



## Creation of MBM file

1. Convert images to bmp and save them as .mbm using function CFbsbitmap::saveL()
2. Get this MBM filepaths into some buffer and pass them to TDesC\* aSources[],
3. For TInt32 aSourceIds[] you can use ids starting from 0.
4. Call this storeL() function ,mbm is ready to use now...

```

_LIT(KMbmFile, "C:\\result.mbm");

_LIT(KBMPFilePath0, "C:\\facebook.mbm");
_LIT(KBMPFilePath1, "C:\\balance.mbm");

TInt32* uniqueIds = new ( ELeave ) TInt32[ 2 ];
CleanupStack::PushL( uniqueIds );
uniqueIds[ 0 ] = 0;
uniqueIds[ 1 ] = 0;

TFileName** filenames = new ( ELeave ) TFileName*[ 2 ];
CleanupStack::PushL( filenames );
filenames[ 0 ] = new (ELeave) TFileName( KBMPFilePath0);
filenames[ 1 ] = new (ELeave) TFileName( KBMPFilePath1);

CFbsBitmap::StoreL( KMbmFile, // Filename for new multi-bitmap mbm
2, // Count of files
( const TDesC** )filenames, // bitmaps to be loaded
uniqueIds ); // id's of the bitmaps in MBM files

// Clean resources
delete filenames[ 0 ];
delete filenames[ 1 ];

CleanupStack::PopAndDestroy( filenames );
CleanupStack::PopAndDestroy( uniqueIds );

```

## Alternate method

This method applies if you already have Bmp files and/or their masks. In this case the MBM files can be created using the mmp file.

### Without mask file

```

START BITMAP      S60Test.mbm
HEADER
TARGETPATH       \system\apps\step6
SOURCEPATH       ..\bitmaps
SOURCE           c12 tlo.bmp
END

```

### With mask file

```

START BITMAP      mbyownmbm.
HEADER
TARGETPATH       system\apps\Jhalak

```

## Creation\_of\_MBM\_file

```
SOURCEPATH      ..\bitmaps
SOURCE          bmp2 check.
SOURCE          c12 check_mask.
SOURCE          c12 uncheck.
SOURCE          c12 uncheck_mask.
END
```

This creates a mbg file that has to be included wherever the images need to be used. The mbg is actually a text file which contains an enum with the image indexes.

e.g.

```
/* This file has been generated, DO NOT MODIFY. */
enum TMbmS60test
{
EMbmS60testT1o
};
```