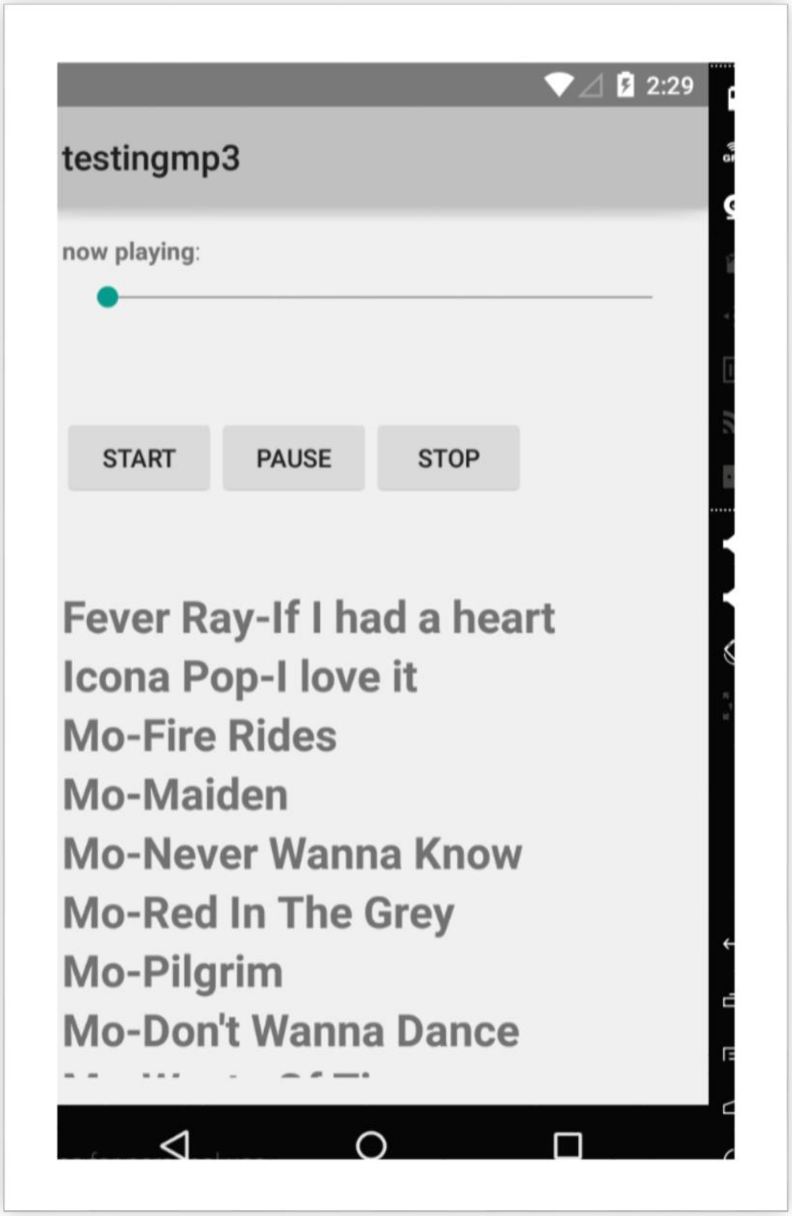

testingmp3

now playing:

Fever Ray-If I had a heart
Icona Pop-I love it
Mo-Fire Rides
Mo-Maiden
Mo-Never Wanna Know
Mo-Red In The Grey
Mo-Pilgrim
Mo-Don't Wanna Dance

MP3 PLAYER

MOBILE APPLICATIONS

Jimmi Andersen, Ada Wiśnlowska and Jan Radil | mob1 | 3. august 2015

Description.

The objective of the task was prepare functional MP3 player with basic control menu such as Play, Pause and Stop.

During development, we had a couple of troubles, which we describe in detail in this documentation.

1) STOP functionality

During the development phase we had a trouble with the stop functionality, we encountered that the program resets buffer while we are using the API's stop function and then API doesn't allow to start playing the song from the beginning again. We found a way to solve this issue by using the pause and seek functions as displayed in the example below.

```
stop.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View v) {
           //because using mp.stop() stopped the music from playing state without possibility to restart it
           mp.pause();
           mp.seekTo(0);
           Toast.makeText(MainActivity.this, "music stopped", Toast.LENGTH_SHORT).show();
        }
     });
```

2) Language issue

Subsequently, we have accounted language problem with API. As API uses the Latin alphabet then it does not contain letters such as æ-ø-å. This was a problem once we wanted add new songs to our player. We tried to set font size for our texts

```
String s="<font size="25/">text</font>
```

This solution did not work, so we decided to use following method, which works properly.

```
t1.setTextSize(TypedValue.COMPLEX_UNIT_SP, 25);
```

3) Seekbar

During the seek bar development phase we discovered that our seekbar did not move because we forgot to add a handler to automatically update the seekbar state according to the song’s current position. We have set 1ms to smoothly update the seekbar position.

```
public void seekbar_update()
  {
     if(mp.isPlaying())
     {   sb.setProgress(mp.getCurrentPosition());
        Runnable r=new Runnable()
        {
           @Override
        public void run()
           {
              seekbar_update();
           }
        };
        new Handler().postDelayed(r,1);
     }
}
```

4) Double Initialization

During the collaboration between the media player and the seekbar we encounter some difficulty with the fact that we apparently have initialization of the media player in two places which does not allow the seekbar to use the right media player when updating the progress bar. As the one which will be used is the last one created media player which will not start with playing the song.

We solved this by making the media player a field variable and initialize it in the OnCreate() method and with this way we use the same media player in the entire program.