图像媒体模块说明文档

图像媒体模块负责提供图像媒体的访问操作.目前实现了对 jpg,bmp,ico 图像文件的读取操作.

源文件目录: lib/source/MultiMedia/
主文件: MM_Image.c
头文件: MM_Image.H

目前支持的格式 JPG BMP ICO PNG

函数接口说明

打开图像文件
MM_IMAGE * MM_image_open (u8 *path);

参数	u8 *path	文件路径
返回值	MM_IMAGE *	图像句柄

1.MM_IMAGE * 是一个内部图像句柄,为了隐藏内部特性,因此对外将 MM_IMAGE 声明成了 void 型数据.这只是一个句柄,你不能对其做任何处理.
2.返回值为 NULL,代表打开失败.
3.打开成功后,必须在使用完毕关闭这个句柄.否则内存资源将不被释放.
4.文件路径使用 ANSI 编码.

读取图像数据
int MM_image_read (MM_IMAGE * image, void *buf, int width, int height)

参数	MM_IMAGE * image	图像句柄	
	void *buf	接收图像数据的缓存起始地址	
	int width	宽度(像素单位)	
	int height	高度(像素单位)	
返回值	int	0	读取成功
		其他	读取失败

1. 图像句柄必须是有效的.
2. 接收图像数据的缓存必须有足够的空间.(宽度*高度*4).
3. 图像格式为 A8R8G8B8 格式,即每像素 32 位(4 字节),从高到低分别是 Alpha,Red,Green,Blue,每通道占 8 位.
4. 图像将按照你给出的宽度和高度输出,不论原来的图像是什么分辨率,函数都会自动缩放到你指定的分辨率输出.

5. 此函数可能需要耗费大量的内存来存储临时数据,因此读取大图(jpg,bmp 等等)的时候,最好使用 MM_image_readline 来一行一行的读取数据. 对于 ico 等小图片, 这个函数就可以应付了.

关闭图像句柄

void　　MM_image_close　(MM_IMAGE * image)

参数	MM_IMAGE * image	图像句柄	
返回值	无		

图像使用完毕之后必须使用这个函数关闭,它将会释放内部使用过的内存空间,并关闭相关的文件.不要关闭无效的句柄,不要重复关闭同一个句柄.

获取图像尺寸

int　　MM_image_size　(MM_IMAGE * image, int * width, int * height　)

输入	MM_IMAGE * image	图像句柄	
	int * width	接收宽度数据的地址	
	int * height	接收高度数据的地址	
输出	int	0	成功
		其他	失败

1. 图像尺寸使用像素单位
2. 返回失败时, *width, *height 的值将不可预知,不要读取或使用他们.
3. 使用这个函数配合 MM_image_read 就可以读取出原始大小的图像了.
4. ICO 文件一般都有由多个分辨率图像组成,且尺寸都已经预知(16*16,32*32,64*64,128*128),因此对 ICO 文件,这个函数无意义.

读取图像指定行

```
int　　MM_image_readline　( MM_IMAGE * image, void *buf, int scanline );
```

参数	MM_IMAGE * image	图像句柄
	void *buf	接受图像数据的地址
	int scanline	读取的扫描线
返回值	int	0 成功,其他失败

1. 并不是所有的图片都支持随机读取的,对于不支持随机读取的图像,你仍然可以使用这个函数,不过 scanline 的值将被忽略,并且从 0 开始,每读取一次都图像行都递增 1.

查询一个图像句柄是否支持随机读取

```
int　　MM_image_is_random_read　( MM_IMAGE * image );
```

参数	MM_IMAGE * image	图像句柄
返回值	int	1 支持随机读取,0 不支持随机读取

1. 使用 MM_image_readline 读取图像之前,请调用这个函数查询改图像是否支持随机读取.