FAQs – Facial Processing

* **What image format is supported by the framework?**
  + For camera preview: YUV NV21. Any other image formats are required conversion to YUV NV21.
  + For static images: Bitmap
* **What should be my ‘frameWidth’ and ‘frameHeight’ in FacialProcessing.setFrame(…); ?** 
  + For camera preview: ‘Camera.getParamters.previewSize’ should give you the respective width and height.
  + For static images: Dimensions of the bitmap image.
* **Why are the face co-ordinates of the result rotated with respect to the preview on the screen.** 
  + This is caused due to the way the camera sensor is mounted on your device. Different devices can have different camera sensor mounting.
  + To fix this use ‘setFrame(byte [] frameData, int frameWidth, int frameHeight, Boolean isMirrored, PREVIEW\_ROTATION\_ANGLE rotationAngle)’ . For rotationAngle, give the desired angle represented by the enum ‘PREVIEW\_ROTATION\_ANGLE’ that will bring the coordinates into alignment with the preview
  + Android API ‘Camera.orientation’ can give you delta angle by which the sensor is rotated from its original position. Use that angle to determine the PREVIEW\_ROTATION\_ANGLE.

* **Why does FacialProcessing.getInstance() return null ?**
  + It could be because of
    - FacialProcessing is not supported. To confirm call method ‘FacialProcessing.isFeatureSupported’. If the method returns false then the Facial Processing feature is not supported on that device.
    - The instance for FacialProcessing is already being used. Use the existing instance or release (FacialProcessing.release( )) the previously created instance and then call ‘FacialProcessing.getInstance( )’ again.
* **What is the benefit of the different processing modes?**
  + Video mode is more efficient due to its prediction algorithm (relies on previous frame data). It is more suitable for scanning camera previews.
  + Still mode is more accurate because since it scans the entire image every time. It is more suitable for static images.
* **In what case would I use the ‘setFrame’ method with the ‘isMirrored’ parameter?**
  + When working with front camera preview frames, set the ‘isMirrored’ parameter to ‘TRUE’.
* **The face coordinates of the result are not aligned with respect to the preview on the screen.** 
  + You will need to normalize the co-ordinates of the result with respect to the size of the drawing surface’s width and height

* + Example:

if rotation angle is 0 or 180 degrees, then

newX = oldX \* surfaceWidth / cameraPreviewWidth   
 newY = oldY \* surfaceHeight / cameraPreviewHeight

if rotation angle is 90 or 270 degrees, then

newX = oldX \* surfaceWidth / cameraPreviewHeight   
 newY = oldY \* surfaceHeight / cameraPreviewWidth