Sofie Andersson, Qasim Ahmad, Simon Gullstrand, Gustav Frigren
2014-10-30

MALMÖ HÖGSKOLA

# Translate The Word

Sofie Andersson, Qasim Ahmad, Simon Gullstrand, Gustav Frigren
2014-10-30

MALMÖ HÖGSKOLA

# Projektarbete

## Sammanfattning av APIer och applikationens användning av dessa

Vår applikation är ett enklare spel som går ut på att gissa översättningen på ord, du får fyra olika alternativ och är översättningen rätt får du poäng. Svarar du fel avslutas omgången och ett resultat presenteras, du kan sedan välja att starta en ny omgång eller att dela resultatet på Facebook. För att det ska vara möjligt att dela med dina vänner måste du logga in med Facebook innan spelet startar.

### Facebook

Används för identifiering av spelaren så resultat kan sparas och delas med dina vänner. Inloggningen sker automatiskt efter en inloggning om Facebook applikationen är installerad på telefonen.

Med hjälp av dela funktionen i APIn så kan vi publicera inlägg på Facebook från applikationen, när någon sedan klickar på inlägget blir de omdirigerade till vår applikations sida. Det går även genom denna funktion att rekommendera eller skicka förfrågningar till dina vänner vilket gör applikationen mer synlig och ger den en större möjlighet att spridas.

Facebook erbjuder också ett enkelt sätt att implementera reklam i din applikation, reklamen anpassas och riktas in på din målgrupp.

### ThesHub

Är en multispråkig API som fungerar som en stor ordbok eller synonymordbok. Med hjälp av APIn kan du snabbt hämta ord, översätta dem, identifiera språk eller som vi, bygga något slags spel av det.

Vi använder APIn för att hämta ett slumpmässigt ord på svenska och dess översättning på engelska. Du får fyra slumpmässiga alternativ på översättningen men bara ett av dem är rätt och gissar du på det rätta får du poäng, när du svarar fel avslutas omgången och resultatet presenteras.

Sofie Andersson, Qasim Ahmad, Simon Gullstrand, Gustav Frigren
2014-10-30

MALMÖ HÖGSKOLA

# Projektmedlemmarnas insatser

## Sofie Andersson

Mitt viktigaste bidrag till projektet har varit skapandet av Controller-klassen och klassdiagrammet. Även dokumentationen är till stor del skriven av mig.
När klassdiagrammet skulle göras fick jag strukturera om i koden för att banta ned den och få bort onödiga relationer, jag skapade då Controllern och ordnade så att alla förfrågningar sker genom den klassen istället.

Utöver det här har jag också ansvarat för designen av GUIt, d.v.s. bakgrunder, texter, knappar, symboler och deras funktioner.

## Qasim Ahmad

Implementerat Facebook SDK och skrivit all kod för att kommunicera med Facebooks API. Så att man kan logga in på facebook och dela sitt resultat som man får i quizet på sin facebook-feed. La ner rätt mycket tid på att bara få in Facebooks SDK. Var mycket kod som skulle ändras i både Manifestet och build filerna för att det skulle funka.

Har också felsökt kod med Gustav för att i slutändan få ihop allt till en app.

Övrigt: Finslipat dokumentationen lite, suttit med MyActivity, konfigurerat på developer.facebook.com för permissions.

## Gustav Frigren

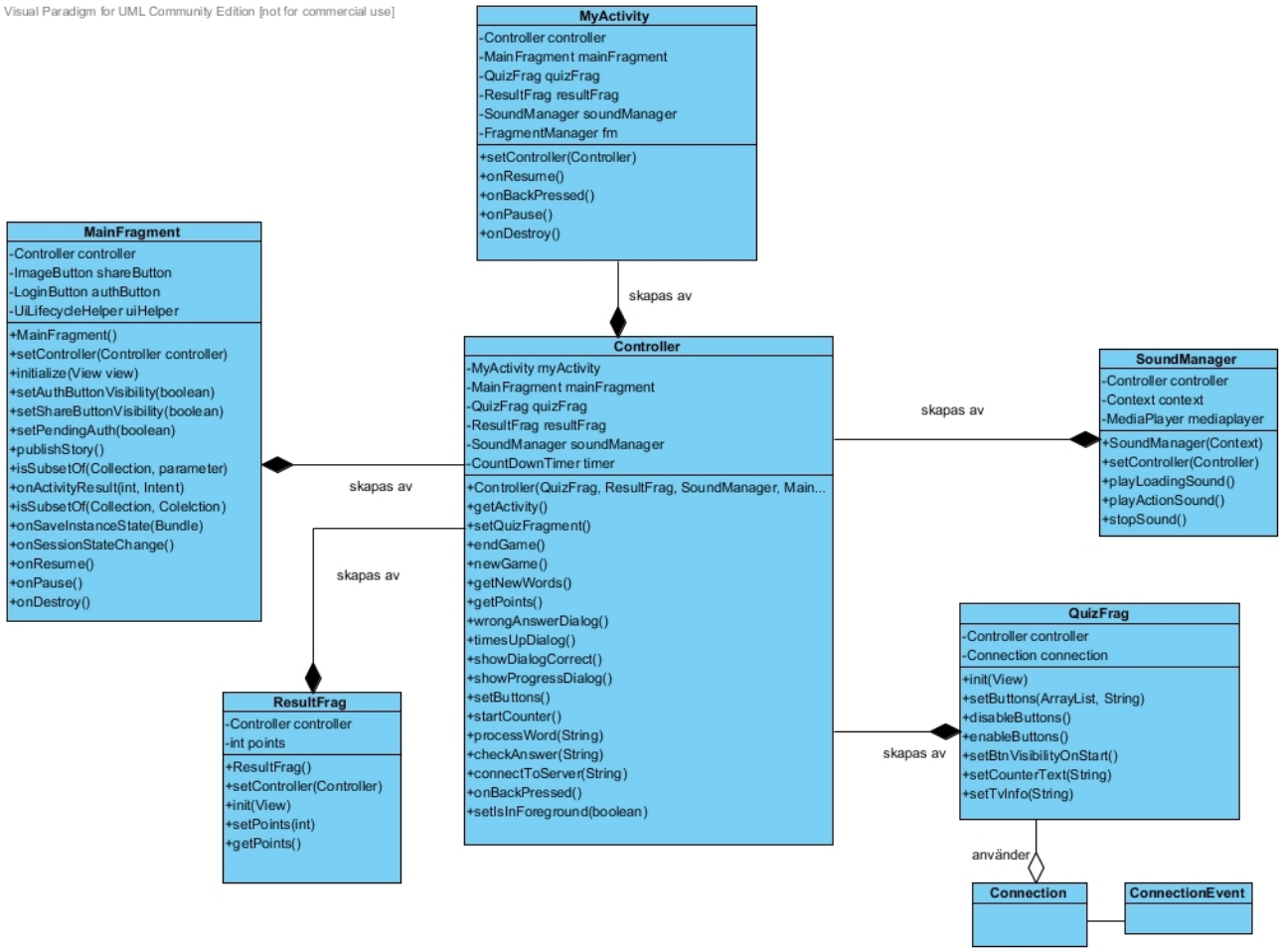
Har gjort största delen av själva logiken i spelet (hur och när appen ska använda API:t för att hämta slumpmässiga ord och översätta ord samt hur detta ska skrivas ut på knappar och i textfält). Skapade även anslutningen och hittade ett sätt att komma runt kravet på Unirest-API:t (som ThesHub normalt använder för kommunikation). Av någon anledning fungerade inte Unirest som vi önskade. Implementerade även ljuduppspelningen i appen.

## Simon Gullstrand

Började med att försöka få in facebook APIn I ett projekt, lyckades starta så jag kunde köra previews av vad APIn kan göra.
Hoppade sen över till att hjälpa Gustav med QuizAPIn och själva logiken bakom spelet. Gjorde tex en metod som formaterar svaret från APIn till en snyggare sträng.
Samt gjorde första steget i mergenings processen där vi satte ihop quizdelen med facebookdelen. Löste det genom att ha facebook delen i en activity runt en Container som vi lade quizet i som ett Fragment.

Sofie Andersson, Qasim Ahmad, Simon Gullstrand, Gustav Frigren
2014-10-30

MALMÖ HÖGSKOLA



Sofie Andersson, Qasim Ahmad, Simon Gullstrand, Gustav Frigren
2014-10-30

MALMÖ HÖGSKOLA

## Connection

```
/**
 * Created by Gustav Frigren on 2014-10-23.
 */
public class Connection extends AsyncTask<String, ConnectionEvent, Boolean> {

    private final String TAG = "CONNECTION";

    private String urlString;
    private int incomingID;
    private String wordToSend;

    public Connection(String urlString) {
        this.urlString = urlString;
        this.execute(urlString);
    }

    @Override
    protected Boolean doInBackground(String... params) {
        try {
            HttpClient client = new DefaultHttpClient();
            HttpGet get = new HttpGet(params[0]);
            get.setHeader("X-Mashape-Key",
"JjRKQLR6pDmshK0xPNDV2anSAINFp1azu2TjsnIcAegmY9ILXu");
            org.apache.http.HttpResponse responseGet = client.execute(get);
            HttpEntity resEntityGet = responseGet.getEntity();
            String res = EntityUtils.toString(resEntityGet);
            splitWord(res);
            wordToSend = wordToSend.replaceAll("\\s", "");
            Log.i("GET ", res);
            publishProgress(new ConnectionEvent(this, incomingID, wordToSend));

        } catch (Exception e) {
            e.printStackTrace();
            publishProgress(new ConnectionEvent(this,
ConnectionEvent.CONNECTION_FAILED, "No connection"));
        }
        return null;
    }

    private void splitWord(String str) {
        Log.d("TEST", str);
        String[] parts = str.split(":");
        String name = parts[0]; // 004
        Log.d("TEST", name);
        if (name.contains("translations")) {
            //cleaning up all symbols
            incomingID = ConnectionEvent.INCOMING_TRANSLATION;
            String print = parts[1];
            print = print.replace('"', ' ');
            print = print.replace('[', ' ');
            print = print.replace(']', ' ');
            print = print.replace('}', ' ');


            Log.d("TEST", print);
```

MALMÖ HÖGSKOLA

```
            //if more than one word choose word nr2 and print it
            if (print.contains(",")) {
                String[] printparts = print.split(",");
                print = printparts[1]; // 004
                Log.d("TEST", print);
                //makeing first letter big
                print = print.toLowerCase();
                wordToSend = Character.toString(print.charAt(1)).toUpperCase() +
print.substring(2);
                //if only one word print it
            } else {
                print = print.toLowerCase();
                wordToSend = Character.toString(print.charAt(1)).toUpperCase() +
print.substring(2);
            }

        } else if (name.contains("word")) {
            //cleaning up all symbols
            incomingID = ConnectionEvent.INCOMING_RANDOM;
            String print = parts[1];
            print = print.replace('"', ' ');
            print = print.replace('[', ' ');
            print = print.replace(']', ' ');
            print = print.replace('}', ' ');
            Log.d("TEST", print);
            //if more than one word choose word nr2 and print it
            if (print.contains(",")) {
                String[] printparts = print.split(",");
                print = printparts[1]; // 004
                Log.d("TEST", print);
                //makeing first letter big
                print = print.toLowerCase();
                wordToSend = Character.toString(print.charAt(1)).toUpperCase() +
print.substring(2);
                //if only one word print it
            } else {
                print = print.toLowerCase();
                wordToSend = Character.toString(print.charAt(1)).toUpperCase() +
print.substring(2);
            }
        }

    }

    protected void onProgressUpdate(ConnectionEvent... event) {
        fireMyEvent(event[0]);
    }

    protected void onPostExecute(Integer... integers) {
    }

    ConnectionListener listener;
    // Listener Interface
    public interface ConnectionListener extends EventListener {
        public void myEventOccurred(ConnectionEvent evt);
    }

    public void addEventListener(ConnectionListener listener) {
        this.listener=listener;
```

```
    }

    public void removeEventEventListener(ConnectionListener listener)
{
        listener=null;
    }

    public void removeEventEventListener() {
        listener=null;
    }

    void fireMyEvent(ConnectionEvent evt) {
        listener. myEventOccurred(evt);
    }

}
```

MALMÖ HÖGSKOLA

## ConnectionEvent

```
/**
 * Created by Gustav Frigren on 2014-10-23.
 */
public class ConnectionEvent extends EventObject {

    int type;
    String message;

    public static final int INCOMING_RANDOM=2;
    public static final int INCOMING_TRANSLATION=3;
    public static final int CONNECTION_FAILED=-1;

    public ConnectionEvent(Object source, int type) {
        super(source);
        this.type=type;
    }
    public ConnectionEvent(Object source, int type, String message) {
        super(source);
        this.type=type;
        this.message = message;
    }
    public int getType(){
        return type;
    }

}
```

Sofie Andersson, Qasim Ahmad, Simon Gullstrand, Gustav Frigren
2014-10-30

MALMÖ HÖGSKOLA

## Controller

```
/**
 * Created by Sofie on 2014-11-03.
 */
public class Controller {

    private Connection connection;
    private QuizFrag quizFrag;
    private ResultFrag resultFrag;
    private SoundManager soundManager;
    private MainFragment mainFragment;
    private MyActivity myActivity;
    private CountDownTimer timer;

    public int points;

    private Dialog d2;
    private ProgressDialog d;

    private ArrayList<String> wordList = new ArrayList<String>();
    private String wordToTranslate;
    private String translatedWord;

    public static Boolean doubleBackToExitPressedOnce = false;
    public static Boolean isInForeground = true;

    private final String TAG = "MYACTIVITY";

    private final String GET_RANDOM_WORD_ENGLISH =
"https://theshub.p.mashape.com/random?language=en";
    private final String GET_RANDOM_WORD_SWEDISH =
"https://theshub.p.mashape.com/random?language=sv";
```

```
    private final String TRANSLATE_SWEDISH_TO_ENGLISH =
"https://theshub.p.mashape.com/translate?from=sv&to=en&word=";


    private android.support.v4.app.FragmentManager fm;


    public Controller(QuizFrag quizFrag, ResultFrag resultFrag, SoundManager
soundManager, MainFragment mainFragment, MyActivity myActivity) {


        this.quizFrag = quizFrag;
        this.resultFrag = resultFrag;
        this.soundManager = soundManager;
        this.mainFragment = mainFragment;
        this.myActivity = myActivity;


        quizFrag.setController(this);
        resultFrag.setController(this);
        soundManager.setController(this);
        mainFragment.setController(this);
    }

    public MyActivity getActivity() {
        return myActivity;
    }



    public void setQuizFragment() {
        fm = myActivity.getSupportFragmentManager();
        fm.beginTransaction()
                .replace(R.id.framlayout, quizFrag)
                .commit();
    }

    public void endGame() {
        soundManager.stopSound();
```

```
        resultFrag.setPoints(points);
        mainFragment.setPoints(points);
        Log.d("POINTS CONTROLLER: ", "" + points);
        points = 0;
        fm = myActivity.getSupportFragmentManager();
        fm.beginTransaction()
                .replace(R.id.framlayout, resultFrag)
                .commit();
    }


    public void newGame() {
        soundManager.stopSound();
        soundManager.playLoadingSound(isInForeground);
        showProgressDialog();
        wordToTranslate = null;
        translatedWord = null;
        quizFrag.setBtnVisibilityOnStart();
        wordList.clear();
        connectToServer(GET_RANDOM_WORD_SWEDISH);
    }


    public void getNewWords() {
        soundManager.stopSound();
        soundManager.playLoadingSound(isInForeground);
        showProgressDialog();
        wordToTranslate = null;
        translatedWord = null;
        wordList.clear();
        connectToServer(GET_RANDOM_WORD_SWEDISH);

    }
```

```
public void wrongAnswerDialog() {

    final Dialog dAnswer = new Dialog(myActivity);
    dAnswer.setTitle("YOUR ANSWER IS WRONG!"); //TODO String
    dAnswer.setCanceledOnTouchOutside(false);

    dAnswer.show();
    CountDownTimer timerThree = new CountDownTimer(3000, 1000) {
        @Override
        public void onTick(long l) {

        }

        @Override
        public void onFinish() {
            dAnswer.dismiss();
            endGame();
        }
    };
    timerThree.start();

}

public void timesUpDialog() {

    final Dialog dTime = new Dialog(myActivity);
    dTime.setTitle("TIME'S UP!"); //TODO String
    dTime.setCanceledOnTouchOutside(false);

    dTime.show();
    CountDownTimer timerThree = new CountDownTimer(3000, 1000) {
```

```
            @Override
            public void onTick(long l) {


            }


            @Override
            public void onFinish() {
                dTime.dismiss();
                endGame();
            }
        };
        timerThree.start();
    }



    public void showDialogCorrect() {
        d2 = new Dialog(getActivity());
        d2.setTitle("CORRECT");  //TODO String
        d2.setCanceledOnTouchOutside(false);
        LayoutInflater factory = LayoutInflater.from(getActivity());
        View infoLayout = factory.inflate(R.layout.dialog, null);
        TextView tv = (TextView) infoLayout.findViewById(R.id.textViewCorrect);
        tv.setText("YOUR ANSWER IS CORRECT");
        tv.setTextColor(Color.WHITE);
        d2.getWindow().setBackgroundDrawable(new
ColorDrawable(Color.TRANSPARENT));


        d2.setContentView(infoLayout);
        d2.show();
        CountDownTimer timerTwo = new CountDownTimer(3000, 1000) {
            @Override
            public void onTick(long l) {
            }
```

```
        @Override
        public void onFinish() {
            d2.dismiss();
            getNewWords();
        }
    };
    timerTwo.start();
}


public void showProgressDialog() {
    d = new ProgressDialog(myActivity);
    d.setMessage("LOADING"); //TODO String
    d.setProgressStyle(ProgressDialog.STYLE_HORIZONTAL);
    d.setCanceledOnTouchOutside(false);
    d.setCancelable(false);

    d.show();
    final int progressTime = 100;
    final Thread t = new Thread() {
        public void run() {
            int jumpTime = 0;
            while (jumpTime < progressTime) {
                if (translatedWord != null) {
                    jumpTime = 25;
                }
                if (wordList.size() == 1) {
                    jumpTime = 50;
                }
                if (wordList.size() == 2) {
                    jumpTime = 75;
                }
                if (wordList.size() == 3) {
```

```
                jumpTime = 100;
            }
            d.setProgress(jumpTime);
        }
    }
};
t.start();
}


public void setButtons() {
    soundManager.stopSound();
    soundManager.playActionSound(true);
    d.dismiss();
    startCounter();
    Log.i(TAG, "SETBUTTONS");

    wordList.add(translatedWord);
    Collections.shuffle(wordList);
    quizFrag.setButtons(wordList, wordToTranslate);
}


private void startCounter() {
    timer = new CountDownTimer(10000, 1000) {
        @Override
        public void onTick(long l) {
            String str = ("" + l / 1000);
            quizFrag.setCounterText(str);
        }

        @Override
        public void onFinish() {
```

```
            String str = "TIMES UP!";
            quizFrag.setCounterText(str);
            timesUpDialog();
        }
    };
    timer.start();
}

private void processWord(String word) {
    Log.i(TAG, "PROCESSWORD: " + word + "...");
    if (wordToTranslate == null) {
        Log.i(TAG, "SETWORDTOTRANSLATE");
        wordToTranslate = word;
        connectToServer(TRANSLATE_SWEDISH_TO_ENGLISH + word);
    } else {
        Log.i(TAG, "PROCESSWORD ELSE: " + word + "...");
        if (wordList.size() < 3) {
            Log.i(TAG, "ADDTOWORDLIST: " + word);
            wordList.add(word);
            if (wordList.size() < 3) {
                Log.i(TAG, "IF WLIST < 3 2");
                connectToServer(GET_RANDOM_WORD_ENGLISH);
            } else {
                setButtons();
            }
        }
    }
}

public void checkAnswer(String answer) {
    timer.cancel();
    quizFrag.disableButtons();
    if (answer.equals(translatedWord)) {
```

```
            points += 1;
            quizFrag.setTvInfo("YOUR ANSWER IS CORRECT!");
            showDialogCorrect();
        } else {
            wrongAnswerDialog();
        }

    }

    public void connectToServer(String str) {
        if (connection != null) {
            connection = null;
        }
        connection = new Connection(str);
        connection.addEventListener(new Connection.ConnectionListener() {
            public void myEventOccurred(ConnectionEvent evt) {
                switch (evt.type) {
                    case ConnectionEvent.INCOMING_RANDOM:
                        Log.i(TAG, "SWITCH INCOMING RANDOM");
                        processWord(evt.message);
                        break;
                    case ConnectionEvent.INCOMING_TRANSLATION:
                        Log.i(TAG, "SWITCH INCOMING TRANSLATION");
                        if (evt.message.equals("")) {
                            wordToTranslate = null;
                            connectToServer(GET_RANDOM_WORD_SWEDISH);
                        } else {
                            translatedWord = evt.message;
                            Log.i(TAG, "SWITCH TRANSLATEDWORD: " + translatedWord);
                            connectToServer(GET_RANDOM_WORD_ENGLISH);
                        }
                        break;
                    case ConnectionEvent.CONNECTION_FAILED:
```

```
                    Toast.makeText(myActivity, "CONNECTION FAILED, TRY AGAIN!", Toast.LENGTH_LONG).show();
                    break;
                default:
                    break;
            }
        }
    });
}


public void onBackPressed() {
    if (doubleBackToExitPressedOnce) {
        soundManager.stopSound();
        System.exit(0);
        return;
    }
    doubleBackToExitPressedOnce = true;
    Toast.makeText(myActivity, "PRESS BACK AGAIN TO EXIT", Toast.LENGTH_SHORT).show();
    new Handler().postDelayed(new Runnable() {
        @Override
        public void run() {
            doubleBackToExitPressedOnce = false;
        }
    }, 2000);
}


public void setIsInForeground(boolean b) {
    isInForeground = b;
}
```

Sofie Andersson, Qasim Ahmad, Simon Gullstrand, Gustav Frigren
2014-10-30

}

MALMÖ HÖGSKOLA

MALMÖ HÖGSKOLA

MainFragment

```
/**
 * created by Qasim Ahmad 2014-10-22
 */
public class MainFragment extends Fragment {
    private static final String TAG = "MainFragment";
    private UiLifecycleHelper uiHelper;
    public static ImageButton shareButton;
    public static LoginButton authButton;

    Controller controller;



    private static final List<String> PERMISSIONS = Arrays.asList("publish_actions");
    private static final String PENDING_PUBLISH_KEY =
"pendingPublishReauthorization";
    private static final String FACEBOOK_APPID = "PUT YOUR FACEBOOK APPID HERE";

    private boolean pendingPublishReauthorization = false;

    Bundle postParams;

    private Session.StatusCallback callback = new Session.StatusCallback() {
        @Override
        public void call(Session session, SessionState state, Exception exception) {
            onSessionStateChange(session, state, exception);
        }
    };



    public MainFragment() {
        // Required empty public constructor
    }
```

```
public void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);

    uiHelper = new UiLifecycleHelper(getActivity(), callback);
    uiHelper.onCreate(savedInstanceState);
}

@Override
public View onCreateView(LayoutInflater inflater, ViewGroup container,
                         Bundle savedInstanceState) {
    // Inflate the layout for this fragment
    View view = inflater.inflate(R.layout.activity_my, container, false);

    authButton = (LoginButton) view.findViewById(R.id.authButton);

    authButton.setFragment(this);



    FrameLayout frameLayout = (FrameLayout)view.findViewById(R.id.framlayout);


    shareButton = (ImageButton) view.findViewById(R.id.shareButton);
    shareButton.setVisibility(View.INVISIBLE);
    shareButton.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View v) {

            Session session = Session.getActiveSession();
            if(session.isOpened()) {
                publishStory();
```

```
                Toast.makeText(getActivity(), "RESULTATET ÄR DELAT",
Toast.LENGTH_LONG).show();
            } else {
                Toast.makeText(getActivity(), "INTE INLOGGAD",
Toast.LENGTH_LONG).show();
            }

        }
    });

    if (savedInstanceState != null) {
        pendingPublishReauthorization =
                savedInstanceState.getBoolean(PENDING_PUBLISH_KEY, false);
    }

    postParams = new Bundle();

    return view;
}


public void onResume() {
    super.onResume();
    AppEventsLogger.activateApp(getActivity());
    Session session = Session.getActiveSession();
    if (session != null &&
            (session.isOpened() || session.isClosed()) ) {
        onSessionStateChange(session, session.getState(), null);
    }
    //  uiHelper.onResume();
}


@Override
```

```
    public void onActivityResult(int requestCode, int resultCode, Intent data)
{
        super.onActivityResult(requestCode, resultCode, data);
        uiHelper.onActivityResult(requestCode, resultCode, data);
    }

    @Override
    public void onPause() {
        super.onPause();
        AppEventsLogger.deactivateApp(getActivity());

        uiHelper.onPause();
    }

    @Override
    public void onDestroy() {
        super.onDestroy();
        uiHelper.onDestroy();
    }

    public void onSaveInstanceState(Bundle outState) {
        super.onSaveInstanceState(outState);
        outState.putBoolean(PENDING_PUBLISH_KEY, pendingPublishReauthorization);
        uiHelper.onSaveInstanceState(outState);
    }

    private void onSessionStateChange(Session session, SessionState state, Exception
exception) {
        if (state.isOpened()) {
            if (pendingPublishReauthorization &&
                    state.equals(SessionState.OPENED_TOKEN_UPDATED)) {
                pendingPublishReauthorization = false;
                Log.d("TEST", "ONSESSIONSTATE..");
                shareButton.setVisibility(View.VISIBLE);
```

```
                publishStory();


            }
//            shareButton.setVisibility(View.VISIBLE);
            Log.i(TAG, "Logged in...");
        } else if (state.isClosed()) {

            shareButton.setVisibility(View.INVISIBLE);
            Log.i(TAG, "Logged out...");
        }
    }

    private boolean isSubsetOf(Collection<String> subset, Collection<String>
superset) {
        for (String string : subset) {
            if (!superset.contains(string)) {
                return false;
            }
        }
        return true;
    }

    private void publishStory() {
        Session session = Session.getActiveSession();

        if (session != null){

            // Check for publish permissions
            List<String> permissions = session.getPermissions();

            if (!isSubsetOf(PERMISSIONS, permissions)) {
                pendingPublishReauthorization = true;
                Session.NewPermissionsRequest newPermissionsRequest = new Session
```

```
                .NewPermissionsRequest(getActivity(), PERMISSIONS);

session.requestNewPublishPermissions(newPermissionsRequest);
            return;
        }




        Log.d("TESTING", String.valueOf(ResultFrag.points));

        postParams.putString("name", "Translate The Word \n" +
                "\n POINTS: " + ResultFrag.points );

        postParams.putString("caption","");
        postParams.putString("description","Simon, Gustav, Qasim, Sofie");
        postParams.putString("picture", "http://i62.tinypic.com/2lihjqs.jpg");

        Request.Callback callback= new Request.Callback() {
            public void onCompleted(Response response) {
                JSONObject graphResponse = response
                        .getGraphObject()
                        .getInnerJSONObject();
                String postId = null;
                try {
                    postId = graphResponse.getString("id");
                } catch (JSONException e) {
                    Log.i(TAG,
                            "JSON error "+ e.getMessage());
                }
                FacebookRequestError error = response.getError();
                if (error != null) {
                    Toast.makeText(getActivity()
```

```
                                .getApplicationContext(),
                        error.getErrorMessage(),
                        Toast.LENGTH_SHORT).show();
                } else {

                }
            }
        };

        Request request = new Request(session, "me/feed", postParams,
                HttpMethod.POST, callback);

        RequestAsyncTask task = new RequestAsyncTask(request);
        task.execute();
    }

}


public void setController(Controller controller) {
    this.controller = controller;
}


}
```

Sofie Andersson, Qasim Ahmad, Simon Gullstrand, Gustav Frigren
2014-10-30

MALMÖ HÖGSKOLA

## MyActivity

```
/**
 * Created by Qasim Ahmad
 */

public class MyActivity extends FragmentActivity {

    private Controller controller;

    private QuizFrag quizFrag;
    private ResultFrag resultFrag;
    private SoundManager soundManager;
    private MainFragment mainFragment;

    private android.support.v4.app.FragmentManager fm;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.fragment_start_up);
        String id = getResources().getString(R.string.facebook_app_id);

        if (savedInstanceState == null) {
            // Add the fragment on initial activity setup
            mainFragment = new MainFragment();
            fm = getSupportFragmentManager();
                    fm.beginTransaction()
                    .add(android.R.id.content, mainFragment)
                    .commit();
        } else {
            // Or set the fragment from restored state info
            mainFragment = (MainFragment) fm
                    .findFragmentById(android.R.id.content);
        }

        quizFrag = new QuizFrag();
        resultFrag = new ResultFrag();
        soundManager = new SoundManager(this);
        mainFragment = new MainFragment();

        controller = new Controller(quizFrag, resultFrag, soundManager,
mainFragment, this);

        setController(controller);
        controller.setQuizFragment();
    }

    public void setController (Controller controller) {
        this.controller = controller;
    }


    @Override
    protected void onResume() {
        super.onResume();
        controller.setIsInForeground(true);
    }
```

```
    @Override
    public void onBackPressed() {
        controller.onBackPressed();
    }

    @Override
    protected void onPause() {
        super.onPause();
        soundManager.stopSound();
        controller.setIsInForeground(false);
    }


    @Override
    protected void onDestroy() {
        controller.setIsInForeground(false);
        soundManager.stopSound();
        super.onDestroy();
    }


}
```

Sofie Andersson, Qasim Ahmad, Simon Gullstrand, Gustav Frigren
2014-10-30

MALMÖ HÖGSKOLA

## QuizFrag

```
/**
 * Created by Simon Gullstrand and Gustav Frigren
 */
public class QuizFrag extends Fragment {

    private Controller controller;

    private TextView tvInfo;
    private TextView tvCounter;

    private Button btnOne;
    private Button btnTwo;
    private Button btnThree;
    private Button btnFour;
    private Button btnNext;

    public QuizFrag(){
    }

    public void setController (Controller controller) {
        this.controller = controller;
    }

    @Override
    public View onCreateView(LayoutInflater inflater, ViewGroup container,
                             Bundle savedInstanceState) {
        // Inflate the layout for this fragment
        View view = inflater.inflate(R.layout.fragment_quiz, container, false);
        init(view);
        return view;
    }

    private void init(View view) {
        tvInfo = (TextView)view.findViewById(R.id.textViewInfo);
        btnNext = (Button)view.findViewById(R.id.buttonNext);
        btnNext.setOnClickListener(listener);
        tvCounter = (TextView)view.findViewById(R.id.textViewCounter);

        btnOne = (Button)view.findViewById(R.id.buttonOne);
        btnTwo = (Button)view.findViewById(R.id.buttonTwo);
        btnThree = (Button)view.findViewById(R.id.buttonThree);
        btnFour = (Button)view.findViewById(R.id.buttonFour);

        btnOne.setOnClickListener(listener);
        btnTwo.setOnClickListener(listener);
        btnThree.setOnClickListener(listener);
        btnFour.setOnClickListener(listener);

        disableButtons();
    }
```

```
public void setButtons(ArrayList<String> wordList, String
wordToTranslate) {
        btnOne.setText(wordList.get(0).toString());
        btnTwo.setText(wordList.get(1).toString());
        btnThree.setText(wordList.get(2).toString());
        btnFour.setText(wordList.get(3).toString());
        tvInfo.setText("TRANSLATE: " + wordToTranslate);
        enableButtons();
    }

    public void disableButtons() {
        btnOne.setEnabled(false);
        btnTwo.setEnabled(false);
        btnThree.setEnabled(false);
        btnFour.setEnabled(false);
    }

    public void enableButtons() {
        btnOne.setEnabled(true);
        btnTwo.setEnabled(true);
        btnThree.setEnabled(true);
        btnFour.setEnabled(true);
    }

    public void setBtnVisibilityOnStart() {
        btnNext.setVisibility(View.INVISIBLE);
        btnNext.setEnabled(false);
    }

    public void setCounterText(String str) {
        tvCounter.setText(str);
    }

    public void setTvInfo(String str){
        tvInfo.setText(str);
    }

    View.OnClickListener listener = new View.OnClickListener() {
        @Override
        public void onClick(View v) {
            if(v.getId() == R.id.buttonNext) {
                controller.newGame();
            } if(v.getId() == R.id.buttonOne) {
                String str = btnOne.getText().toString();
                controller.checkAnswer(str);
            } if(v.getId() == R.id.buttonTwo) {
                String str = btnTwo.getText().toString();
                controller.checkAnswer(str);
            } if(v.getId() == R.id.buttonThree) {
                String str = btnThree.getText().toString();
                controller.checkAnswer(str);
            } if(v.getId() == R.id.buttonFour) {
                String str = btnFour.getText().toString();
                controller.checkAnswer(str);
            }
        }
    };
}
```

MALMÖ HÖGSKOLA

## ResultFrag

```
/*
 * Created by Simon Gullstrand
 */
public class ResultFrag extends Fragment {

    private Controller controller;
    int points;

    public ResultFrag() {
        // Required empty public constructor
    }

    public void setController (Controller controller) {
        this.controller = controller;
    }

    @Override
    public View onCreateView(LayoutInflater inflater, ViewGroup container,
                             Bundle savedInstanceState) {
        View view = inflater.inflate(R.layout.fragment_result, container,
false);
        init(view);
        return view;
    }

    private void init(View view) {
        TextView res = (TextView)view.findViewById(R.id.resTV);
        res.setText(String.valueOf(points));

        controller.setShareButtonVisibility(true);
        controller.setAuthButtonVisibility(true);

        Button newGame = (Button)view.findViewById(R.id.btnewGame);
        newGame.setOnClickListener(new Listener());
    }

    public void setPoints(int points) {
        this.points = points;
    }

    public int getPoints() {
        return points;
    }

    private class Listener implements View.OnClickListener {
        @Override
        public void onClick(View view) {
            controller.setQuizFragment();
            controller.setAuthButtonVisibility(false);
            controller.setShareButtonVisibility(false);
        }
    }
}
```

## SoundManager

Sofie Andersson, Qasim Ahmad, Simon Gullstrand, Gustav Frigren
2014-10-30

```
/**
 * Created by Gustav Frigren on 2014-10-31.
 */
public class SoundManager {

    private Controller controller;
    private Context context;
    private MediaPlayer mediaPlayer;

    public SoundManager(Context context) {
        this.context = context;
    }

    public void setController (Controller controller) {
        this.controller = controller;
    }

    public void playLoadingSound(Boolean isInForeground) {
        if(isInForeground = true) {
            mediaPlayer = MediaPlayer.create(context,
R.raw.bensoundtheelevatorbossanova);
            mediaPlayer.start();
        }
    }

    public void playActionSound(Boolean isInForeground) {
        if(isInForeground == true) {
            mediaPlayer = MediaPlayer.create(context,
R.raw.bensoundextremeaction);
            mediaPlayer.start();
        }
    }

    public void stopSound() {
        if(mediaPlayer != null) {
            mediaPlayer.release();
            mediaPlayer = null;
        }
    }
}
```