Function

vImageBuffer_Init(_:_:_:_:_:)

Initializes a vImage buffer with a specified width, height, and bits per pixel.

iOS 7.0+ | iPadOS 7.0+ | Mac Catalyst 13.1+ | macOS 10.9+ | tvOS 7.0+ | visionOS 1.0+ | watchOS 1.0+

```swift
func vImageBuffer_Init(
    _ buf: UnsafeMutablePointer<vImage_Buffer>,
    _ height: vImagePixelCount,
    _ width: vImagePixelCount,
    _ pixelBits: UInt32,
    _ flags: vImage_Flags
) -> vImage_Error
```

Parameters

`buf`
A valid empty `vImage_Buffer` structure.

`height`
The height of the image.

`width`
The width of the image.

`pixelBits`
The number of bits in a pixel of image data. If `pixelBits` isn't divisible by 8, `vImageBuffer_Init(_:_:_:_:_:)` pads each row of pixels to a multiple of a byte. This ensures that two rows don't share the same byte, and all rows start at the beginning of a byte.

`flags`

The options to use when performing this operation.

Return Value

`kvImageNoError`; otherwise, one of the error codes in Data Types and Constants.

Discussion

This function provides two different operations, depending on whether you pass the `kvImageNoAllocate` flag.

Without the `kvImageNoAllocate` flag, this function initializes a `vImage_Buffer` structure with memory that the vImage library sizes and aligns for the best performance. For example, the following code initializes a vImage buffer with the optimal memory size and alignment for a 10 x 5 image:

```
var buffer = vImage_Buffer()

vImageBuffer_Init(
    &buffer,
    5,     // height
    10,    // width
    8,     // bits per pixel
    vImage_Flags(kvImageNoFlags))

// Prints "vImage_Buffer(data: Optional(...), height: 5, width: 10, rowBytes: 16)".
print(buffer)
```

With the `kvImageNoAllocate` flag, the function returns the memory alignment and overwrites the buffer's `rowBytes` property with values that provide the best performance. For example, the following code calls `vImageBuffer_Init(_:_:_:_:_:)` and uses the computed values to allocate an `UnsafeMutableRawPointer`:

```
let width = 10
let height = 5

var buffer = vImage_Buffer()

let alignment = vImageBuffer_Init(
    &buffer,
    vImagePixelCount(height),
    vImagePixelCount(width),
    8,
    vImage_Flags(kvImageNoAllocate))

guard alignment > 0 else {
    // A negative value indicates an error.
    fatalError()
}

let data = UnsafeMutableRawPointer.allocate(
    byteCount: buffer.rowBytes * height,
    alignment: alignment)

buffer.data = data

// Prints "vImage_Buffer(data: Optional(...), height: 5, width: 10, rowBytes: 16)".
print(buffer)
```

In both cases, this function sets the buffer's width, height, and row bytes properties.

See Also

Initializing vImage buffers

`struct vImage_Buffer`

An image buffer that stores an image's pixel data, dimensions, and row stride.