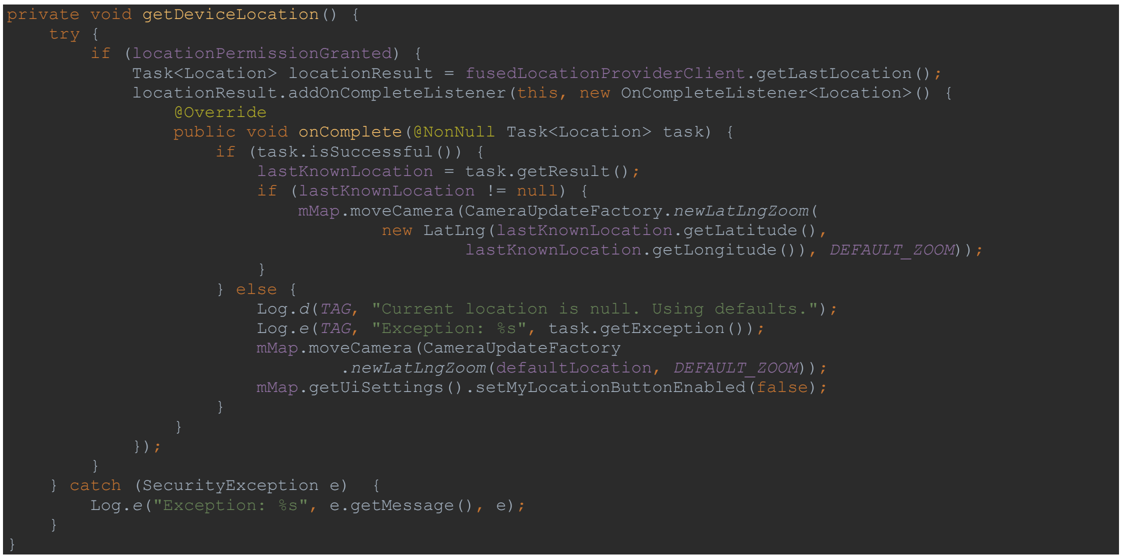
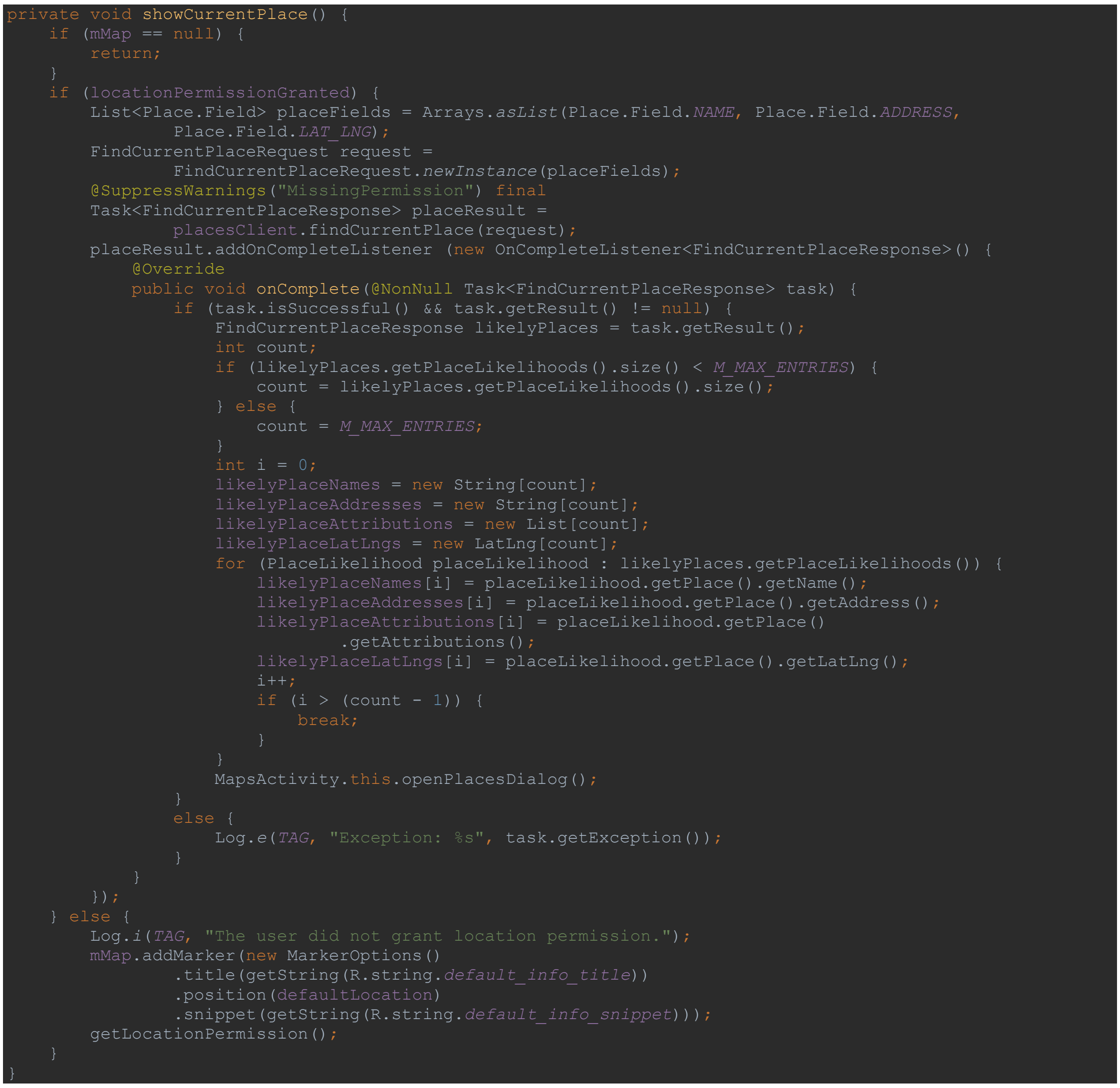
1. getDeviceLocation()



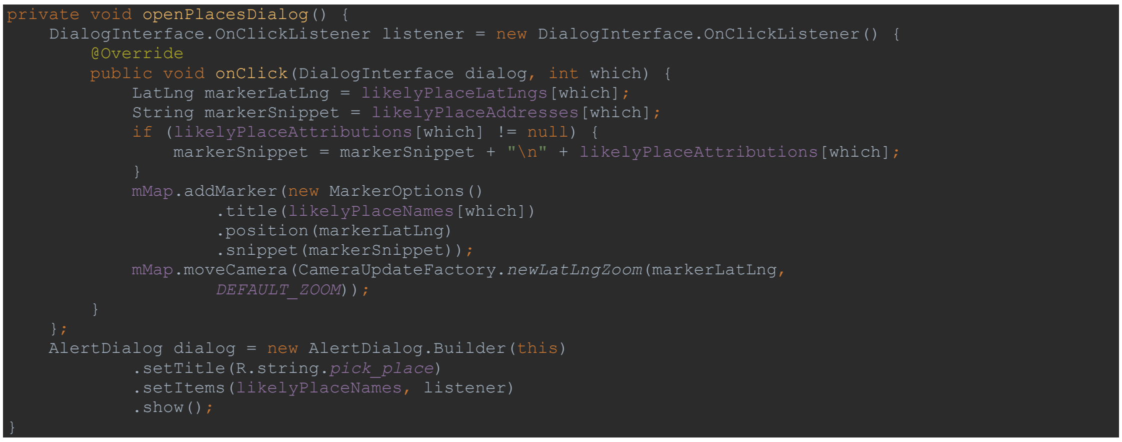
```
private void getDeviceLocation() {
    try {
        if (locationPermissionGranted) {
            Task<Location> locationResult = fusedLocationProviderClient.getLastLocation();
            locationResult.addOnCompleteListener(this, new OnCompleteListener<Location>() {
                @Override
                public void onComplete(@NonNull Task<Location> task) {
                    if (task.isSuccessful()) {
                        lastKnownLocation = task.getResult();
                        if (lastKnownLocation != null) {
                            mMap.moveCamera(CameraUpdateFactory.newLatLngZoom(
                                    new LatLng(lastKnownLocation.getLatitude(),
                                            lastKnownLocation.getLongitude()), DEFAULT_ZOOM));
                        }
                    } else {
                        Log.d(TAG, "Current location is null. Using defaults.");
                        Log.e(TAG, "Exception: %s", task.getException());
                        mMap.moveCamera(CameraUpdateFactory
                                .newLatLngZoom(defaultLocation, DEFAULT_ZOOM));
                        mMap.getUiSettings().setMyLocationButtonEnabled(false);
                    }
                }
            });
        }
    } catch (SecurityException e)  {
        Log.e("Exception: %s", e.getMessage(), e);
    }
}
```

2. showCurrentPlace()



```
private void showCurrentPlace() {
    if (mMap == null) {
        return;
    }
    if (locationPermissionGranted) {
        List<Place.Field> placeFields = Arrays.asList(Place.Field.NAME, Place.Field.ADDRESS,
                Place.Field.LAT_LNG);
        FindCurrentPlaceRequest request =
                FindCurrentPlaceRequest.newInstance(placeFields);
        @SuppressWarnings("MissingPermission") final
        Task<FindCurrentPlaceResponse> placeResult =
                placesClient.findCurrentPlace(request);
        placeResult.addOnCompleteListener (new OnCompleteListener<FindCurrentPlaceResponse>() {
            @Override
            public void onComplete(@NonNull Task<FindCurrentPlaceResponse> task) {
                if (task.isSuccessful() && task.getResult() != null) {
                    FindCurrentPlaceResponse likelyPlaces = task.getResult();
                    int count;
                    if (likelyPlaces.getPlaceLikelihoods().size() < M_MAX_ENTRIES) {
                        count = likelyPlaces.getPlaceLikelihoods().size();
                    } else {
                        count = M_MAX_ENTRIES;
                    }
                    int i = 0;
                    likelyPlaceNames = new String[count];
                    likelyPlaceAddresses = new String[count];
                    likelyPlaceAttributions = new List[count];
                    likelyPlaceLatLngs = new LatLng[count];
                    for (PlaceLikelihood placeLikelihood : likelyPlaces.getPlaceLikelihoods()) {
                        likelyPlaceNames[i] = placeLikelihood.getPlace().getName();
                        likelyPlaceAddresses[i] = placeLikelihood.getPlace().getAddress();
                        likelyPlaceAttributions[i] = placeLikelihood.getPlace()
                                .getAttributions();
                        likelyPlaceLatLngs[i] = placeLikelihood.getPlace().getLatLng();
                        i++;
                        if (i > (count - 1)) {
                            break;
                        }
                    }
                    MapsActivity.this.openPlacesDialog();
                }
                else {
                    Log.e(TAG, "Exception: %s", task.getException());
                }
            }
        });
    } else {
        Log.i(TAG, "The user did not grant location permission.");
        mMap.addMarker(new MarkerOptions()
                .title(getString(R.string.default_info_title))
                .position(defaultLocation)
                .snippet(getString(R.string.default_info_snippet)));
        getLocationPermission();
    }
}
```

3. openPlacesDialog()



```
private void openPlacesDialog() {
    DialogInterface.OnClickListener listener = new DialogInterface.OnClickListener() {
        @Override
        public void onClick(DialogInterface dialog, int which) {
            LatLng markerLatLng = likelyPlaceLatLngs[which];
            String markerSnippet = likelyPlaceAddresses[which];
            if (likelyPlaceAttributions[which] != null) {
                markerSnippet = markerSnippet + "\n" + likelyPlaceAttributions[which];
            }
            mMap.addMarker(new MarkerOptions()
                    .title(likelyPlaceNames[which])
                    .position(markerLatLng)
                    .snippet(markerSnippet));
            mMap.moveCamera(CameraUpdateFactory.newLatLngZoom(markerLatLng,
                    DEFAULT_ZOOM));
        }
    };
    AlertDialog dialog = new AlertDialog.Builder(this)
            .setTitle(R.string.pick_place)
            .setItems(likelyPlaceNames, listener)
            .show();
}
```