



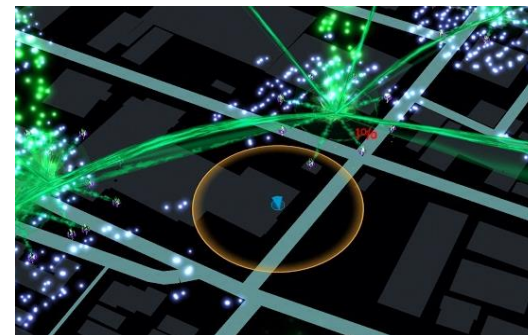
Android Geogame Workshop

Thomas Heinz
University of Bamberg

Location-based Games

- Prominent Examples:

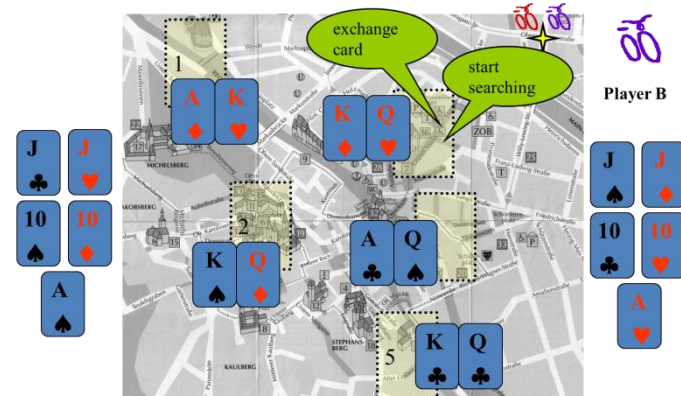
- (Geocaching)
- Google Ingress
- Pokemon Go



Ingress

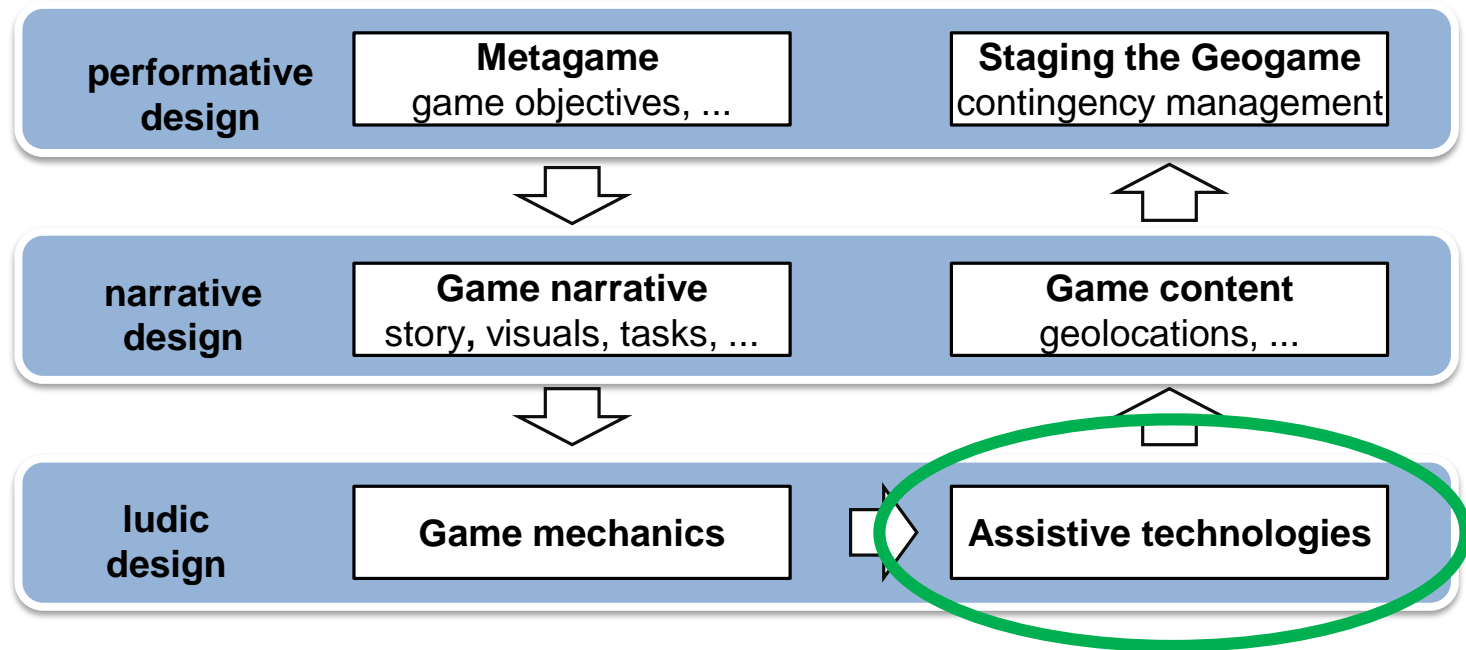
- Our Geogames

- Mostly strategic, competitive multiplayer games



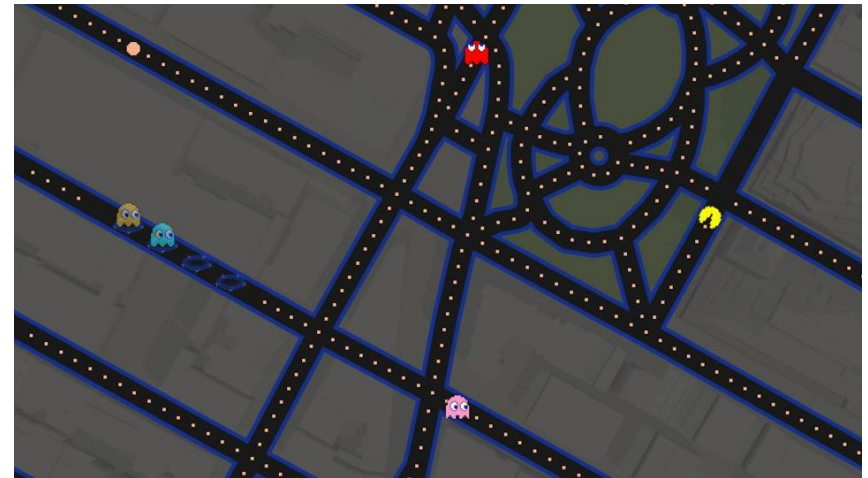
CityPoker

This Workshops Focus



Goal: GeoPacman

- Goal for this workshop:
 - Create a location-based Pacman game
 - Playable anywhere
 - No server required
- Play Pacman:
 - <http://freepacman.org>



<http://www.techradar.com/news/internet/pac-man-goes-wakka-wakka-all-over-google-maps-1289912>

Complimentary Files

- Complimentary files, as well as, solutions for the exercises can be found at:
<http://geogames-team.org/files/uji/>
- Android Studio project for the Guesstimate Geogame can be found at:
<http://geogames-team.org/files/uji/Guesstimate.zip>



Android & Geogame Architecture

Important Android Components - Activities

- Activities
 - An activity is a single, focused thing that the user can do
 - Almost all activities interact with the user
 - An Activity class takes care of creating a window for you in which you can place your UI
 - For example displaying a map
- An android application can consist of multiple activities
- Activities in the system are managed as an *activity stack*
- When a new activity is started, it is placed on the top of the stack and becomes the running activity

Important Android Components – Intents & Services

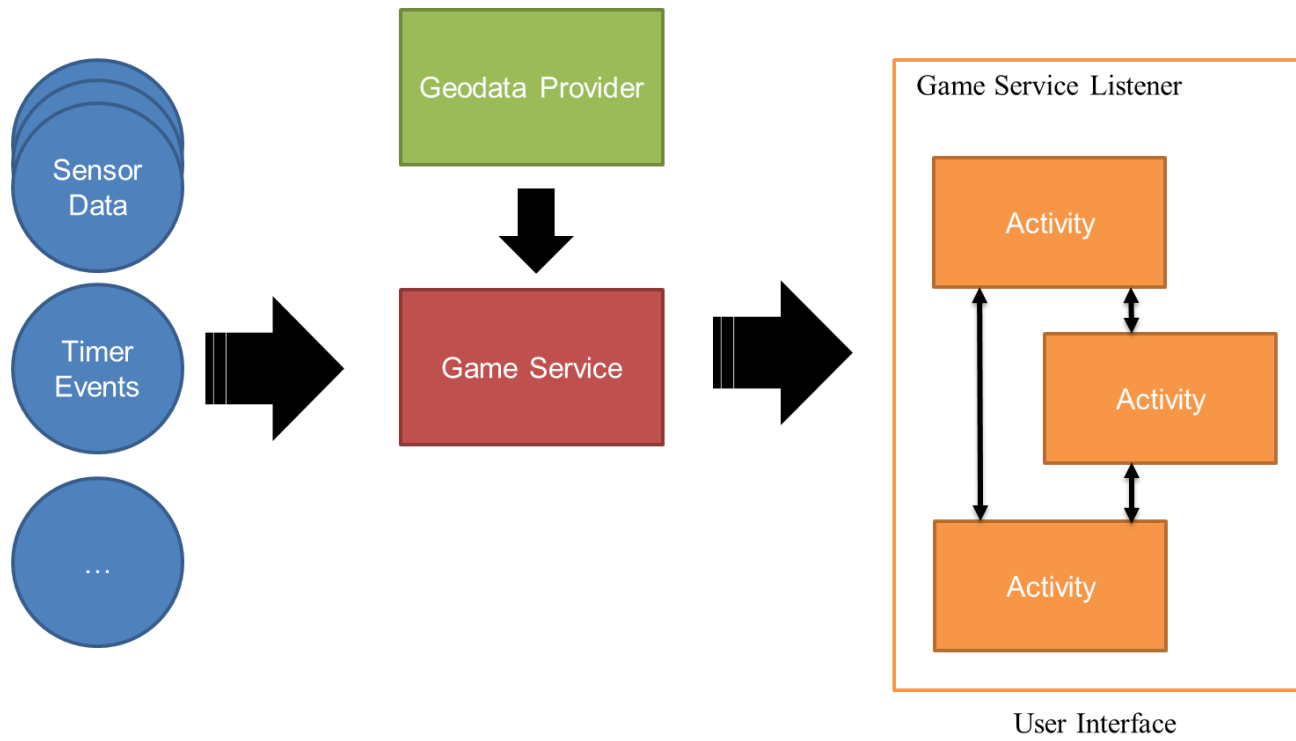
■ Intents

- Messages passed between applications and the Android environment
- Your applications should listen out for relevant events and respond appropriately
- Could be answering a phone call, responding to changes in application state or reacting to an GPS update

■ Services

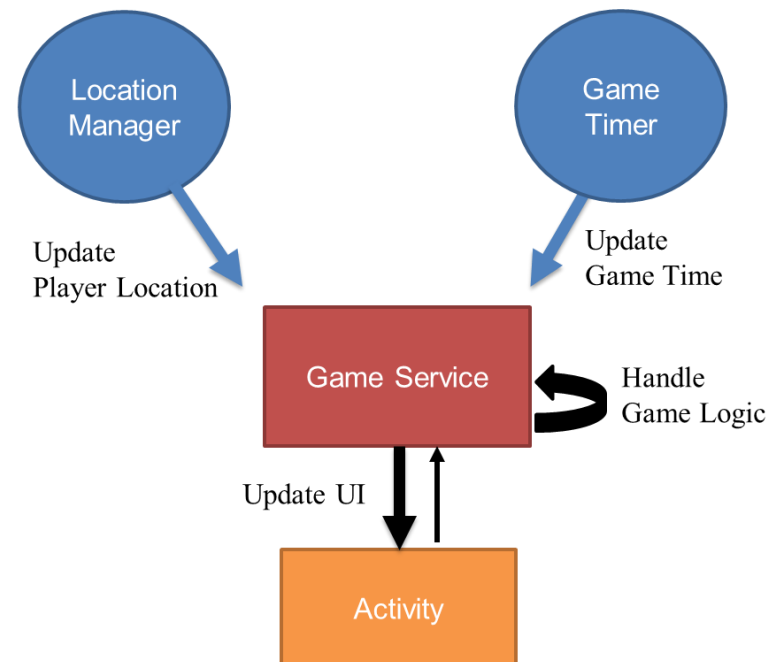
- Background applications that provide features, abilities, or actions on which your applications (and their activities) can rely
- Often long-lived and generally run without a visible user interface

Geogame Architecture



Geogame Architecture

1. Main menu activity is started
2. Background game service is initialized
3. Player selects game configuration
4. Game UI activity is shown
5. Game service listens for events and updates the currently active UI accordingly





Building a Native Android App

Native vs Web Applications

■ Native:

- Target each specific device OS

■ Hybrid

- Target multiple mobile OS, but rely on existing web content

■ Web Apps

- Target the device's browser

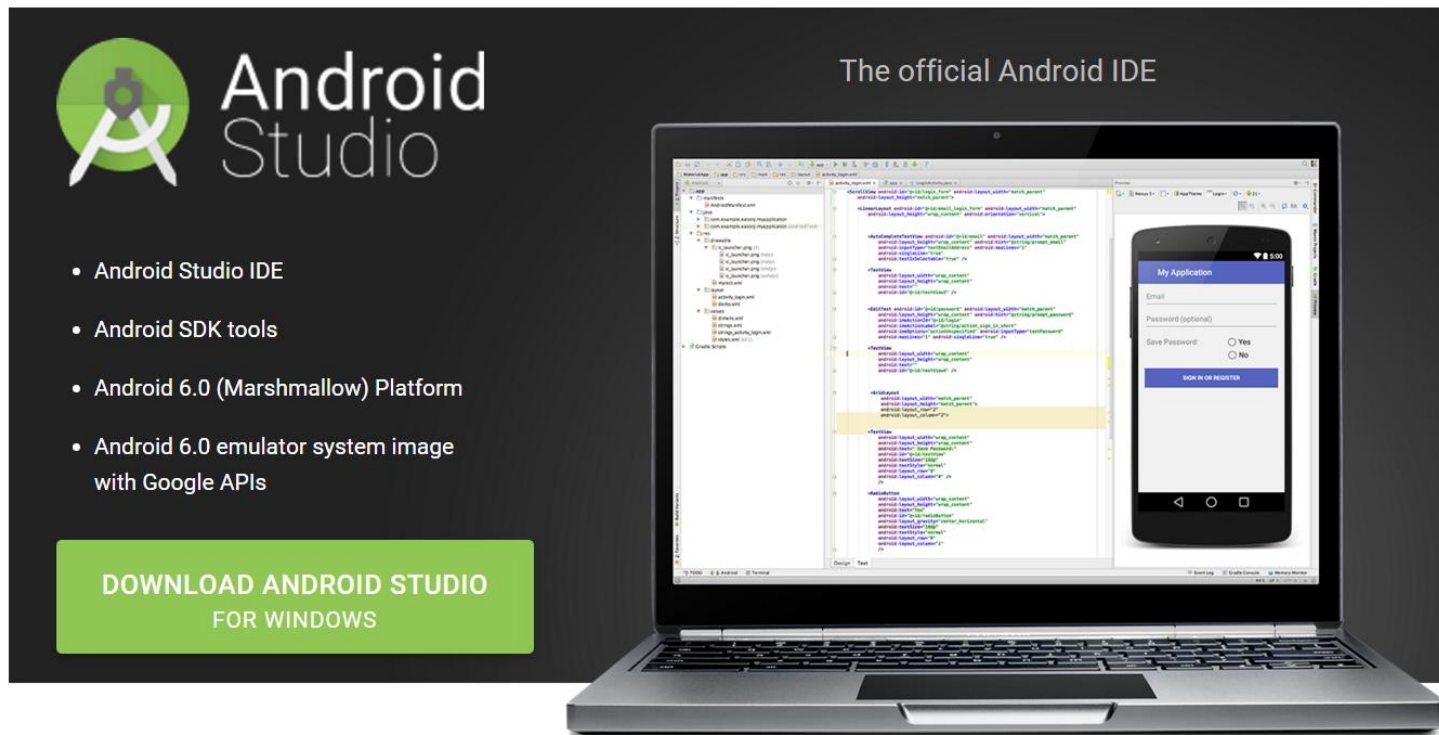
■ Pro native development:

- Usually the best user experience
- Access to all device utilities
 - GPS, compass, calendar, media, contacts, camera, ...
- Distribution through a store is supported
- Powerful offline libraries (e.g. GIS functionality)

Android

- Initial release: September 23, 2008
- Current version: 6.0.1 "Marshmallow" December 9, 2015
- Latest reports mention it has an 84.7% market share (as of Q3 2015) on the smartphone OS market
- Runs on phones, wrist watches, televisions

Android Development with Android Studio



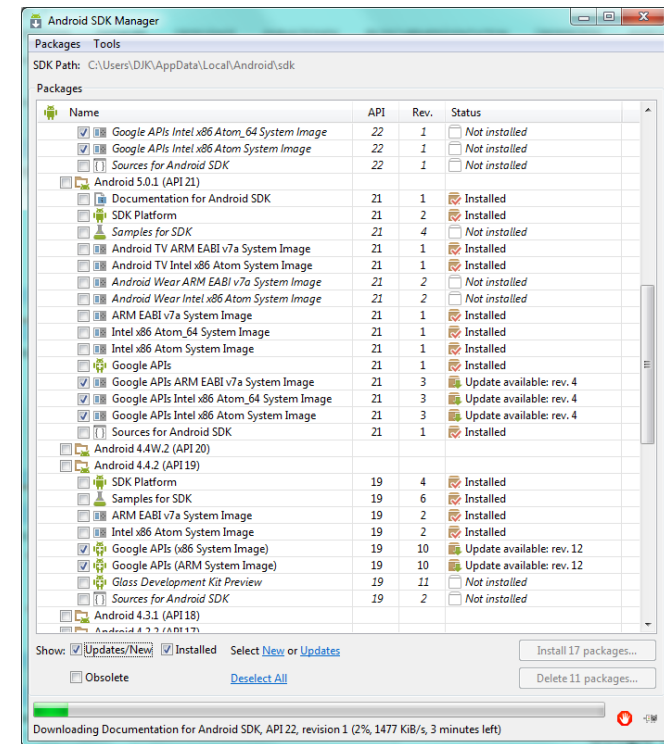
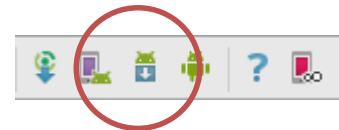
The official Android IDE

- Android Studio IDE
- Android SDK tools
- Android 6.0 (Marshmallow) Platform
- Android 6.0 emulator system image with Google APIs

DOWNLOAD ANDROID STUDIO FOR WINDOWS

SDK Manager

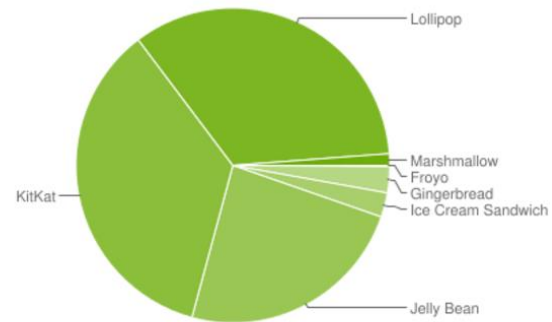
- Tracks and manages all of the components of various Android SDKs that are available
- By default, it installs the latest packages and tools
- Click the checkbox next to each additional SDK



Should be installed:

- For this workshop you should have installed from the Android 4.0 or newer section
 - SDK Platform
 - Sources for the Android SDK
 - Documentation for Android SDK
 - The ARM EABI System Image
- From the Extras section:
 - Android Support Library
 - Google USB Driver

Platform Versions



- Android SDKs are backward compatible
- Newer versions bring more features
- Minimal and targeted SDK versions can be configured to reduce testing needs or enable newer features

Version	Codename	API	Distribution
2.2	Froyo	8	0.1%
2.3.3 - 2.3.7	Gingerbread	10	2.7%
4.0.3 - 4.0.4	Ice Cream Sandwich	15	2.5%
4.1.x	Jelly Bean	16	8.8%
4.2.x		17	11.7%
4.3		18	3.4%
4.4	KitKat	19	35.5%
5.0	Lollipop	21	17.0%
5.1		22	17.1%
6.0	Marshmallow	23	1.2%

Data collected during a 7-day period ending on February 1, 2016.

Development on ...

■ a physical Device

- Developer mode has to be enabled:
- Open *Settings* > *About* > *Software Information* > *More*.
- Tap “*Build number*” seven times to enable Developer options
- Go back to Settings menu and now you'll be able to see “Developer options” there.
- Tap it and turn on *USB Debugging* from the menu on the next screen.

■ on an Emulator

- Included in Android Studio



■ Alternative VM: Genymotion

- Better performance
- Requires registration
- Google Services not available
- <https://www.genymotion.com/>
<https://www.genymotion.com/#!/developers/user-guide>

Creating a Virtual Android Device



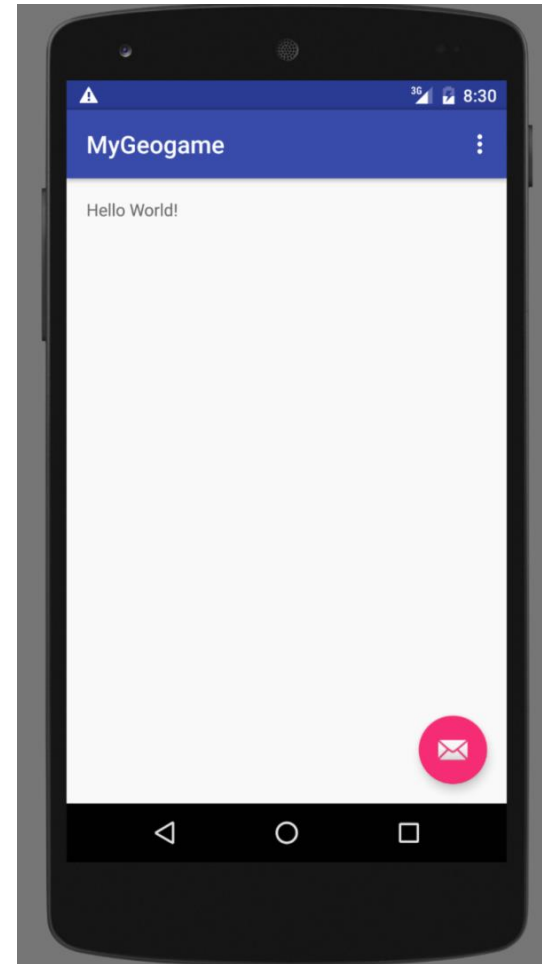
- Open the AVD Manager
 - Click „*Create Virtual Device*“
 - Choose a „Phone“ device from the list
 - Resolution should be lower then your desktop resolution

Nexus 5	4,0"	480x800	hdpi
Nexus One	3,7"	480x800	hdpi
Nexus 6P	5,7"	1440x2560	560dpi
Nexus 6	5,96"	1440x2560	560dpi
Nexus 5X	5,2"	1080x1920	420dpi
Nexus 5	4,95"	1080x1920	xxhdpi
Nexus 4	4,7"	768x1280	xhdpi
Galaxy Nexus	4,65"	720x1280	xhdpi
5.4" FWVGA	5,4"	480x854	mdpi
5.1" WVGA	5,1"	480x800	mdpi
4.7" WXGA	4,7"	720x1280	xhdpi

Hello World!

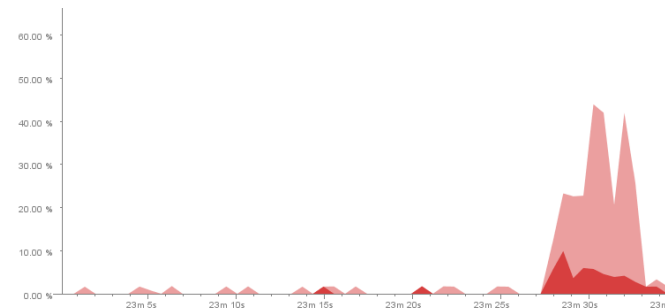
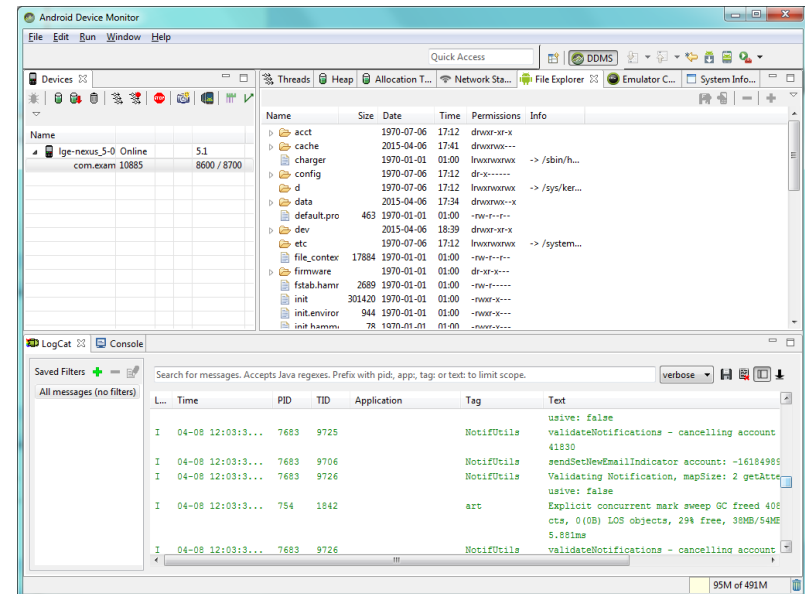
■ Tasks:

- Create a new Android-Project (File > New > New Project)
- Create a run configuration for this project
- Launch your application without any code changes on an emulated or a physical device



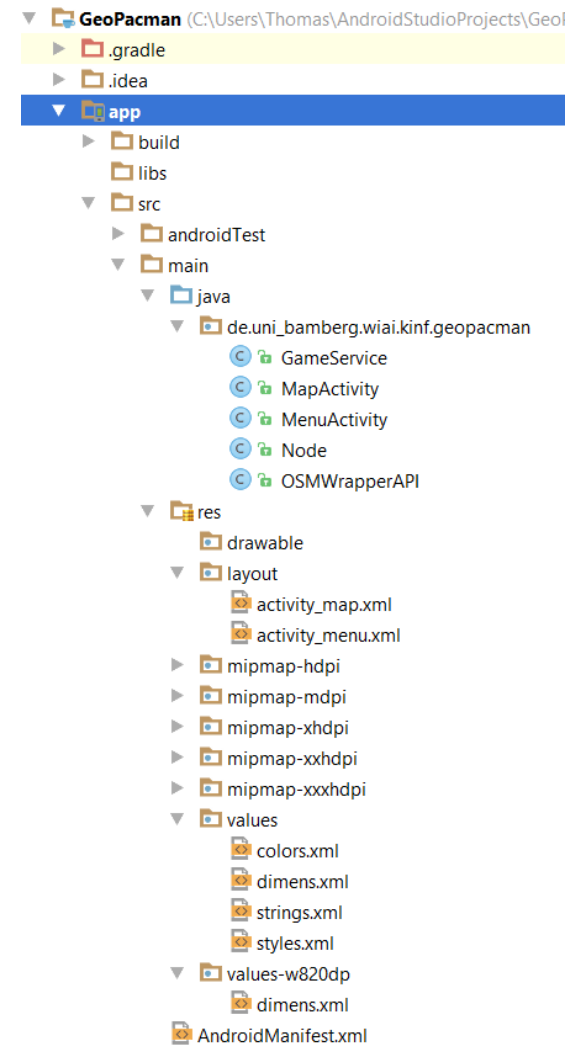
Debugging

- Logging, LogCat
 - Use android.util.Log to
 - LogCat displays Log-Messages
- Android Device Monitor
 - Displays other usefull debugging information like memory, cpu and network usage



Project Structure

- *src Folder*
 - Contains the java source code files
- *res Folder*
 - Mostly XML configuration files
 - E.G.: layout, strings
- *R.java class*
 - Automatically generated
 - All resource IDs are defined in this class
 - Provides access to resources
 - `String helloString = getString(R.string.hello);`
 - `setContentView(R.activity_main);`



Gradle

- Open source build automation system
- Uses a Groovy-based domain-specific language
- Designed for multi-project builds which can grow to be quite large
- Android Studio projects contain a top-level build file and a build file for each module, both called *build.gradle*
- Android-specific build options as well as app configurations and dependencies can be adjusted



```
android {  
    compileSdkVersion 23  
    buildToolsVersion "23.0.2"  
  
    defaultConfig {  
        applicationId "de.geopacman"  
        minSdkVersion 15  
        targetSdkVersion 23  
        versionCode 1  
        versionName "1.0"  
    }  
}
```



Displaying a Map

Map Providers

■ OSMdroid

- Open source
- Requires SLF4J (Logging)
- Multitouch support
- Multiple basemaps available
- Not very well documented

<https://github.com/osmdroid/osmdroid/wiki/How-to-use-the-osmdroid-library>

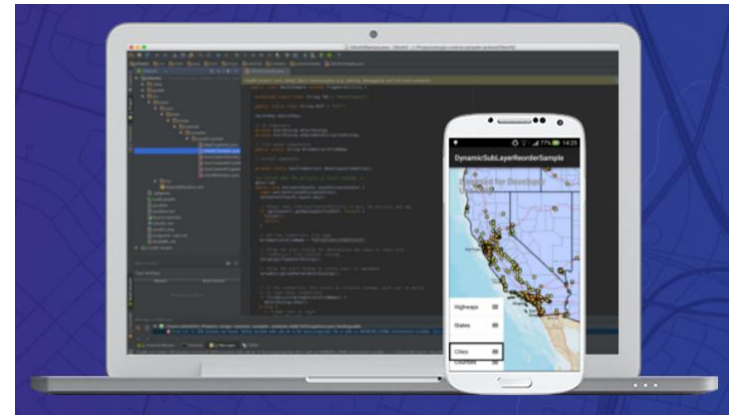
■ Google Maps

- Cumbersome registration required
- Requires Google PlayS ervices dependency
- Can be configured easily
- Supports multitouch + rotation
- Extensive documentation and code samples

<https://developers.google.com/maps/documentation/android/>

ArcGIS Runtime SDK for Android

- Provides mapping and GIS capabilities to your Android apps
- Works offline with basemaps and operational data support
- Query content from ArcGis Online
- Online and offline routing applications
- Perform advanced geometric and spatial analysis
-



- Consists of a bunch jar files providing a rich mapping
- Extensive API Reference and Guides and available online
 - <https://developers.arcgis.com/android/api-reference/>
 - <https://developers.arcgis.com/android/guide/>

Installing the ArcGIS Runtime SDK

■ Task: Install the SDK

- Follow the instructions:
<https://developers.arcgis.com/android/>
- Adjust your Android Studio Gradle files
- Add the ArcGIS Android Bintray maven repository to your projects *build.gradle* file
- Add *arcgis-android* as a dependency to the app
- Set the *packaging-exclude* options
- Let Android Studio sync your changes

The project *build.gradle* file:

```
allprojects {
    repositories {
        jcenter()
        // Added Arcis maven repository
        maven {
            url 'http://esri.bintray.com/arcgis'
        }
    }
}
```

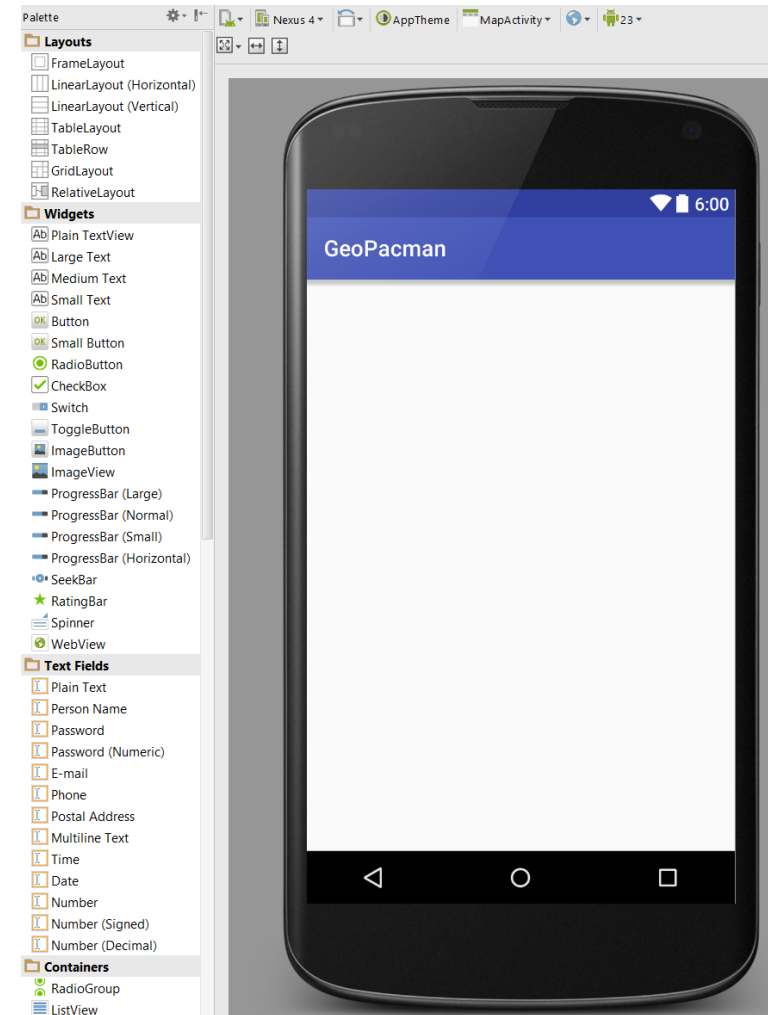
The apps *build.gradle* file:

```
dependencies {
    compile fileTree(dir: 'libs', include: ['*.jar'])
    testCompile 'junit:junit:4.12'
    compile 'com.android.support:appcompat-v7:23.1.1'
    // Added Arcis dependency
    compile 'com.esri.arcgis.android:arcgis-android:10.2.7'
}

defaultConfig {
    // ...
    // Added ArcGIS Android exclusion settings
    packagingOptions {
        exclude 'META-INF/LGPL2.1'
        exclude 'META-INF/LICENSE'
        exclude 'META-INF/NOTICE'
    }
}
```

Android Views/Widgets

- Basic building block for user interface components
- *Widgets* are used to create interactive UI components
 - Buttons, Text Fields, Containers, ...
- Arranged in a single tree



UI Layouting

- Layout defines the visual structure for a user interface
- Can declare a layout in two ways
 - Declare UI elements in XML
 - Instantiate layout elements at runtime

```
<?xml version="1.0" encoding="utf-8" ?>
<android.support.design.widget.CoordinatorLayout xmlns:android="http://schemas.android.com/apk/res-auto"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:fitsSystemWindows="true"
    tools:context="de.uni_bamberg.wiai.kinf.mygeogame.MapActivity"

    <android.support.design.widget.AppBarLayout
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:theme="@style/AppTheme.AppBarOverlay">

        <android.support.v7.widget.Toolbar
            android:id="@+id/toolbar"
            android:layout_width="match_parent"
            android:layout_height="?attr/actionBarSize"
            android:background="?attr/colorPrimary"
            app:popupTheme="@style/AppTheme.PopupOverlay" />

    </android.support.design.widget.AppBarLayout>

    <com.esri.android.map.MapView
        android:id="@+id/map"
        android:layout_width="fill_parent"
        android:layout_height="fill_parent"
        mapoptions.MapType="Streets"
        mapoptions.center="34.056215, -117.195668"
        mapoptions.ZoomLevel="16">

    </com.esri.android.map.MapView>

</android.support.design.widget.CoordinatorLayout>
```

Adding Widgets Programmatically

```
//Add a LinearLayout
LinearLayout layout = (LinearLayout) findViewById(R.id.LinearLayout);
LinearLayout.LayoutParams params = new LinearLayout.LayoutParams(
    LinearLayout.LayoutParams.WRAP_CONTENT,
    LinearLayout.LayoutParams.WRAP_CONTENT);
layout.setOrientation(LinearLayout.VERTICAL);

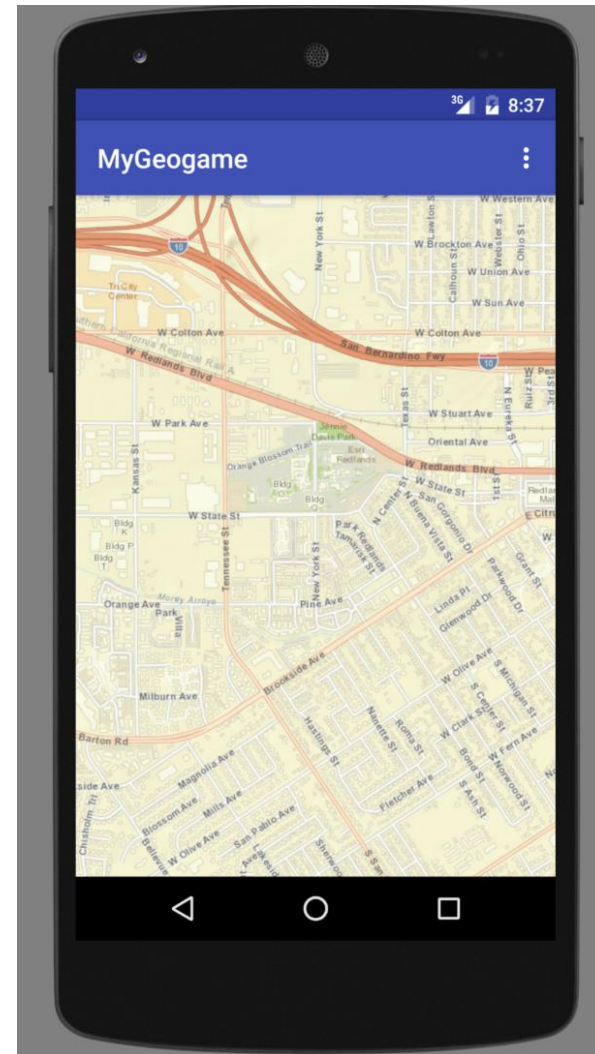
// Add a button
Button button = new Button(this);
button.setText("Ok");

Button startGameBtn = (Button) findViewById(R.id.start_btn);
startGameBtn.setOnClickListener((v) -> { startGame(v); });
```

Displaying a Map

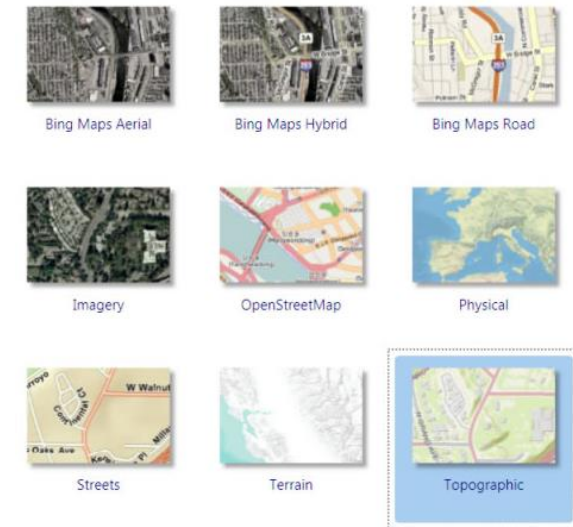
- **Task:** Embed a Esri MapView widget into your app
 - Open the activity_map.xml
 - Replace the text label with the following code:

```
<com.esri.android.map.MapView
    android:id="@+id/map"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent"
    mapoptions.MapType="Satellite"
    mapoptions.ZoomLevel="16">
</com.esri.android.map.MapView>
```



Adjusting the Map

- **Task:** Adjust the map (activity) to your liking by changing the activity layout
 - Center the map on a location nearby
 - Zoom in closely
 - Use another basemap
 - Remove the gaps around the map



ArcGis Basemap Options

- **API-References:**
 - <https://developers.arcgis.com/android/api-reference/reference/com/esri/android/map/MapView.html>
 - <https://developers.arcgis.com/android/api-reference/reference/com/esri/android/map/MapOptions.html>



Adding a Marker

Adding Graphics to the Map

- A map is a canvas that draws layers of geographic data
- Different layer types are used to draw different data types
- A basemap layer is already present on our map
- Graphics layers allow you to dynamically display graphics on a map

- To add a marker graphic to the map, we need to

- Inflate the MapView

```
mapView = (MapView) findViewById(R.id.map);
```

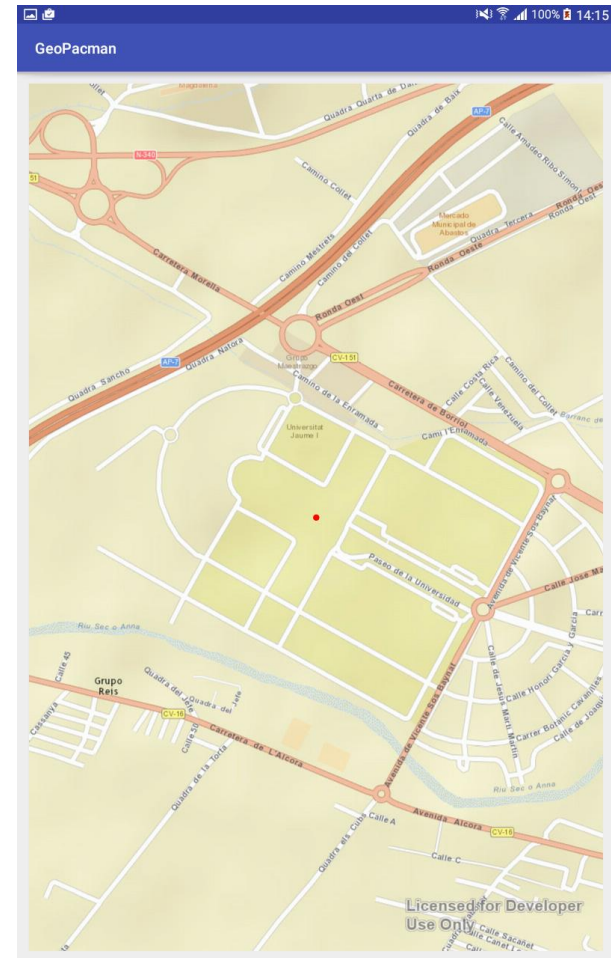
- Add a graphics layer and add it to the map

```
gLayer = new GraphicsLayer();  
mapView.addLayer(gLayer);
```

- Create and add a graphic

Adding a Marker

- Task: Add to the onCreate method of your activity
 - Look up coordinates nearby
 - Create a point geometry for these coordinates
 - Create a symbol for the marker
 - E.g. A *SimpleMarkerSymbol*
 - Create a graphic for the symbol and add it to the map
 - `Graphic pointGraphic = new Graphic(pointGeometry, simpleMarker);`
 - `graphicsLayer.addGraphic(pointGraphic);`



Did it work?

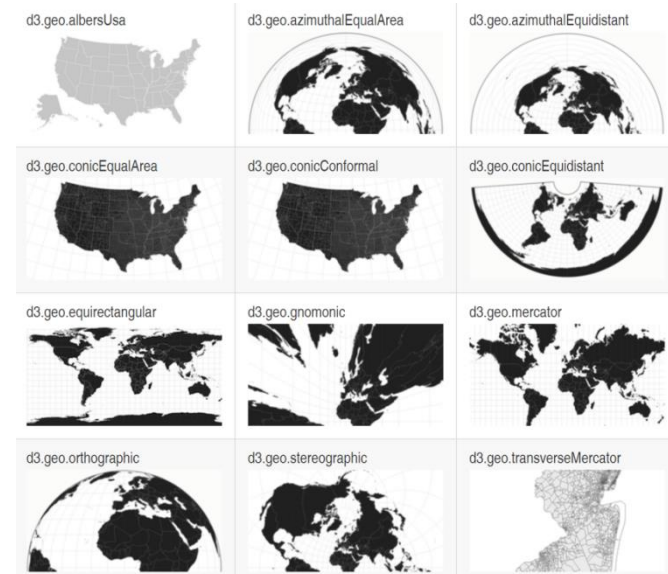
```
public class MapActivity extends AppCompatActivity {  
    private MapView mapView;  
    private GraphicsLayer gLayer;  
  
    @Override  
    protected void onCreate(Bundle savedInstanceState) {  
        super.onCreate(savedInstanceState);  
        setContentView(R.layout.activity_map);  
  
        // Inflate Mapview from XML  
        mapView = (MapView) findViewById(R.id.map);  
  
        // Create a graphics layer and add it to the map  
        gLayer = new GraphicsLayer();  
        mapView.addLayer(gLayer);  
  
        // Create a point geometry  
        Point pointGeometry = new Point(-0.069336, 39.994760);  
  
        // Project it to the maps spatial reference system  
        pointGeometry = (Point) GeometryEngine.project(pointGeometry,  
            SpatialReference.create(SpatialReference.WKID_WGS84),  
            mapView.getSpatialReference());  
  
        // Add a graphic to the graphics layer  
        SimpleMarkerSymbol simpleMarker = new SimpleMarkerSymbol(Color.RED, 10,  
            SimpleMarkerSymbol.STYLE.CIRCLE);  
        Graphic pointGraphic = new Graphic(pointGeometry, simpleMarker);  
        gLayer.addGraphic(pointGraphic);  
    }  
}
```



Geographic Projections

- Coordinates are associated with a coordinate system, which is a frame of reference around a model of the earth's surface
- Not all coordinates and their associated coordinate systems are the same; they can be in different units or they can be based on different types of models.
- To move coordinates from one coordinate system to another, mathematical transformations are used

- Esri MapView uses *WGS_1984_Web_Mercator* as a default



<https://github.com/mbostock/d3/wiki/Geo-Projections>

Event Listener

- When instantiating the *MapView*, you cannot assume that the object is initialized immediately
- This is due to the normal life cycle of an Activity
- Things go wrong if you attempt to get work with the map immediately after you call the constructor
- The proper way is to set a *OnStatusChangeListener*
- Be careful: When initializing a *MapView*, the *ChangeListener* notifies you of the status changes from both the *MapView* and Layers

Adding a Marker the Proper Way

```

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

    // Inflate Mapview from XML
    mapView = (MapView) findViewById(R.id.map);

    // Create a graphics layer and add it to the map
    gLayer = new GraphicsLayer();
    mapView.addLayer(gLayer);

    mapView.setOnStatusChangeListener(new OnStatusChangeListener() {
        @Override
        public void onStatusChanged(Object source, STATUS status) {
            if (OnStatusChangeListener.STATUS.INITIALIZED == status &&
                source == mapView && !mapInitialized) {
                mapInitialized = true;

                Point pointGeo = new Point(-0.0726841, 39.989929);
                pointGeo = (Point) GeometryEngine.project(pointGeo,
                    SpatialReference.create(SpatialReference.WKID_WGS84),
                    mapView.getSpatialReference());
                SimpleMarkerSymbol pointSymbol = new SimpleMarkerSymbol(Color.RED, 8,
                    SimpleMarkerSymbol.STYLE.CIRCLE);
                Graphic from = new Graphic(pointGeo, pointSymbol);
            }
        }
    });
}

```





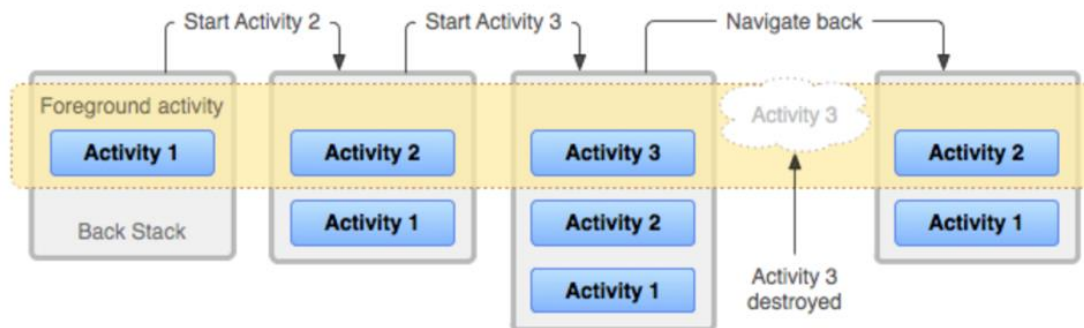
Adding a Second Activity

Intents

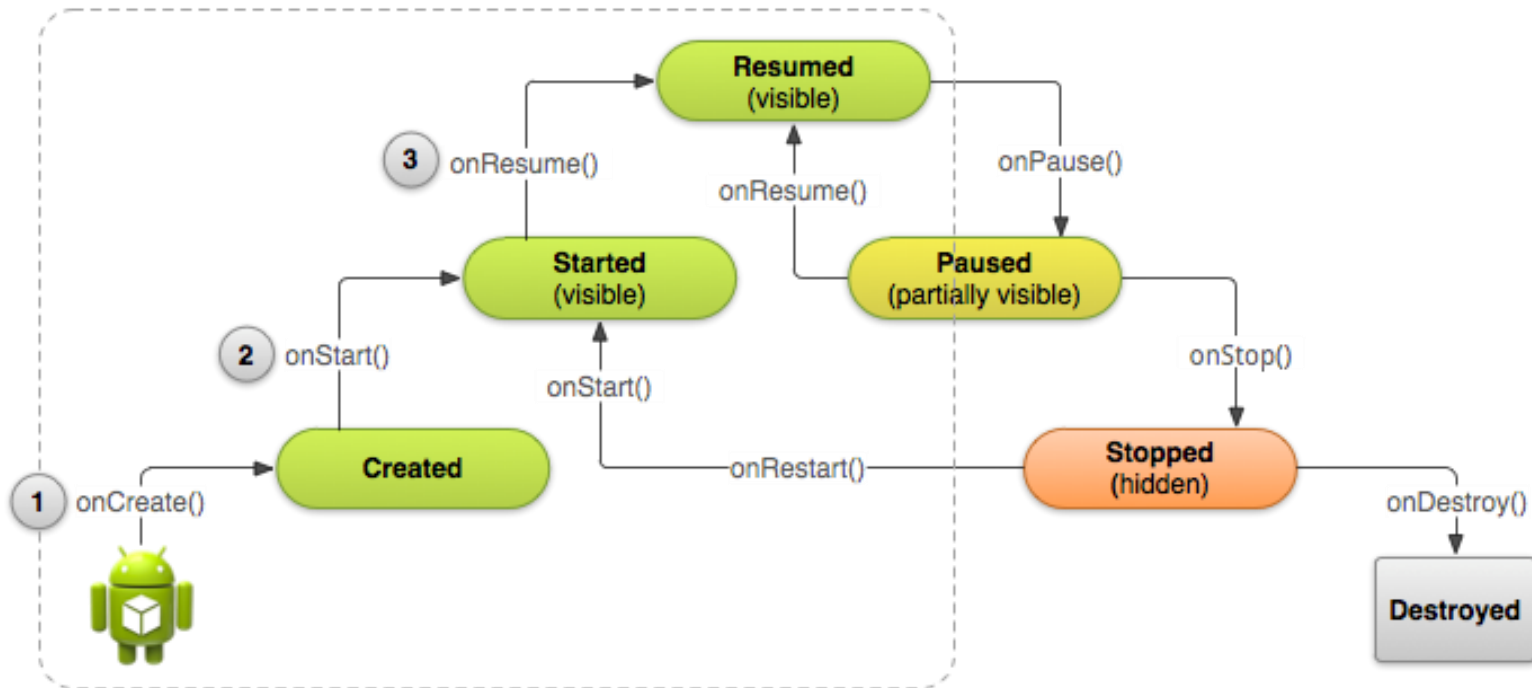
- An intent is a messaging object you can use to request an action from another app component
- Three fundamental use-cases
 - Start an activity
 - Start a service
 - Deliver a broadcast
- Two types of intents
 - **Explicit** intents specify the component to start by name
 - **Implicit** intents do not name a specific component, but instead declare a general action to perform

The Activity Stack

- Applications usually contain multiple activities
- Activity should be designed around a specific kind of action the user can perform and can start other activities
- When the current activity starts another, the new activity is pushed on the top of the stack and takes focus

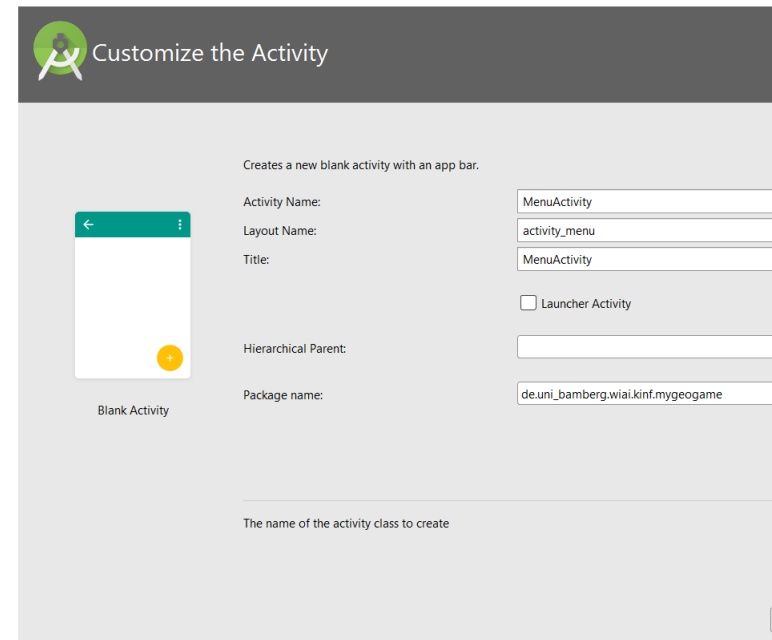


The Activity Lifecycle



Creating a MainMenu

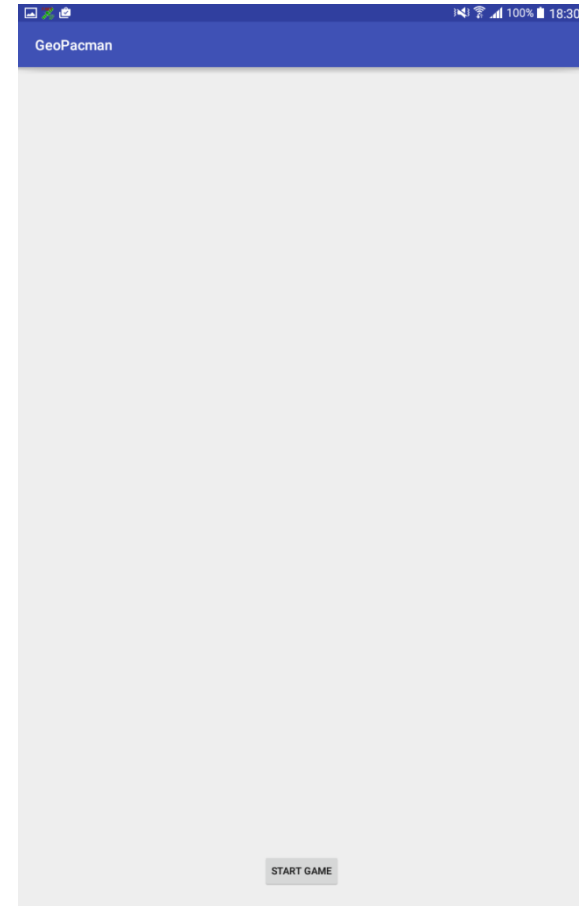
- Task: Create a second activity
 - Will act as a main menu
 - Should at least contain a “Start Game” Button
 - Pressing the button should switch to the map activity



A Simplistic Menu Activity

```
public class MenuActivity extends AppCompatActivity {  
  
    @Override  
    protected void onCreate(Bundle savedInstanceState) {  
        super.onCreate(savedInstanceState);  
        setContentView(R.layout.activity_menu);  
    }  
}
```

```
<?xml version="1.0" encoding="utf-8"?>  
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"  
    android:layout_width="match_parent"  
    android:layout_height="match_parent">  
  
    <Button  
        android:layout_width="wrap_content"  
        android:layout_height="wrap_content"  
        android:text="Start Game"  
        android:id="@+id/start_btn"  
        android:layout_alignParentBottom="true"  
        android:layout_centerHorizontal="true"  
        android:layout_marginBottom="18dp" />  
</RelativeLayout>
```



http://www.geogames-team.org/files/uij/solutions/2_menu/

Launcher Activities

- The main activity for your app must be declared in the manifest with an `<intent-filter>` that includes the *MAIN action* and *LAUNCHER category*
- Multiple launcher activities are allowed
- If either the MAIN action or LAUNCHER category are not declared for one of your activities, then your app icon will not appear in the Home screen

```
<application
    android:allowBackup="true"
    android:icon="@mipmap/ic_launcher"
    android:label="GeoPacman"
    android:supportsRtl="true"
    android:theme="@style/AppTheme">

    <activity android:name=".MapActivity">
        <!--Removed the Launcher category-->
    </activity>

    <activity android:name=".MenuActivity">
        <!--Added the Launcher category-->
        <intent-filter>
            <action android:name="android.intent.action.MAIN" />
            <category android:name="android.intent.category.LAUNCHER" />
        </intent-filter>
    </activity>

    <service android:name=".GameService" />

</application>
```

Switching between Activities

```
public class MenuActivity extends AppCompatActivity {

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

        Button startGameBtn = (Button) findViewById(R.id.start_btn);
        startGameBtn.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                startGame();
            }
        });
    }

    private void startGame() {
        Intent gameIntent = new Intent(getApplicationContext(), MapActivity.class);
        startActivity(gameIntent);
    }
}
```

```
<?xml version="1.0" encoding="utf-8" ?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="de.uni_bamberg.wiai.kinf.geopacman">

    <!-- Added permission to use the internet -->
    <uses-permission android:name="android.permission.INTERNET" />

    <application
        android:allowBackup="true"
        android:icon="@mipmap/ic_launcher"
        android:label="@string/app_name"
        android:supportsRtl="true"
        android:theme="@style/AppTheme">

        <activity android:name=".MapActivity">
            <!--Removed the Launcher category-->
        </activity>

        <activity android:name=".MenuActivity">
            <!--Added the Launcher category-->
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />
                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>

    </application>
</manifest>
```


Switching between Activities (2)

```
public class MenuActivity extends AppCompatActivity {  
  
    @Override  
    protected void onCreate(Bundle savedInstanceState) {  
        super.onCreate(savedInstanceState);  
        setContentView(R.layout.activity_menu);  
  
        // Button startGameBtn = (Button) findViewById(R.id.start_btn);  
        // startGameBtn.setOnClickListener(new View.OnClickListener() {  
        //     @Override  
        //     public void onClick(View v) {  
        //         startGame();  
        //     }  
        // });  
    }  
  
    public void startGame(View v) {  
        Intent gameIntent = new Intent(getApplicationContext(), MapActivity.class);  
        startActivity(gameIntent);  
    }  
}
```

```
<?xml version="1.0" encoding="utf-8"?>  
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"  
    android:layout_width="match_parent"  
    android:layout_height="match_parent">  
  
    <Button  
        android:layout_width="wrap_content"  
        android:layout_height="wrap_content"  
        android:text="Start Game"  
        android:id="@+id/start_btn"  
        android:layout_alignParentBottom="true"  
        android:layout_centerHorizontal="true"  
        android:layout_marginBottom="18dp"  
        android:onClick="startGame"/>  
</RelativeLayout>
```



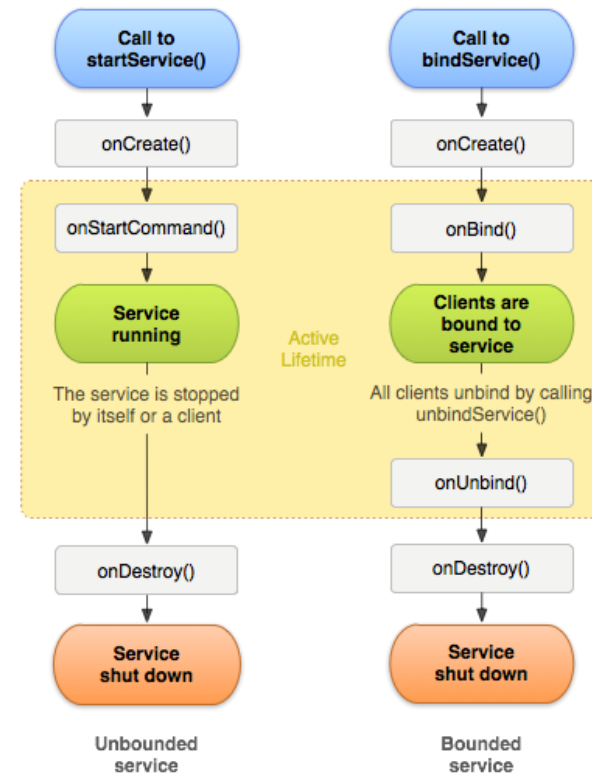
Adding a Service

Benefits of Using a Service

- Switching to another app will not pause the game logic loop
- Closing the activity will not stop the game (Only remote service)
- Game logic is separated from the user interface
- A service can communicate with activities
- Local Service:
 - Runs in the same process as the activity
 - Killing this process will also kill the activity
- Remote Service:
 - The service runs independently from the activity

Service Interaction

- Start a service (e.g. in a activity)
 - `startService(intent)`
- Stopp Service
 - `stopService(intent)`
- `startService` calls implicitly `onStartCommand`
- For interaction with an activity a service needs to be bound



<http://developer.android.com/guide/components/services.html>

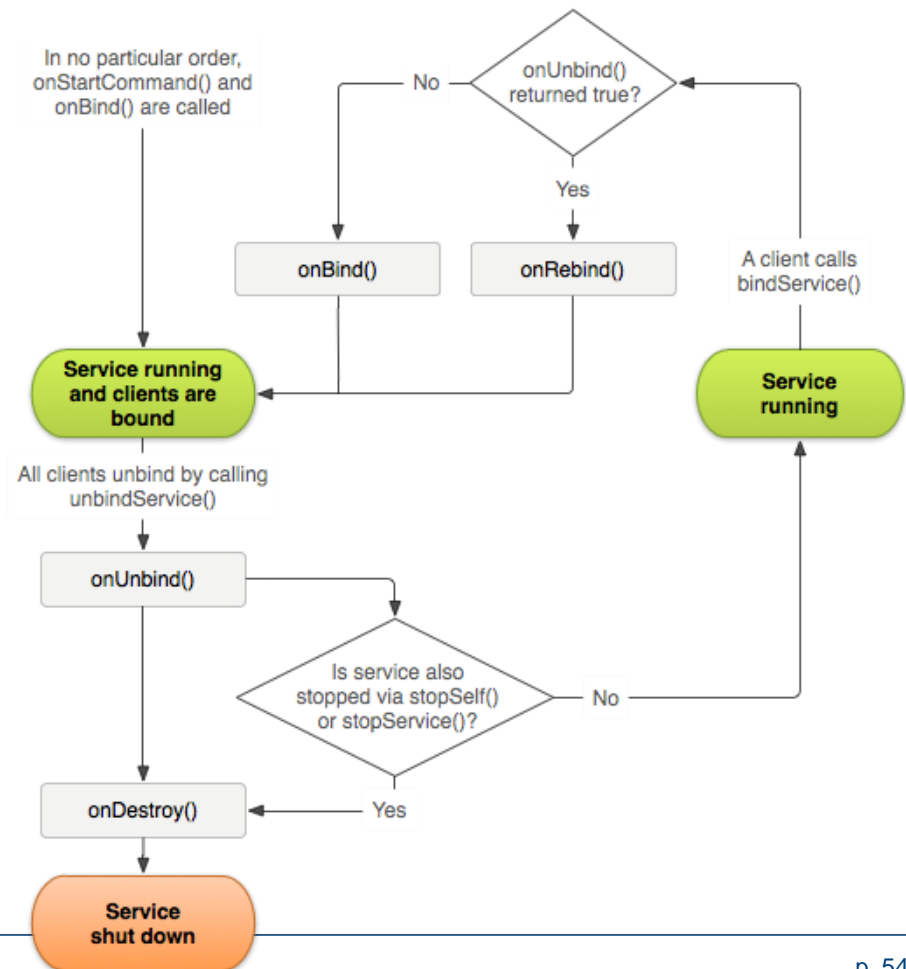
Binding a Service

■ Bind

- Binds a service
- Returns a Binder-Interface for interaction
- Activity defines Callback as a ServiceConnection-Object

■ Unbind

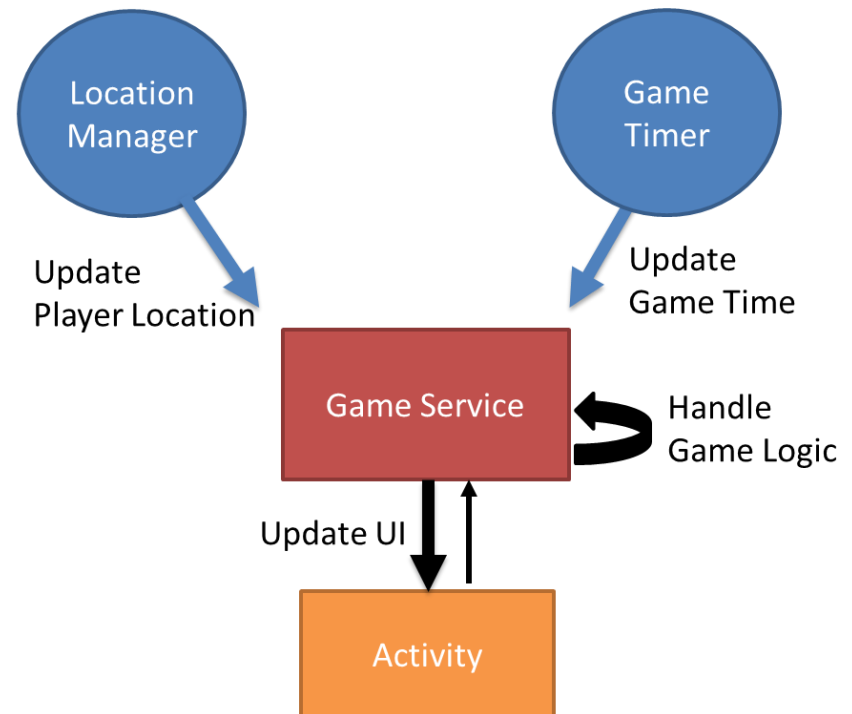
- Terminates the connection
- Service continues to run and can be rebound



Implementing a Game Service

■ Task:

- Add a game service to your project
- Files can be found at:
<http://www.geogames-team.org/files/uji/files/service/>
- Start the service when the “*Start Game*” Button is pressed



Location Manager

- A system service
- Able to obtain periodic updates of the device's geographical location
- Requires the `ACCESS_COARSE_LOCATION` or `ACCESS_FINE_LOCATION` permissions.
- If your application only has the coarse permission then it will not have access to the GPS or passive location providers

```
LocationManager locService = (LocationManager) getSystemService(Context.LOCATION_SERVICE);
LocationListener locListener = new LocationListener() {
    @Override
    public void onLocationChanged(Location location) {
    }

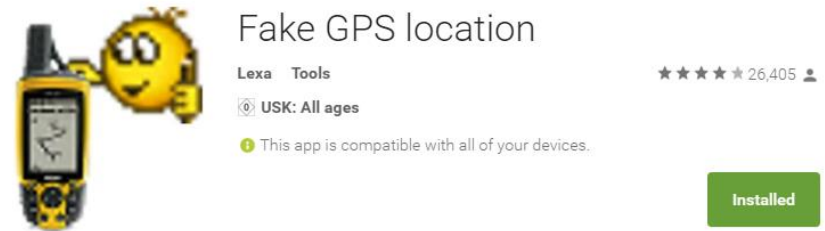
    @Override
    public void onStatusChanged(String provider, int status, Bundle extras) {}

    @Override
    public void onProviderEnabled(String provider) {}

    @Override
    public void onProviderDisabled(String provider) {}
};
if (ActivityCompat.checkSelfPermission(this,
    Manifest.permission.ACCESS_FINE_LOCATION) != PackageManager.PERMISSION_GRANTED &&
    ActivityCompat.checkSelfPermission(this,
    Manifest.permission.ACCESS_COARSE_LOCATION) != PackageManager.PERMISSION_GRANTED) {
    return;
}
locService.requestLocationUpdates(LocationManager.GPS_PROVIDER,
    MINIMUM_TIME_BETWEEN_UPDATE, MINIMUM_DISTANCECHANGE_FOR_UPDATE, locListener);
```

Simulating GPS

- On the device
 - With the help of third party apps (e.g. Fake GPS)
- With the Device Monitor
 - DDMS -> Emulator Control -> Location Controls
- Mock locations have to be enabled in the developer options

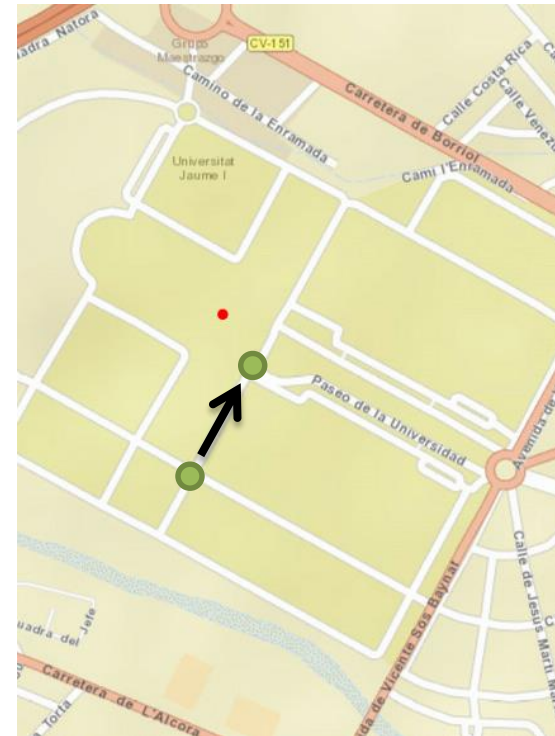


<https://play.google.com/store/apps/details?id=com.lexa.fakegps&hl=en>

Update Marker on Location Change

■ Tasks:

- Add a *location manager listener* to your *game service*
- On location changes, notify *GameService Listeners*
- Update the marker position on the map activity



Updating the Player Position

- Get the location manager service
- Create a location listener
- Updating the player position for all listeners

```
// ===== Event Handling =====

private void initLocationService() {
    LocationManager locService = (LocationManager) getSystemService(Context.LOCATION_SERVICE);
    LocationListener locListener = new LocationListener() {
        @Override
        public void onLocationChanged(Location location) {
            playerPosition = location;
            // Alert listeners about changed player position
            for (GameServiceListener listener : listeners) {
                listener.updatePlayerPosition(location);
            }
        }

        @Override
        public void onStatusChanged(String provider, int status, Bundle extras) {}

        @Override
        public void onProviderEnabled(String provider) {}

        @Override
        public void onProviderDisabled(String provider) {}
    };
    if (ActivityCompat.checkSelfPermission(this,
        Manifest.permission.ACCESS_FINE_LOCATION) != PackageManager.PERMISSION_GRANTED &&
        ActivityCompat.checkSelfPermission(this,
            Manifest.permission.ACCESS_COARSE_LOCATION) != PackageManager.PERMISSION_GRANTED) {
        return;
    }
    locService.requestLocationUpdates(LocationManager.GPS_PROVIDER,
        MINIMUM_TIME_BETWEEN_UPDATE, MINIMUM_DISTANCECHANGE_FOR_UPDATE, locListener);
}
```

Updating the Map

- The player graphic has to be removed before being added again

```
@Override
public void updatePlayerPosition(Location location) {
    if(playerPosGraphicId != -1) {
        gLayer.removeGraphic(playerPosGraphicId);
    }
    if(location != null) {
        final Point p = new Point(location.getLongitude(), location.getLatitude());
        final Symbol symbol = new SimpleMarkerSymbol(Color.GREEN, 15,
            SimpleMarkerSymbol.STYLE.CIRCLE);
        final Point pgeo = (Point) GeometryEngine.project(p, SpatialReference.create(4326),
            mMapView.getSpatialReference());
        Graphic pointGraphic = new Graphic(pgeo, symbol);
        playerPosGraphicId = gLayer.addGraphic(pointGraphic);
    }
}
```

Timer & Score

■ Tasks:

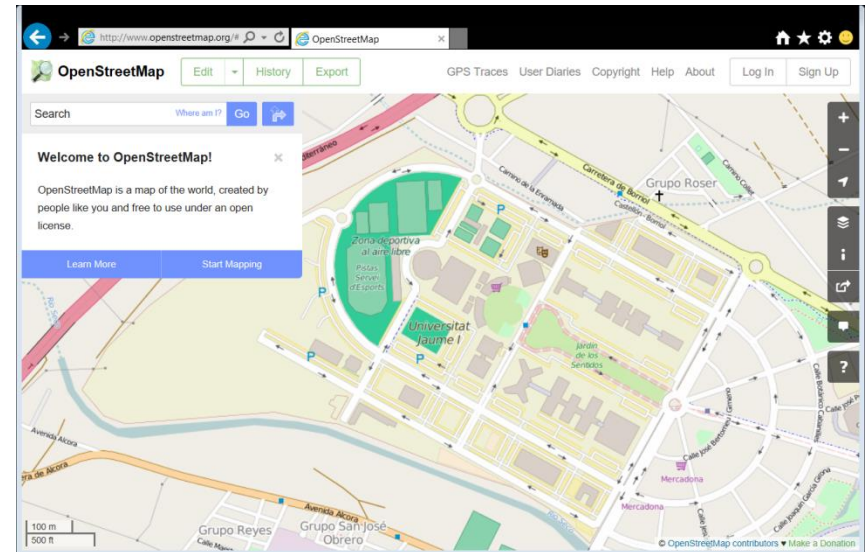
- Add a timer in the service that keeps track of the passed game time
- Add a player-score to the service
- Add interface elements to the map activity to visualize the game time and the player score



Querying and working with public geographic data

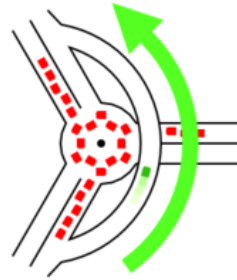
OSM

- OpenStreetMap (OSM) is a collaborative project to create a free editable map of the world
- Prominent example of volunteered geographic information
- Rather than the map itself, the data generated by the are considered its primary output
- Has an Editing API for fetching and saving raw geodata from/to the OpenStreetMap database



<http://www.openstreetmap.org>

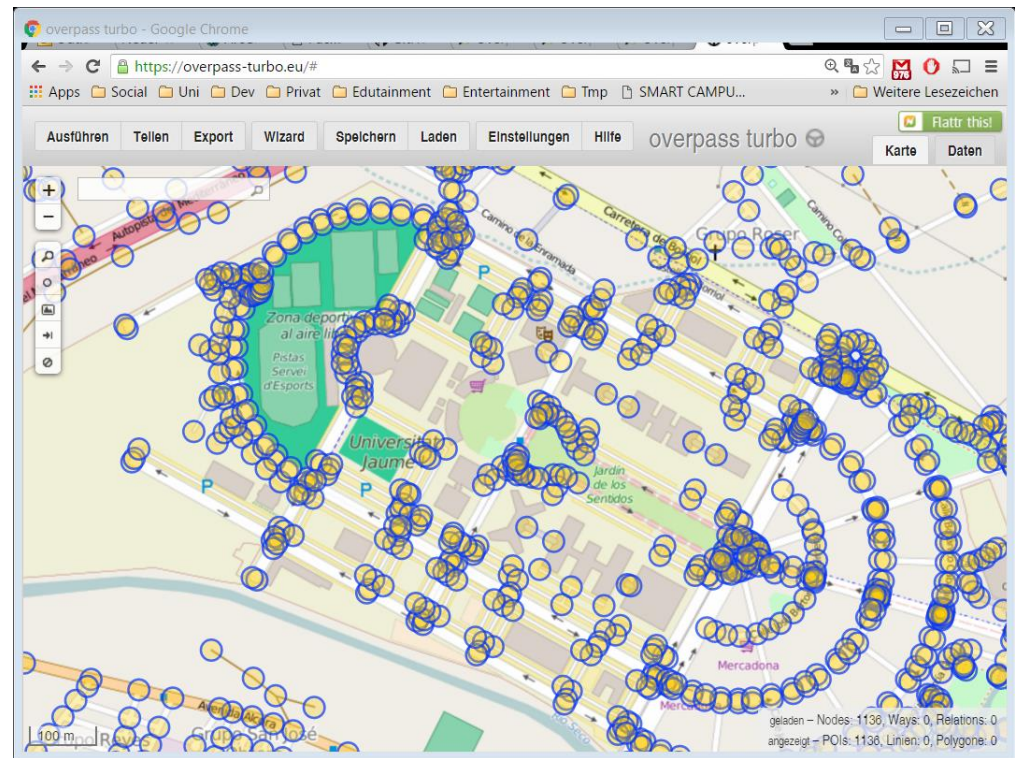
Overpass API



- A read-only API that serves up custom selected parts of the OSM map data
- Acts as a database over the web
- Client sends a query to the API and gets back the data set that corresponds to the query
- Overpass API is optimized for data consumers that need a few elements within a glimpse

Overpass Turbo

- **Task:** Try to generate a good query that fetches nodes for location-based pacman game
- **Optional:** Generate a query that takes the players position into account



<https://overpass-turbo.eu/>

Query Examples:

http://wiki.openstreetmap.org/wiki/Overpass_API/Language_Guide

http://wiki.openstreetmap.org/wiki/Overpass_turbo/Examples

Ingame Locations

■ Tasks:

- Get these locations into the app
- Use the following OSM API Wrapper files, available here:

<http://www.geogames-team.org/files/uji/files/osm/>



Triggering Location-Based Events

■ The easy way:

```
locService = (LocationManager) getSystemService(Context.LOCATION_SERVICE);
this.locListener = new LocationListener() {
    @Override
    public void onLocationChanged(Location location) {
        Log.d(TAG, "Location changed");
        playerPosition = location;
        float distance = playerPosition.distanceTo(targetLocation);
        if (distance < PROXY_RADIUS) {
            // Do something
        }
    }
};
```

■ Using the location manager:

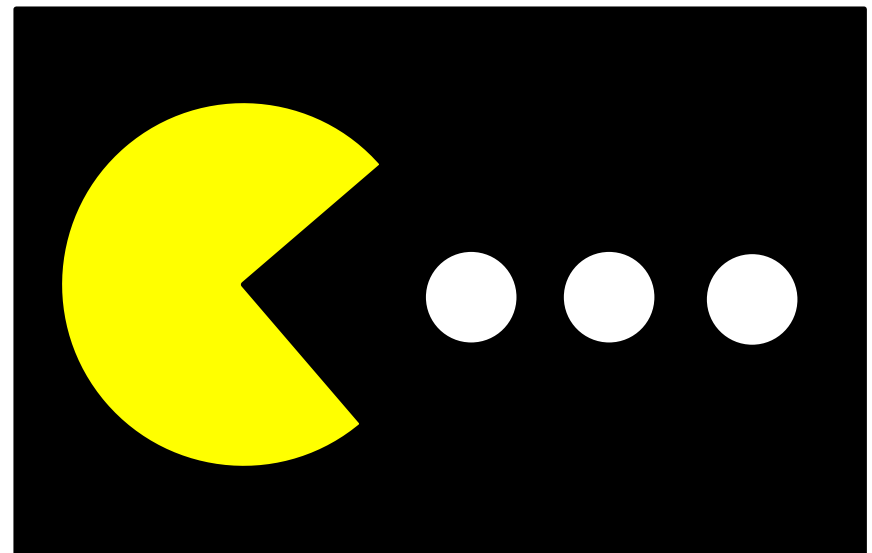
```
void addProximityAlert(double latitude, double longitude, float radius, long expiration, PendingIntent intent)
    Set a proximity alert for the location given by the position (latitude, longitude) and the given radius.
```

■ Geotrigger API

- runs in the cloud
- reduces battery drain when running location-based apps
- Precision is dependent on tracking profile:
 - neighborhood level to GPS precision

Making Nodes Collectible

- Tasks: Modify your game, so that the player “collects” nodes when she is within a 10m radius
 - Implement this logic within the game service
 - Update the map accordingly
 - Update the score
 - Test your implementation





Routing

Routing Options

- OSM Online Router
 - E.g. OpenRouteService, YOURS, BRouter
 - Comparison:
http://wiki.openstreetmap.org/wiki/Routing/online_routers
- Graphhopper
 - Offline routing with OSM-Data
- ESRI (Online / Offline)

Routing with the ESRI SDK

- Allows you to calculate point-to-point and multi-point routes using ArcGIS
- Requires the authentication with ArcGis
- Impedance specifies a cost that should be minimized in the resulting route. For example, impedance can be set to Time in order to calculate the fastest route or Length to calculate the shortest route

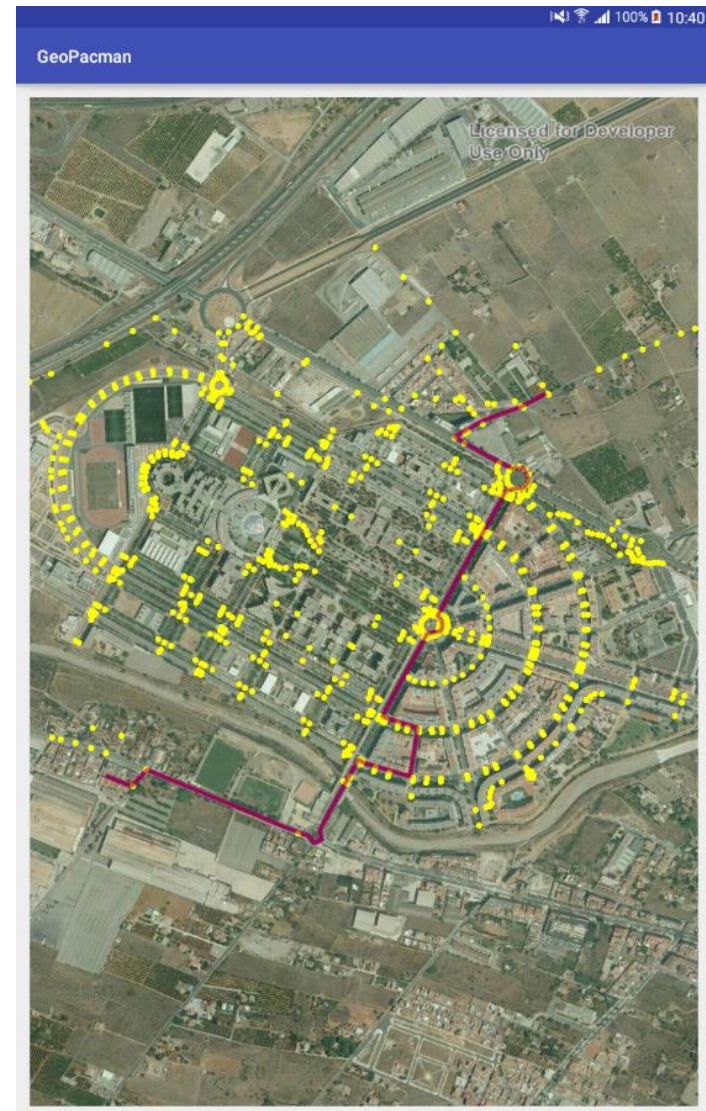
Calculating a Route Online

- Steps to get a route between to locations:
 - Authenticate with ArcGis Online
 - Set up a routing task
 - Create a network analysis layer that contains all stops for that route
 - Let the task calculate the route
 - Optional: Create a graphic and add it to the graphics layer

```
String routeTaskURL = "http://route.arcgis.com/arcgis/rest/services/World/Route/NAS";
try {
    UserCredentials creds = new UserCredentials();
    creds.setUserAccount("name", "password");
    RouteTask routeTask = RouteTask.createOnlineRouteTask(routeTaskURL, creds);
    RouteParameters routeParams = routeTask.retrieveDefaultRouteTaskParameters();
    routeParams.setOutSpatialReference(mapView.getSpatialReference());
    NAFeaturesAsFeature stops = new NAFeaturesAsFeature();
    stops.setSpatialReference(mapView.getSpatialReference());
    stops.addFeature(from);
    stops.addFeature(to);
    routeParams.setStops(stops);
    RouteResult mResults = routeTask.solve(routeParams);
    Route route = mResults.getRoutes().get(0);
    Log.d("Route", route.toString());
    Geometry routeGeo = route.getRouteGraphic().getGeometry();
    gLayer.addGraphic(new Graphic(routeGeo, new SimpleLineSymbol(0x99990055, 5)));
} catch (Exception e) {
    Log.e("Route", "Could not create route");
    e.printStackTrace();
}
```

Routing Exercise

- **Task:** Create a route between two arbitrary locations on your map
- Visualize this route
- Research how you can access the nodes of this route

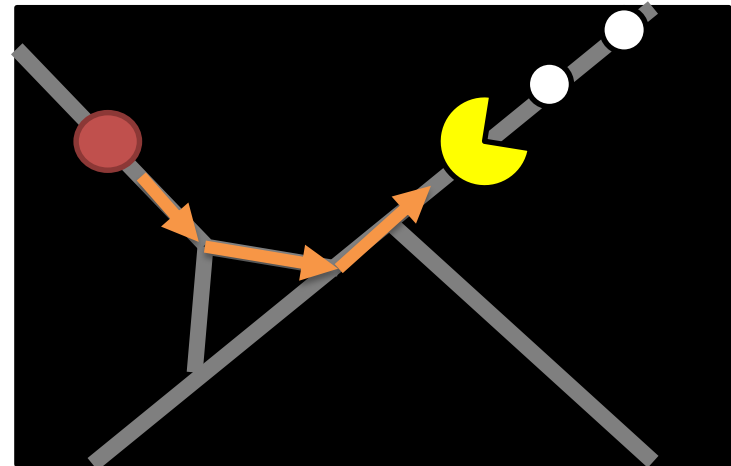


Accessing Route Nodes

- Route geometries are always polylines
 - Cast the geometry
 - Iterate over its points

```
Geometry routeGeo = route.getRouteGraphic().getGeometry();
Polyline line = (Polyline) routeGeo;
for (int i = 0; i < line.getPointCount(); i++) {
    Point routeNode = line.getPoint(i);
    Log.d("Route Node", routeNode.toString());
}
```

- To make an entity move along a route translate it towards the next node





Putting It All Together

GeoPacman

- **Task:** Create a location-based Pacman game
- **Inspiration:**
 - Enemies
 - Support game field creation through ArcGis Online
 - Make the game playable anywhere
 - Multiplayer with the help of ArcGis Services

