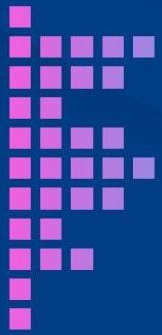


OTUS

Android Basics

Sensors & Camera

REC

Меня хорошо видно & слышно?

Ставим “+”, если все хорошо
“-”, если есть проблемы

Тема вебинара

Sensors & Camera

Кирьяков Максим

Android разработчик

Об опыте :

4+ лет опыта коммерческой разработки

@Crafto

Правила вебинара

Активно
участвуем

Задаем вопрос
в чат или голосом

Вопросы вижу в чате,
могу ответить не сразу

Маршрут вебинара

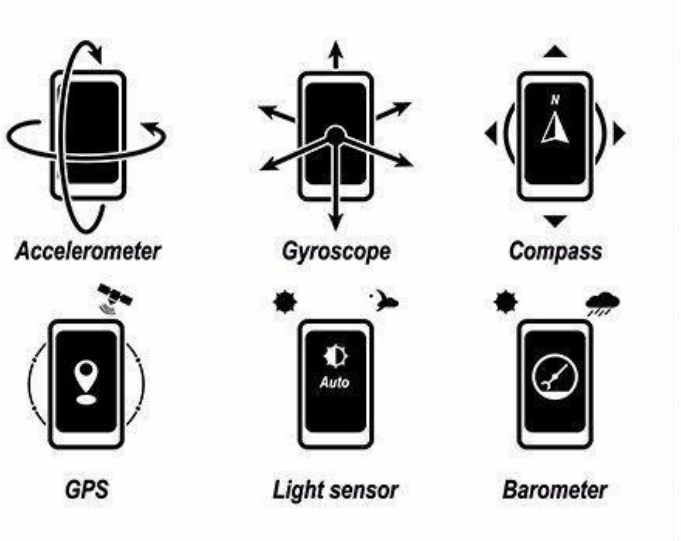
Цели вебинара

1. Научиться получать данные с датчиков
2. Научиться работать с камерой

Напишите в чат, в каких приложениях вы могли наблюдать работу датчиков?

Датчики

Большинство Android устройств имеют встроенные датчики движения, ориентации в пространстве и окружающей среды.
Позволяют получать необработанные и точные данные, что позволяет отслеживать движение устройства или его позицию в пространстве, а такжи изменения окружающей среды.



Область применения

1. Отслеживание наклонов, поворотов в играх
2. Работа компаса
3. Отключение экран во время звонков
4. Нотификации об инцидентах

Категории датчиков

- Датчики движения
  Измерение скорости движения и вращения по трём осям координат
  - Акселерометр
  - Датчик силы тяжести
  - Гироскоп

Категории датчиков

- Датчики движения
  Измерение скорости движения и вращения по трём осям координат
  - Акселерометр,
  - Датчик силы тяжести
  - Гироскоп
- Датчики окружающей среды
  Измерение температуры окружающей среды, давления, освещенности и влажности
  - Барометр
  - Фотометр
  - Термометр

Категории датчиков

- Датчики движения
  Измерение скорости движения и вращения по трём осям координат
  - Акселерометр,
  - Датчик силы тяжести
  - Гироскоп
- Датчики окружающей среды
  Измерение температуры окружающей среды, давления, освещенности и влажности
  - Барометр
  - Фотометр
  - Термометр
- Датчики положения
  Измерение физической позиции устройства
  - Датчик ориентации
  - Магнитометр

Типы датчиков

- Аппаратные
  Физические компоненты встроенные в устройства

- Программные
  Не физические устройства, но имитирующие их.
  Могут получать данные от одно или нескольких аппаратных устройств (называют виртуальными или синтетическими)

Ознакомиться с таблицей датчиков поддерживаемых системой Android можно здесь

Sensor Framework

Sensor Framework

SensorManager – получение служб датчиков, их перечня, доступности
Sensor – непосредственно класс для работы с определенным датчиком
SensorEvent – содержит данные о событии датчика
SensorEventListener – интерфейс, который получает оповещения о событиях

Доступность датчиков

Sensor	Android 4.0 (API Level 14)	Android 2.3 (API Level 9)	Android 2.2 (API Level 8)	Android 1.5 (API Level 3)
TYPE_ACCELEROMETER	Yes	Yes	Yes	Yes
TYPE_AMBIENT_TEMPERATURE	Yes	n/a	n/a	n/a
TYPE_GRAVITY	Yes	Yes	n/a	n/a
TYPE_GYROSCOPE	Yes	Yes	n/a[1]	n/a[1]
TYPE_LIGHT	Yes	Yes	Yes	Yes
TYPE_LINEAR_ACCELERATION	Yes	Yes	n/a	n/a
TYPE_MAGNETIC_FIELD	Yes	Yes	Yes	Yes
TYPE_ORIENTATION	Yes[2]	Yes[2]	Yes[2]	Yes
TYPE_PRESSURE	Yes	Yes	n/a[1]	n/a[1]
TYPE_PROXIMITY	Yes	Yes	Yes	Yes
TYPE_RELATIVE_HUMIDITY	Yes	n/a	n/a	n/a
TYPE_ROTATION_VECTOR	Yes	Yes	n/a	n/a
TYPE_TEMPERATURE	Yes[2]	Yes	Yes	Yes

Использование

Инициализация и получение списка сервисов

```
val sensorManager = getSystemService(Context.SENSOR_SERVICE) as SensorManager

// получение всех доступных датчиков
val deviceSensors = sensorManager.getSensorList(Sensor.TYPE_ALL)

// получение магнитометра
val sensor = sensorManager.getDefaultSensor(Sensor.TYPE_MAGNETIC_FIELD)
```

Использование

Регистрация слушателя

```
private val sensorEventListener = object : SensorEventListener {
  override fun onSensorChanged(event: SensorEvent?) {
     //do smth
  }
  override fun onAccuracyChanged(sensor: Sensor?, accuracy: Int) {
     //do smth
  }
}
sensorManager.registerListener(sensorEventListener,  sensor,

SensorManager.SENSOR_DELAY_NORMAL)   sensorManager.unregisterListener(sensorEventListener)
```

Использование

Дискретизация

```
/** 0 микросекунд */
SENSOR_DELAY_FASTEST
/** 20000 микросекунд */
SENSOR_DELAY_GAME
/** 60000 микросекунд */
SENSOR_DELAY_UI
/** 200000 микросекунд */
SENSOR_DELAY_NORMAL
```

Фильтрация устройств

Runtime

```
val lightSensor = sensorManager.getDefaultSensor(Sensor.TYPE_LIGHT)
if (lightSensor != null) {
        // todo smth
} else {
        // nothing todo
}
```

Фильтрация устройств

Runtime

```
val lightSensor = sensorManager.getDefaultSensor(Sensor.TYPE_LIGHT)
if (lightSensor != null) {
        // todo smth
} else {
        // nothing todo
}
```

Google Play

```
<uses-feature
  android:name="android.hardware.sensor.accelerometer"
  android:required="true" />
```

Camera

Camera

- Camera (deprecated)
  Самое первое и оригинальное API, сейчас не поддерживается и считается устаревшим.

Camera

- Camera (deprecated)
  Самое первое и оригинальное API, сейчас не поддерживается и считается устаревшим.

- Camera2
  Основной API для работы с камерой, довольно сложный, для специфических вариантов использования. Всю обработку различий аппаратного обеспечения нужно делать самостоятельно.

Camera

- Camera (deprecated)
  Самое первое и оригинальное API, сейчас не поддерживается и считается устаревшим.

- Camera2
  Основной API для работы с камерой, довольно сложный, для специфических вариантов использования. Всю обработку различий аппаратного обеспечения нужно делать самостоятельно.

- CameraX
  Входит в Jetpack library, работает с Android 5.0 и выше. Библиотека имеет высокоуровневое API общих случаев использования и обрабатывает различия в аппаратном обеспечении устройств. Это надстройка над Camera2

Camera Intents

Сделать фото

```
val takePictureIntent = Intent(MediaStore.ACTION_IMAGE_CAPTURE)
try {
    startActivityForResult(takePictureIntent, REQUEST_IMAGE_CAPTURE)
} catch (e: ActivityNotFoundException) {
    // display error state to the user
}
```

Camera Intents

Сделать видео

```
Intent(MediaStore.ACTION_VIDEO_CAPTURE).also { takeVideoIntent ->
  takeVideoIntent.resolveActivity(packageManager)?.also {
     startActivityForResult(takeVideoIntent, REQUEST_VIDEO_CAPTURE)
  } ?: run {
     //display error state to the user
  }
}
```

Camera Intents

Обработать результат

```
override fun onActivityResult(requestCode: Int, resultCode: Int, data: Intent?) {
  super.onActivityResult(requestCode, resultCode, data)
  Log.d("CAMERA", "Result data ${data.toString()}")
}
```

Практическая часть

Вопросы?

Ставим “+”,
если вопросы есть

Ставим “–”,
если вопросов нет

Рефлексия

Цели вебинара

Проверка достижения целей

1. Научились получать данные с датчиков
2. Научились работать с камерой

Спасибо за внимание! Поделитесь обратной связью