NGN_IMAGE.H

NGN_TextureData* ConvertRawToTextureData(NGN_RawImage* raw);

Convierte una imagen en formato RAW a datos de textura. Devuelve NULL en caso de error.

```
NGN_RawImage* pixels = ngn->load->SpriteAsRaw("data/coin.spr", 3);
NGN_TextureData* coin_data = ngn->image->ConvertRawToTextureData(pixels);
```

bool CutOutMask(
    NGN_RawImage* source,   // Imagen RAW de origen
    NGN_RawImage* mask,   // Imagen RAW de mascara
    NGN_RawImage* destination   // Imagen RAW de destino
);

Aplica la máscara de recorte especificada a la imagen de origen y almacena el resultado en la imagen de destino. Si los pixeles de la máscara contienen información de transparencia (canal alpha), esta también se aplica. Devuelve TRUE en caso de éxito y FALSE en caso de error.

```
NGN_RawImage* stone = ngn->load->PngAsRaw("data/stone.png");
NGN_RawImage* mask = ngn->load->PngAsRaw("data/cutmask.png");
NGN_RawImage* pixels = new NGN_RawImage();
ngn->image->CutOutMask(stone, mask, pixels);
```

bool HollowMask(
    NGN_RawImage* source,   // Imagen RAW de origen
    NGN_RawImage* mask,   // Imagen RAW de mascara
    NGN_RawImage* destination   // Imagen RAW de destino
);

Aplica la máscara de vaciado especificada a la imagen de origen y almacena el resultado en la imagen de destino. Si los pixeles de la máscara contienen información de transparencia (canal alpha), esta también se aplica. Devuelve TRUE en caso de éxito y FALSE en caso de error.

```
NGN_RawImage* stone = ngn->load->PngAsRaw("data/stone.png");
NGN_RawImage* mask = ngn->load->PngAsRaw("data/cutmask.png");
NGN_RawImage* pixels = new NGN_RawImage();
ngn->image->HollowMask(stone, mask, pixels);
```

bool AdvancedMask(
**   NGN_RawImage* source,   // Imagen de origen**
**   NGN_RawImage* mask,   // Imagen de la máscara**
**   NGN_RawImage* destination,   // Imagen de destino**
**   Vector2I32 offset = {0, 0}, // Offset de la máscara**
**   uint8_t mode = NGN_MASKMODE_CUTOUT   // Modo de la máscara**
);

Aplica la máscara especificada a la imagen de origen y almacena el resultado en la imagen de destino. Si los pixeles de la máscara contienen información de transparencia (canal alpha), esta también se aplica. Puede especificarse el modo de aplicación de la máscara, NGN_MASKMODE_CUTOUT o NGN_MASKMODE_HOLLOW (recorte o vaciado), así como el desplazamiento de la máscara (offset). Devuelve TRUE en caso de éxito y FALSE en caso de error.

```
NGN_RawImage* stone = ngn->load->PngAsRaw("data/stone.png");
NGN_RawImage* mask = ngn->load->PngAsRaw("data/cutmask.png");
NGN_RawImage* pixels = new NGN_RawImage();
Vector2I32 offset = {15, 25};
ngn->image->AdvancedMask(stone, mask, pixels, offset, NGN_MASKMODE_CUTOUT);
```

bool RendererToSurface(NGN_RendererSurface* destination);

Guarda el contenido actual del renderer en el objeto tipo NGN_RendererSurface especificado. Devuelve TRUE en caso de existo o FALSE en caso de error.

```
RendererSurface* renderer_surface = new NGN_RendererSurface();
ngn->image->RendererToSurface(renderer_surface);
```

bool SurfaceToRaw(
**   NGN_RendererSurface* source,   // Surface de origen**
**   NGN_RawImage* destination,   // Imagen RAW de destino**
**   NGN_RawImage* mask = NULL   // Mascara cutout (opcional)**
);

Convierte el Surface especificado a una imagen en formato RAW y, opcionalmente, le aplica una máscara de recorte si esta se especifica. Este método devuelve TRUE en caso de éxito o FALSE en caso de error.

```
RendererSurface* renderer_surface = new NGN_RendererSurface();
ngn->image->RendererToSurface(renderer_surface);
NGN_RawImage* mask = ngn->load->PngAsRaw("data/cutmask.png");
NGN_RawImage* pixels = new NGN_RawImage();
ngn->image->SurfaceToRaw(renderer_surface, pixels, mask);
```