

**Correction TP06****Matière : ATELIER DEVELOPPEMENT MOBILE AVANCE****Classes : SEM31****Exercice1**

```
public class MainActivity extends Activity {  
    private TextView tvListe;  
    private SensorManager smg;  
    @Override  
    protected void onCreate(Bundle savedInstanceState) {  
        super.onCreate(savedInstanceState);  
        setContentView(R.layout.activity_main);  
        init();  
    }  
    private void init() {  
        tvListe = (TextView) findViewById(R.id.tvListe);  
        smg = (SensorManager) getSystemService(SENSOR_SERVICE);  
        listerCapteurs();  
    }  
    private void listerCapteurs() {  
        List<Sensor> lCapteur = smg.getSensorList(Sensor.TYPE_ALL);  
        String s = "";  
        int i = 1;  
        for (Sensor c : lCapteur) {  
            s += ("Capteur " + i + ":\n");  
            s += "\t Nom: " + c.getName() + "\n";  
            s += "\t Version: " + c.getVersion() + "\n";  
            s += "\t Type: " + getType(c.getType()) + "\n";  
            s += "\t Vendeur: " + c.getVendor() + "\n";  
            s += "\t Consommation: " + c.getPower() + " mA\n";  
            s += "\t Résolution: " + c.getResolution() + "\n";  
            s += "\t Maximum range: " + c.getMaximumRange() + "\n";  
            i++;  
        }  
        tvListe.setText(s);  
    }  
    private String getType(int type) {  
        String strType;  
        switch (type) {  
            case Sensor.TYPE_ACCELEROMETER: strType = "TYPE_ACCELEROMETER"; break;  
            case Sensor.TYPE_GRAVITY: strType = "TYPE_GRAVITY"; break;  
            case Sensor.TYPE_GYROSCOPE: strType = "TYPE_GYROSCOPE"; break;  
            case Sensor.TYPE_LIGHT: strType = "TYPE_LIGHT"; break;  
            case Sensor.TYPE_LINEAR_ACCELERATION: strType = "TYPE_LINEAR_ACCELERATION"; break;  
            case Sensor.TYPE_MAGNETIC_FIELD: strType = "TYPE_MAGNETIC_FIELD"; break;  
            case Sensor.TYPE_ORIENTATION: strType = "TYPE_ORIENTATION"; break;  
            case Sensor.TYPE_PRESSURE: strType = "TYPE_PRESSURE"; break;  
            case Sensor.TYPE_PROXIMITY: strType = "TYPE_PROXIMITY"; break;  
            case Sensor.TYPE_ROTATION_VECTOR: strType = "TYPE_ROTATION_VECTOR"; break;  
            case Sensor.TYPE_TEMPERATURE: strType = "TYPE_TEMPERATURE"; break;  
            default: strType = "TYPE_UNKNOW"; break;  
        }  
        return strType;  
    }  
}
```

Exercice2

```
public class MainActivity extends Activity implements SensorEventListener {  
    private TextView tvX; private TextView tvY; private TextView tvZ;  
    private Button btnAjouter; private Button btnVider; private EditText edList;  
    private SensorManager smg;  
    private Sensor acc;  
    @Override  
    protected void onCreate(Bundle savedInstanceState) {  
        super.onCreate(savedInstanceState);  
        setContentView(R.layout.activity_main);  
        init();  
    }  
    private void init() {  
        tvX = (TextView) findViewById(R.id.tvX);  
        tvY = (TextView) findViewById(R.id.tvY);  
        tvZ = (TextView) findViewById(R.id.tvZ);  
        btnAjouter = (Button) findViewById(R.id.btnAjouter);  
        btnVider = (Button) findViewById(R.id.btnVider);  
        edList = (EditText) findViewById(R.id.edList);  
        smg = (SensorManager) getSystemService(SENSOR_SERVICE);  
        acc = smg.getDefaultSensor(Sensor.TYPE_ACCELEROMETER);  
        ajouterEcouteurs();  
    }  
    private void ajouterEcouteurs() {  
        btnAjouter.setOnClickListener(new OnClickListener() {  
            @Override  
            public void onClick(View arg0) { ajouter(); } });  
        btnVider.setOnClickListener(new OnClickListener() {  
            @Override  
            public void onClick(View arg0) { vider(); } });  
    }  
    protected void vider() {  
        edList.setText("");  
    }  
    protected void ajouter() {  
        String ligne= "(" + tvX.getText().toString() + ","  
            + tvY.getText().toString() + "," + tvZ.getText().toString() + ")\n";  
        edList.setText(edList.getText() + ligne);  
    }  
    @Override  
    protected void onPause() {  
        smg.unregisterListener(this, acc);  
        super.onPause();  
    }  
    @Override  
    protected void onResume() {  
        smg.registerListener(this, acc, SensorManager.SENSOR_DELAY_UI);  
        super.onResume();  
    }  
    @Override  
    public void onAccuracyChanged(Sensor arg0, int arg1) {}  
    @Override  
    public void onSensorChanged(SensorEvent arg0) {  
        if (arg0.sensor.getType() == Sensor.TYPE_ACCELEROMETER) {  
            tvX.setText(arg0.values[0] + "m/s2");  
            tvY.setText(arg0.values[1] + "m/s2");  
            tvZ.setText(arg0.values[2] + "m/s2");  
        }  
    }  
}
```

Exercice3

```
public class MainActivity extends AppCompatActivity {
    public static final int[] TAB_PRES = new
int[]{R.drawable.pres00,R.drawable.pres01,R.drawable.pres02,R.drawable.pres03};
    public static final int[] TAB_LOIN = new
int[]{R.drawable.loin00,R.drawable.loin01,R.drawable.loin02,R.drawable.loin03};
    private TextView tvProx;
    private ImageView imgPL;
    private SensorManager mg;
    private Sensor prox;
    private int courant;
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        init();
    }
    private void init() {
        tvProx=findViewById(R.id.tvProx);
        imgPL = findViewById(R.id.imgPL);
        mg = (SensorManager) getSystemService(SENSOR_SERVICE);
        prox = smg.getDefaultSensor(Sensor.TYPE_PROXIMITY);
        courant=0;
        ajouterEcouteur();
    }
    private void ajouterEcouteur() {
        imgPL.setOnClickListener(new OnClickListener() {
            @Override
            public void onClick(View v) {
                suivant();
            }
        });
    }
    @Override
    protected void onResume() {
        mg.registerListener(this, prox, SensorManager.SENSOR_DELAY_UI);
        super.onResume();
    }
    @Override
    protected void onPause() {
        mg.unregisterListener(this, prox);
        super.onPause();
    }
    @Override
    public void onAccuracyChanged(Sensor arg0, int arg1) {
    }
    @Override
    public void onSensorChanged(SensorEvent event) {
        if (event.sensor.getType() == Sensor.TYPE_PROXIMITY) {
            float prox = event.values[0];
            if(prox==0)
                imgPL.setImageResource(TAB_PRES[courant]);
            else
                imgPL.setImageResource(TAB_LOIN[courant]);
        }
    }
    private void suivant() {
        courant=(courant+1)%TAB_PRES.length;
    }
}
```