

ImageJCV Functions

To use these functions, the [ImageJCV](#) Feature is required.

convertCV(ImageCV i, Number newPixelFormat)

Converts the input image to another [pixel format](#).

Example

```
output = MAP({expressions=[['convertCV(image, 28)', 'converted']]}, input)
```

getWidthCV(ImageCV i)

Returns the width of the given image

Example

```
output = MAP({expressions=[['getWidth(image)', 'width']]}, input)
```

getHeightCV(ImageCV i)

Returns the height of the given image

Example

```
output = MAP({expressions=[['getHeight(image)', 'height']]}, input)
```

getDepthCV(ImageCV i)

Returns the depth of the given image

Example

```
output = MAP({expressions=[['getDepth(image)', 'depth']]}, input)
```

getNumChannelsCV(ImageCV i)

Returns the number of channels of the given image

Example

```
output = MAP({expressions=[['getNumChannels(image)', 'numChannels']]}, input)
```

getPixelFormatCV(ImageCV i)

Returns the [pixel format](#) of the given image

Example

```
output = MAP({expressions=[['getPixelFormat(image)', 'pixelFormat']]}, input)
```

resizeCV(ImageCV i, Number width, Number height)

- `convertCV(ImageCV i, Number newPixelFormat)`
- `getWidthCV(ImageCV i)`
- `getHeightCV(ImageCV i)`
- `getDepthCV(ImageCV i)`
- `getNumChannelsCV(ImageCV i)`
- `getPixelFormatCV(ImageCV i)`
- `resizeCV(ImageCV i, Number width, Number height)`
- `stretchContrastCV(ImageJCV i, Number oldMin, Number oldMax, Number newMin, Number newMax)`
- `toImageCV(Number width, Number height, Number pixelFormat)`
- `toImageCV(Number width, Number height, Number depth, Number numChannels, Number pixelFormat)`
- `toImageCV(Image i)`
- `reinterpretCV(ImageJCV i, Number newWidth, Number newHeight, Number newDepth, Number newNumChannels, Number newPixelFormat)`

Resizes the image to the given width and height

Example

```
output = MAP({expressions=[[ 'resizeCV(image, 100, 100)', 'smallImage' ]]], input)
```

stretchContrastCV(ImageJCV i, Number oldMin, Number oldMax, Number newMin, Number newMax)

Converts an 16-bit 1-channel image (f.ex. a temperature map) to an 24-bit RGB grayscale image. The contrast of the new image is calculated with this formula for each pixel:

$$\text{newValue} = (\text{oldValue} - \text{oldMin}) / (\text{oldMax} - \text{oldMin}) * (\text{newMax} - \text{newMin}) + \text{newMin}$$

The example converts a temperature map to a grayscale image such that input 1000 maps to black and input 5000 maps to white.

Example

```
output = MAP({expressions=[[ 'stretchContrastCV(image, 1000, 5000, 0, 255)', 'grayscale' ]]], input)
```

toImageCV(Number width, Number height)

Creates a new 32-bpp RGBA image with the given width and height

Example

```
output = MAP({expressions=[[ 'toImageCV(512, 512)', 'image' ]]], input)
```

toImageCV(Number width, Number height, Number depth, Number numChannels, Number pixelFormat)

Creates a new image with the given width, height, [depth](#), channel count and [pixel format](#).

Example

```
output = MAP({expressions=[[ 'toImageCV(512, 512, 8, 4, 28)', 'image' ]]], input) /// 32bpp RGBA image
```

toImageCV(Image i)

Copies the contents of the input image (from the [Image feature](#)) into a new ImageJCV image.

Example

```
output = MAP({expressions=[[ 'toImageCV(inputImage)', 'image' ]]], input)
```

reinterpretCV(ImageJCV i, Number newWidth, Number newHeight, Number newDepth, Number newNumChannels, Number newPixelFormat)

Reinterprets the content of an image (argument 0) as another image with different width, height, [depth](#), number of channels or [pixel format](#). Any parameter which is set to 1 will be replaced by the corresponding value of the original image. Buffer sizes of original and reinterpreted image must match!

Example: Convert a 16-bit grayscale image to a 32-bit RGBA image with half of the width:

Example

```
output = MAP({expressions=[['reinterpret(image, getWidth(image)/2, -1, 8, 4, 28)', 'image']]}, input)
```

Parameter explanation:

- image: Original image
- getWidth(image)/2: Half width
- -1: Original height
- 8: 8 bit per pixel channel
- 4: 4 channels per pixel
- 28: 32 Bit RGBA [pixel format](#)