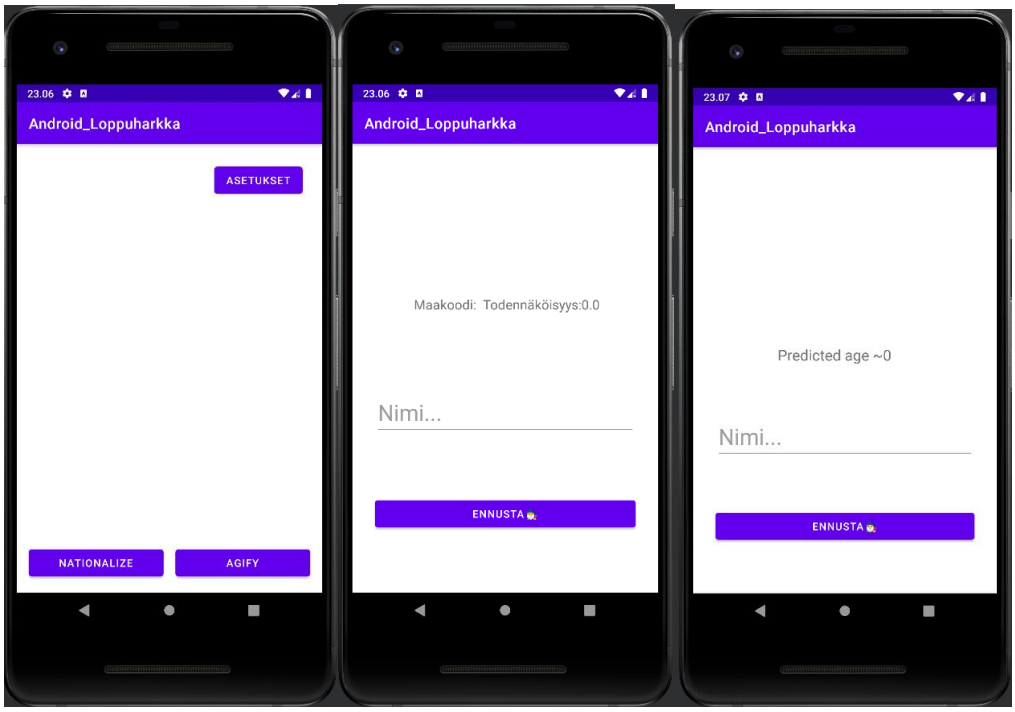
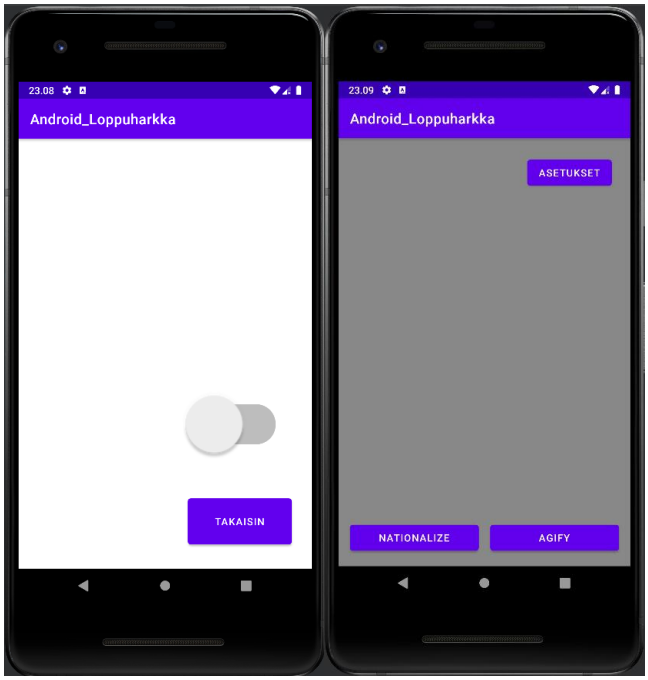
1 ANDROID LOPPUHARJOITUS

1.1 Sovelluksen dokumentointi

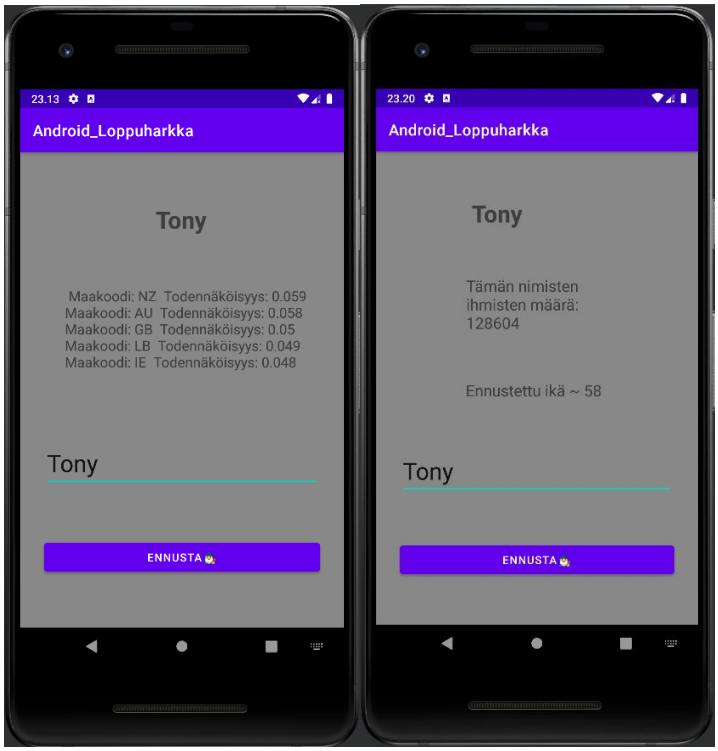
Tein sovelluksen, jossa on neljä aktiviteettia, jotka olivat main, settings ja kaksi aktiviteettiä, joilla oli webbi rajapinta.



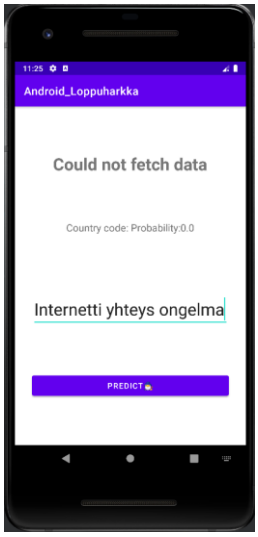
Settingsissä on yksi switch, jolla saa päätettyä onko sovellus vaaleassa vai tummassa tilassa. Tila vaihtuu switchin asennon mukaan, kun asetuksista poistutaan.



Main aktiviteetistä pääsee navigoimaan agify ja nationalize näkymiin, joissa käyttäjä voi ennustaa ikää tai kansallisuutta nimen perusteella.



Näkymissä on käytetty constraint layouttia ja järkevästi käännettävät tekstit on käännetty myös suomeksi. Jos dataa ei saada haettua rajapinnalta antaa sovellus tuloksien tilalle virheviestin.



Harjoitusta tehtäessä sain hyvää kertausta ja rutiinia aikaisemmissa harjoituksissa tehtyihin asioihin, kuten instanssien käyttäminen.

1.2 Koodit

1.2.1 AgifyActivity

```
package com.example.android_loppuharkka;

import ...


public class AgifyActivity extends AppCompatActivity {
    String name = " ";
    int count;
    int age;
    String bgcolor =" ";

    private String mUrl = "https://api.agify.io?name=";
    private RequestQueue mQueue;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_agify);
        mQueue = Volley.newRequestQueue( context: this);

        if(savedInstanceState != null) {
            name = savedInstanceState.getString( key: "NAME");
            count = savedInstanceState.getInt( key: "COUNT");
            age = savedInstanceState.getInt( key: "AGE");
            bgcolor = savedInstanceState.getString( key: "color");

        }
        TextView nameAgifyTextView = (TextView) findViewById(R.id.nameAgifyTextView);
        nameAgifyTextView.setText("  " + name + "  ");

        TextView ageTextView = (TextView) findViewById(R.id.ageTextView);
        ageTextView.setText("Predicted age ~" + age + "");


        if(savedInstanceState != null) {
            bgcolor = savedInstanceState.getString( key: "color");
        }

        if(getIntent().getStringExtra( name: "color")!= null){
            bgcolor = getIntent().getStringExtra( name: "color");
        }
```

```
        ConstraintLayout layoutti = findViewById(R.id.agifyLayout);
        if(bgcolor.equals("black")){
            layoutti.setBackgroundColor(Color.GRAY);
        }
        if(bgcolor.equals("white")){
            layoutti.setBackgroundColor(Color.WHITE);
        }
    }

    @Override
    public void onSaveInstanceState(Bundle savedInstanceState){
        //Tallennetaan olion tila
        savedInstanceState.putString("NAME", name);
        savedInstanceState.putInt("COUNT", count);
        savedInstanceState.putInt("AGE", age);
        savedInstanceState.putString("color", bgcolor);
        super.onSaveInstanceState(savedInstanceState);
    }



    public void fetchAgifyData(View view) {
        EditText nameEditText = findViewById(R.id.nameEditTextAgify);
        JsonObjectRequest jsonObjectRequest = new JsonObjectRequest(Request.Method.GET,  url: mUrl + nameEditText.getText(),  jsonRequest: null,
                new Response.Listener<JSONObject>() {
                    @Override
                    public void onResponse(JSONObject response) {
                        //Toast.makeText(getApplicationContext(), response.toString(), Toast.LENGTH_LONG).show();
                        parseJsonAndUpdateUi(response);
                    }
                }, new Response.ErrorListener() {
            @Override
            public void onErrorResponse(VolleyError error) {
                TextView nameAgifyTextView = findViewById(R.id.nameAgifyTextView);
                nameAgifyTextView.setText(R.string.error);
            }
        });
```

```
        // Lisätään request volley queueen
        mQueue.add(jsonObjectRequest);
    }

    private void parseJsonAndUpdateUi(JSONObject ageObject) {
        // Kaivetaan jsonista data käyttöliittymäkomponentteihin
        TextView ageTextView =  findViewById(R.id.ageTextView);
        TextView nameCountTextView =  findViewById(R.id.nameCountTextView);
        TextView nameAgifyTextView =  findViewById(R.id.nameAgifyTextView);
        nameCountTextView.setText( " ");
        nameAgifyTextView.setText( " ");
        ageTextView.setText( " ");

        try {

            name = ageObject.getString( name: "name");
            nameAgifyTextView.setText(" " + name + " ");

            age = ageObject.getInt( name: "age");
            ageTextView.setText(getString(R.string.predicted_age)+ " " + age + " ");

            count = ageObject.getInt( name: "count");
            nameCountTextView.setText(getString(R.string.Countofpeople)+ (" ")+ count + " ");
        } catch (JSONException e) {
            e.printStackTrace();
        }
    }
}
```

1.2.2 NationalizeActivity

```java
package com.example.android_loppuharkka;
import ...
public class NationalizeActivity extends AppCompatActivity {
    String name = " ";
    String countri = " ";
    String main = " ";
    Double prob = 0.0;
    String bgcolor =" ";
    private String mUrl = "https://api.nationalize.io?name=";
    private RequestQueue mQueue;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_nationalize);
        mQueue = Volley.newRequestQueue( context: this);

        if(savedInstanceState != null) {
            name = savedInstanceState.getString( key: "NAME");
            countri = savedInstanceState.getString( key: "COUNTRI");
            main = savedInstanceState.getString( key: "MAIN");
            prob = savedInstanceState.getDouble( key: "PROB");
            bgcolor = savedInstanceState.getString( key: "color");
        }
        TextView nameNationalizeTextView =  findViewById(R.id.nameNationalizeTextView);
        nameNationalizeTextView.setText(" " + name + " ");

        TextView countryCodeTextView =  findViewById(R.id.countryCodeTextView);
        countryCodeTextView.setText(  countri + getString(R.string.countrycode) + main + getString(R.string.probability) + prob + "\n");

        if(savedInstanceState != null) {
            bgcolor = savedInstanceState.getString( key: "color");

        }
        if(getIntent().getStringExtra( name: "color")!= null){
            bgcolor = getIntent().getStringExtra( name: "color");
        }
        ConstraintLayout layoutti = findViewById(R.id.nationalizeLayout);
        if(bgcolor.equals("black")){
            layoutti.setBackgroundColor(Color.GRAY);
```

```
        }
        if(bgcolor.equals("white")){
            layoutti.setBackgroundColor(Color.WHITE);
        }

    }

    @Override
    public void onSaveInstanceState(Bundle savedInstanceState){
        //Tallennetaan olion tila
        savedInstanceState.putString("NAME", name);
        savedInstanceState.putString("COUNTRI", countri);
        savedInstanceState.putString("MAIN", main);
        savedInstanceState.putDouble("PROB", prob);
        savedInstanceState.putString("color", bgcolor);

        super.onSaveInstanceState(savedInstanceState);
    }

    public void fetchData(View view) {

        EditText nameEditTextNationalize = findViewById(R.id.nameEditTextNationalize);
        JsonObjectRequest jsonObjectRequest = new JsonObjectRequest(Request.Method.GET,
                url: mUrl + nameEditTextNationalize.getText(),  jsonRequest: null,
                new Response.Listener<JSONObject>() {
                    @Override
                    public void onResponse(JSONObject response) {
                        parseJsonAndUpdateUi(response);

                    }
                }, new Response.ErrorListener() {
            @Override
            public void onErrorResponse(VolleyError error) {
                TextView nameNationalizeTextView = (TextView) findViewById(R.id.nameNationalizeTextView);
                nameNationalizeTextView.setText(R.string.error);
            }
        });
```

```
        // Lisätään request volley queueen
        mQueue.add(jsonObjectRequest);
    }

    private void parseJsonAndUpdateUi(JSONObject nationObject) {
        // Kaivetaan jsonista data käyttöliittymäkomponentteihin

        TextView nameNationalizeTextView = (TextView) findViewById(R.id.nameNationalizeTextView);
        TextView countryCodeTextView = (TextView) findViewById(R.id.countryCodeTextView);
        countryCodeTextView.setText( " ");
        try {

            name = nationObject.getString( name: "name");
            nameNationalizeTextView.setText(" " + name + " ");


            JSONArray country = nationObject.getJSONArray( name: "country");
            for(int i = 0; i < country.length(); i++) {
                countri= countryCodeTextView.getText().toString();
                JSONObject nationobj = country.getJSONObject(i);
                main = nationobj.getString( name: "country_id");
                prob = nationobj.getDouble( name: "probability");
                countryCodeTextView.setText(  countri + " " + getString(R.string.countrycode)
                        + " " + main + " " + getString(R.string.probability) + " " + prob + "\n");

            }
        } catch (JSONException e) {
            e.printStackTrace();
        }
    }
}
```

1.2.3 MainActivity

```java
package com.example.android_loppuharkka;

import ...


public class MainActivity extends AppCompatActivity {

    String bgcolor = "";
    boolean state = false;
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);

        if(savedInstanceState != null) {
            bgcolor = savedInstanceState.getString( key: "color");

        }

        if(getIntent().getStringExtra( name: "color")!= null){
            bgcolor = getIntent().getStringExtra( name: "color");
        }

        ConstraintLayout layoutti = findViewById(R.id.mainLayout);
        if(bgcolor.equals("black")){
        layoutti.setBackgroundColor(Color.GRAY);
        state = true;
        }
        if(bgcolor.equals("white")){
            layoutti.setBackgroundColor(Color.WHITE);
        }

    }

    @Override
    public void onSaveInstanceState(Bundle savedInstanceState){
        //Tallennetaan olion tila
        savedInstanceState.putString("color", bgcolor);
        super.onSaveInstanceState(savedInstanceState);
    }
```

```java
@Override
protected void onStart() {

    super.onStart();
    //Aktiviteetti on käynnistymässä
}

@Override
protected void onResume() {

    super.onResume();
    //Aktiviteetti on tulossa näkyviin
}

@Override
protected void onStop() {

    super.onStop();
    //Aktiviteetti on pois näkyvistä
}

@Override
protected void onPause() {

    super.onPause();
    //Aktiviteetti on poistumassa näkyvistä
}

@Override
protected void onDestroy() {

    super.onDestroy();
    //Aktiviteetti on tuhottu
}
```

```java
public void openNationalize(View view) {
    Intent intent = new Intent( packageContext: this, NationalizeActivity.class);
    if (bgcolor.equals("black")) {
        intent = intent.putExtra( name: "color",  value: "black");


    }else{
        intent = intent.putExtra( name: "color",  value: "white");

    }
    startActivity(intent);
}

public void openAgify(View view) {

    Intent intent = new Intent( packageContext: this, AgifyActivity.class);
    if (bgcolor.equals("black")) {
        intent = intent.putExtra( name: "color",  value: "black");


    }else{
        intent = intent.putExtra( name: "color",  value: "white");

    }


    startActivity(intent);

}

public void openSettings(View view) {
    Intent intent = new Intent( packageContext: this, Settings.class);


    if (bgcolor.equals("black")) {
        intent = intent.putExtra( name: "color",  value: "black");
```

```
        }else{
            intent = intent.putExtra( name: "color",  value: "White");

        }

        if (bgcolor.equals("black")) {
            intent = intent.putExtra( name: "booli",  value: "true");


        }else{
            intent = intent.putExtra( name: "booli",  value: "false");

        }

        startActivity(intent);

    }

}
```

1.2.4 Settings

```
package com.example.android_loppuharkka;

import ...

public class Settings extends AppCompatActivity {

    boolean switchState;
    private Switch Switch;
    String buul;
    String bgcolor =" ";

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_settings);
        Switch = findViewById(R.id.switch1);

        if(savedInstanceState != null) {
            switchState = savedInstanceState.getBoolean( key: "state");
            Switch.setChecked(savedInstanceState.getBoolean( key: "switsi"));
            bgcolor = savedInstanceState.getString( key: "color");
        }

        if(getIntent().getStringExtra( name: "booli")!= null){
            buul = getIntent().getStringExtra( name: "booli");
        }

        if(buul.equals("true")){
            Switch.setChecked(true);
        }else if(buul.equals("false")){
            Switch.setChecked(false);
        }

        if(savedInstanceState != null) {
            bgcolor = savedInstanceState.getString( key: "color");

        }
```

```
    if(getIntent().getStringExtra( name: "color")!= null){
        bgcolor = getIntent().getStringExtra( name: "color");
    }

    ConstraintLayout layoutti = findViewById(R.id.settingsLayout);
    if(bgcolor.equals("black")){
        layoutti.setBackgroundColor(Color.GRAY);

    }
    if(bgcolor.equals("white")){
        layoutti.setBackgroundColor(Color.WHITE);
    }
}

@Override
public void onSaveInstanceState(Bundle savedInstanceState){
    //Tallennetaan olion tila
    savedInstanceState.putBoolean("state", switchState);
    savedInstanceState.putBoolean("switsi", Switch.isChecked());
    savedInstanceState.putString("color", bgcolor);
    super.onSaveInstanceState(savedInstanceState);
}

@Override
protected void onStart() {

    super.onStart();
    //Aktiviteetti on käynnistymässä

}

@Override
protected void onResume() {

    super.onResume();
    //Aktiviteetti on tulossa näkyviin
}


@Override
protected void onStop() {
```

```
        super.onStop();
        //Aktiviteetti on pois näkyvistä
    }

    @Override
    protected void onPause() {

        super.onPause();
        //Aktiviteetti on poistumassa näkyvistä
    }

    @Override
    protected void onDestroy() {

        super.onDestroy();
        //Aktiviteetti on tuhottu
    }



    public void backToMain(View view){


        //Siirrytään takaisin
        Intent intent = new Intent( packageContext: this,MainActivity.class);

        if (Switch.isChecked()) {
            switchState = true;
            intent = intent.putExtra( name: "color",  value: "black");


        }else{
            switchState = false;
            intent = intent.putExtra( name: "color",  value: "white");

        }
        startActivity(intent);
    }
}
```

2 QML -loppuharjoitus

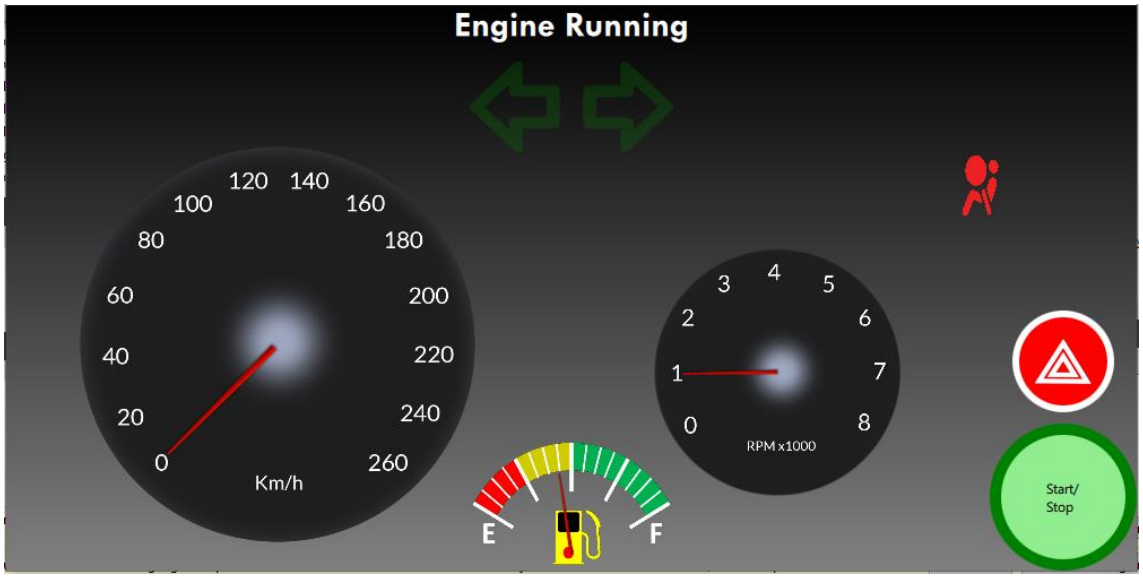
2.1 Sovelluksen dokumentointi

Päätin tehdä harjoitustehtävän jatkamalla kojelauta tehtävää pidemmälle.
Kojelaudalla on

- nopeusmittari, jonka viisari osoittaa annettua nopeutta (oletuksena 0) ja käynnistäessä viisari pyörähtää mittarin ympäri
- polttoainemittari, joka saa satunnaisen arvon, jos polttoaine on vähissä, syttyy vikavalojen joukkoon polttoainevalo.
- vilkut, joilla ei ole muuta toiminnallisuutta kuin olla hätävilkkuina hätävilkun nappia painettaessa.
- kierroslukumittari, joka osoittaa saamaansa arvoa ja käynnistäessä nousee vaihdettuun tyhjäkäynti arvoon.
- vikavalot, jotka syttyvät, jos saavat signaalin (Math.random) polttoaineen valoa lukuun ottamatta joka ottaa signaalinsa siitä mitä polttoainemittari näyttää.

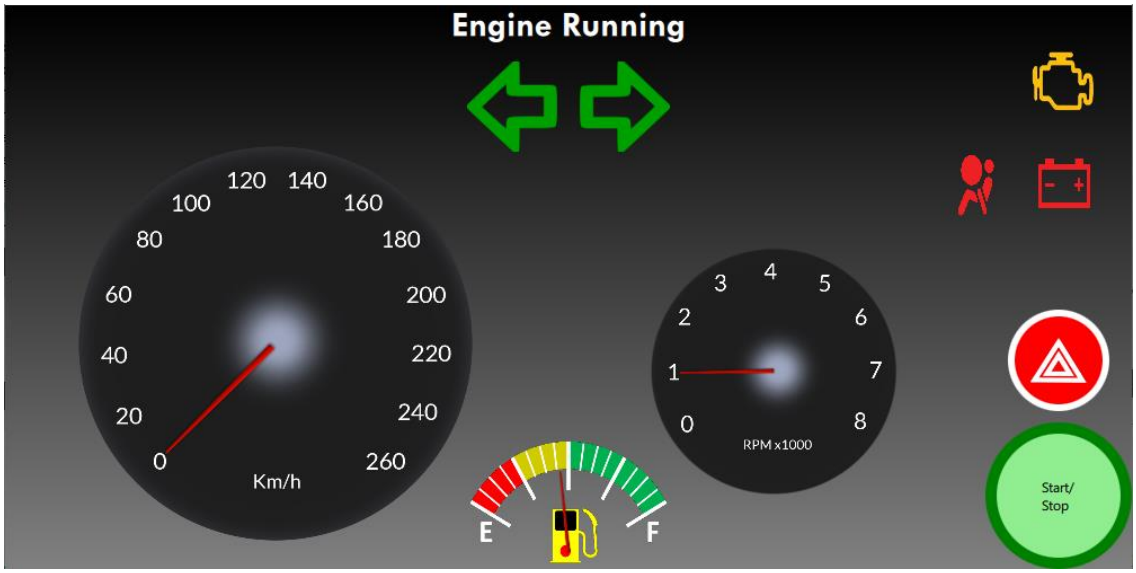
- Kuvat sovelluksen ajamisesta

(Moottori sammutettu)



(Moottori käynnistetty)

(Muutama lisäkäynnistys, joissa on saatu eri arvoja vikavaloille)



(Hätävilkut painettu päälle.)

2.2 Koodit

2.2.1 main.qml

```
import QtQuick 2.15
import QtQuick.Window 2.15

Window {
    width: 1000
    height: 500
    visible: true
    title: qsTr("Citroen Berlingo")
    Rectangle{
        property bool engineRunning: false // start stop
        id: background
        anchors.fill: parent
        color: "#000000"
        gradient: Gradient {
            GradientStop {
                id:gradientStop1
                position: 0.00;
                color: "#000000";

            }
            GradientStop {
                id:gradientStop2
                position: 1.00;
                color: "#000000";
                Behavior on color{
                    PropertyAnimation{
                    easing.type: Easing.InOutCubic
                    duration: 1500
                            }
                 }
            }
        }
        Timer{
             id: timer
             interval: 500
             running: false
             repeat: true
             onTriggered: indicator.opacity = indicator.opacity === 0.2 ? 1 : 0.2
                    }
        Tachometer{
            id: tacho
            rpm: 0
            opacity: 0
            Behavior on opacity{
                PropertyAnimation{
                    easing.type: Easing.InCirc
                    duration: 2000
                }
```

```
        }
        x:560
        y:200
    }
    Speedometer{
        id: speedo
        speed: 260
        opacity: 0
        Behavior on opacity{
            PropertyAnimation{
                easing.type: Easing.InCirc
                duration: 2000
            }
        }
        x:40
        y:100
    }
    Fuelgauge{
        id: fuelgauge
        fuel: 0
        opacity: 0
        Behavior on opacity{
            PropertyAnimation{
                easing.type: Easing.InCirc
                duration: 2000
            }
        }
        x:400
        y:370
    }

    Indicators{
        id: indicator
        opacity: 0
        Behavior on opacity{
            PropertyAnimation{
                easing.type: Easing.InCirc
                duration: 50
            }
        }
        x:400
        y:0
    }
```

```
Warninglights{
    id: warnings
    opacity: 0
    Behavior on opacity{
        PropertyAnimation{
            easing.type: Easing.InCirc
            duration: 2000
        }
    }
    x:800
    y:10
}



Text{
    id: motorStatusText
    text: ""
    opacity: 0
    color: "#ffffff"
    font.family: "Tw Cen MT"
    font.bold: true
    font.pointSize: 24
    anchors.horizontalCenter: parent.horizontalCenter
    Behavior on opacity{
        PropertyAnimation{
            easing.type: Easing.InCirc
            duration: 2000
        }
    }
    Behavior on y {
        PropertyAnimation{
            easing.type: Easing.InQuint
            duration: 2000
        }
    }

    }
```

```
Button{
    buttonText: "Start/\nStop"
    backgroundColor: "lightgreen"
    border.color: "green"
    border.width: 10
    anchors.right: parent.right
    anchors.bottom: parent.bottom
    onButtonClicked: {
        if(background.engineRunning == false ){
            motorStatusText.text = "Engine Running"
            motorStatusText.opacity = 1
            tacho.rpm = 1000
            speedo.speed = 0
            speedo.opacity = 1
            tacho.opacity = 1
            fuelgauge.opacity = 1
            fuelgauge.fuel = (Math.ceil(Math.random()*60))
            indicator.opacity = 0.2
            warnings.opacity = 1
            hazard.opacity= 1
            gradientStop2.color = "grey"
            //Warninglight randomizer
            warnings.oil.opacity =  Math.ceil(Math.random()*2)-1
            warnings.engine.opacity = Math.ceil(Math.random()*2)-1
            warnings.airbag.opacity = Math.ceil(Math.random()*2)-1
            warnings.battery.opacity = Math.ceil(Math.random()*2)-1
            if(fuelgauge.fuel < 5){warnings.fuellight.opacity = 1}else{warnings.fuellight.opacity = 0}
            background.engineRunning =true;

            }
            else{
            motorStatusText.opacity = 0
            tacho.rpm = 0
            speedo.opacity = 0
            tacho.opacity = 0
            fuelgauge.opacity = 0
            fuelgauge.fuel = 0
            indicator.opacity = 0
            warnings.opacity = 0
            hazard.opacity= 0
            background.engineRunning =false;
            gradientStop2.color = "black"
            }
        }
    }
```

```
            Emergencyblinkers{
                id: hazard
                border.color: "white"
                x:890
                y:270
                state: false
                opacity:0
                Behavior on opacity{
                    PropertyAnimation{
                        easing.type: Easing.InCirc
                        duration: 2000
                    }
                }
                onButtonClicked: {
                    if(hazard.state == false){
                        timer.running = true
                        hazard.state = true
            } else {
                        hazard.state = false
                        timer.running = false
                        indicator.opacity = 0.2

                    }
            }
        }
    }
}
```

2.2.2 Warninglights.qml

```
import QtQuick 2.15

Item {
    property var oil: oil
    property var engine: engine
    property var airbag: airbag
    property var battery: battery
    property var fuellight: fuellight

    Rectangle{
        width:160
        height:400
        color:"transparent"
        Image {
            id: oil
            opacity: 1
            width: 100
            height: 80
            source: "oljy.png"
            x:10
            y:20
            }
```

(loput vikavalot tehty samalla kaavalla kuin ”*Oil*”)

2.2.3 Tachometer.qml

```
import QtQuick 2.15

Item {
    property int rpm: 0
    Image {
        width: 250
        height: 250
        source: "tacho.png"
        Image{
            id: redNeedle
            source: "needlered.png"
            width: parent.width * 0.03
            height: parent.height * 0.35
            anchors.centerIn: parent
            anchors.verticalCenterOffset: -0.5 * height
            rotation: -121 + rpm / 1000 * 30

            transformOrigin: Item.Bottom
            Behavior on rotation{
                PropertyAnimation{
                    duration: 3000
                    easing.type: Easing.InBounce

                    }
                }
            }
        }
}
```

2.2.4 Speedometer.qml

```
import QtQuick 2.15

Item {
    property int speed: 0
    Image {
        width: 400
        height: 400
        source: "speedo.png"
        Image{
            id: redNeedle
            source: "needlered.png"
            width: parent.width * 0.03
            height: parent.height * 0.35
            anchors.centerIn: parent
            anchors.verticalCenterOffset: -0.5 * height
            rotation: -135+ speed * 1.05
            transformOrigin: Item.Bottom
            Behavior on rotation{
                PropertyAnimation{
                    duration: 3000
                    easing.type: Easing.InExpo

                    }
                }
            }

    }
}
```

2.2.5 Indicators.qml

```
import QtQuick 2.15

Item {

    Rectangle{
        width:200
        height:200
        color:"transparent"
        Image {
            id: left
            width: 80
            height: 80
            rotation: 180
            source: "vilkku.png"
            x:10
            y:50

            }
        Image {
            id: right
            width: 80
            height: 80

            source: "vilkku.png"
            x:110
            y:50

            }
    }
}
```

2.2.6 Fuelgauge.qml

```
import QtQuick 2.15

Item {
    property int fuel: 0
    Image {
        width: 200
        height: 150
        source: "fuel.png"
        Image{
            id: redNeedle
            source: "needlered.png"
            width: parent.width * 0.03
            height: parent.height * 0.5
            anchors.centerIn: parent
            anchors.verticalCenterOffset: 0.06 * height
            rotation: -60 + fuel * 2
            transformOrigin: Item.Bottom
            Behavior on rotation{
                PropertyAnimation{
                    duration: 3000
                    easing.type: Easing.CosineCurve

                    }
                }
            }
        }
}
```

2.2.7 Emergencyblinkers

```
import QtQuick 2.15

Rectangle{
    property color backgroundColor: "red"
    property bool state
    signal buttonClicked
    width: 90
    height: 90
    radius: 100
    color: backgroundColor
    border.width:6
    Image {
        id: blinkers
        source: "hazard.png"
        width: 50
        height: 40
        anchors.verticalCenter: parent.verticalCenter
        anchors.horizontalCenter: parent.horizontalCenter

    }
    MouseArea{
        anchors.fill: parent
        onPressed:{
            border.width = 10
            blinkers.width = 40
            blinkers.height = 32
        }
        onReleased:{
            border.width = 6
            blinkers.width = 50
            blinkers.height = 40
            parent.buttonClicked()
        }
    }
}
```

2.2.8 Button

```
import QtQuick 2.15

Rectangle{
    property string buttonText: ""
    property color backgroundColor: "green"
    signal buttonClicked

    width: 130
    height: 130
    radius: 100
    color: backgroundColor
    Text{
    text: parent.buttonText
    anchors.centerIn: parent
    }
    MouseArea{
        anchors.fill: parent
        onPressed:{
            border.width = 15
        }
        onReleased:{
            border.width = 10
            parent.buttonClicked()
        }

    }
}
```