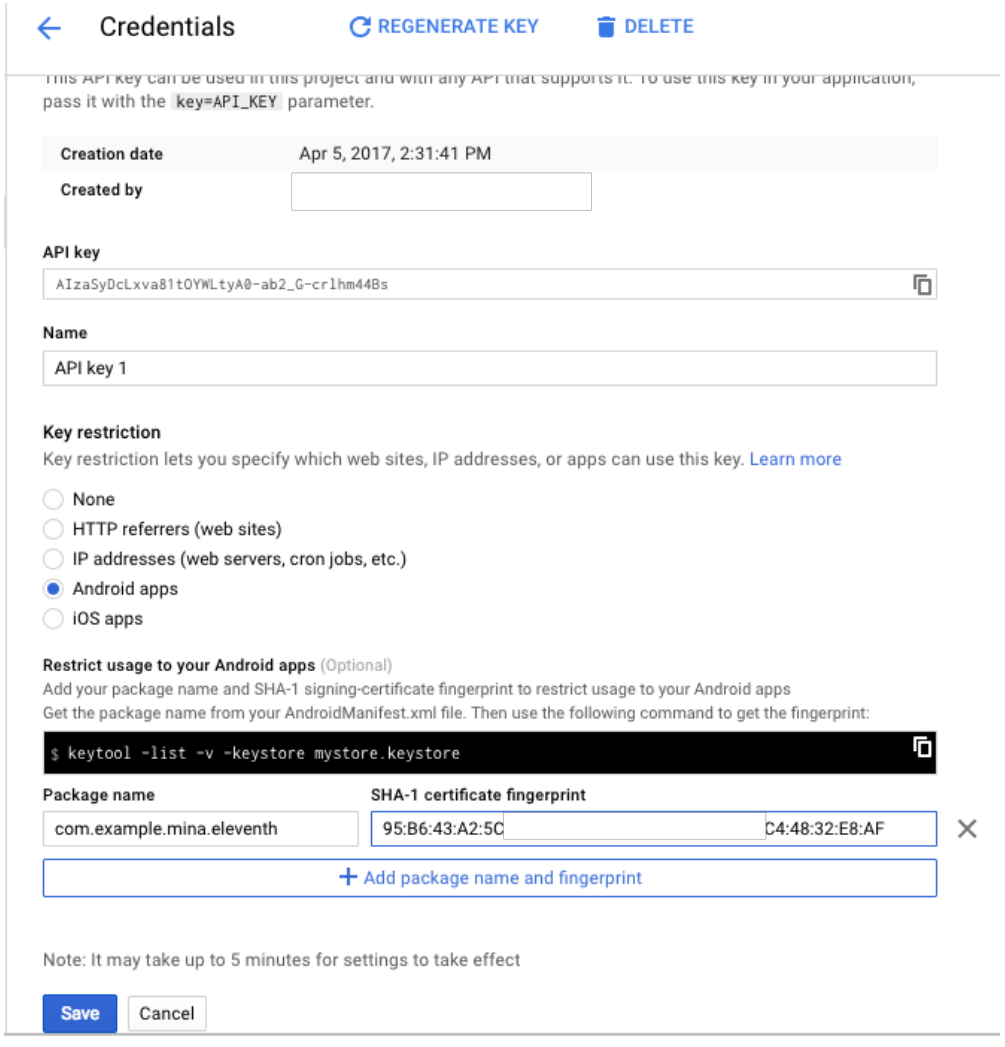
4. enable API


Google Maps Android API
ENABLE
About this API
Add maps based on Google Maps data to your Android application with the Google Maps Android API. The API automatically handles access to Google Maps servers, map display and res
gestures such as clicks and drags.
Using credentials with this API
Using an API key
To use this API you need an API key. An API key identifies your project to check quotas and access.
Go to the Credentials page to get an API key. You'll need a key for each platform, such as Web,
Android, and iOS. Learn more
Your application
API key
Google service

5. add credentials



6. add key restriction



1. Copy your API key and add it into string value (strings.xml)

```
<string name="google_maps_key"
    templateMergeStrategy="preserve">YOUR_KEY_HERE</string>
```

2. Update your app gradle

```
...
dependencies {
    compile fileTree(dir: 'libs', include: ['*.jar'])
    androidTestCompile('com.android.support.test.espresso:espresso-core
        {
        exclude group: 'com.android.support', module:
            'support-annotations'
    })
    compile 'com.android.support:appcompat-v7:25.3.1'
    compile 'com.android.support:design:25.3.1'
    testCompile 'junit:junit:4.12'

    compile 'com.google.android.gms:play-services-maps:10.2.1'
    compile 'com.google.android.gms:play-services-location:10.2.1'
    compile 'com.google.android.gms:play-services-places:10.2.1'

    compile 'com.google.android.gms:play-services:10.2.1'
    compile 'com.android.support:multidex:1.0.1'
    compile files('libs/YouTubeAndroidPlayerApi.jar')
```

```
    compile 'noman.placesapi:placesAPI:1.1.3'
}
```

3. Required permissions in AndroidManifest.xml to access the location of the device

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="com.example.mina.eleventh">

    <uses-permission
        android:name="android.permission.ACCESS_FINE_LOCATION" />
    <uses-permission
        android:name="android.permission.ACCESS_COARSE_LOCATION" />
    <uses-permission android:name="android.permission.INTERNET" />

    <application
        android:allowBackup="false"
        android:icon="@mipmap/ic_launcher"
        android:label="@string/app_name"
        android:supportsRtl="true"
        android:theme="@style/AppTheme">
        <activity
            android:name=".MainActivity"
            android:label="@string/app_name"
```

```
            android:theme="@style/AppTheme.NoActionBar">
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />

                <category
                    android:name="android.intent.category.LAUNCHER"
                    />
            </intent-filter>
        </activity>

        <meta-data
            android:name="com.google.android.geo.API_KEY"
            android:value="@string/google_maps_key" />

    </application>

</manifest>
```

4. Creating a Google API Client

```
private GoogleApiClient mGoogleApiClient;
...

// Activity's onCreate method
mGoogleApiClient = new GoogleApiClient.Builder(this)
        .addConnectionCallbacks(this)
        .addOnConnectionFailedListener(this)
        .addApi(LocationServices.API)
        .build();
```

5. Implementing Location Callbacks

```
ThisActivity ... implements
        GoogleApiClient.ConnectionCallbacks,
        GoogleApiClient.OnConnectionFailedListener {
...

@Override
public void onConnected(Bundle bundle) {
    Log.i(TAG, "Location services connected.");
}

@Override
public void onConnectionSuspended(int i) {
    Log.i(TAG, "Location services suspended. Please reconnect.");
}
```

6. Connecting and Disconnecting

```
@Override
protected void onResume() {
    super.onResume();
    setUpMapIfNeeded();
    mGoogleApiClient.connect();
}
...

@Override
protected void onPause() {
    super.onPause();
    if (mGoogleApiClient.isConnected()) {
        mGoogleApiClient.disconnect();
    }
}
```

7. Handling Errors

```
@Override
public void onConnectionFailed(ConnectionResult connectionResult) {
    if (connectionResult.hasResolution()) {
        try {
            // Start an Activity that tries to resolve the error
            connectionResult.startResolutionForResult(this,
                CONNECTION_FAILURE_RESOLUTION_REQUEST);
        } catch (IntentSender.SendIntentException e) {
            e.printStackTrace();
        }
    } else {
        Log.i(TAG, "Location services connection failed with code "
            + connectionResult.getErrorCode());
    }
}
```

8. Requesting Location Updates: implement LocationListener interface

```
@Override
public void onLocationChanged(Location location) {
    handleNewLocation(location);
}

...

// create request
private LocationRequest mLocationRequest;

// start request in Activity's onCreate method
mLocationRequest = LocationRequest.create()
        .setPriority(LocationRequest.PRIORITY_HIGH_ACCURACY)
        .setInterval(10 * 1000)         // 10 seconds, in
            milliseconds
        .setFastestInterval(1 * 1000); // 1 second, in milliseconds

...

@Override
public void onConnected(Bundle bundle) {
    Location location =
         LocationServices.FusedLocationApi.getLastLocation(mGoogleApiCl
    if (location == null) {
        LocationServices.FusedLocationApi.requestLocationUpdates(mGoogl
            mLocationRequest, this);
```

```
        }
        else {
            handleNewLocation(location);
        }
    }

    ...


    @Override
    protected void onPause() {
        super.onPause();
        if (mGoogleApiClient.isConnected()) {
            LocationServices.FusedLocationApi.removeLocationUpdates(mGoogle
                this);
            mGoogleApiClient.disconnect();
        }
    }

    ...

    private void handleNewLocation(Location location) {
        Log.d(TAG, location.toString());

        double currentLatitude = location.getLatitude();
        double currentLongitude = location.getLongitude();
```

```
        LatLng latLng = new LatLng(currentLatitude, currentLongitude);

        MarkerOptions options = new MarkerOptions()
            .position(latLng)
            .title("I am here!");
        mMap.addMarker(options);
        mMap.moveCamera(CameraUpdateFactory.newLatLng(latLng));
    }
```

Part II

YouTube Player in your App

Outline I

1. Go to Google Developer Console and select or create a new project (or select an existing project)
2. On the left sidebar, select Library and choose YouTube Data API (version 3)
3. On the left sidebar, select Credentials and Create new key
4. Paste the SHA-1 key and your project's package name
5. You should see the API KEY on the dashboard

Download the latest of version of YouTube Android Player API and extract it. Once extracted, you can find YouTubeAndroidPlayerApi.jar file inside libs folder.

YouTube Player

```
//Opens in the StandAlonePlayer, defaults to fullscreen
if (YouTubeIntents.canResolvePlayVideoIntent(this)) {
  //Opens in the StandAlonePlayer, defaults to fullscreen
  startActivity(YouTubeStandalonePlayer.createVideoIntent(this,
       getString(R.string.google_maps_key), "68A_HPYGdlk", 50000,
       true, true));
            }
```

```
Intent intentStartYoutube =
      YouTubeIntents.createPlayVideoIntent(getApplicationContext(),
          VIDEO_ID);
    startActivity(intentStartYoutube);
```

- an alternative to using the YouTubePlayerFragment
- your activity needs to extend YouTubeBaseActivity

```
<com.google.android.youtube.player.YouTubePlayerView
  android:id="@+id/youtube_view"
  android:layout_width="match_parent"
  android:layout_height="wrap_content"
  android:layout_marginBottom="30dp" />
```

```
public class Youtube1Activity extends YouTubeBaseActivity implements
    YouTubePlayer.OnInitializedListener, View.OnClickListener {

        private static final int RECOVERY_DIALOG_REQUEST = 1;

        // YouTube player view
        private YouTubePlayerView youTubeView;

        @Override
        protected void onCreate(Bundle savedInstanceState) {
            super.onCreate(savedInstanceState);
            requestWindowFeature(Window.FEATURE_NO_TITLE);
            getWindow().setFlags(WindowManager.LayoutParams.FLAG_FULLSCREEN
                    WindowManager.LayoutParams.FLAG_FULLSCREEN);

            setContentView(R.layout.activity_youtube1);

            youTubeView = (YouTubePlayerView)
                findViewById(R.id.youtube_view);

            // Initializing video player with developer key
            youTubeView.initialize(getString(R.string.google_maps_key),
                this);

        }
```

```
@Override
public void onInitializationFailure(YouTubePlayer.Provider
     provider,
         YouTubeInitializationResult errorReason) {
     if (errorReason.isUserRecoverableError()) {
         errorReason.getErrorDialog(this,
              RECOVERY_DIALOG_REQUEST).show();
     } else {
         String errorMessage = String.format(
                  getString(R.string.error_player),
                       errorReason.toString());
         Toast.makeText(this, errorMessage,
              Toast.LENGTH_LONG).show();
     }
}

@Override
public void onInitializationSuccess(YouTubePlayer.Provider
     provider,
         YouTubePlayer player, boolean wasRestored) {
     if (!wasRestored) {

         // loadVideo() will auto play video
         // Use cueVideo() method, if you don't want to play
              it automatically
         player.loadVideo(getString(R.string.video_code));
```

```
                // Hiding player controls
                player.setPlayerStyle(YouTubePlayer.PlayerStyle.CHROMELESS)
            }
        }

        @Override
        protected void onActivityResult(int requestCode, int
            resultCode, Intent data) {
            if (requestCode == RECOVERY_DIALOG_REQUEST) {
                // Retry initialization if user performed a
                    recovery action
                getYouTubePlayerProvider().initialize(getString(R.string.go
                    this);
            }
        }

    private YouTubePlayer.Provider getYouTubePlayerProvider() {
        return (YouTubePlayerView) findViewById(R.id.youtube_view);
    }

    @Override
    public void onClick(View v) {

    }
```

```
}
```

- fragment containing a YouTubePlayerView
- activity does not need to extend an activity provided by the library, as is the case with using the YouTubePlayerView directly

```
<fragment
 android:name="com.google.android.youtube.player.YouTubePlayerSupportFragme
 android:id="@+id/moviePlayer"
 android:layout_width="match_parent"
 android:layout_height="wrap_content"/>
```

```
public class MainActivity extends YouTubeBaseActivity
 implements YouTubePlayer.OnInitializedListener{

 public static final String DEVELOPER_KEY = "replace your own API
      Key here";
 private static final int RECOVERY_DIALOG_REQUEST = 1;
 private static final String VIDEO_ID = "fhWaJi1Hsfo";

 YouTubePlayerFragment myYouTubePlayerFragment;

 @Override
 protected void onCreate(Bundle savedInstanceState) {
  super.onCreate(savedInstanceState);
  setContentView(R.layout.activity_main);
  myYouTubePlayerFragment =
       (YouTubePlayerFragment)getFragmentManager()
    .findFragmentById(R.id.youtubeplayerfragment);
  myYouTubePlayerFragment.initialize(DEVELOPER_KEY, this);
 }

 @Override
 public void onInitializationFailure(YouTubePlayer.Provider
      provider,
   YouTubeInitializationResult errorReason) {
  if (errorReason.isUserRecoverableError()) {
   errorReason.getErrorDialog(this, RECOVERY_DIALOG_REQUEST).show();
```

```
 } else {
  String errorMessage = String.format(
    "There was an error initializing the YouTubePlayer (%1$s)",
    errorReason.toString());
  Toast.makeText(this, errorMessage, Toast.LENGTH_LONG).show();
 }
}

@Override
public void onInitializationSuccess(Provider provider,
     YouTubePlayer player,
  boolean wasRestored) {
 if (!wasRestored) {
       player.cueVideo(VIDEO_ID);
 }
}

@Override
protected void onActivityResult(int requestCode, int resultCode,
     Intent data) {

 if (requestCode == RECOVERY_DIALOG_REQUEST) {
  // Retry initialization if user performed a recovery action
  getYouTubePlayerProvider().initialize(DEVELOPER_KEY, this);
 }
}
```

```
    protected YouTubePlayer.Provider getYouTubePlayerProvider() {
     return (YouTubePlayerView)findViewById(R.id.moviePlayer);
    }

}
```

```
<com.google.android.youtube.player.YouTubeThumbnailView
        android:id="@+id/thumbnailview"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content" />
```

```
public class MainActivity extends Activity
 implements YouTubeThumbnailView.OnInitializedListener{

 public static final String DEVELOPER_KEY = "replace your own API
     Key here";
 private static final String VIDEO_ID = "fhWaJi1Hsfo";

 private YouTubeThumbnailLoader youTubeThumbnailLoader;
 private YouTubeThumbnailView thumbnailView;

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

  thumbnailView =
      (YouTubeThumbnailView)findViewById(R.id.thumbnailview);
  thumbnailView.initialize(DEVELOPER_KEY, this);
```

```
}

@Override
public void onInitializationFailure(YouTubeThumbnailView
     thumbnailView,
  YouTubeInitializationResult errorReason) {

 String errorMessage =
   String.format("onInitializationFailure (%1$s)",
     errorReason.toString());
 Toast.makeText(this, errorMessage, Toast.LENGTH_LONG).show();
}

@Override
public void onInitializationSuccess(YouTubeThumbnailView
     thumbnailView,
  YouTubeThumbnailLoader thumbnailLoader) {

 Toast.makeText(getApplicationContext(),
        "onInitializationSuccess", Toast.LENGTH_SHORT).show();

 youTubeThumbnailLoader = thumbnailLoader;
    thumbnailLoader.setOnThumbnailLoadedListener(new
         ThumbnailListener());

    youTubeThumbnailLoader.setVideo(VIDEO_ID);
```

```
    }

    private final class ThumbnailListener implements
        YouTubeThumbnailLoader.OnThumbnailLoadedListener {

      @Override
      public void onThumbnailLoaded(YouTubeThumbnailView thumbnail,
          String videoId) {
        Toast.makeText(getApplicationContext(),
          "onThumbnailLoaded", Toast.LENGTH_SHORT).show();
      }

      @Override
      public void onThumbnailError(YouTubeThumbnailView thumbnail,
          YouTubeThumbnailLoader.ErrorReason reason) {
       Toast.makeText(getApplicationContext(),
         "onThumbnailError", Toast.LENGTH_SHORT).show();
      }
    }

}
```