ISET SFAX **AU 2022/2023 S1**	**DEPARTEMENT TECHNOLOGIE** **DE L'INFORMATIQUE**

Correction TP07

Matière : ATELIER DEVELOPPEMENT MOBILE AVANCE	**Classes : SEM31**

Exercice1

```
package com.serv;
public class MainActivity extends Activity {
  private Button btnDemarrer;
  private Button btnArreter;
  private TextView tvTemps;
  private BroadcastReceiver receiver;
  @Override
  protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_main);
    init();
  }
  private void init() {
    btnDemarrer = (Button) findViewById(R.id.btnDemarrer);
    btnArreter = (Button) findViewById(R.id.btnArreter);
    tvTemps = (TextView) findViewById(R.id.tvTemps);
    ajouterEcouteur();
  }
  private void ajouterEcouteur() {
    btnDemarrer.setOnClickListener(new OnClickListener() {
      @Override
      public void onClick(View arg0) {
        demarrer();
      }
    });
    btnArreter.setOnClickListener(new OnClickListener() {
      @Override
      public void onClick(View arg0) {
        arreter();
      }
    });
    receiver = new BroadcastReceiver() {
      @Override
      public void onReceive(Context context, Intent intent) {
        afficher(intent);
      }
    };
  }
  protected void afficher(Intent intent) {
    String temps = intent.getStringExtra("temps");
    tvTemps.setText(temps);
  }
  protected void arreter() {
    Intent i = new Intent(this, PremierService.class);
    stopService(i);
  }
  protected void demarrer() {
    Intent i = new Intent(this, PremierService.class);
    startService(i);
  }
  @Override
  protected void onResume() {
    super.onResume();
    registerReceiver(receiver, new IntentFilter(PremierService.NOTIFICATION));
  }
  @Override
  protected void onPause() {
    unregisterReceiver(receiver);
    super.onPause();
  }
}
package com.serv;
public class PremierService extends Service {
  public static final String NOTIFICATION = "com.serv.temps";
  private Thread th;
  private boolean actif;
  @Override
  public IBinder onBind(Intent intent) {
    Log.i("Service", "onBind");
    return null;
  }
  @Override
  public void onCreate() {
    Log.i("Service", "onCreate");
    actif = true;
    super.onCreate();
    afficherTemps();
  }
  private void afficherTemps() {
    Runnable r = new Runnable() {
      @Override
      public void run() {
        while (actif) {
          afficher();
        }
      }
    };
    th = new Thread(r);
    th.start();
  }
  protected void afficher() {
    try {
      String s = DateFormat.format("MM/dd/yy hh:mm:ss", System.currentTimeMillis()).toString();
      Log.i("", s);
      envoyerMessage(s);
      th.sleep(1000); // ou SystemClock.sleep(1000);
    } catch (InterruptedException e) {
      e.printStackTrace();
    }
  }
  private void envoyerMessage(String temps) {
    Intent intent = new Intent(NOTIFICATION);
    intent.putExtra("temps", temps);
    sendBroadcast(intent);
  }
  @Override
    public void onDestroy() {
      Log.i("", "onDestroy");
      actif = false;
      super.onDestroy();
    }
}
```

Exercice2

```
package com.compass;
public class MainActivity extends Activity {
  private Button btnTourner;
  private ImageView imgCompass;
  private float dernierAngle;
  private float angleTest;
  private BroadcastReceiver receiver;
  @Override
  protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_main);
    init();
  }
  private void init() {
    btnTourner = (Button) findViewById(R.id.btnTourner);
    imgCompass = (ImageView) findViewById(R.id.imgCompass);
    dernierAngle = 0;
    angleTest = 0;
    ajouterEcouteur();
  }
  private void ajouterEcouteur() {
    btnTourner.setOnClickListener(new OnClickListener() {
      @Override
      public void onClick(View v) {
        tourner30();
      }
    });
    receiver = new BroadcastReceiver() {
      @Override
      public void onReceive(Context context, Intent intent) {
        actualiser(intent);
      }
    };
  }
  protected void actualiser(Intent intent) {
    float azimutDegree = intent.getFloatExtra("azimut", 0);
    tourner(-azimutDegree);
  }
  protected void demarrerService() {
    Intent i = new Intent(this, OrientService.class);
    startService(i);
  }
  protected void arreterService() {
    Intent i = new Intent(this, OrientService.class);
    stopService(i);
  }
  @Override
  protected void onResume() {
    super.onResume();
    demarrerService();
    registerReceiver(receiver, new IntentFilter(OrientService.NOTIFICATION));
  }
  @Override
  protected void onPause() {
    arreterService();
    unregisterReceiver(receiver);
    super.onPause();
  }
  protected void tourner30() {
    angleTest += 30;
    if (angleTest > 360)
```

```
      angleTest = 30;
    tourner(angleTest);
  }
  private void tourner(float angle) {
    // create a rotation animation (reverse turn degree degrees)
    RotateAnimation ra = new RotateAnimation(dernierAngle, angle, Animation.RELATIVE_TO_SELF, 0.5f,
        Animation.RELATIVE_TO_SELF, 0.5f);
    // how long the animation will take place
    ra.setDuration(210);
    // set the animation after the end of the reservation status
    ra.setFillAfter(true);
    // Start the animation
    imgCompass.startAnimation(ra);
    dernierAngle = angle;
  }
}
package com.compass;
public class OrientService extends Service implements SensorEventListener {
  public static final String NOTIFICATION = "com.serv.azimut";
  private SensorManager mg;
  private Sensor orient;
  @Override
  public IBinder onBind(Intent intent) {
    Log.i("Service", "onBind");
    return null;
  }
  @Override
  public void onCreate() {
    Log.i("Service", "onCreate");
    super.onCreate();
    init();
  }
  private void init() {
    mg = (SensorManager) getSystemService(SENSOR_SERVICE);
    orient = mg.getDefaultSensor(Sensor.TYPE_ORIENTATION);
    mg.registerListener(this, orient, SensorManager.SENSOR_DELAY_UI);
  }
  private void envoyerMessage(float azimutDegree) {
    Intent intent = new Intent(NOTIFICATION);
    intent.putExtra("azimut", azimutDegree);
    sendBroadcast(intent);
  }
  @Override
  public void onDestroy() {
    mg.unregisterListener(this, orient);
    Log.i("", "onDestroy");
    super.onDestroy();
  }
  @Override
  public void onSensorChanged(SensorEvent event) {
    float azimutDegree = event.values[0];
    envoyerMessage(azimutDegree);
  }
  @Override
  public void onAccuracyChanged(Sensor sensor, int accuracy) {
  }
}
```

Exercice3

"MainActivity"

```
private void ajouterEcouteur() {
        br = new BroadcastReceiver() {
            @Override
            public void onReceive(Context context, Intent intent) {
                actualiser(intent);
            }
        };
    }
private void demarrerService() {
        Intent i = new Intent(this, RouteService.class);
        startService(i);
    }
    private void arreterService() {
        Intent i = new Intent(this, RouteService.class);
        stopService(i);
    }
    @Override
    protected void onResume() {
        demarrerService();
        registerReceiver(br, new IntentFilter(RouteService.ACTION_ETAT));
        super.onResume();
    }
    @Override
    protected void onPause() {
        arreterService();
        unregisterReceiver(br);
        super.onPause();

    }
protected void actualiser(Intent intent) {
        double accZ = intent.getDoubleExtra("ACC_Z", 0);
        if (accZ < 13) {
            tvEtat.setText(accZ + " => "+"BON");
            imgEtat.setImageResource(IMG_REV_BON);
        }
        else if (accZ < 16) {
            tvEtat.setText(accZ + " => "+"MAUVAIS");
            imgEtat.setImageResource(IMG_REV_MAUVAIS);
        }
        else {
            tvEtat.setText(accZ + " => "+"TRES MAUVAIS");
            imgEtat.setImageResource(IMG_REV_TRES_MAUVAIS);
        }
    }
```

"ProxService"

```
private void init() {
        smg = (SensorManager) getSystemService(SENSOR_SERVICE);
        acc = smg.getDefaultSensor(Sensor.TYPE_ACCELEROMETER);
        smg.registerListener(this, acc, SensorManager.SENSOR_DELAY_UI);
    }
    @Override
    public void onDestroy() {
        smg.unregisterListener(this, acc);
        super.onDestroy();
    }

@Override
    public void onSensorChanged(SensorEvent event) {
        if (event.sensor.getType() == Sensor.TYPE_ACCELEROMETER) {
            double accZ = Math.abs(event.values[2]);

            Intent i = new Intent();
            i.setAction(ACTION_ETAT);
            i.putExtra("ACC_Z", accZ);

            sendBroadcast(i);
        }
    }
```