

Image Manipulation

Image manipulation

Image manipulations are performed using the S60 multimedia architecture. Dedicated APIs are provided for image rotation, image scaling, and converting to and from a variety of industry-standard file types, such as JPEG and GIF.

From an implementation perspective, the key difference between the alternative S60 versions is handling the asynchronous nature of the image manipulations. Under S60 1st Edition an observer mechanism, based on the **MMdaImageUtilObserver** class, is used. By deriving from this class and implementing its virtual functions, applications are informed of the completion of a manipulation and can respond appropriately.

When developing for S60 2nd Edition, a **CActive**-derived class should be created and the **TRequestStatus** member **iStatus** passed into the function performing the manipulation. This will result in the **RunL()** of the **CActive**-derived class being called when the manipulation is finished.

Image conversion

Conversion can take place to and from images stored in files or descriptors. Following a decode operation, if an image is to be displayed onto the device screen, a bitmap object of type **CFbsBitmap** must be created.

Conversely, if the image is already displayed on screen—for example, within a camera application—and needs to be saved as a JPEG, its data will currently be contained within a **CFbsBitmap** object. Consequently, the **CFbsBitmap** class plays an important role in the conversion process, acting as a bridging point between both encoding and decoding.

Decoding

The principles for decoding an image are the same, regardless of the S60 version being developed for. First, a channel to the data store, such as a GIF file, is opened, and this provides essential image attributes such as the image size.

When this is complete, a **CFbsBitmap** object can be instantiated with respect to the attributes obtained; however, it is still lacking the image data that will be displayed. The conversion operation takes place, resulting in the image data being stored inside the bitmap object, ready for use by application code.