

JSImage

Method Summary

[JSImage](#) [#flip](#)(type)
Flips the image vertically (type param=0) or horizontally (type param=1).

[String](#) [#getContentType](#)()
Gets the contenttype (image/jpeg) of this image.

[byte\[\]](#) [#getData](#)()
Gets the bytes of this image, so that they can be saved to disk or stored the database.

[Number](#) [#getHeight](#)()
Gets the height of this image.

[String](#) [#getMetaDataDescription](#)()
Gets the description of a metadata property from the image.

[Object](#) [#getMetaDataObject](#)()
Gets the real object of a metadata property from the image.

[String\[\]](#) [#getMetaDataProperties](#)()
Gets the available metadata properties from the image.

[Number](#) [#getWidth](#)()
Gets the width of this image.

[JSImage](#) [#resize](#)(width, height)
Resizes the image to the width/height given, keeping aspect ratio.

[JSImage](#) [#rotate](#)(degrees)
Rotates the image the number of degrees that is given.

Method Details

[flip](#)

[JSImage](#) [flip](#)(type)

Flips the image vertically (type param=0) or horizontally (type param=1). A new JSImage is returned.

Parameters

type

Returns

[JSImage](#)

Sample

```
var image = plugins.images.getImage(byteArray_or_file_or_filename);//loads the image
image = image.flip(0);//flip vertically
var bytes = image.getData();//gets the image bytes
plugins.file.writeFile('filename',bytes);//saves the image bytes
```

[getContentType](#)

[String](#) [getContentType](#)()

Gets the contenttype (image/jpeg) of this image.

Returns

[String](#)

Sample

```
var image = plugins.images.getImage(byteArray_or_file);
var width = image.getWidth();
var height = image.getHeight();
var contentType = image.getContentType();
```

[getData](#)

[byte\[\]](#) [getData](#)()

Gets the bytes of this image, so that they can be saved to disk or stored the database.

Returns

[byte\[\]](#)

Sample

```
var image = plugins.images.getImage(byteArray_or_file_or_filename);//loads the image
image = image.resize(200,200);//resizes it to 200,200
var bytes = image.getData();//gets the image bytes
plugins.file.writeFile('filename',bytes);//saves the image bytes
```

getHeight

Number getHeight()

Gets the height of this image.

Returns

Number

Sample

```
var image = plugins.images.getImage(byteArray_or_file);
var width = image.getWidth();
var height = image.getHeight();
var contentType = image.getContentType();
```

getMetaDataDescription

String getMetaDataDescription()

Gets the description of a metadata property from the image. Currently only jpg is supported.

Returns

String

Sample

```
var image = plugins.images.getImage(byteArray_or_file_or_filename);//loads the image
// get the available metadata properties from the image, currently only jpg is supported
var propertiesArray = image.getMetaDataProperties();
for(var i=0;i<propertiesArray.length;i++)
{
    var property = propertiesArray[i]
    application.output("property: " + property);
    application.output("description (string): " + image.getMetaDataDescription(property))
    application.output("real object: " + image.getMetaDataObject(property))
}
// Thumbnail data is stored under property 'Exif - Thumbnail Data', extract that and set it in a dataprovider
thumbnail = image.getMetaDataObject("Exif - Thumbnail Data"); // gets thumbnail data from the image
```

getMetaDataObject

Object getMetaDataObject()

Gets the real object of a metadata property from the image. Currently only jpg is supported.

Returns

Object

Sample

```
var image = plugins.images.getImage(byteArray_or_file_or_filename);//loads the image
// get the available metadata properties from the image, currently only jpg is supported
var propertiesArray = image.getMetaDataProperties();
for(var i=0;i<propertiesArray.length;i++)
{
    var property = propertiesArray[i]
    application.output("property: " + property);
    application.output("description (string): " + image.getMetaDataDescription(property))
    application.output("real object: " + image.getMetaDataObject(property))
}
// Thumbnail data is stored under property 'Exif - Thumbnail Data', extract that and set it in a dataprovider
thumbnail = image.getMetaDataObject("Exif - Thumbnail Data"); // gets thumbnail data from the image
```

getMetaDataProperties

String[] getMetaDataProperties()

Gets the available metadata properties from the image. Currently only jpg is supported.

Returns

[String\[\]](#)

Sample

```
var image = plugins.images.getImage(byteArray_or_file_or_filename);//loads the image
// get the available metadata properties from the image, currently only jpg is supported
var propertiesArray = image.getMetaDataProperties();
for(var i=0;i<propertiesArray.length;i++)
{
    var property = propertiesArray[i]
    application.output("property: " + property);
    application.output("description (string): " + image.getMetaDataDescription(property))
    application.output("real object: " + image.getMetaDataObject(property))
}
// Thumbnail data is stored under property 'Exif - Thumbnail Data', extract that and set it in a dataprovider
thumbnail = image.getMetaDataObject("Exif - Thumbnail Data"); // gets thumbnail data from the image
```

[getWidth](#)

[Number](#) **getWidth()**

Gets the width of this image.

Returns

[Number](#)

Sample

```
var image = plugins.images.getImage(byteArray_or_file);
var width = image.getWidth();
var height = image.getHeight();
var contentType = image.getContentType();
```

[resize](#)

[JSImage](#) **resize**(width, height)

Resizes the image to the width/height given, keeping aspect ratio. A new JSImage is returned.

Parameters

width

height

Returns

[JSImage](#)

Sample

```
var image = plugins.images.getImage(byteArray_or_file_or_filename);//loads the image
image = image.resize(200,200);//resizes it to 200,200
var bytes = image.getData();//gets the image bytes
plugins.file.writeFile('filename',bytes);//saves the image bytes
```

[rotate](#)

[JSImage](#) **rotate**(degrees)

Rotates the image the number of degrees that is given. A new JSImage is returned.

Parameters

degrees

Returns

[JSImage](#)

Sample

```
var image = plugins.images.getImage(byteArray_or_file_or_filename);//loads the image
image = image.rotate(90);//rotate the image 90 degrees
var bytes = image.getData();//gets the image bytes
plugins.file.writeFile('filename',bytes);//saves the image bytes
```