Bitmap class for direct screen access

This article shows a bitmap class that can be used to store images for quick rendering to the screen buffer when using direct screen access. This might be used, for example, in rendering sprites in a DSA enabled game

Remarks for the sample

- It assumes, that images are stored in an mbm file in the application's private directory
- It reserves space on the heap to store the image and converts the bitmap in a way it can be directly rendered in the screen buffer
- Remember to use /c24 switch when creating mbm file with mifconv tool to ensure right color depth

Links

- Anti-tearing with CDirectScreenBitmap
- How to draw image to screen directly

Code

Header file (.h)

```
#define TPixel TUint32

class CMyBitmap : public CBase
 {
 ...

 TPixel* iData;
        TSize iSize;

 ...
    };
```

Source file (cpp)

```
...

_LIT( KMyBitmapFile,"MyGame.mbm" );

// CMyBitmap class represents a bitmap ready for rendering
// with direct screen access
// aIndex is the image's index in the mbm file
void CMyBitmap::ConstructL( TInt aIndex )
    {
    CFbsBitmap* image = new( ELeave ) CFbsBitmap;
    CleanupStack::PushL( image );

    // Finding out the location of the mbm file
    TFileName path;

#ifdef __WINS__
    path.Append( _L("z:\\resource\\apps\\") );
#else
    path.Copy(CEikonEnv::Static()->EikAppUi()->Application()->AppFullName().Left(2));
    TFileName relPath;
    CEikonEnv::Static()->FsSession().PrivatePath( relPath );
    path.Append( relPath);
#endif

    path.Append( KMyBitmapFile );

    // Load bitmap
    User::LeaveIfError( image->Load( path, aIndex, EFalse ) );

    // Creating space on heap
    iSize = image->SizeInPixels();
    // Remember to delete it in the destructor
    iData = new ( ELeave ) TPixel[ iSize.iWidth * iSize.iHeight ];

    // TBitmapUtil helps accessing pixels safely
    TBitmapUtil sourceUtil( image );

    for ( TInt y = 0; y < iSize.iHeight; y++ )
     {
     sourceUtil.Begin( TPoint( 0, y ) );
     for ( TInt x = 0; x < iSize.iWidth; x++ )
      {
      *(iData + y * iSize.iWidth + x) = sourceUtil.GetPixel();
      sourceUtil.IncXPos();
      }
     sourceUtil.End();
     }
    CleanupStack::PopAndDestroy();
    }

...
```


(C) Copyright Nokia 2013. All rights reserved.

(C) Copyright Nokia 2013. All rights reserved.