Correction TP08	
Matière : ATELIER DEVELOPPEMENT MOBILE AVANCE	Classes : SEM31

Exercice1

Capture

```
package com.capture;
//imports
public class MainActivity extends Activity {
  private ImageView imgCapture;
  private Button btnCaptureImage;
  private VideoView vidCapture;
  private Button btnCaptureVideo;
  private final static int ACTION_CAPTURE_IMAGE = 1;
  private final static int ACTION_CAPTURE_VIDEO = 2;
  @Override
  protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_main);
    init();
  }
  private void init() {
    imgCapture = (ImageView) findViewById(R.id.imgCapture);
    btnCaptureImage = (Button) findViewById(R.id.btnCaptureImage);
    vidCapture = (VideoView) findViewById(R.id.vidCapture);
    MediaController mediaController = new
                MediaController(this);
    mediaController.setAnchorView(vidCapture);
    vidCapture.setMediaController(mediaController);
    btnCaptureVideo = (Button) findViewById(R.id.btnCaptureVideo);
    ajouterEcouteur();
  }
  private void ajouterEcouteur() {
    btnCaptureImage.setOnClickListener(new OnClickListener() {
      @Override
      public void onClick(View arg0) {      capturerImage();      }
    });
    btnCaptureVideo.setOnClickListener(new OnClickListener() {
      @Override
      public void onClick(View arg0) {      capturerVideo();      }
    });
  }
  protected void capturerImage() {
    Intent i = new Intent(android.provider.MediaStore.ACTION_IMAGE_CAPTURE);
    startActivityForResult(i, ACTION_CAPTURE_IMAGE);
  }
  protected void capturerVideo() {
    Intent takeVideoIntent = new Intent(MediaStore.ACTION_VIDEO_CAPTURE);
    startActivityForResult(takeVideoIntent, ACTION_CAPTURE_VIDEO);
  }
  @Override
  protected void onActivityResult(int requestCode, int resultCode, Intent data) {
    if (requestCode == ACTION_CAPTURE_IMAGE && resultCode == RESULT_OK) {
      Bundle extras = data.getExtras();
      Bitmap imageBitmap = (Bitmap) extras.get("data");
      imgCapture.setImageBitmap(imageBitmap);
    } else if (requestCode == ACTION_CAPTURE_VIDEO && resultCode == RESULT_OK) {
      Uri videoUri = data.getData();
      vidCapture.setVideoURI(videoUri);
      vidCapture.start();
    }
  }
}
```

Exercice2

Détection Visage

```
package com.visage;
//imports
public class MainActivity extends Activity {
  private Button btnSuivant;
  private ToggleButton tglVisible;
  private FaceView fView;
  private int[] tImage = new int[] { R.raw.image0, R.raw.image1, R.raw.image2, R.raw.image3,
R.raw.image4 };
  private int indiceCourant;
  @Override
  protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_main);
    init();
  }
  private void init() {
    btnSuivant = (Button) findViewById(R.id.btnSuivant);
    tglVisible = (ToggleButton) findViewById(R.id.tglVisible);
    fView = (FaceView) findViewById(R.id.fView);
    indiceCourant = 0;
    detecter(tglVisible.isChecked());
    ajouterEcouteur();
  }
  private void ajouterEcouteur() {
    btnSuivant.setOnClickListener(new OnClickListener() {
      @Override
      public void onClick(View arg0) {
        suivant();
      }
    });
    tglVisible.setOnCheckedChangeListener(new OnCheckedChangeListener() {
      @Override
      public void onCheckedChanged(CompoundButton arg0, boolean arg1) {
        detecter(arg1);
      }
    });
  }
  private void detecter(boolean afficherInfo) {
    InputStream stream = getResources().openRawResource(tImage[indiceCourant]);
    Bitmap bitmap = BitmapFactory.decodeStream(stream);
    if (afficherInfo) {
      Builder b = new Builder(this);
      b.setTrackingEnabled(true);
      b.setLandmarkType(FaceDetector.ALL_LANDMARKS);
      b.setClassificationType(FaceDetector.ALL_CLASSIFICATIONS);
      FaceDetector detector = b.build();
      Frame frame = new Frame.Builder().setBitmap(bitmap).build();
      SparseArray<Face> faces = detector.detect(frame);
      fView.setContent(bitmap, faces);
    } else
      fView.setContent(bitmap, null);
  }
  private void suivant() {
    indiceCourant = (indiceCourant + 1) % tImage.length;
    detecter(tglVisible.isChecked());
  }
}
package com.visage;

//imports
import com.google.android.gms.vision.face.Face;
import com.google.android.gms.vision.face.Landmark;

public class FaceView extends View {

  private Bitmap mBitmap;
  private SparseArray<Face> mFaces;

  public FaceView(Context context, AttributeSet attrs) {
    super(context, attrs);
  }
  public void setContent(Bitmap bitmap, SparseArray<Face> faces) {
    mBitmap = bitmap;
    mFaces = faces;
    invalidate();
  }
  @Override
  protected void onDraw(Canvas canvas) {
    super.onDraw(canvas);
    if ((mBitmap != null)) {
      double scale = drawBitmap(canvas);
      if (mFaces != null)
        afficherFaces(canvas, scale);
    }
  }
  private double drawBitmap(Canvas canvas) {
    double viewWidth = canvas.getWidth();
    double viewHeight = canvas.getHeight();
    double imageWidth = mBitmap.getWidth();
    double imageHeight = mBitmap.getHeight();
    double scale = Math.min(viewWidth / imageWidth, viewHeight / imageHeight);
    Rect destBounds = new Rect(0, 0, (int) (imageWidth * scale), (int) (imageHeight * scale));
    canvas.drawBitmap(mBitmap, null, destBounds, null);
    return scale;
  }
  private void afficherFaces(Canvas canvas, double scale) {
    Paint paint = new Paint();

    for (int i = 0; i < mFaces.size(); ++i) {
      Face face = mFaces.valueAt(i);
      afficherCadre(canvas, scale, paint, face);
      afficherPointInterest(canvas, scale, paint, face);
      afficherEtat(canvas, scale, paint, face);
    }
  }
  private void afficherCadre(Canvas canvas, double scale, Paint paint, Face face) {
    paint.setStyle(Paint.Style.STROKE);
    paint.setColor(Color.BLUE);
    paint.setStrokeWidth(3);

    float xTL = (float) (face.getPosition().x * scale);
    float yTL = (float) (face.getPosition().y * scale);

    float width = (float) (face.getWidth() * scale);
    float height = (float) (face.getHeight() * scale);
    float xBR = xTL + width;
    float yBR = yTL + height;
    canvas.drawRect(xTL, yTL, xBR, yBR, paint);
  }
  private void afficherPointInterest(Canvas canvas, double scale, Paint paint, Face face) {
    paint.setStyle(Paint.Style.FILL);
    paint.setColor(Color.RED);
    paint.setStrokeWidth(3);
    for (Landmark landmark : face.getLandmarks()) {
      int cx = (int) (landmark.getPosition().x * scale);
      int cy = (int) (landmark.getPosition().y * scale);
      canvas.drawCircle(cx, cy, 3, paint);
    }
  }

  private void afficherEtat(Canvas canvas, double scale, Paint paint, Face face) {
    paint.setStyle(Paint.Style.FILL);
    paint.setColor(Color.YELLOW);
    paint.setStrokeWidth(1);
    paint.setTextSize(20.0f);

    float xTL = (float) (face.getPosition().x * scale);
    float yTL = (float) (face.getPosition().y * scale);

    String desc = "id: " + face.getId();
    desc += " h:" + String.format("%.2f", face.getIsSmilingProbability());
    desc += " r:" + String.format("%.2f", face.getIsRightEyeOpenProbability());
    desc += " l:" + String.format("%.2f", face.getIsLeftEyeOpenProbability());

    canvas.drawText(desc, xTL, yTL, paint);
  }
}
```

Exercice3
Détection code à barre

```
package com.code;
//imports
import com.google.android.gms.vision.barcode.Barcode;
import com.google.android.gms.vision.barcode.BarcodeDetector;
import com.google.android.gms.vision.barcode.BarcodeDetector.Builder;

public class MainActivity extends Activity {
  private Button btnSuivant;
  private ToggleButton tglVisible;
  private BarcodeView bView;
  private int[] tImage = new int[] { R.raw.code0, R.raw.code1, R.raw.code2 , R.raw.code3 , R.raw.code4,
R.raw.code5 , R.raw.code6  };
  private int indiceCourant;
  @Override
  protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_main);
    init();
  }
  private void init() {
    btnSuivant = (Button) findViewById(R.id.btnSuivant);
    tglVisible = (ToggleButton) findViewById(R.id.tglVisible);
    bView = (BarcodeView) findViewById(R.id.bView);
    indiceCourant = 0;
    detecter(tglVisible.isChecked());
    ajouterEcouteur();
  }
  private void ajouterEcouteur() {
    btnSuivant.setOnClickListener(new OnClickListener() {
      @Override
      public void onClick(View arg0) {
        suivant();
      }
    });
    tglVisible.setOnCheckedChangeListener(new OnCheckedChangeListener() {
      @Override
      public void onCheckedChanged(CompoundButton arg0, boolean arg1) {
        detecter(arg1);
      }
    });

  }

  private void detecter(boolean afficherInfo) {

    InputStream stream = getResources().openRawResource(tImage[indiceCourant]);
    Bitmap bitmap = BitmapFactory.decodeStream(stream);

    if (afficherInfo) {

      BarcodeDetector.Builder b = new Builder(this);
      b.setBarcodeFormats(Barcode.ALL_FORMATS);


      BarcodeDetector detector = b.build();


      Frame frame = new Frame.Builder().setBitmap(bitmap).build();
      SparseArray<Barcode> codes = detector.detect(frame);
      bView.setContent(bitmap, codes);
    } else
      bView.setContent(bitmap, null);
  }

  private void suivant() {
    indiceCourant = (indiceCourant + 1) % tImage.length;
    detecter(tglVisible.isChecked());
  }

}
package com.code;
//imports
public class BarcodeView extends View {

  private Bitmap mBitmap;
  private SparseArray<Barcode> mCodes;

  public BarcodeView(Context context, AttributeSet attrs) {
    super(context, attrs);
  }
  public void setContent(Bitmap bitmap, SparseArray<Barcode> codes) {
    mBitmap = bitmap;
    mCodes = codes;
    invalidate();
  }
  @Override
  protected void onDraw(Canvas canvas) {
    super.onDraw(canvas);
    if ((mBitmap != null)) {
      double scale = drawBitmap(canvas);
      if (mCodes != null)
        afficherCodes(canvas, scale);
    }
  }
  private double drawBitmap(Canvas canvas) {
    double viewWidth = canvas.getWidth();
    double viewHeight = canvas.getHeight();
    double imageWidth = mBitmap.getWidth();
    double imageHeight = mBitmap.getHeight();
    double scale = Math.min(viewWidth / imageWidth, viewHeight / imageHeight);

    Rect destBounds = new Rect(0, 0, (int) (imageWidth * scale), (int) (imageHeight * scale));
    canvas.drawBitmap(mBitmap, null, destBounds, null);
    return scale;
  }

  private void afficherCodes(Canvas canvas, double scale) {
    Paint paint = new Paint();

    for (int i = 0; i < mCodes.size(); ++i) {
      Barcode bar = mCodes.valueAt(i);
      afficherCadre(canvas, scale, paint, bar);

    }
  }

  private void afficherCadre(Canvas canvas, double scale, Paint paint, Barcode bar) {
    paint.setStyle(Paint.Style.STROKE);
    paint.setColor(Color.BLUE);
    paint.setStrokeWidth(3);
    paint.setTextSize((float) (40.0f*scale));
    Rect r = bar.getBoundingBox();

    float xTL = (float) (r.left * scale);
    float yTL = (float) (r.top * scale);

    float xBR = (float) (r.right * scale);
    float yBR = (float) (r.bottom * scale);

    canvas.drawRect(xTL, yTL, xBR, yBR, paint);

    String desc = bar.rawValue;
    canvas.drawText(desc, xTL, (yTL+yBR)/2, paint);

  }

}
```

Exercice4

Analyse Visage

```
private void analyser() {
        Bitmap bitmap = ((BitmapDrawable) imgVisages.getDrawable()).getBitmap();
        FaceDetector.Builder b = new FaceDetector.Builder(this);
        b.setTrackingEnabled(true);
        b.setLandmarkType(FaceDetector.ALL_LANDMARKS);
        b.setClassificationType(FaceDetector.ALL_CLASSIFICATIONS);
        FaceDetector detector = b.build();
        Frame frame = new Frame.Builder().setBitmap(bitmap).build();
        SparseArray<Face> faces = detector.detect(frame);
        int nbV=faces.size();
        int nbVS=0;
        int nbVI=0;
        double dymMax=0;

        for (int i = 0; i < faces.size(); ++i) {
            Face face = faces.valueAt(i);
            if(face.getIsSmilingProbability()>0.5f)
                nbVS++;
            if(face.getEulerZ()>20)
                nbVI++;


            double xLE = 0, yLE = 0, xRE = 0, yRE = 0;

            for (Landmark l : face.getLandmarks()) {
                if (l.getType() == Landmark.LEFT_EYE) {
                    xLE = l.getPosition().x;
                    yLE = l.getPosition().y;
                }
                if (l.getType() == Landmark.RIGHT_EYE) {
                    xRE = l.getPosition().x;
                    yRE = l.getPosition().y;
                }
            }
            double dym = Math.sqrt(Math.pow(xLE - xRE, 2) + Math.pow(yLE - yRE, 2));
            if(dym>dymMax)
                dymMax=dym;
        }
        tvNbV.setText(nbV+"");
        tvNbVS.setText(nbVS+"");
        tvNbVI.setText(nbVI+"");
        tvDYM.setText(dymMax+"");
    }
```