TUKORO

JUST TALK

אפליקציה להזנת טקסט בדיבור

JusTalk

אפליקציה להזנת טקסט בדיבור

מילוי טפסים דורש מהמשתמש זמן רב. החל מבקשת הטופס המבוקש, מילוי כל הפרטים הנצרכים וכמובן שליחת הטופס.

כיום גם הפלטפורמות שכן מאפשרות מילוי טופס ע"י דיבור
אינן ממלאות אחר צרכי המשתמש
זיהוי המילים מוטעה רוב הפעמים ואין אפשרות שליחה ותיקון.

האפליקציה שפיתחנו מספקת זיהוי טוב יותר של המילים הנקלטות
מאפשרת שליחה מידית או מאוחרת ותיקון הטופס לאחר המילוי
כל זאת עם תכנון חווית המשתמש
ובניית ממשק משתמש נוח נאה וברור למשתמש

במוצר זה פיתחנו ממשק קולי ייעודי למילוי טפסים בדיבור,

כאשר מטרתנו היא:

כתיבה מחדש של מוצר הprototype הקיים של החברה.

על מנת להגיע למוצר משופר יותר מכל הבחינות הנ"ל.

עמדנו ביעדים אותם הצבנו לעצמנו

יתרונות האפליקציה

- חיסכון זמן במילוי טופס מבלי להתעסק בניירת מיותרת.
- קלות שליחת הטופס הרצוי בלחיצת כפתור.
- מהירות וממשק נוח ברור ונאה.
- מספקת אפשרות לשמירה ועדכון טופס במועד הרצוי בתוך האפליקציה.
- מותאמת לאנשים המתקשים בקריאה וכתיבה.
- אפשרות תמיכה באנגלית עברית וכל שפה אפשרית כפי הגדרת הטופס.

מבנה המערכת

נתאר את הרכיבים העיקריים

Classes

- **Field** מכילה את השדות:

id, filedName, required, order, dataType, filedAnswer

- **Layout** מכילה את השדות:

id, ayoutName, fields;

- **User** מכילה את השדות:

id, userEmail, password, serviceId, clientId, layouts

- **AppAdapter** מכילה את הפונקציות:

ArrayList<Layout> getObjectsLayoutsForUser(String arrayOfLayoutInfo

מקבלת את תשובת השרת במחרוזת ומחזירה אובייקט המכיל את נתוני השדות לפי משתמש.

boolean submitLayoutForUser(Layout layout

מקבלת טופס מהמשתמש שולחת אותו לשרת ומחזירה true/false לפי סטטוס השליחה.

- **FormMessage** מכילה את השדות:

id, isMe,message; userId,dateTime

Services

- **AudioRecordService** -bound service פונקציות עיקריות:

void startRecordAudio()

הפונקציה יוצרת AudioRecord ומתחילה הקלטה.

void writeAudioDataToFile()

הפונקציה מבצעת האזנה לקול שנקלט, מקליטה את הקול וכותבת לקובץ. וכאשר מזהה שקט קוראת לפונקציה העוצרת את התהליך.זיהוי השקט נעשה ע"י קבלת אמפליטודות הקול שנקלטו ובדיקה האם הן נמצאות בטווח הנחשב ל-"טווח שקט". ישנו הסבר מפורט יותר בהמשך- בחלק של קטעי קוד.

void stopRecording()

הפונקציה עוצרת את ההקלטה וקוראת לפונקציית ההמרה.

void convertWAVToFLAC()

הפונקציה ממירה את קובץ ה WAV שנוצר ושומרת אותו בקובץ חדש מסוג FLAC.

AsyncTasks

- **AsyncCall** פונקציות עיקריות:

void doInBackground()

פונקציה המתבצעת ברקע - קריאה ל AudioRecordService על מנת להקליט את תשובת המשתמש.
בפונקציה זו ממומש ה Task העיקרי.

void onPostExecute()

פונקציה המתבצעת לאחר סיום ביצוע ה Task לאחר שתשובת המשתמש נקלטה במלואה ונשמרה בקובץ.
בפונקציה זו מתרחשות מספר נקודות חשובות:

- קריאה ל AsyncTask - SendRecordSoap האחראי על שליחת קובץ ה Audio לשרת.
- המתנה עד לקבלת תשובה (בעזרת הוספה של execute().get), ולאחר שמתקבלת תשובה- הצגת Message המייצג את התשובה ועדכון התשובה על המסך.
- קריאה לפונקציה -doSpeak() על מנת להקריא את השדה הבא.

void onPreExecute()

פונקציה המתבצעת לפני הרצת ה Task קריאה לפונקציה showProgressBar()
המציגה dottedProgressBar המופיע בזמן הקלטת המשתמש (על מנת לשפר את חווית המשתמש).

- **SendRecordSoap**

אחראי על שליחת ההקלטה לשרת והחזרת התשובה המתקבלת
מכיל את הפונקציה encodeAudio האחראית על המרה BASE64
בפונקציה זו השתמשנו בSoapObject וב envelope על מנת לשלוח את קידוד של קובץ ההקלטה.
מפורט בהמשך. בחלק של קטעי קוד.

Activites

- **FormActivity**

כאן מתנהל כל תהליך מילוי הטופס:
נבצע פה קריאה ל TTS Service בכדי להקריא את השדות,
נשתמש בפונקציה onUtteranceCompleted בכדי לדעת מתי הסתיימה הקראת השדה ורק לאחר מכן
נבצע קריאה AsyncCall המבצע את ההקלטה - כפי שפירטנו בשקופית הקודמת.
בסוף מילוי כל השדות מוצגים למשתמש שתי בחירות:

- שליחה מידית - קריאה לפונקציה sendForm() -הטופס ישלח במלאו לאימייל המוגדר. אם הטופס שמור ב sharedPreferences - ביצוע מחיקה.
- שמירה בטפסים הממתינים לאישור - קריאה לפונקציה saveForm() -שמירת הטופס ב sharedPreferences על מנת להציג בכרטיסיית 'טפסים הממתינים לאישור' לעריכה , עדכון הטופס ושליחתו במועד הרצוי.

בנוסף, מתבצעת כאן הצגה דינמית של bubbles המייצגים שאלה ותשובה עבור כל שדה, ע"י הפונקציות:
displayMessage, generateRightMessage, generateLeftMessage
וכן הצגת dottedProgressBar מתוך הספרייה (.om.github.igortrncic.dotted-progress-bar:library:1.0)
בכל זמן שמתבצעת הקלטה ע"י הפונקציות: hideProgressBar, showProgressBar

- **LoginActivity**

אימות משתמש מתבצע באמצעות מסך כניסה הדורש אימייל וסיסמא – תוך אימות הנתונים עם השרת.
למשתמש ישנם 3 ניסיונות כניסה, כאשר המשתמש טועה מוצגים לו מספר הניסיונות שנותרו ולאחר 3 כאלה התחברויות ננעלת.

- **MainActivity**

ניתוב לדפים User profile, Forms, Waiting forms בלחיצה על התפריט הצדדי או בסלייד יועבר המשתמש ל Fragment המתאים.

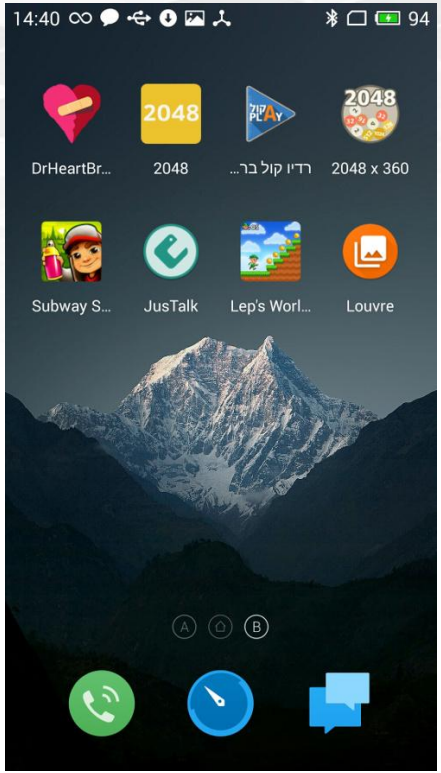
- **SingleWaitingForm**

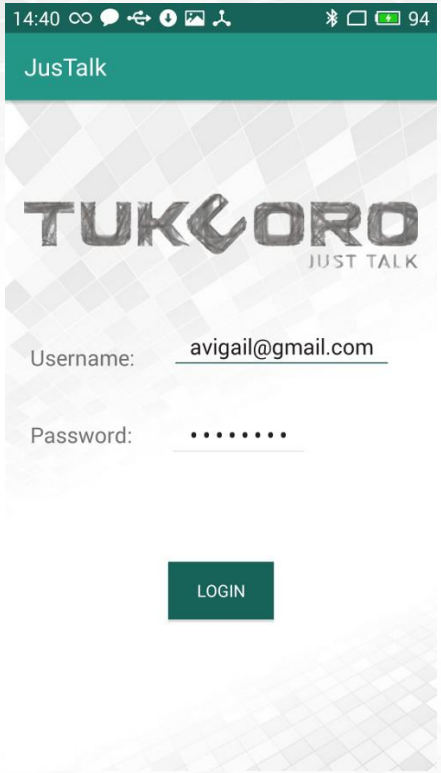
לאחר בחירת טופס מתוך הטפסים הממתינים לאישור הטופס ייפתח לעריכת המשתמש כאן יוכל לשמור אותו ולשלוח במועד הרצוי.

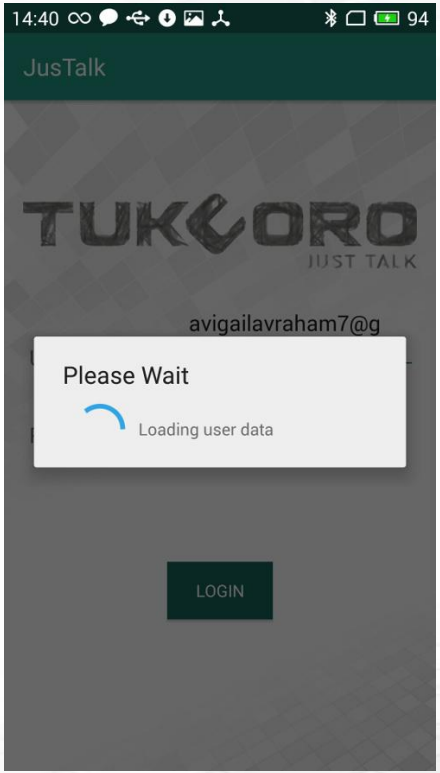
בנוסף...

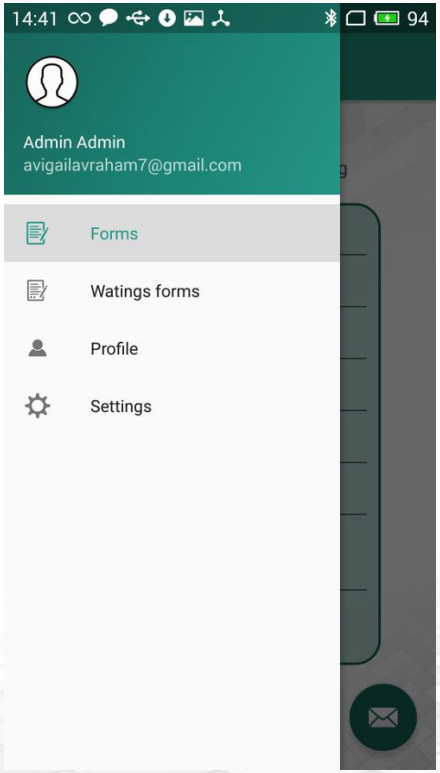
- **FormMessagesAdapter**
  אחראי על הצגת ה Messages בתהליך מילוי הטופס
- **MyBounceInterpolator**
  אחראי על אנימציית הכפתור

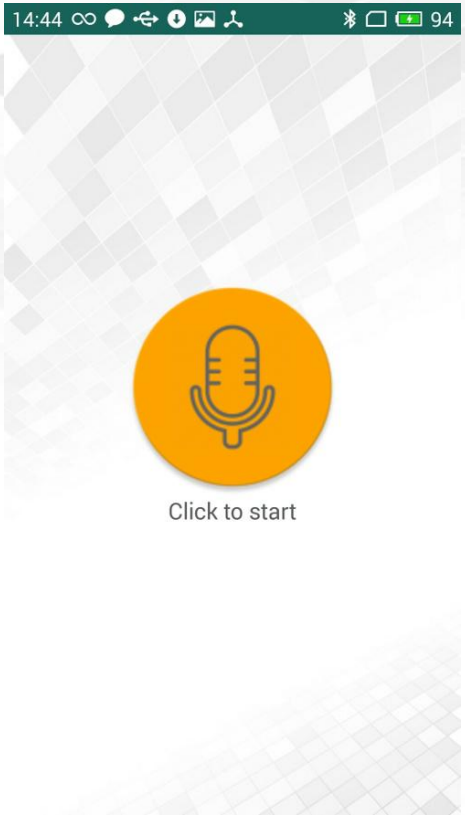
ScreenShots

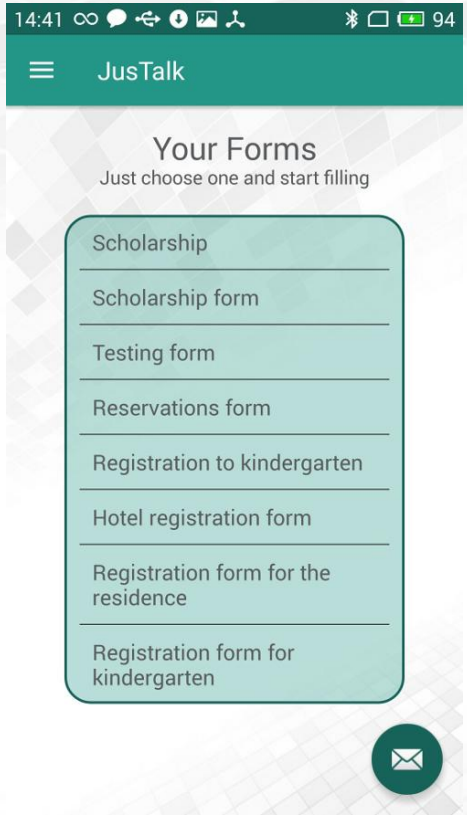


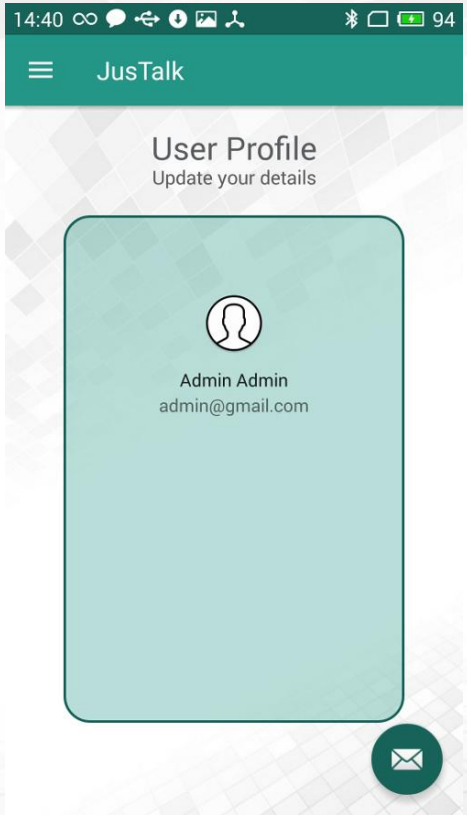


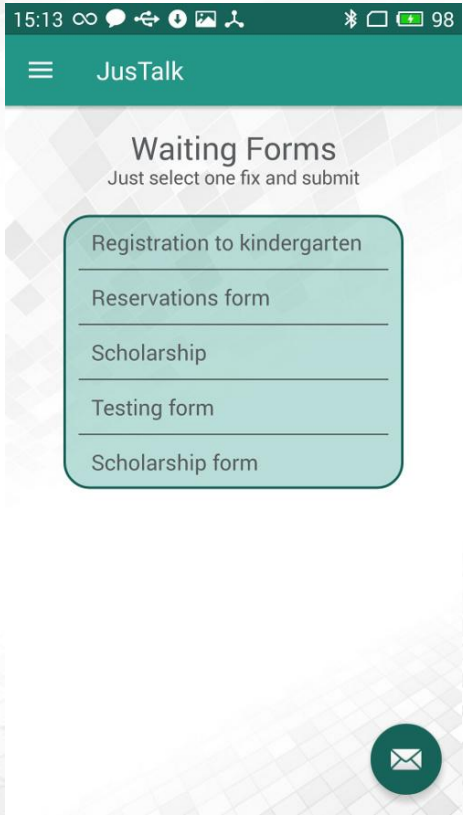


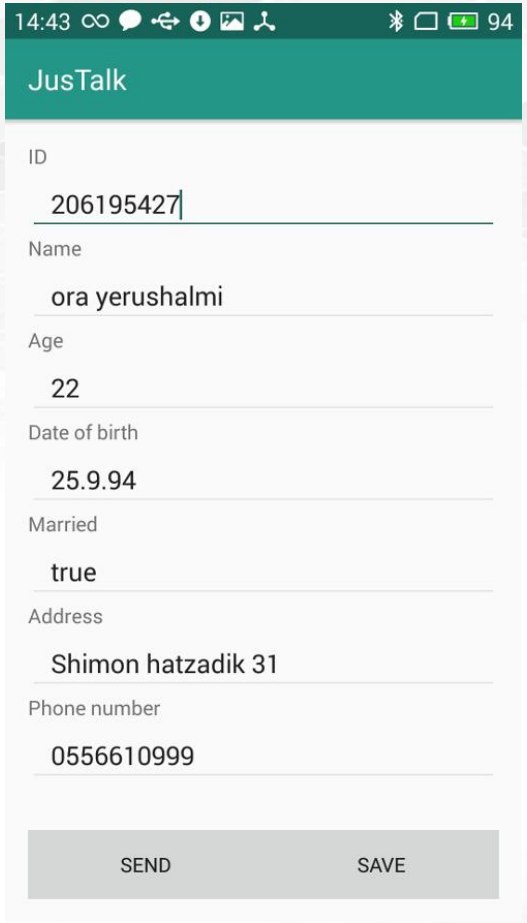



14:44
94
Click to start


14:41
94
JusTalk
Your Forms
Just choose one and start filling
Scholarship
Scholarship form
Testing form
Reservations form
Registration to kindergarten
Hotel registration form
Registration form for the residence
Registration form for kindergarten


14:40
94
JusTalk
User Profile
Update your details
Admin Admin
admin@gmail.com


15:13
98
JusTalk
Waiting Forms
Just select one fix and submit
Registration to kindergarten
Reservations form
Scholarship
Testing form
Scholarship form


14:43
94
JusTalk
ID
206195427
Name
ora yerushalmi
Age
22
Date of birth
25.9.94
Married
true
Address
Shimon hatzadik 31
Phone number
0556610999
SEND
SAVE


14:45
94
1223
Mar 31, 2017 14:44:47
Name
Mar 31, 2017 14:44:55
or
Garden number


15:00
96
Name
Mar 31, 2017 14:59:32
Ariella
Mar 31, 2017 14:59:32
A number of invited
Mar 31, 2017 14:59:44
2567
Mar 31, 2017 14:59:44
Number of rooms
SEND NOW
SAVE IN WAITIGS FORMS

קטעי קוד עיקריים

ניתוח הקול הנקלט

```
  private static final int RECORDER_SAMPLERATE = 16000;
  private static final int RECORDER_CHANNELS = AudioFormat.CHANNEL_IN_MONO;
  private static final int RECORDER_AUDIO_ENCODING = AudioFormat.ENCODING_PCM_16BIT
bufferSize = AudioRecord.getMinBufferSize(8000,
            AudioFormat.CHANNEL_CONFIGURATION_MONO,
            AudioFormat.ENCODING_PCM_16BIT);
public void startRecordAudio(){
  recorder = new AudioRecord(MediaRecorder.AudioSource.MIC,
        RECORDER_SAMPLERATE, RECORDER_CHANNELS,RECORDER_AUDIO_ENCODING, bufferSize);

  int i = recorder.getState();
  if(i==1)
     recorder.startRecording();
  writeAudioDataToFile();

}
 private void writeAudioDataToFile(){
      float tempFloatBuffer[] = new float[3];
      int tempIndex          = 0;
      byte data[] = new byte[bufferSize];
      String filename = getTempFilename();
      FileOutputStream os = null;
      try {
         os = new FileOutputStream(filename);
      } catch (FileNotFoundException e) {
// TODO Auto-generated catch block
         e.printStackTrace();
      }
```

יצירת אובייקט מסוג AudioRecord ואתחול סוג הערוץ, תדר הדגימה,פורמט קידוד וגודל הbuffer.

התחלת ההקלטה

יצירת קובץ לתוכו נקליט את הקול

```
int read = 0;
 int count=0;
 if(null != os){

 while(true) {
    float totalAbsValue = 0.0f;
    short sample        = 0;
    read = recorder.read(data, 0, bufferSize);
   // Analyze Sound.
  for( int i=0; i<bufferSize; i+=2 )
 {
    sample = (short)( (data[i]) | data[i + 1] << 8 );
    totalAbsValue += Math.abs( sample ) / (bufferSize/2);
 }
// Analyze temp buffer.
tempFloatBuffer[tempIndex % 3] = totalAbsValue;

float amplitude = 0.0f;
for (int i = 0; i < 3; ++i)
    amplitude += tempFloatBuffer[i];

//Silence detect
if ((amplitude >= 0 && amplitude <= SLINCE_RANGE) && isRecording == false) {
    Log.i("TAG", "1");
    if(currentTime==null) {
      currentTime = System.currentTimeMillis();
    }
   else{
       //Set seconds of silence
       distance = System.currentTimeMillis()-currentTime;

        //if there no word - add more time for silence
       if(count==0){
                 silenceLimit = FIRST_WORD_LIMIT_TIME ;
       }
```

קריאת המידע שנקלט ב recorder לתוך data- מערך של bytes

סכימת ערכי המידע הנקלט (בערך מוחלט)

בדיקה אם אמפליטודת הקול הינה בטווח נמוך וכן לא מתבצעת הקלטה

חישוב מספר המילישניות שזוהה בהן שקט

במקרה שהמשתמש לא אמר אף מילה עד כה - נגדיל את טווח השקט האפשרי

```
        //if the silence is too long -stop recording
        if(distance> silenceLimit){
            currentTime = null;
            stopRecording();
            Log.d("very long time","stop recording!!");
            break;
        }
    }
}
//Speech detect and we start recording
if (amplitude > SLINCE_RANGE && isRecording == false) {
    silenceLimit = silenceLimit*0.8 + (System.currentTimeMillis() - currentTime)*0.2;
    Log.i("TAG", "2");
    currentTime = System.currentTimeMillis();
     isRecording = true;
}
//Silence detect - after say word and maybe before say next word
 if ((amplitude >= 0 && amplitude <= SLINCE_RANGE) && isRecording == true) {
      count++;
      isRecording=false;
   }
    tempIndex++;
    }
 }
  try {
       os.close();
   } catch (IOException e) {
       e.printStackTrace();
   }
```

בדיקה אם אורך הזמן שזוהה בו שקט ארוך מזמן הגבלת השקט, עצירת ההקלטה

האמפליטודה גבוהה מספיק, ז"א שזוהה דיבור, התחלת ההקלטה

חישוב ערך הגבלת השקט תוך התחשבות בערך הקיים עד וכן התחשבות באורך השקט שבין מילה למילה שנאמרה.

ההקלטה עדיין מתבצעת והאמפליטודה נמצאת בטווח נמוך, ז"א שזוהה שקט, המשתמש סיים לומר מילה, עצירת ההקלטה.

שליחת קובץ ההקלטה

```
public class SendRecordSoap extends AsyncTask<Object, Object, Object> {
private static final String SOAP_ACTION = "http://www.tukuoro.com/ParseImmidiateSingleFromAudio";
private static final String METHOD_NAME = "ParseImmidiateSingleFromAudio";
private static final String NAMESPACE = "http://www.tukuoro.com/";
private static final String URL = "https://wili.tukuoro.com/tukwebservice/tukwebservice_app.asmx";
@Override
protected Object doInBackground(Object... arg0) {
    SoapObject request = new SoapObject(NAMESPACE, METHOD_NAME);
    //init property of request
    request.addProperty("audio",encodeAudio());
    request.addProperty("fieldName",this.fieldName.toLowerCase());
    request.addProperty("fieldType",this.fieldType);
    request.addProperty("language",this.language);
    request.addProperty("clientId","68174861");
    request.addProperty("serviceId","58469251");
    SoapSerializationEnvelope envelope = new SoapSerializationEnvelope(SoapEnvelope.VER11);
    envelope.dotNet=true;
    envelope.setOutputSoapObject(request);
    HttpTransportSE httpTransport = new HttpTransportSE(URL);
    httpTransport.debug = true;
```

אתחול הפרמטרים של השליחה

הוספת התכונות הרצויות

הכנסת החבילה שאותה רוצים לשלוח לתוך "מעטפת"- envelope

```
try {
        httpTransport.call(SOAP_ACTION, envelope);
    } catch (IOException e) {
        e.printStackTrace();
    } catch (XmlPullParserException e) {
        e.printStackTrace();
    } //send request
    SoapObject result;
    try {
        result = (SoapObject)envelope.getResponse();
        Log.d("App", "" + envelope.getResponse());
        return result;
    } catch (SoapFault e) {
        e.printStackTrace();
        Log.d("faild to send","!!");
    }
    Log.d("faild to send","!!");
    return null;
}
```

שליחת החבילה

קבלת תשובה ושליחתה

נראה סרטון קצר


07:43
99
DrHeartBr...
2048
2048
2048 x 360
Subway S...
JusTalk
Lep's Worl...
Louvre

תודות

למנחה האקדמי- מר אסף שפנייר

על הליווי המקצועי וההכוונה.

לאחראי התעשייתי- מר ניר פנחס

על כל העזרה.

לכל המעודדים והמסייעים!