ImageControl

INHERITS FROM Object

DECLARED IN ImageControl.h

CLASS DESCRIPTION

This object is the object you should be talking to the most. It's job it to be the go between for you program and the converters stored on disk. Like the converters, however, this object is also stored on disk, so feel free to modify the code and place a new copy in the Converters directory as Bitmap.controls. I do ask a couple of things. First, if you think a change warrants it, such as a major bug fix, please send me the modified code, so that I can consider it for release in a future release or a patch. Secondly, I ask that you do not distribute modified version. This has the possibility of creating a lot of confusion and difficulty for the user.

Using this object is basically fairly simple. You simply need to instantiate a new version via the use of the ControlLoader object. From there, you ask it to get a new filename via it's open panel. It can then use this filename to open a new bitmap. Doing the reverse by calling the save panel, and then passing the filename bask to the program can save an image.

INSTANCE VARIABLES

Inherited from Object	Class	isa;
Declared in Converter	(none)	

METHOD TYPES

Creating and Freeing a Converter	+ new: - free
Getting Pixels	- getXYFunction - resetNext - getNextFunction
Converting Colors	- getToGrayFunction - getToRGBFunction
Converting Formats	- ditherImage - convertToPalette:andReturn:andPalettes:::

CLASS METHODS

new

+ **new**: *image*

Returns the current error state of the converter. The type of error state return can be found in Converter.h, and must be CONVERT_ERR_NONE, CONVERT_ERR_WARNING, or CONVERT_ERR_FATAL. The image should be returned unless the error state reported is CONVERT_ERR_FATAL.

See also: - **errorMessage**
- **errorStringMessage**

INSTANCE METHODS

convertToPalette:andReturn:andPalettes:::
- **convertToPalette:** (int)*size*
andReturn: (unsigned char **)*outPic*
andPalettes: (unsigned char *)*r* **:** (unsigned char *)*g* **:** (unsigned char *)*b*

Applies a simple color mapping scheme to take a twenty four bit image and convert it to an image with a palette. Note that will also handle one bit and gray scale images, so you do not need to check the configuration of the input image before calling the method.

When calling this image, *size* indicates the desired palette size, but because this only applies simple color mapping, you will probably not want to pass in a size less then 64 colors. It will work for as few as 8 colors, however, if you don't mind severe color banding. *outPic* is an array of unsigned chars containing the returned bitmap data. There will be one byte per pixel, despite the value in *size*, so you'll have to take this into account when specifying less than 256 colors. *r, g,* and *b* are arrays where the palette values will be put. They must be at least as large as *size* in length.

Return self.

ditherImage
- **ditherImage**

Applies a Floyd Steinberg ditherizing algorhythm to it's image and returns an id to a new **NXBitmapImageRep**, or **nil** if it is not successful. When done, the object is still valid for it's original image. You will need a new instantiation for the returned image if you wish to do more with the image.

free

- **free**

  Free the object. Note that this does not free the image that was passed to it. You'll have to do that yourself.

resetNext

- **resetNext**

  This prepares the image to return sequential pixel values. This method must be called prior to using a **GetPixelNextFunc**.

  Returns self.

getNextFunction

- (GetPixelNextFunc)**getNextFunction**

  This returns a pointer for the appropiate get next function for the control object's image type. It will be optimized for getting pixels from the image type.

  When making calls to the returned pointer, the function takes no parameters, and returns a pointer to a **Pixel.**

getToGrayFunc

- (ToGrayFunc)**getToGrayFunc**

  Returns a pointer to a function that can convert **Pixel** values into gray scale **Pixel** values. The returned function accepts a pointer to and returns a pointer to a **Pixel**. Also note, if the original image contains alpha, the return **Pixel** will contain alpha, although you can choose to ignore it.

getToGrayFunc

- (ToRGBFunc)**getToGrayFunc**

This is identical to the *getToGrayFunc* except that the returned function will convert **Pixel**s into RGB pairs.

getXYFunc

- (GetPixelXYFunc)**getXYFunc**

This is esentially the same as the *getNextFunction* except that the returned function accepts (x, y) pairs as arguments and returns the pixel value at that location.

CONSTANTS AND DEFINED TYPES

```
typedef struct {
    pixel  values[5];
    int    count;
} Type

typedef    Pixel * (*GetPixelXYFunc)(int, int)

typedef    Pixel * (*GetPixelNextFunc)(void)

typedef    Pixel * (*ToGrayFunc)(Pixel *)

typedef    Pixel * (*ToRGBFunc)(Pixel *)
```