Programmierung mobiler Geräte

SoSe 2015

Native Entwicklung mit Android

Multimedia: Video und Sprache

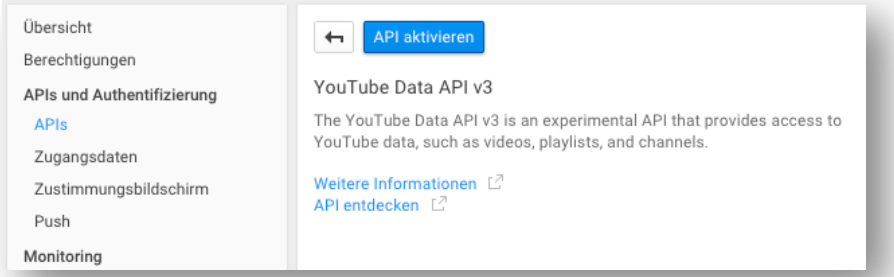
Markus Berg
Hochschule Wismar
Fakultät für Ingenieurwissenschaften
Bereich Elektrotechnik und Informatik
http://mmberg.net

Rückblick

- Letzte Woche: Internetzugriff
  - Permissions
  - REST
  - AsyncTask
- Heute:
  - Multimedia
    - YouTube
    - Spracherkennung

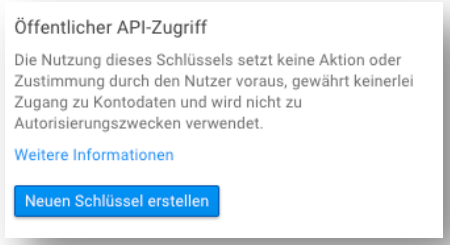
YouTube: Zugriff aktivieren

- API aktivieren und API Key beantragen (Google Account notwendig)
  - Siehe: https://developers.google.com/youtube/android/player/register
  - In *Google Developer Console* unter *APIs und Authentifizierung* YouTube Data API v3 aktivieren



  - Dann unter Zugangsdaten *Android-Schlüssel* erstellen
    - Hierzu wird lokaler Keystore benötigt

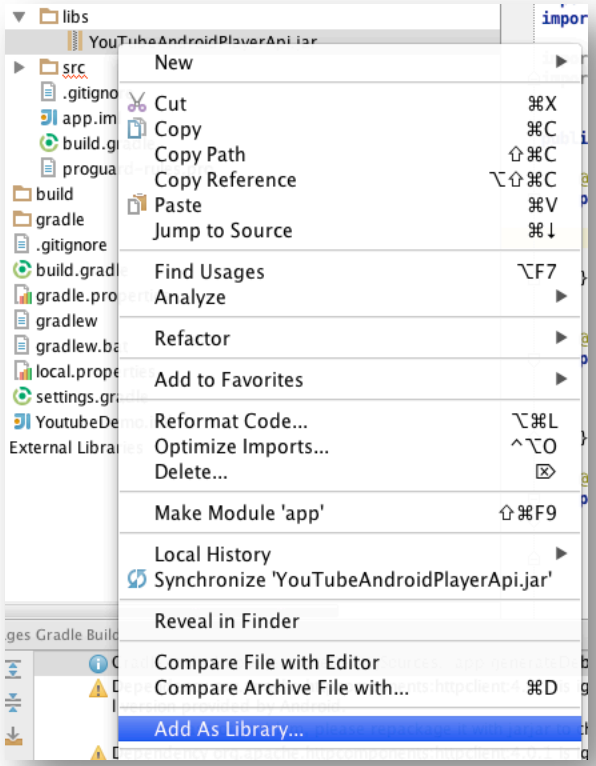
```
cd ~/.android/debug.keystore
keytool -exportcert -alias androiddebugkey -keystore
   ./debug.keystore -list -v
(Password: android)
```



    - SHA1 Fingerprint kopieren

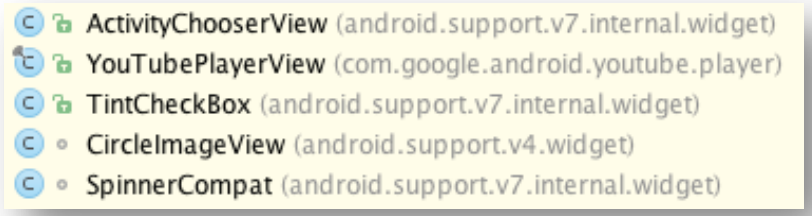
YouTube: Vorbereitungen

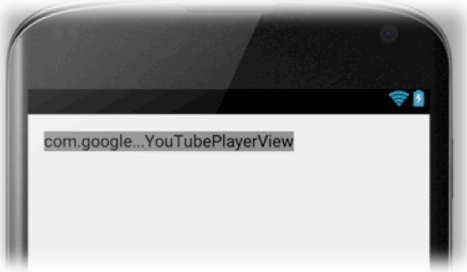
- Player API runterladen: https://developers.google.com/youtube/android/player/downloads/
- Bibliothek in den libs-Ordner des Projektes kopieren
  - Dem Projekt hinzufügen per rechter Maustaste: „Add as library“
- Im Manifest Internet-Permission hinzufügen



Layout anpassen

- Dem Layout eine *YouTubePlayerView* hinzufügen
  - Dazu im grafischen Editor *CustomView* anklicken
  - *YouTubePlayerView* auswählen





  - Rendering-Probleme ignorieren ;-)
  - Manuell ID hinzufügen im XML

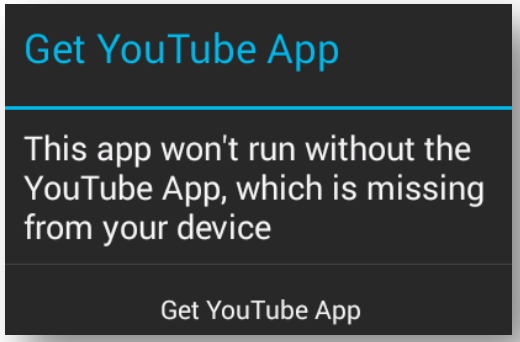
Activity anpassen

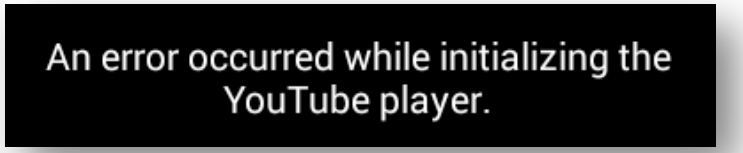
- Ableiten von `YouTubeBaseActivity`
- String-Konstante mit API-Key hinzufügen
- Youtube-View per `findViewById` ermitteln
- Youtube-View initialisieren über `initialize()` und Inline-Deklaration eines Listeners:
  - Videos werden über IDs geladen (ersichtlich in URLs, z.B. https://www.youtube.com/watch?v=***Hg212A_aPJ4***)

```
youtubeview.initialize(KEY,new YouTubePlayer.OnInitializedListener() {
            @Override
            public void onInitializationSuccess(YouTubePlayer.Provider provider,
                      YouTubePlayer youTubePlayer, boolean b) {
                youTubePlayer.cueVideo("Hg212A_aPJ4");

            }

            //...
 });
```

Ausführen

- YouTube App muss installiert sein
  - Sonst:

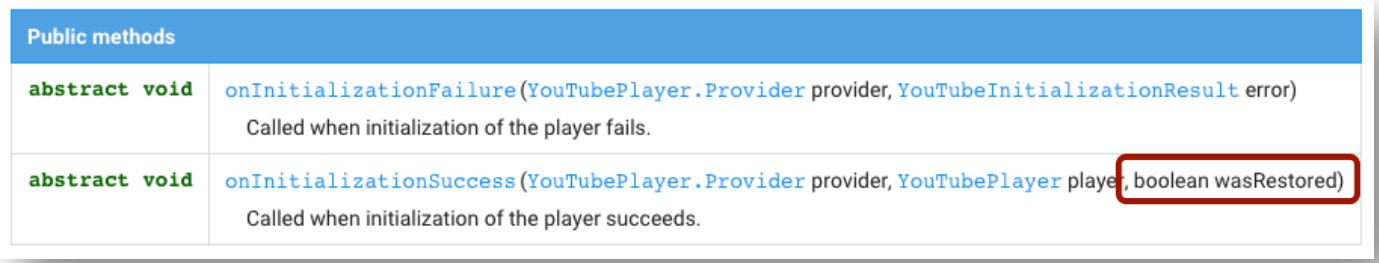




  - Am besten mit echtem Gerät (statt Emulator) testen
    - Installation von Apps im Emulator nicht direkt unterstützt und nur über Umwege möglich

Activity anpassen

- Fehlermeldung ergänzen und erneutes Laden verhindern bei Restoration (z.B. Wechsel in Vollbildmodus)



Public methods	
abstract void	onInitializationFailure(YouTubePlayer.Provider provider, YouTubeInitializationResult error) Called when initialization of the player fails.
abstract void	onInitializationSuccess(YouTubePlayer.Provider provider, YouTubePlayer player, boolean wasRestored) Called when initialization of the player succeeds.

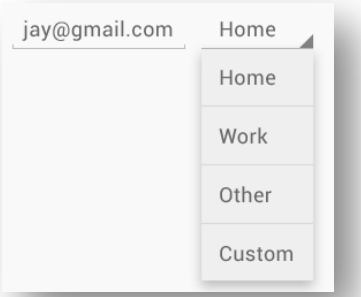
```
youtubeview.initialize(key,new YouTubePlayer.OnInitializedListener() {
     @Override
     public void onInitializationSuccess(YouTubePlayer.Provider provider,
      YouTubePlayer youTubePlayer, boolean b) {
            if(!b){
                  youTubePlayer.cueVideo("Hg212A_aPJ4");
             }
     }

     @Override
     public void onInitializationFailure(YouTubePlayer.Provider provider,
      YouTubeInitializationResult youTubeInitializationResult) {
         Toast.makeText(getApplicationContext(), "Fehler beim Initialisieren",Toast.LENGTH_SHORT);
     }
});
```

Video auswählen

- Zwei Activities:
  - Erste Activity: Dropdownmenü (Spinner) zur Auswahl des Videos und Button zum Wechseln in die Video-Activity hinzufügen
    - Liste mit Videos vorhalten (Beispiele)
  - Zweite Activity: Video anzeigen
  - Kommunikation per Intent (Übermittlung der ausgewählten ID)

Video auswählen: Spinner

- Ein Spinner ist vergleichbar mit einem Dropdownmenü
  - http://developer.android.com/guide/topics/ui/controls/spinner.htm



- Daten per Adapter binden
  - Liste mit Video IDs

```
private String[] videos = {"Hg212A_aPJ4","dQ2ufIOm7u0"};
```

  - ArrayAdapter
    - Zwei Layout-Ressourcen: Anzeige des ausgewählten Elements + Anzeige der Optionen

```
spinnerview = (Spinner)findViewById(R.id.spinner);
ArrayAdapter<String> adapter = new ArrayAdapter<>(this,android.R.layout.simple_spinner_item, videos);
adapter.setDropDownViewResource(android.R.layout.simple_spinner_dropdown_item);
spinnerview.setAdapter(adapter);
```

Video auswählen: Intent

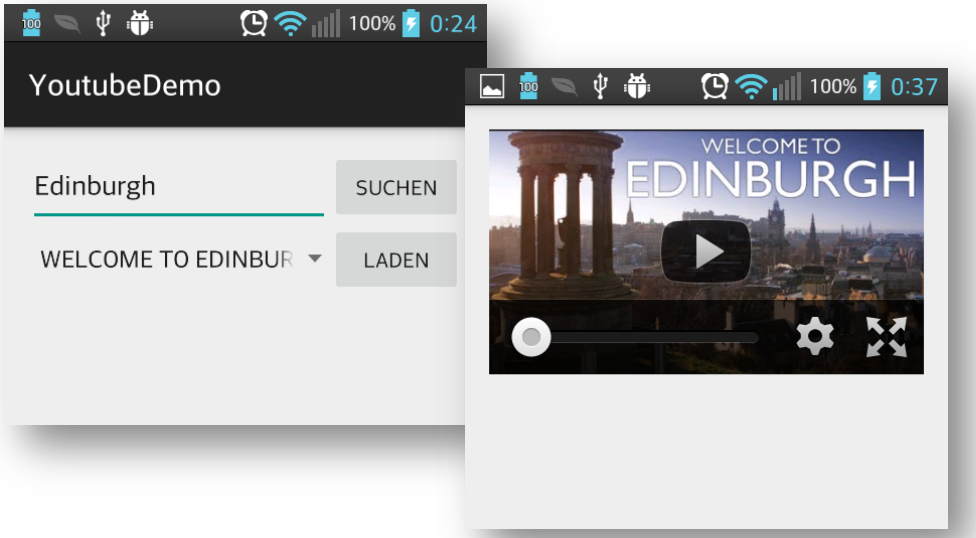
- Bei Klick auf Button
  - per Intent Video-Activity starten und Video-ID übermitteln (anhand aktuell ausgewähltem Spinner-Eintrag)

```
Button button = (Button) findViewById(R.id.button);
button.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {

        Intent startYoutubeIntent = new Intent(MainActivity.this,YouTubeActivity.class);
        startYoutubeIntent.putExtra(YouTubeActivity.YOUTUBE_VIDEO_ID,
                videos[spinnerview.getSelectedItemPosition()]);
        startActivity(startYoutubeIntent);
    }
});
```

Video suchen

- Liste der statischen Videos ersetzen durch Suchergebnisse
  - Eingabefeld + Button zum Starten der Suche
  - Durchführen einer YouTube-Suche
  - Anzeigen der Ergebnisse im Spinner
  - Auswählen eines Ergebnisses und Bestätigung per Button führt zu zweiter Activity, die das Video anzeigt

Im Designer...



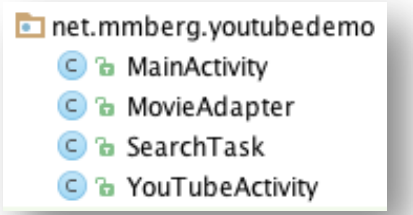
Video suchen: Vorbereitung

- Über die *YouTube Data API Client Library*
  - https://developers.google.com/api-client-library/java/apis/youtube/v3
  - Dem Projekt die Dependency '***com.google.apis:google-api-services-youtube:v3-rev137-1.20.0***' hinzufügen oder alternativ Bibliothek manuell runterladen
- Zur Nutzung einen *Browser-Key* generieren
  - Mit dem bisherigen Android-Key geht's nicht und später käme folgender Fehler:


```
{
"code" : 403,
"errors" : [ {
"domain" : "usageLimits",
"message" : "There is a per-IP or per-Referer restriction configured on your API key
"reason" : "ipRefererBlocked",
"extendedHelp" : "https://console.developers.google.com"
} ],
```

Video suchen

- Die API greift über HTTP-Request auf YouTube zu
  - Daher auch der Browser-Key
  - Funktioniert auch aus „herkömmlichem“ Java (nicht Android-spezifisch) bzw. auch im Browser (GET)
    - Rückgabewert: JSON → wird von API geparsed
  - Netzwerkzugriffe nicht im UI-Thread durchführen!
    - AsyncTask nutzen
  - 4 Klassen:
    - Activity für Eingabe des Suchbegriffs und Ausgabe der Ergebnisse
    - Eigener Adapter für Binding der Suchergebnisse an den Spinner
    - Suche in separatem Thread
    - Activity zur Anzeige des Videos



Video suchen: Nutzung der API

- SearchTask
  - ... ist ein AsyncTask
  - Schritt 1: Initialisieren der API

```
@Override
protected List<SearchResult> doInBackground(String... params) {

    List<SearchResult> videos=new ArrayList<>();
    try{
        //init YouTube:
        YouTube youtube = new YouTube.Builder(new NetHttpTransport(),
                new JacksonFactory(), new HttpRequestInitializer() {
            @Override
            public void initialize(HttpRequest httpreq) throws IOException {}
        }).setApplicationName("myapp").build();

       //create query:
       //...
    }
    catch(IOException e){
        Log.d(TAG, "Could not initialize: "+e);
    }

    return videos;
}
```

Video suchen: Nutzung der API

- Schritt 2: Query absetzen:

```
@Override
    protected List<SearchResult> doInBackground(String... params) {

            //...

            //create query:
            YouTube.Search.List query = youtube.search().list("id,snippet");
            query.setKey(key);
            query.setType("video");
            query.setFields("items(id/videoId,snippet/title)");
            query.setQ(params[0]);
            SearchListResponse response=query.execute();
            videos = response.getItems();
        }

        catch(IOException e){
            Log.d("YC", "Could not initialize: "+e);
        }

        return videos;
    }
```

Sucheergebnisse mit folgenden Attributen (parts) → Ressourcen sparen

Nur Videos suchen

Ergebnisse noch fein-granularer weiter einschränken (fields)
→ Ein Part hat mehrere Fields

Setzen des Suchbegriffs. Hier: Erster Parameter

Resultiert in GET-Request:

https://www.googleapis.com/youtube/v3/search?key=API_KEY&fields=items(id(videoId),snippet(title))&part=id,snippet&type=video&q=SEARCH_KEYWORD

Mehr Infos: https://developers.google.com/youtube/v3/getting-started

Video suchen: MovieAdapter

- Eigener Adapter zur Bindung der Suchergebnisse an den Spinner
  - Ableiten von BaseAdapter und Implementieren der üblichen Methoden (siehe vorletzte Vorlesung)
  - Konstruktor:
    - Merken der Ressourcen zur Erzeugung der Views sowie Kontext und Liste der anzuzeigenden Suchergebnisse

```
public MovieAdapter(Context context, int spinnerResource, int dropdownResource, List<SearchResult> results) {
    this.context=context;
    this.results=results;
    this.spinnerResource=spinnerResource;
    this.dropdownResource=dropdownResource;
}
```

  - Überschreiben von:
    - getView:

```
@Override
public View getView(int position, View convertView, ViewGroup parent) {
    if(convertView==null){
        LayoutInflater inflater = (LayoutInflater)context.getSystemService(context.LAYOUT_INFLATER_SERVICE);
        convertView = inflater.inflate(spinnerResource, parent, false);
    }

    ((TextView)convertView).setText(results.get(position).getSnippet().getTitle());

    return convertView;
}
```

    - getDropdownView
      - Wie `getView()`, allerdings mit `dropdownResource`

Video suchen: MainActivity

- Adapter instanziieren:
  - Übergeben des Kontexts
  - Übergeben der Ressourcen-IDs
  - Übergeben der Suchergebnisvariable (`List<SearchResult>`)
    - Anfangs leer

```
List<SearchResult> results = new ArrayList<>();
```

```
MovieAdapter adapter = new MovieAdapter(this, android.R.layout.simple_spinner_item, android.R.layout.simple_spinner_dropdown_item, results);
spinnerview.setAdapter(adapter);
```

- 2 Buttons:
  - Durchführen der Suche
    - Und aktualisieren der `results`-Variable
  - Laden der YouTubeActivity

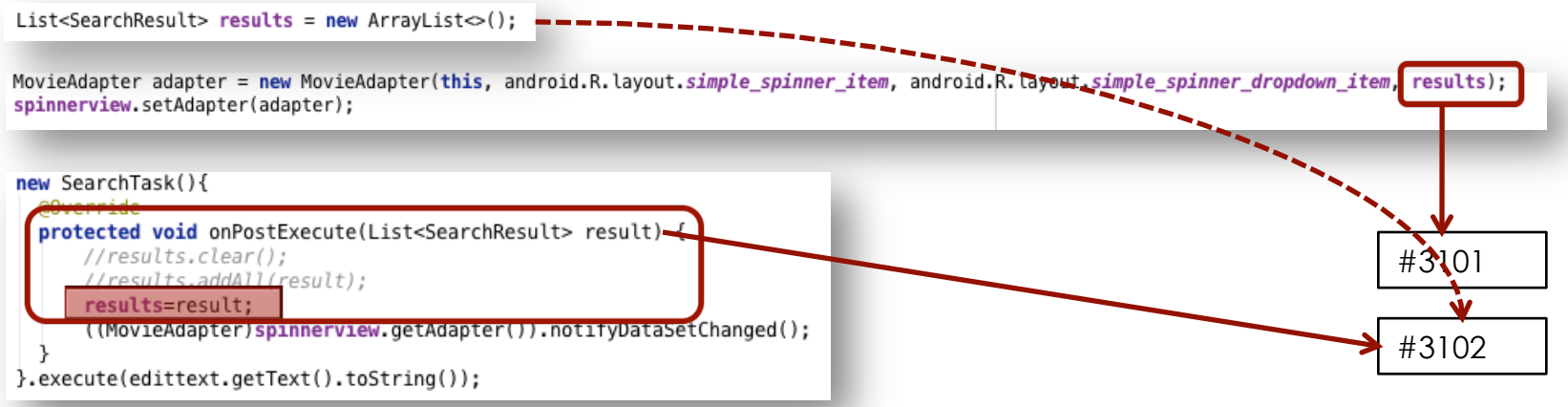
MainActivity: Starten der Suche

- Starten der Suche, d.h. starten des AsyncTasks
- In Ergebnishandler:
  - Variable mit den Suchergebnissen aktualisieren
  - Über `notifyDataSetChanged()` mitteilen, dass sich die Daten geändert haben

```
Button button2 = (Button) findViewById(R.id.button2);
button2.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {
        EditText edittext = (EditText) findViewById(R.id.editText);
        new SearchTask(){
          @Override
          protected void onPostExecute(List<SearchResult> result) {
              results.clear();
              results.addAll(result);
              ((MovieAdapter)spinnerview.getAdapter()).notifyDataSetChanged();
          }
        }.execute(edittext.getText().toString());
    }
});
```

Denkaufgabe

- Warum funktioniert folgender Code nicht?



Achtung: Nicht einfach „Pointer umbiegen“! Unbedingt die ursprünglich dem Adapter übergebene Variable nutzen, da der Speicherbereich, der dem Adapter bekannt ist, sonst nur die alten Daten beinhaltet.

MainActivity: Laden des Videos

- Bei Klick Intent auslösen
  - Als Extra die YouTube-ID des aktuell ausgewählten Videos übergeben

```
Button button = (Button) findViewById(R.id.button);
button.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {

        Intent startYoutubeIntent = new Intent(MainActivity.this,YouTubeActivity.class);
        int selected=spinnerview.getSelectedItemPosition();
        startYoutubeIntent.putExtra(YouTubeActivity.YOUTUBE_VIDEO_ID,results.get(selected).getId().getVideoId());
        startActivity(startYoutubeIntent);
    }
});
```

  - Die YouTubeActivity haben wir bereits erstellt
  - Fertig! → Ausprobieren

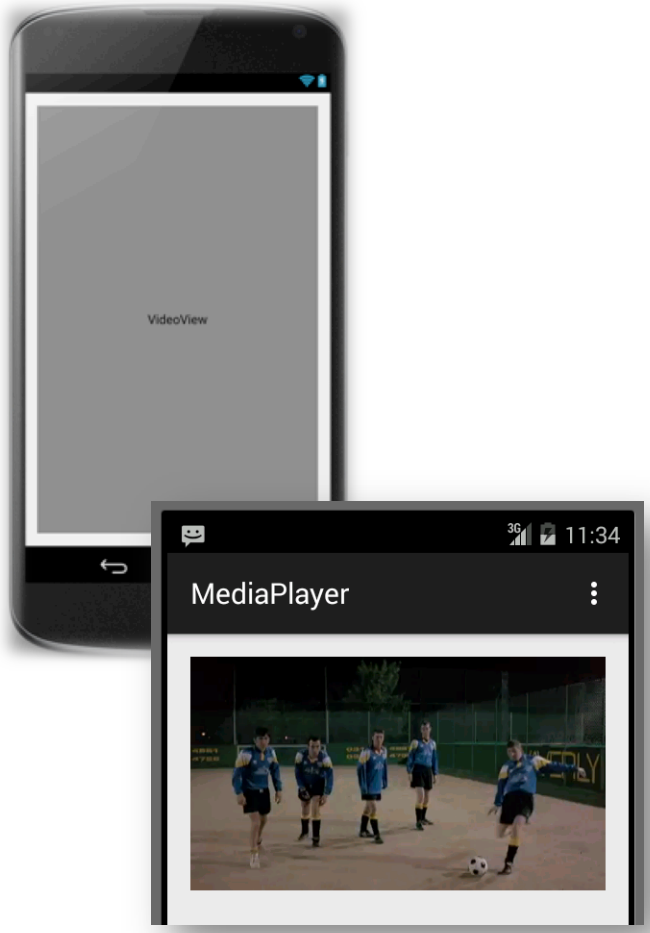
Alternative: VideoView

- Falls es sich um keine YouTube-Videos handelt, man keinen API-Key beantragen möchte oder man im Emulator Videos darstellen möchte: *Video View* benutzen
- Video View kapselt Media Player und ist besonders leicht zu handhaben
- Das größte Problem ist es Videos außerhalb von YouTube zu finden ;-)
  - Und zwar als mp4 o.ä. und nicht embedded in Flash etc.

VideoView: Kurzanleitung

- Video View hinzufügen
- Internet Permission hinzufügen
- URI setzen
  - z.B. http://progressive.totaleclips.com.edgesuite.net/981/e98140_257.mp4?eclipId=e98140&bitrateId=461&vendorId=102&type=.mp4
  - *oder:* http://download.wavetlan.com/SVV/Media/HTTP/H264/Talkinghead_Media/H264_test1_Talkinghead_mp4_480x360.mp4
- Controls hinzufügen
- Starten

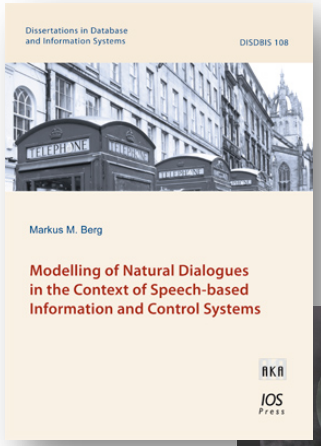
```
VideoView videoView = (VideoView) findViewById(R.id.videoView);
Uri videoUri = Uri.parse("http://progressive.totaleclips.com.edgesuite.
videoView.setVideoURI(videoUri);

MediaController videoControl = new MediaController(this);
videoControl.setAnchorView(videoView);
videoView.setMediaController(videoControl);

videoView.start();
```



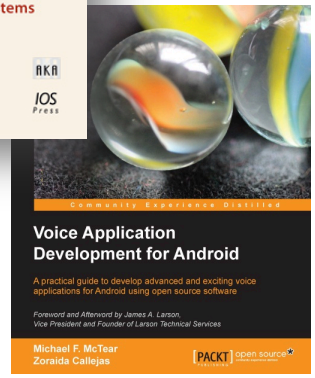
Ausschnitt aus „Trainspotting“

Sprachverarbeitung

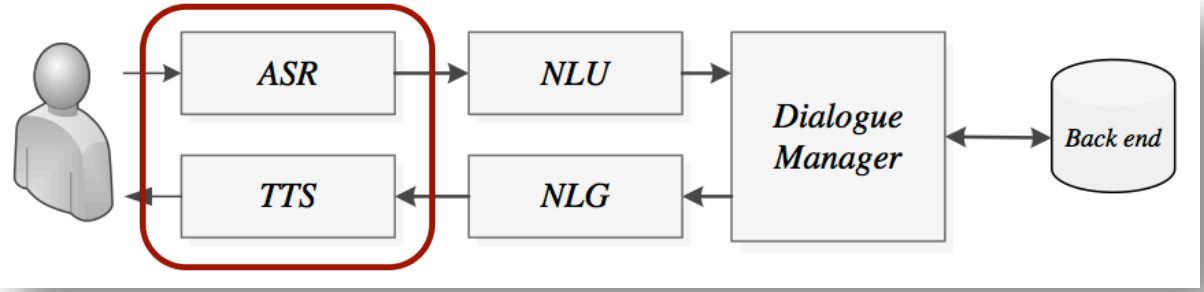
- Hier nur ein sehr kurzer Überblick
- Mehr zur Theorie von Dialogsystemen im Modul „*Human Language Technology*“
- Lesetipps:
  - *Markus M. Berg*
    „Modelling of Natural Dialogues in the Context of Speech-based Information and Control Systems“
  - *ISBN: 978-3-89838-508-4*
  - *http://mmberg.net/go/phd (PDF)*
    bzw. http://www.iospress.nl/book/modelling-of-natural-dialogues-in-the-context-of-speech-based-information-and-control-systems/ (print)



  - *Michael F. McTear, Zoraida Callejas*
    „Voice Application Development for Android“
  - *https://www.packtpub.com/application-development/voice-application-development-android*



Dialogsystem: Ein Überblick



Markus Berg, 2014: Modelling of Natural Dialogues in the Context of Speech-based Information and Control Systems

- Wir realisieren an dieser Stelle nur zwei sehr rudimentäre Beispiele für TTS (Text-to-Speech) und ASR (Automatic Speech Recognition)
- Beide Themengebiete haben in den letzten Jahrzehnten enorme Verbesserungen erfahren und es bis zur Produktreife geschafft
- Der heutige Fokus liegt in der Wissenschaft vor allem auf dem Sprachverstehen, dem Dialogmanagement und der Sprachgenerierung

Dialogsystem: Ein Beispiel

- Dialogmanagement
  - Entscheidung welche Informationen das System zur Erfüllung einer Aufgabe noch benötigt und in welcher Reihenfolge sie erfragt werden
- Sprachgenerierung
  - Erzeugung der Sätze in einer Form, die der Situation und dem Nutzer entspricht
- Sprachverstehen
  - Nicht zu verwechseln mit Spracherkennung
  - Inhaltliches Verstehen der Aussage des Nutzers und Ableiten der gewünschten Aktion
  - Erkennen von Subdialogen/ Nachfragen etc.

S: *How may I help you?*
U: *I'd like to book a trip*
S: *Where do you want to start?*
U: *In Edinburgh*
S: *And where do you want to go?*
U: *How is the weather in Inverness?*
S: *20°C. And what is your destination?*
U: *Umm, how long do I need from Edinburgh to Inverness?*
S: *2:56 hrs. And where do you want to go?*
U: *Ok, to Inverness then.*
S: *When do you want to depart?*
U: *I'd like to travel from 01/08/2013 until 14/08/2013.*
S: *And with how many persons?*
U: *3*
S: *The cheapest flight is 500 Euro.*

Markus Berg, 2014: Modelling of Natural Dialogues in the Context of Speech-based Information and Control Systems, p. 192

Dialogsystem:

- Sprachverarbeitung ist mehr als nur das Überführen von Audiosignalen in Text
- Intention des Nutzers muss verstanden werden
  - Semantik
    - *„Ich möchte mit meinen beiden Kindern und meiner Frau über Pfingsten ins Warme reisen“*
  - Pragmatik
    - *„Kannst du bitte das Licht ausmachen?“*
      - *Ja**
      - [Licht ausschalten]
  - Überbeantwortung
    - *S: „Wann möchten Sie starten?“*
    - *U: „Morgen nach Köln“*
  - Subdialoge
    - *S: „Wohin möchten Sie reisen?“*
    - *U: „Wie ist denn das Wetter in Paris?“*
  - ...

Sprachausgabe: Theorie

- Menschliche Samples (concatenative synthesis)
  - Aufgezeichnete Phrasen, Wörter, Silben, Phoneme,..
  - Unit Selection
    - Größte passende Einheit wird ausgewählt
  - Diphonsynthese
    - Nur Lautübergänge
- Künstliche Generierung
  - Formantsynthese
    - Erzeugt Laute auf Grundlage eines Akustikmodells
    - Geringe Dateigröße da komplett berechnet ohne Datenbank mit Samples

Android-Beispiel: Sprachausgabe

- Quick and dirty:

```
tts = new TextToSpeech(this, new TextToSpeech.OnInitListener() {
    @Override
    public void onInit(int status) {
        tts.setLanguage(Locale.ENGLISH);
        tts.speak("Hello students, I hope you are still awake!",TextToSpeech.QUEUE_ADD,null);
    }
```

Wichtig, sonst „leaked Service Connection“-Exception!

```
@Override
protected void onDestroy(){
    super.onDestroy();
    tts_active=false;
    if (tts!=null) tts.shutdown();
    tts=null;
}
```

- Sauberer:

```
private TextToSpeech tts;
private static final int TTS_DATA_CHECK=1;
private static final String TAG = "net.mmberg.log";

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

    //Testen ob TTS Ressourcen vorhanden, Ergebnis in onActivityResultHandler
    Intent checkIntent = new Intent();
    checkIntent.setAction(TextToSpeech.Engine.ACTION_CHECK_TTS_DATA);
    startActivityForResult(checkIntent, TTS_DATA_CHECK);
}

@Override
protected void onActivityResult(int requestCode, int resultCode, Intent data) {

    if (requestCode == TTS_DATA_CHECK && resultCode == TextToSpeech.Engine.CHECK_VOICE_DATA_PASS) {
        Log.i(TAG, "TTS available");
        tts = new TextToSpeech(this, new TextToSpeech.OnInitListener() {
            @Override
            public void onInit(int status) {
                if (status == TextToSpeech.SUCCESS) {
                    Log.i(TAG, "TTS init successful");
                    if(tts.isLanguageAvailable(Locale.ENGLISH)==TextToSpeech.LANG_AVAILABLE) {
                        Log.i(TAG, "Language available");
                        tts.setLanguage(Locale.ENGLISH);
                        tts.speak("Hello students, I hope you are still awake!", TextToSpeech.QUEUE_ADD, null);
                    }
                    else Toast.makeText(getApplicationContext(),"Sprache nicht verfügbar",Toast.LENGTH_SHORT);
                }
            }
        });
    }
    else if  (requestCode == TTS_DATA_CHECK && resultCode == TextToSpeech.Engine.CHECK_VOICE_DATA_FAIL) {
        Toast.makeText(getApplicationContext(),"TTS nicht verfügbar",Toast.LENGTH_SHORT);
    }
}
```

Sprachausgabe

- Convenience-Methode speak()

```
if(tts.isLanguageAvailable(Locale.ENGLISH)==TextToSpeech.LANG_AVAILABLE) {
    Log.i(TAG, "Language available");
    tts.setLanguage(Locale.ENGLISH);
    tts_active=true;
}
```

```
public void speak(String text){
    if (tts_active) tts.speak(text, TextToSpeech.QUEUE_ADD, null);
}
```

Spracherkennung: Theorie

- Akustikmodell
  - Hidden-Markov-Modell: Hypothesen welchem Buchstaben der vom Sprecher geäußerte Laut entsprechen könnte
  - Daraus entsteht Hypothese über gesamtes Wort (mehrere aufeinanderfolgende Laute)
- Sprachmodell
  - Welche Wörter können aufeinander folgen (deklarativ) bzw. welche Wörter folgen häufig aufeinander (Statistik)
  - „I want **to** buy **two** apples“ (to vs. two)
  - Verfahren:
    - Grammatikbasiert (deklarativ)
      - Oft sehr restriktiv, nahezu perfekte Erkennung
      - Problem: out-of-vocabulary-Fehler
    - Free-Form / statistische Sprachmodelle (n-Gramm)
      - Google besitzt riesige Datenmengen und wertet diese im Hinblick auf Sprachmodelle aus (auch völlig „unbekannte“ Wörter wie „Wonnemar“ oder „FC Anker Wismar“ werden erkannt)
      - Funktioniert erstaunlich gut!

Android-Beispiel: Spracherkennung

- Achtung: Nicht im Emulator unterstützt!
- Wir nutzen ein Free-Form-Modell
  - Ohne Grammatik
  - Versteht nicht nur Englisch, sondern auch Deutsch
  - Formatiert Ergebnisse korrekt: Pro 7 ist nicht Pro sieben

```
public void asr(View v){
    Intent intent = new Intent(RecognizerIntent.ACTION_RECOGNIZE_SPEECH);
    intent.putExtra(RecognizerIntent.EXTRA_LANGUAGE_MODEL, RecognizerIntent.LANGUAGE_MODEL_FREE_FORM);
    startActivityForResult(intent, SPEECH_REQUEST_CODE);
}
```

```
@Override
protected void onActivityResult(int requestCode, int resultCode, Intent data) {

    //TTS
    //...

    //ASR
    if (requestCode == SPEECH_REQUEST_CODE && resultCode == RESULT_OK) {
        List<String> results = data.getStringArrayListExtra(RecognizerIntent.EXTRA_RESULTS);
        for(String result : results){
            Log.i(TAG,"Result: "+result);
        }
        speak(results.get(0));
    }
}
```

Spracherkennung: Einstellungen

- Optionale Parameter (dem Intent hinzufügen)
  - Anzahl der Ergebnisse (N-best): EXTRA_MAX_RESULTS
  - Text, der zur Spracheingabe auffordert: EXTRA_PROMPT
  - Sprache: EXTRA_LANGUAGE
- In der Regel werden mehrere Ergebnisse mit verschiedenen Konfidenzwerten sortiert zurückgegeben
  - Gibt Erkennungswahrscheinlichkeit an (mit welcher Wahrscheinlichkeit handelt es sich bei den vom Nutzer geäußerten Lauten um das erkannte Wort?)
  - In onActivityResult-Eventhandler auswerten:

```
List<String> results =
 data.getStringArrayListExtra(RecognizerIntent.EXTRA_RESULTS);
//...
List<String> confidence=
 data.getStringArrayListExtra(RecognizerIntent.
      EXTRA_CONFIDENCE_SCORES);
```

Spracherkennung: Ergebnisse auswerten

- Sobald Sprache erkannt wurde, soll eine bestimmte Aktion ausgeführt werden
  - z.B. Schalten einer Lampe, Durchführen einer Websuche, Starten einer App, Laden einer Activity,...
  - Hierzu muss die Nutzeraussage „verstanden“ werden
  - Einfachster Ansatz: Keywords
    - „Wohin möchten Sie reisen?“
      - „Ich will nach Madrid.“
      - „Ich möchte nach Rom reisen.“
      - „Nach Paris soll‘s gehen.“
    - Realisierung: z.B. Java-Methode `contains()`
      - Prüfung ob erkannter Text ein bestimmtes Wort enthält

Kleine Demo

- Aufbauend auf der REST-Übung zur Anzeige der Zitate
- Vorlesen eines Zitats eines bestimmten Autors nachdem dieser per Sprachbefehl genannt wurde („Command and Control“ → nur Nennung des Namens; keine ganzen Sätze → Abhilfe: `contains()`)

```
@Override
protected void onActivityResult(int requestCode, int resultCode, Intent data) {

    //ASR
    if (requestCode == SPEECH_REQUEST_CODE && resultCode == RESULT_OK) {
        List<String> results = data.getStringArrayListExtra(RecognizerIntent.EXTRA_RESULTS);
        for(String result : results){
            Log.i(TAG,"Result: "+result);

            //process result
            Quote resultQuote = findQuoteByAuthor(result);

            if(resultQuote!=null) {
                speak(resultQuote.getAuthor() + ": "+resultQuote.getQuote());

                Intent detailsIntent = new Intent(MainActivity.this, DetailsActivity.class);
                detailsIntent.putExtra(QUOTE, resultQuote.getQuote());
                startActivity(detailsIntent);

                break;
            }
        }

    }
}

private Quote findQuoteByAuthor(String author){
    for(int i=0; i<quotesList.size();i++){
        Quote q=quotesList.get(i);
        if (q.getAuthor().equalsIgnoreCase(author)){
            return q;
        }
    }
    return null;
}
```

Praktikum

- Die App aus dem letzten Praktikum soll erweitert werden
  - Bei Klick auf eine Sendung soll ein (passender) Trailer von YouTube abgespielt werden
    - Suche anhand des Sendungstitels durchführen und erstbestes Video auf Detailactivity (wo bisher nur Sender und Uhrzeit stehen) anzeigen
  - Per Sprachbefehl sollen die Details der Sendung eines bestimmten Senders angezeigt werden
    - *„Pro 7“ bzw. „Was läuft auf Pro 7?“*
- Deadline: 23.06.2015