Общая информация про акселерометр

Акселерометр или G-сенсор – это датчик, определяющий угол наклона электронного устройства по отношению к земной поверхности. На основании данных от датчика программное обеспечение понимает положение смартфона, и поворачивает изображение на дисплее. Иными словами, именно акселерометр способствует автоматическому повороту экрана в альбомную ориентацию при повороте телефона.

Также этот датчик фиксирует ускорение перемещения устройства в пространстве, одновременно сопоставив три пространственные координаты. Можно сказать, что сенсор измеряет разницу между проекциями абсолютного и гравитационного ускорения.

Акселерометр в смартфоне позволяет использовать:

1. Шагомеры или другие подобные сервисы. Благодаря возможности измерять положение устройства в пространстве, а также его ускорение, сенсор обеспечивает корректную работу шагомера. Это незаменимый помощник для поклонников пробежек или прогулок. Нет необходимости покупать отдельный фитнес-трекер, поскольку в телефон можно установить приложение и использовать его в конкретных целях.

2. Игры. Благодаря G-сенсору процесс игры становится настоящим удовольствием, ведь датчик мгновенно реагирует на минимальную смену положения телефона. Можно отказаться от классической консольной системы управления, поскольку корректировать положение можно путем изменения положения телефона в пространстве.

3. Удобный интерфейс. При смене положения смартфона датчик сразу повернет интерфейс устройства в нужное положение. Эксплуатация устройства максимально удобная и комфортная. Особенно удобен автоповорот экрана при просмотре видео или фильмов.

Подробнее по ссылке:

https://v-androide.com/instruktsii/obsluzhivanie/kak-rabotaet-akselerometr-v-smartfone.html

Создаем новый проект – gravy2

1. B MainActivity

Создаем SensorManager и инициализируем его:

lateinit var sManager: SensorManager
sManager = getSystemService(Context.SENSOR SERVICE) as SensorManager

Получим сам сенсор, который будем использовать, в скобках указываем сенсор который хотим использовать – Sensor.TYPE ACCELEROMETER:

val sensor = sManager.getDefaultSensor(Sensor.TYPE ACCELEROMETER)

Необходимо чтобы в Активити мы могли полчать данные с сенсора sensor

object подчеркнут красным цветом – добавляем оба метода – Inplement Members

2		-		
3	import			
9				
10 븛	class	MainActivity	Manoanat Aativit. () (Multiplement Members	
11	la	teinit var <u>sM</u>	R 🔲 হ 🍝	
12 🞯	ι ονι	erride fun <mark>on</mark>	✓ I android.hardware.SensorEventListener	le?) {
13		super.onCre	onSensorChanged(p0: SensorEvent/): Unit	
14		setContentV	and a maccuracy nangeupo sensor, p : inj: onit	
15		<u>sManager</u> =		SERVICE) as SensorManager
16		val sensor		sor.TYPE_ACCELEROMETER)
17		val sListen		er{
18				
19		}		
20	A }			
21	- A			
			Copy JavaDoc OK Select None Cance	

Получается:

3 Eimport
10
11 🚜 🖯 class MainActivity : AppCompatActivity() {
12 lateinit var <u>sManager</u> : SensorManager
13 of override fun onCreate(savedInstanceState: Bundle?) {
14 super.onCreate(savedInstanceState)
15 setContentView(R.layout.activity_main)
16 <u>sManager</u> = getSystemService(Context.SENSOR_SERVICE) as SensorManager
17 val sensor = <u>sManager</u> .getDefaultSensor(Sensor.TYPE_ACCELEROMETER)
18 val sListener = object : SensorEventListener{
19 of override fun onSensorChanged(p0: SensorEvent?) {
20 TODO(reason: "Not yet implemented")
21 🗘 }
22
23 of c override fun onAccuracyChanged(p0: Sensor?, p1: Int) {
24 TODO(reason: "Not yet implemented")
25 🖓 }
26
27 🖨 }

В onAccuracyChanged точность измерения будет указываться.

В onSensorChanged будут приходить данные.

Теперь необходимо получать данные и что-то с ними делать.

2. Переходим в activity_main

Делаем размер существующего TextView 24sp



Делаем текст черным.

Указываем id – tvSensor

		🗮 Code 💷 Split	🔤 Des		
🜀 AccelerationTestApp ~ 🛛 🛇 Default (en-us) ~		c	. *		
I.				ed>	C 1a
		tvSensor			
¥	> Declared Attributes				
1	∼ Layout				
	Constraint Widget				
· · · · · · · · · · · · · · · · · · ·	> Constraints (4)				
· · · · · · · · · · · · · · · · · · ·	layout_width	wrap_content			
	layout_height	wrap_content			
••••••••••••••••••••••••••••••••••••••	visibility				
• <u> </u>	≁ visibility				
· · · · · · · · · · · · · · · · · · ·	✓ Transforms				
· · · · · · · · · · · · · · · · · · ·					
		View			

3. Возвращаемся в MainActivity

Пишем код для поиска созданного tvSensor

va	1	tvSer	nsor	<pre>= findViewById<textview>(R.id.tvSensor)</textview></pre>
12		clas	ss Mai	.nActivity : AppCompatActivity() {
13			latei	nit var <u>sManager</u> : SensorManager
14	0		overn	<pre>ide fun onCreate(savedInstanceState: Bundle?) {</pre>
15				uper.onCreate(savedInstanceState)
16			s	etContentView(R.layout.activity_main)
17				<pre>ral tvSensor = findViewById<textview>(R.id.tvSensor)</textview></pre>
18				
19				Manager = getSystemService(Context.SENSOR_SERVICE) as SensorManager
20			N	<pre>val sensor = <u>sManager</u>.getDefaultSensor(Sensor.TYPE_ACCELEROMETER)</pre>
21				<pre>val sListener = object : SensorEventListener{</pre>
22	0			override fun onSensorChanged(p0: SensorEvent?) {
23				
24				}
25				
26	0			override fun onAccuracyChanged(p0: Sensor?, p1: Int) {
27				
28				}
29				
30			}	
31			}	

Необходимо указать что мы ждем данные с сенсора, для этого используем сенсор менеджер – sManager

sManager.registerListener(sListener, sensor, SensorManager.SENSOR_DELAY_NORMAL)

В строке кода выше: передаем наш sListener, сенсор с которого будем ждать данные sensor, время обновление – SensorManager.SENSOR_DELAY_NORMAL

12	()	cla	ISS Mainactivity : Applompatactivity() {	@ 3 🗛 2 🛆							
13			lateinit var <u>sManager</u> : SensorManager								
14		override fun onCreate(savedInstanceState: Bundle?) {									
15			<pre>super.onCreate(savedInstanceState)</pre>								
16			<pre>setContentView(R.layout.activity_main)</pre>								
17			<pre>val tvSensor = findViewById<textview>(R.id.tvSensor)</textview></pre>								
18											
19			<u>sManager</u> = getSystemService(Context. <i>SENSOR_SERVICE</i>)								
20			val sensor = <u>sManager</u> .getDefaultSensor(Sensor. <i>TYPE_ACCELE</i>	Int							
21			val sListener = object : SensorEventListener{								
22			override fun onSensorChanged(p0: SensorEvent?) {								
23			10 AXIS_MINUS_Z								
24)- DAXIS_X								
25			DAXIS_Y								
26	01		override fun onAccuracyChanged(p0: Sensor?, p1: Int)	Int							
27			# SENSUR_DELAY_FASTEST								
28			B SENSOR DELAY LIT								
29			D SENSOR STATUS ACCURACY HIGH	Int							
30				T 0*:							
31			sManager. registerListener (sListener, sensor, SensorManager.)								
32											

B onSensorChanged поменяем переменную p0 на sEvent.

Мы будем брать значения с позиции 0, 1 и 2. Где с позиции 0 будет значение с оси X, на позиции 1 будет значение с оси Y, на позиции 2 будет значение с оси Z.

Эти значение будем брать и все показывать в наше TextView.

Пишем для оси Х:

val value = sEvent?.values
val sData = "X: \${value?.get(0)}"
tvSensor.text = sData

Получаем:

12	o	clas	is Ma	inAct	: ivity	AppCor	npatAd	ctivit	:y() {					
			late	init	var <u>sMa</u>	nager:	Senso	orMana	iger					
	0		over	ride	fun onC	reate(saved]	Instan	iceSta	te: B	undle	?) {		
				super	.onCrea	te(save	edInst	tanceS	tate)					
				setCo	ontentVi	.ew(R.la	ayout.	.activ	ity_m	ain)				
				val t	vSensor	= find	dView8	ById <t< td=""><td>extVi</td><td>ew>(R</td><td>.id.t</td><td>/Senso</td><td>r)</td><td></td></t<>	extVi	ew>(R	.id.t	/Senso	r)	
				<u>sMana</u>	<u>ager</u> = g	etSyste	emServ	vice(C	ontex	t.SEA	ISOR_SE	RVICE) as s	SensorMa
				val s	sensor =	<u>sManac</u>	<u>ger</u> .ge	etDefa	ultSe	nsor(Sensor	. TYPE	ACCEL	EROMET
				val s	Listene	r = ob	ject :	: Sens	orEve	ntLis	tener	{		
	01			0	override	fun or	nSenso	orChan	iged(s	Event	: Sens	sorEve	ent?) {	[
					val	value	= sE\	vent?.	value	s				
					val	sData	= "X:	\${va	lue?.	get(0)}"			
25					tvS	ensor.	text =	= sDat	a					
				}										
	01			0	override	fun or	nAccur	racyCh	anged	(p0:	Sensor	r?, p1	: Int)	} {
				}										
				}										
				<u>sMana</u>	ager.reg	isterL:	istene	er(sLi	stene	r, se	ensor,	Senso	orManag	jer.SEN
			}											
		_}												

Теперь если запустим приложение мы увидим, как меняются значения по оси Х от О до 9,81

Дописываем код для оси Y и Z. Ставим символы \n чтобы значения на экране отображались с новой строки (т.е. в столбик):

Получаем:

<pre>15 super.onCreate(savedInstanceState) 16 setContentView(R.layout.activity_main) 17 val tvSensor = findViewById<textview>(R.id.tvSensor) 18 19 sManager = getSystemService(Context.SENSOR_SERVICE) as SensorManager 20 val sensor = sManager.getDefaultSensor(Sensor.TYPE_ACCELEROMETER) 20 val sensor = sManager.getDefaultSensor(Sensor.TYPE_ACCELEROMETER) 20 val sensor = sManager.getDefaultSensor(Sensor.TYPE_ACCELEROMETER) 21 val sensor = sManager.getDefaultSensor(Sensor.TYPE_ACCELEROMETER) 21 val sensor = sManager.getDefaultSensor(Sensor.TYPE_ACCELEROMETER)</textview></pre>	
16 setContentView(R.layout.activity_main) 17 val tvSensor = findViewById <textview>(R.id.tvSensor) 18 </textview>	
17 val tvSensor = findViewById <textview>(R.id.tvSensor) 18 </textview>	
18 19 <u>sManager</u> = getSystemService(Context.SENSOR_SERVICE) as SensorManager 20 val sensor = <u>sManager</u> .getDefaultSensor(Sensor.TYPE_ACCELEROMETER)	
19 <u>sManager</u> = getSystemService(Context.SENSOR_SERVICE) as SensorManager 20 val sensor = <u>sManager</u> .getDefaultSensor(Sensor.TYPE_ACCELEROMETER)	
20 val sensor = <u>sManager</u> .getDefaultSensor(Sensor.TYPE_ACCELEROMETER)	
21 🕞 🔰 val slistener = object : SensorEventListener{	
<pre>22 of override fun onSensorChanged(sEvent: SensorEvent?) {</pre>	
23 val value = sEvent?.values	
24 val sData = "X: \${value?.get(0)}\nY: \${value?.get(1)}\nZ: \${value?	?.get(2)}"
25 tvSensor. <u>text</u> = sData	
26 😝 }	
27	
28 of coverride fun onAccuracyChanged(p0: Sensor?, p1: Int) {	
29	
30 🕒 }	
31	
32 🖨 }	
33 sManager.registerListener(sListener, sensor, SensorManager.SENSOR_DELAY_NOR	RMAL)
34 🕒 }	

<mark>Тестируем на телефоне</mark>