CS 193A

Text-to-speech and Speech-to-text

Text-to-Speech

- **text-to-speech**: Allows Android device to speak an audible message based on a text string.
  - Not installed by default on some devices. To install, click Settings → Language and Input → Text to speech output → Google text-to-speech engine "settings" icon → Install voice data → Languages

- In general, text-to-speech on Android is simple:
  - create a `TextToSpeech` object
  - call its `speak` method

- But there are a few details that require us to discuss some advanced features of Java.

TextToSpeech class

- `TextToSpeech` constructor and methods:
  - `new TextToSpeech(activity, listener)` - constructor
  - `getVoice`, `getVoices`, `setVoice` - change speaking voice
  - `getLanguage`, `setLanguage` - sets language used
  - `getPitch`, `setPitch` - sets vocal tone used
  - `isSpeaking` - returns true if speaking
  - `shutdown` - kills TTS engine
  - `speak(text, mode, params)` - speaks given text aloud
  - `stop` - halts any speech
  - `synthesizeToFile(text, params, filename)` - speaks to file

Initialization and listener

- The `TextToSpeech` service can take a while to initialize.
  - So its constructor forces you to pass a listener object.
  - The listener will be notified when the TTS service is done loading.
  - This helps keep the main UI from freezing up during TTS load time.

  *(The code below uses a Java **anonymous inner class**.)*

```
TextToSpeech tts = new TextToSpeech(this,
        new TextToSpeech.OnInitListener() {
    @Override
    public void onInit(int status) {
        // code to run when done loading
    }
});
```

Waiting for initialization

- You must wait until the text-to-speech listener's `onInit` method has been called before trying to speak any text.
  - Otherwise the app will crash with an exception.

- Typical usage pattern:
  - create a boolean flag in your activity
  - have your listener set it to true when the initialization is complete
  - only call speak on the TTS object if the flag is set to true

The speak method

- The `speak` method accepts three parameters *:
  - the text to speak aloud, as a String.
  - the mode to use for speaking, one of:

    `TextToSpeech.QUEUE_ADD` : Speak after any other text is done.

    `TextToSpeech.QUEUE_FLUSH` : Stop any other text and speak immediately.
  - a `Map` of parameters (we don't need any, so we can pass `null`).

    ** (Android 5 introduces a different version of speak(), but we'll ignore it.)*

```
// speak text aloud, if my init boolean flag is set
if (myTTSisReady) {
    tts.speak("Hello, world!",
              TextToSpeech.QUEUE_FLUSH, null);
}
```

Speech-to-text

- **speech-to-text**: User talks; Android records, turns into a String.
- Similar to the camera, Android has a built-in activity for capturing speech into text.
- You can call it using an `Intent` and wait for the result.

```
Intent intent = new Intent(
        RecognizerIntent.ACTION_RECOGNIZE_SPEECH);
intent.putExtra(RecognizerIntent.EXTRA_LANGUAGE,
                Locale.getDefault());
intent.putExtra(RecognizerIntent.EXTRA_LANGUAGE_MODEL,
                RecognizerIntent.LANGUAGE_MODEL_FREE_FORM);
// prompt text is shown on screen to tell user what to say
intent.putExtra(RecognizerIntent.EXTRA_PROMPT, "text");
startActivityForResult(intent, requestCode);
```

Receiving speech-to-text data

- When the speech-to-text activity comes back, its `Intent` gives you all text spoken by the user in an `ArrayList`.
  - Usually the first element (index 0) contains the string you want.

```
@Override
protected void onActivityResult(int requestCode,
        int resultCode, Intent intent) {
    super.onActivityResult(requestCode, resultCode, intent);
    ArrayList<String> list = intent.getStringArrayListExtra(
                                RecognizerIntent.EXTRA_RESULTS);
    String spokenText = list.get(0);
    // ...
}
```

Robust speech-to-text code

- Some devices do not have speech-to-text capability.
  - In these cases, it will throw an exception when you try to use it.
  - To handle such situations, you can **`try/catch`** the exception.

```
Intent intent = new Intent(
        RecognizerIntent.ACTION_RECOGNIZE_SPEECH);
...
try {
    startActivityForResult(intent, requestCode);
} catch (ActivityNotFoundException anfe) {
    // code to handle the exception
}
```