APLIKACJE MOBILNE

Wykład 08

GPS
GOOGLE MAPS

dr Artur Bartoszewski

GPS
ODCZYTYWANIE POZYCJI

Aplikacje mobilne

LocationManager - klasa ta zapewnia dostęp do usług lokalizacji systemu.

Usługi te umożliwiają aplikacjom uzyskiwanie okresowych aktualizacji lokalizacji geograficznej urządzenia lub uruchamianie akcji określonej przez aplikację, gdy urządzenie znajdzie się w pobliżu danej lokalizacji.

LocationListener – klasa, która obsługuje zdarzenia wysyłane przez LocationMenagera-a. Najważniejsza jej metoda to .onLocationChanged() wywoływana przy zmianie lokalizacji.

Aplikacje mobilne

Przygotowanie aplikacji pobierającej lokalizację wymaga:

1. utworzenia referencji do klasy LocationMenager,
2. utworzenia menadżera lokalizacji – pobieramy usługę systemową: getSystemService(***LOCATION_SERVICE***).
3. utworzenia słuchacza zmian lokalizacji LocationListener.
4. wywołania żądania odczytania pozycji: .requestLocationUpdates().
5. oprogramowanie metody .onLocationChanged() słuchacza.

1. Utworzenie referencji do klasy LocationMenager,
2. Utworzenie menadżera lokalizacji – pobieramy usługę systemową: getSystemService(***LOCATION_SERVICE***)

```
LocationManager locationManager01;
@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_main);
    locationManager01 = (LocationManager)getSystemService(LOCATION_SERVICE);
}
```

3. Utworzenie słuchacza zmian lokalizacji LocationListener

```
LocationManager locationManager01;
LocationListener locationListener01;
@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_main);
    locationManager01 = (LocationManager)getSystemService(LOCATION_SERVICE);
    locationListener01 = new LocationListener() {
        @Override
        public void onLocationChanged(Location location) {}
        @Override
        public void onStatusChanged(String provider, int status, Bundle extras) {}
        @Override
        public void onProviderEnabled(String provider) {}
        @Override
        public void onProviderDisabled(String provider) {}
    };
```

4. Wywołanie żądania odczytywania pozycji: .requestLocationUpdates()

Start śledzenia pozycji

```
public void GPSstart(View view) {
    locationManager01.requestLocationUpdates
            (LocationManager.GPS_PROVIDER, minTime: 0, minDistance: 0,locationListener01);
    locationManager01.requestLocationUpdates
            (LocationManager.NETWORK_PROVIDER, minTime: 0, minDistance: 0,locationListener01);
}
```

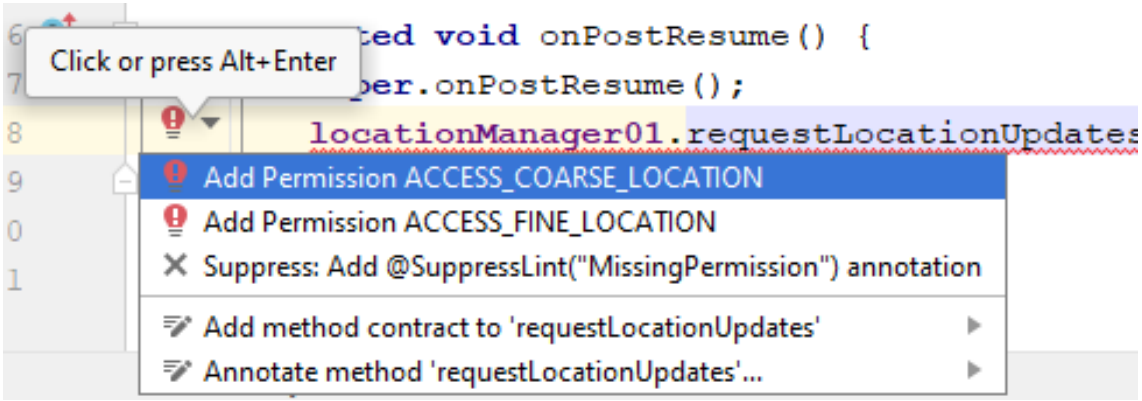
Parametry metody requestLocationUpdates():

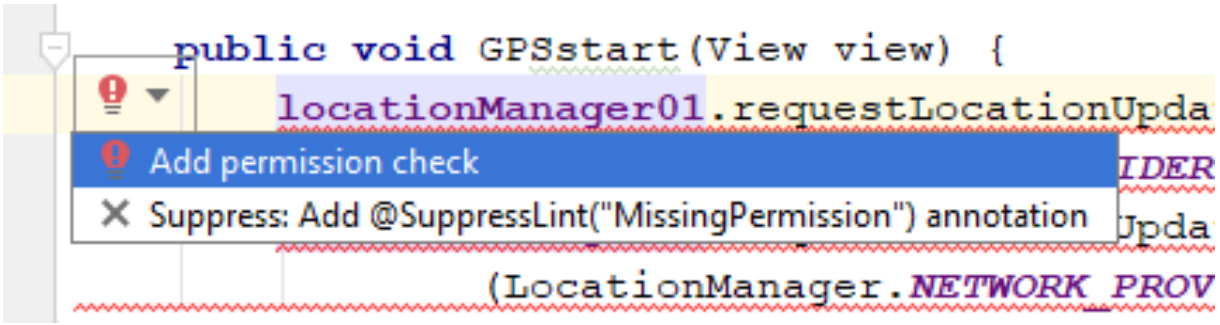
1. Dostawca lokalizacji (GPS lub sieć komórkowa – można używać obu)
2. Czas pomiędzy odczytami
3. Dystans pomiędzy wywołaniami zdarzenia .onLocationChanged()
4. Słuchacz lokalizacji (reaguje na zdarzenia wysłane przez menagera)

Uwaga: parametry ,0,0, oznaczają minimalny możliwy czas i dystans – użytkownik nas za to nie polubi – zużycie zasobów i energii ☺

4. Wywołanie żądania odczytania pozycji: .requestLocationUpdates()

Na poprzednim slajdzie polecenia .requestLocationUpdates() są podkreślone, gdyż brakuje uprawnień aplikacji oraz kontroli uprawnień.





4. Wywołanie żądania odczytywania pozycji: .requestLocationUpdates()

```
public void GPSstart(View view) {
    if (checkSelfPermission(Manifest.permission.ACCESS_FINE_LOCATION) !=
            PackageManager.PERMISSION_GRANTED &&
            checkSelfPermission(Manifest.permission.ACCESS_COARSE_LOCATION)
                    != PackageManager.PERMISSION_GRANTED) {
        // TODO: Consider calling
        //    Activity#requestPermissions
        // here to request the missing permissions, and then overriding
        //   public void onRequestPermissionsResult(int requestCode, String[] permissions,
        //                                          int[] grantResults)
        // to handle the case where the user grants the permission. See the documentation
        // for Activity#requestPermissions for more details.
        return;
    }
    locationManager01.requestLocationUpdates
            (LocationManager.GPS_PROVIDER, minTime: 0, minDistance: 0, locationListener01);
    locationManager01.requestLocationUpdates
            (LocationManager.NETWORK_PROVIDER, minTime: 0, minDistance: 0,locationListener01);
}
```

Sprawdź też uprawnienia w pliku manifest.xml – w razie potrzeby dodaj brakujące.

Wstrzymanie żądania odczytywania pozycji:

```
public void GPSstop(View view) {
    locationManager01.removeUpdates(locationListener01);
}
```

5. oprogramowanie metody .onLocationChanged() słuchacza.

```
locationListener01 = new LocationListener() {
    @Override
    public void onLocationChanged(Location location) {
        opis.setText(
          "Szerokość: "+String.valueOf(location.getLatitude())+
        "\nDługość: "+String.valueOf(location.getLongitude())+
        "\nWysokość: "+String.valueOf(location.getAltitude())+
        "\nDostawca: "+String.valueOf(location.getProvider())
        );
    }
```

Aktualna pozycja przekazywana jest do metody w obiekcie location

Możemy z niego wyciągnąć interesujące nas informacje i wstawić do TextView

ZADANIE PRAKTYCZNE:

Test GPS

```java
public class MainActivity extends AppCompatActivity {
    LocationManager locationManager01;
    LocationListener locationListener01;
    TextView opis;
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        opis = findViewById(R.id.textView01);
        locationManager01 = (LocationManager) getSystemService(LOCATION_SERVICE);
        locationListener01 = new LocationListener() {
            @Override
            public void onLocationChanged(Location location) {
                opis.setText(
                  "Szerokość: "+String.valueOf(location.getLatitude())+
                "\nDługość: "+String.valueOf(location.getLongitude())+
                "\nWysokość: "+String.valueOf(location.getAltitude())+
                "\nDostawca: "+String.valueOf(location.getProvider()));
            }
            @Override
            public void onStatusChanged(String provider, int status, Bundle extras) {}
            @Override
            public void onProviderEnabled(String provider) {}
            @Override
            public void onProviderDisabled(String provider) {}
        };
    }
```

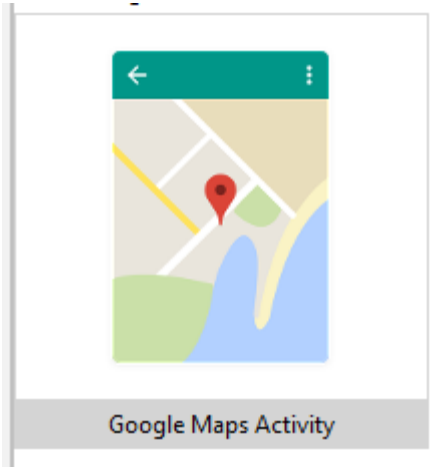
```
public void GPSstart(View view) {
    if (checkSelfPermission(Manifest.permission.ACCESS_FINE_LOCATION)
            != PackageManager.PERMISSION_GRANTED &&
            checkSelfPermission(Manifest.permission.ACCESS_COARSE_LOCATION)
            != PackageManager.PERMISSION_GRANTED) {
        return;
    }
    locationManager01.requestLocationUpdates
            (LocationManager.GPS_PROVIDER, minTime: 0, minDistance: 0, locationListener01);
    locationManager01.requestLocationUpdates
            (LocationManager.NETWORK_PROVIDER, minTime: 0, minDistance: 0, locationListener01);
}
public void GPSstop(View view) {
    locationManager01.removeUpdates(locationListener01);
}
```

Manifest.xml

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

GOOGLE MAPS

PODSTAWY:
KORZYSTANIE Z SZABLONU
GOOGLE MAPS ACTIVITY



Aplikacje mobilne

Aby korzystać z API Google Maps należy uzyskać klucz API.

Po utworzeniu aplikacji z szablonu otwarty jest plik google_maps_api.xml

klikamy link i postępujemy zgodnie z instrukcjami, tzn. musimy zalogować się na swoje konto Google, wybrać lub utworzyć nowy projekt, aktywować dla niego API i na stronie „Credentials” powinniśmy dostać nasz klucz.

Klucz wpisujemy w odpowiednie miejsce:

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

Na starcie otrzymamy mapę ze znacznikiem ustawionym na Sydney

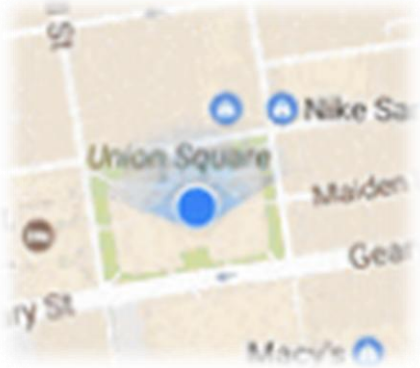
```
@Override
public void onMapReady(GoogleMap googleMap) {
    mMap = googleMap;

    // Add a marker in Sydney and move the camera
    LatLng sydney = new LatLng( v: -34,  v1: 151);
    mMap.addMarker(new MarkerOptions().position(sydney).title("Marker in Sydney"));
    mMap.moveCamera(CameraUpdateFactory.newLatLng(sydney));
}
```

Kilka podstawowych zrobić na mapie - reprezentowanej przez obiekt **mMap**

1. Uruchomienie znacznika aktualnej pozycji i kierunku

```
mMap.setMyLocationEnabled(true);
```

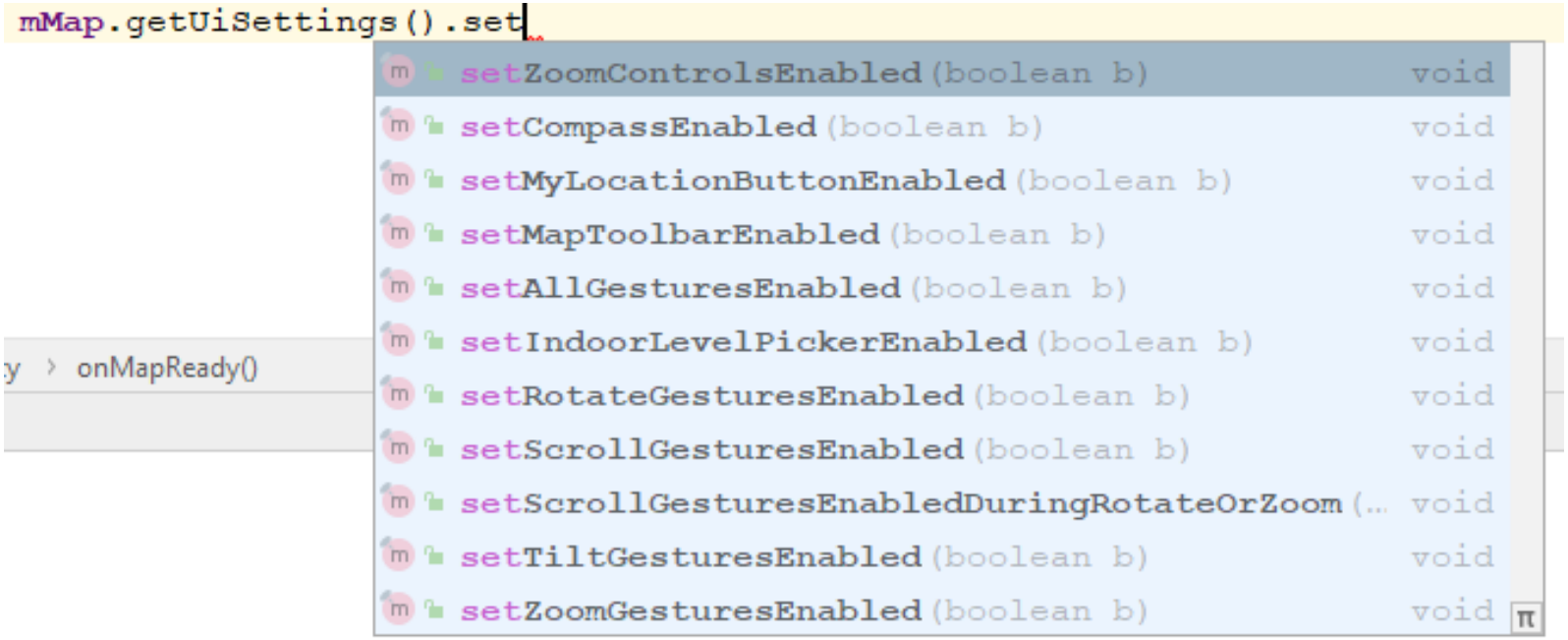


2. Dodanie przycisku przenoszącego kamerę do aktualnej lokalizacji

```
mMap.getUiSettings().setMyLocationButtonEnabled(true);
```

Ustawienia mapy

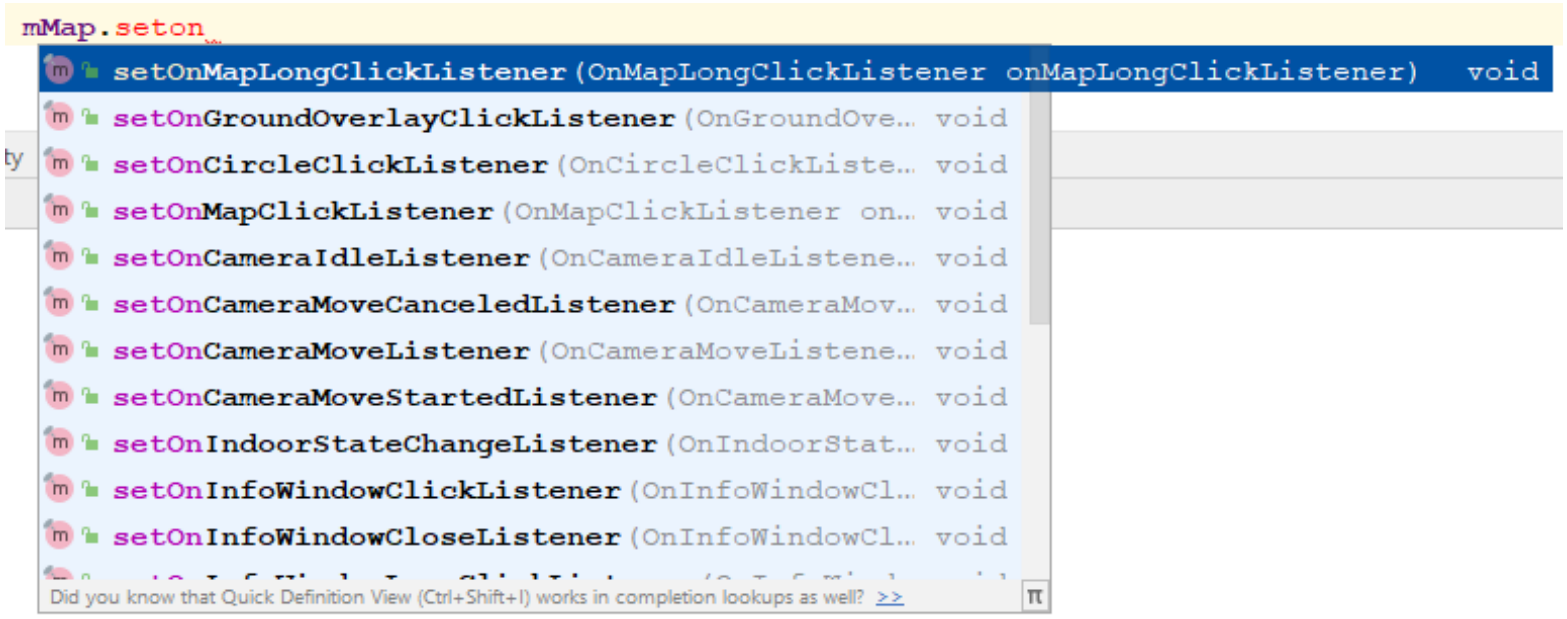
Obiekt **.getUiSettings()** pozwala włączać lub wyłączać wiele innych elementów i ustawień mapy. Można np. zablokować zoom, lub możliwość obrotu.



Zdarzenia mapy

Obiekt mapy przechwytuje różnego rodzaju zdarzenia.

1. Kliknięcie
2. Długie kliknięcie
3. Kliknięcie na aktualną pozycję, klikniecie na marker
4. Itd..



Słuchacz długiego kliknięcia

```
mMap.setOnMapLongClickListener(new GoogleMap.OnMapLongClickListener() {
    @Override
    public void onMapLongClick(LatLng latLng) {
        mMap.addMarker(new MarkerOptions().position(latLng).title("nowy marker"));
        mMap.moveCamera(CameraUpdateFactory.newLatLng(latLng));
    }
});
```

Ustawiamy market w miejscu długiego kliknięcia (przytrzymania) i przesuwamy w to miejsce kamerę.

Metoda onMapLongClick otrzymuje w parametrze pozycję w formacie **LatLang** (Lattitude –Longittude – szerokość – długość geograficzna).

Słuchacz kliknięcia na marker

```
mMap.setOnMarkerClickListener(new GoogleMap.OnMarkerClickListener() {
    @Override
    public boolean onMarkerClick(Marker marker) {
        marker.setVisible(false);
        return false;
    }
});
```

Usuwamy marker, na który klikniemy.

Słuchacz kliknięcia aktualną pozycję

```
mMap.setOnMyLocationClickListener(new GoogleMap.OnMyLocationClickListener() {
    @Override
    public void onMyLocationClick(@NonNull Location location) {
        LatLng tuJestes = new LatLng(location.getLatitude(), location.getLongitude());
        mMap.addMarker(new MarkerOptions().position(tuJestes).title("Pozycja"));
    }
});
```

Ustawiamy marker na aktualnej pozycji

Pozycja przekazana jest za pomocą obiektu klasy **Location**.
Aby zamienić go na pozycję w formacie **LatLang** można ze zmiennej location wyciągnąć długość i szerokość i wstawić je w konstruktor obiektu LatLang

ZADANIE PRAKTYCZNE:

„Okruszek”

Aplikacja zaznacza na mapie punkt startowy. Pokazuje aktualną pozycję. Długim kliknięciem na ekran dodawać możemy „pinezki”

Aplikacje mobilne

1. Zaczynamy od szablonu Google Maps
2. Usuwamy ustawienie przykładowego markera
3. Uruchamiamy wyświetlanie aktualnej pozycji
4. Dodajemy obsługę GPS (jak w poprzednim przykładzie
5. W metodzie .onLocationChanged() słuchacza dodajemy marker – jeżeli nie był już wcześniej ustawiony (nie możemy tego zrobić w onCreate – GPS nie zdąży złapać namiaru
6. Do mapy dodajemy słuchacza zdarzenń – długiego naciśnięcia i umieszczamy w nim wstawienie markera.

```
public class MapsActivity extends FragmentActivity implements OnMapReadyCallback {

    private GoogleMap mMap;
    LocationManager locationMenager01;
    SupportMapFragment mapFragment;
    LocationListener locationListener01;
    Double longit=0.0, latit=0.0;
    LatLng start;
    boolean fix = false;
```

```
@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_maps);
    // Obtain the SupportMapFragment and get notified when the map is ready to be used.
    mapFragment = (SupportMapFragment) getSupportFragmentManager()
            .findFragmentById(R.id.map);
    mapFragment.getMapAsync( onMapReadyCallback: this);
    start = new LatLng(latit,longit);
    locationMenager01 = (LocationManager) getSystemService(LOCATION_SERVICE);
    locationListener01 = new LocationListener() {
        @Override
        public void onLocationChanged(Location location) {
            longit = location.getLongitude();
            latit = location.getLatitude();
            start =  new LatLng(latit,longit);
            if (!fix){
                mMap.addMarker(new MarkerOptions().position(start).title("Punkt startowy"));
                mMap.moveCamera(CameraUpdateFactory.newLatLng(start));
                fix= true;
            }
        }
        @Override
        public void onStatusChanged(String provider, int status, Bundle extras) {}
        @Override
        public void onProviderEnabled(String provider) {}
        @Override
        public void onProviderDisabled(String provider) {}
    };
}
```

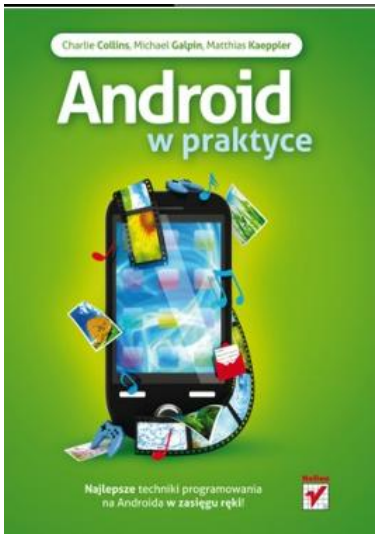
Aplikacje mobilne

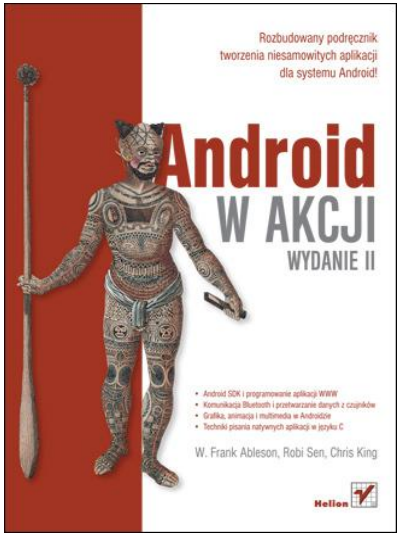
```
@RequiresApi(api = Build.VERSION_CODES.M)
@Override
protected void onResume() {
    super.onResume();
    if (checkSelfPermission(Manifest.permission.ACCESS_FINE_LOCATION)
            != PackageManager.PERMISSION_GRANTED &&
            checkSelfPermission(Manifest.permission.ACCESS_COARSE_LOCATION)
            != PackageManager.PERMISSION_GRANTED) {
        return;
    }
    locationMenager01.requestLocationUpdates
            (LocationManager.GPS_PROVIDER, minTime: 0, minDistance: 0, locationListener01);
    locationMenager01.requestLocationUpdates
            (LocationManager.NETWORK_PROVIDER, minTime: 0, minDistance: 0, locationListener01);
}
```

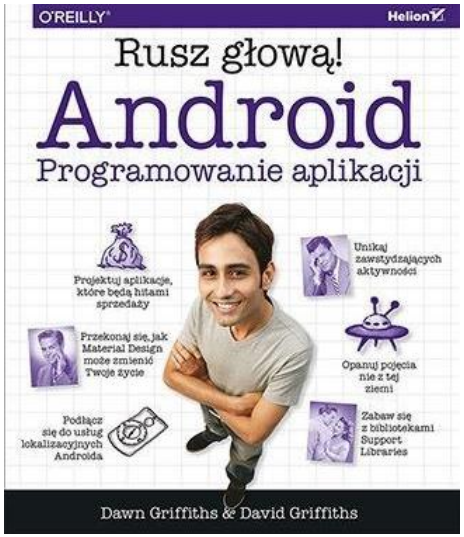
Aplikacje mobilne

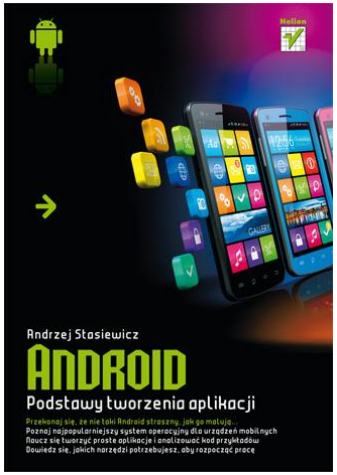
```
@Override
public void onMapReady(GoogleMap googleMap) {
    mMap = googleMap;
    mMap.setMapType(GoogleMap.MAP_TYPE_NORMAL);
    mMap.setMyLocationEnabled(true);
    mMap.setOnMapLongClickListener(new GoogleMap.OnMapLongClickListener() {
        @Override
        public void onMapLongClick(LatLng latLng) {
            mMap.addMarker(new MarkerOptions().position(latLng).title("Tu byłeś"));
        }
    });
    mMap.getUiSettings().setMyLocationButtonEnabled(true);
    mMap.getUiSettings().setZoomControlsEnabled(true);
    mMap.getUiSettings().setCompassEnabled(true);
}
```

Literatura


Charlie Collins, Michael Galpin, Matthias Kaeppler
Android
w praktyce
Najlepsze techniki programowania na Androida w zasięgu ręki!


Rozbudowany podręcznik tworzenia niesamowitych aplikacji dla systemu Android!
Android
W AKCJI
WYDANIE II
W. Frank Ableson, Robi Sen, Chris King
Helion


O'REILLY
Helion
Rusz głową!
Android
Programowanie aplikacji
Projektuj aplikacje, które będą hitami sprzedaży
Unikaj zawstydzających aktywności
Przekonaj się, jak Material Design może zmienić Twoje życie
Opanuj pojęcia nie z tej ziemi
Podłącz się do usług lokalizacyjnych Androida
Zabaw się z bibliotekami Support Libraries
Dawn Griffiths & David Griffiths


Andrzej Stasiewicz
ANDROID
Podstawy tworzenia aplikacji