MOBILE APPLICATION DEVELOPMENT

ANDROID (2017)

LECTURE 25: SENSORS

SENSORS

- The primary way that Android can interact with the world around the device it is running on is via sensors.
- 'Sensors' is a loose term which may refer to a number of components, such as:
  - Cameras and microphones.
  - Positioning sensors like accelerometers/gyroscopes.
  - Pressure and light sensors.
  - Health information sensors.
  - Radio receivers.

INDIRECT CAMERA USAGE

- Interaction with the camera on Android may be either a direct or indirect process.
- To indirectly use the camera, an Android application may simply use an intent action to launch another application's camera-related Activity.
  - The result once the camera Activity finishes may contain image thumbnail data. It may also contain a URI which points to a full image or video file.
  - There is no guarantee that the system has a camera Activity available, though.
- If an application absolutely must capture images or video, it may be better to use the camera directly.

DIRECT CAMERA USAGE

- Applications which must guarantee that they capture images or video with the camera (or which must use the camera for other reasons) may use the CameraDevice class from the `camera2` package.
- Requires a lot more setup than simply calling another Activity and getting a result, but enables full control of the camera, including customization of output formats, exposure levels, and more.
- The `camera2` package models the image capture process as a pipeline, in which many stages must be processing simultaneously to ensure framerates are maintained in the capture/preview process.

RECORDING AUDIO

- The MediaRecorder class may be used to record audio or video on Android.
- Recording audio requires that the programmer specify an audio source, output format, output file, and encoder, and call `prepare()` to set up the MediaRecorder.
- Audio may be recorded from multiple places, but a common input is MediaRecorder.AudioSource.MIC, which captures the microphone input.
- Call `start()` and `stop()` to add to a recording by appending to a file as recording progresses.
- Must clean up the recorder when finished by calling `release()`.

SENSORS

- The SensorManager class enumerates sensor hardware that the system provides, and allows programmers to monitor those sensors. The manager may be obtained by calling `getSystemService(Context.SENSOR_SERVICE)`.
- The SensorManager:
  - Supplies sensors matching particular types via `getSensorList(<TYPE>)`.
  - Manages listeners for sensor events with `registerListener()` / `unregisterListener()`.
  - Listeners provide the `onSensorChanged()` and `onAccuracyChanged()` calls, which allow programmers to see when sensors change their values and to measure the accuracy level of the sensors.