```
ImageEventsBitzFourColors = '4clr' /* four colors */,
    ImageEventsBitzColor = 'colr' /* color */,
    ImageEventsBitzBlackWhite = b\&w / *black \& white */,
    ImageEventsBitzBest = 'best' /* best */,
enum ImageEventsBitz {
typedef enum ImageEventsEdfm ImageEventsEdfm;
};
    ImageEventsEdfmWebDAVFormat = 'dfwd' /* WebDAV format */
    ImageEventsEdfmUnknownFormat = 'df$$' /* unknown format */,
    ImageEventsEdfmUFSFormat = 'dfuf' /* UFS format */,
    ImageEventsEdfmUDFFormat = 'dfud' /* UDF format */,
    ImageEventsEdfmQuickTakeFormat = 'dfqt' /* QuickTake format */,
    ImageEventsEdfmProDOSFormat = 'dfpr' /* ProDOS format */,
    ImageEventsEdfmNFSFormat = 'dfnf' /* NFS format */,
    ImageEventsEdfmMSDOSFormat = 'dfms' /* MSDOS format */,
    ImageEventsEdfmMacOSFormat = 'dfhf' /* Mac OS format */,
    ImageEventsEdfmMacOSExtendedFormat = 'dfh+' /* Mac OS Extended format */,
    ImageEventsEdfmISO9660Format = 'df96' /* ISO 9660 format */,
    ImageEventsEdfmHighSierraFormat = 'dfhs' /* High Sierra format */,
    ImageEventsEdfmAudioFormat = 'dfau' /* audio format */,
    ImageEventsEdfmAppleShareFormat = 'dfas' /* AppleShare format */,
    ImageEventsEdfmApplePhotoFormat = 'dfph' /* Apple Photo format */,
enum ImageEventsEdfm {
typedef enum ImageEventsPrintingErrorHandling ImageEventsPrintingErrorHandling;
}:
PostScript errors */
    ImageEventsPrintingErrorHandlingDetailed = 'lwdt' /* print a detailed report of
handling */,
    ImageEventsPrintingErrorHandlingStandard = 'lwst' /* Standard PostScript error
enum ImageEventsPrintingErrorHandling {
typedef enum ImageEventsSaveOptions ImageEventsSaveOptions;
}:
*\
    ImageEventsSaveOptionsAsk = 'ask ' /* Ask the user whether or not to save the file.
    ImageEventsSaveOptionsNo = 'no ' /* Do not save the file. */,
    ImageEventsSaveOptionsYes = 'yes ' /* Save the file. */,
enum ImageEventsSaveOptions {
ImageEventsImage, ImageEventsMetadataTag, ImageEventsProfile;
ImageEventsSystemDomainObject, ImageEventsUserDomainObject, ImageEventsDisplay,
ImageEventsFolder, ImageEventsLocalDomainObject, ImageEventsNetworkDomainObject,
ImageEventsClassicDomainObject, ImageEventsFile, ImageEventsFilePackage,
ImageEventsDiskItem, ImageEventsAlias, ImageEventsDisk, ImageEventsDomain,
@class ImageEventsApplication, ImageEventsDocument, ImageEventsWindow,
#import <ScriptingBridge/ScriptingBridge.h>
#import <AppKit/AppKit.h>
*\
 * ImageEvents.h
\*
```

```
typedef enum ImageEventsTypz ImageEventsTypz;
};
    ImageEventsTypzTIFF = 'TIFF' /* TIFF */
    ImageEventsTypzTGA = 'tga ' /* TGA */,
    ImageEventsTypzText = 'TEXT' /* Text */,
    ImageEventsTypzSGI = '.SGI' /* SGI */,
    ImageEventsTypzQuickTimeImage = 'qtif' /* QuickTime Image */,
    ImageEventsTypzPSD = 'psd ' /* PSD */,
    ImageEventsTypzPNG = 'PNGf' /* PNG */,
    ImageEventsTypzPICT = 'PICT' /* PICT */,
    ImageEventsTypzPhotoshop = '8BPS' /* Photoshop */,
    ImageEventsTypzPDF = 'PDF ' /* PDF */,
    ImageEventsTypzMacPaint = 'PNTG' /* MacPaint */,
    ImageEventsTypzJPEG2 = 'jpg2' /* JPEG2 */,
    ImageEventsTypzJPEG = 'JPEG' /* JPEG */,
    ImageEventsTypzGIF = 'GIF ' /* GIF */,
    ImageEventsTypzBMP = 'BMPf' /* BMP */,
enum ImageEventsTypz {
typedef enum ImageEventsCmlv ImageEventsCmlv;
}:
    ImageEventsCmlvMedium = 'medi' /* Medium compression */
    ImageEventsCmlvLow = 'low ' /* Low compression */,
    ImageEventsCmlvHigh = 'high' /* High compression */,
enum ImageEventsCmlv {
typedef enum ImageEventsPPCS ImageEventsPPCS;
};
    ImageEventsPPCSXYZ = 'XYZ' /* XYZ */
    ImageEventsPPCSLab = 'Lab ' /* Lab */,
enum ImageEventsPPCS {
typedef enum ImageEventsPCla ImageEventsPCla;
}:
    ImageEventsPClaOutput = 'prtr' /* output device */
    ImageEventsPClaNamed = 'nmcl' /* named color space profile */,
    ImageEventsPClaMonitor = 'mntr' /* display device */,
    ImageEventsPClaLink = 'link' /* device-link profile */,
    ImageEventsPClaInput = 'scnr' /* input device */,
    ImageEventsPClaColorspace = 'spac' /* colorspace profile */,
    ImageEventsPClaAbstract = 'abst' /* abstract profile */,
enum ImageEventsPCla {
typedef enum ImageEventsBitz ImageEventsBitz;
}:
    ImageEventsBitzTwoHundredFiftySixGrays = '256g' /* two hundred fifty six grays */
    ImageEventsBitzTwoHundredFiftySixColors = '256c' /* two hundred fifty six colors */,
    ImageEventsBitzThousandsOfColors = 'thou' /* thousands of colors */,
    ImageEventsBitzSixteenGrays = '16gr' /* sixteen grays */,
    ImageEventsBitzSixteenColors = '16cl' /* sixteen colors */,
    ImageEventsBitzMillionsOfColorsPlus = 'mil+' /* millions of colors plus */,
    ImageEventsBitzMillionsOfColors = 'mill' /* millions of colors */,
    ImageEventsBitzGrayscale = 'gray' /* grayscale */,
    ImageEventsBitzFourGrays = '4gry' /* four grays */,
```

```
ImageEventsTypvPSD = 'psd ' /* PSD */,
    ImageEventsTypvPNG = 'PNGf' /* PNG */,
    ImageEventsTypvPICT = 'PICT' /* PICT */,
    ImageEventsTypvJPEG2 = 'jpg2' /* JPEG2 */,
    ImageEventsTypvJPEG = 'JPEG' /* JPEG */,
    ImageEventsTypvBMP = 'BMPf' /* BMP */,
enum ImageEventsTypv {
typedef enum ImageEventsQual ImageEventsQual;
}:
    ImageEventsQualMedium = 'medi' /* medium */
    ImageEventsQualLow = 'low ' /* low */,
    ImageEventsQualLeast = 'leas' /* least */,
    ImageEventsQualHigh = 'high' /* high */,
    ImageEventsQualBest = 'best' /* best */,
enum ImageEventsQual {
typedef enum ImageEventsSavo ImageEventsSavo;
}:
    ImageEventsSavoYes = 'yes ' /* Save the image. */
    ImageEventsSavoNo = 'no ' /* Do not save the image. */,
enum ImageEventsSavo {
typedef enum ImageEventsPRdr ImageEventsPRdr;
};
    ImageEventsPRdrSaturationIntent = 'Rdr2' /* saturation */
    ImageEventsPRdrRelativeColorimetricIntent = 'Rdr1' /* relative colorimetric */,
    ImageEventsPRdrPerceptualIntent = 'Rdr0' /* perceptual */,
    ImageEventsPRdrAbsoluteColorimetricIntent = 'Rdr3' /* absolute colorimetric */,
enum ImageEventsPRdr {
typedef enum ImageEventsPSpc ImageEventsPSpc;
};
    ImageEventsPSpcXYZ = 'XYZ' /* XYZ */
    ImageEventsPSpcSixColor = '6CLR' /* Six color */,
    ImageEventsPSpcSixChannel = 'MCH6' /* Six channel */,
    ImageEventsPSpcSevenColor = '7CLR' /* Seven color */,
    ImageEventsPSpcSevenChannel = 'MCH7' /* Seven channel */,
    ImageEventsPSpcRGB = 'RGB ' /* RGB */,
    ImageEventsPSpcNamed = 'NAME' /* Named */,
    ImageEventsPSpcLab = 'Lab ' /* Lab */,
    ImageEventsPSpcGray = 'GRAY' /* Gray */,
    ImageEventsPSpcFiveColor = '5CLR' /* Five color */,
    ImageEventsPSpcFiveChannel = 'MCH5' /* Five channel */,
    ImageEventsPSpcEightColor = '8CLR' /* Eight color */,
    ImageEventsPSpcEightChannel = 'MCH8' /* Eight channel */,
    ImageEventsPSpcCMYK = 'CMYK' /* CMYK */,
enum ImageEventsPSpc {
typedef enum ImageEventsPQua ImageEventsPQua;
};
    ImageEventsPQuaNormal = 'Qua0' /* normal */
    ImageEventsPQuaDraft = 'Qua1' /* draft */,
    ImageEventsPQuaBest = 'Qua2' /* best */,
enum ImageEventsPQua {
```

```
@property (copy, readonly) NSString *version; // The version number of the application.
@property (readonly) BOOL frontmost; // Is this the active application?
@property (copy, readonly) NSString *name; // The name of the application.
- (SBElementArray<ImageEventsWindow *> *) windows;
- (SBElementArray<ImageEventsDocument *> *) documents;
@interface ImageEventsApplication : SBApplication
// The application's top-level scripting object.
*\
* Standard Suite
@end
- (void) unembed; // Remove any embedded ICC profiles from an image
- (void) scaleByFactor:(double)byFactor toSize:(NSInteger)toSize; // Scale an image
Save an image to a file in one of various formats
PackBits:(BOOL)PackBits withCompressionLevel:(ImageEventsCmlv)withCompressionLevel; //
- (ImageEventsAlias *) saveAs:(ImageEventsTypv)as icon:(BOOL)icon in:(NSString *)in_
- (void) rotateToAngle:(double)toAngle; // Rotate an image
(NSArray<NSNumber *> *)withPadColor; // Pad an image
- (void) padToDimensions:(NSArray<NSNumber *> *)toDimensions withPadColor:
- (void) matchToDestination:(ImageEventsProfile *)toDestination; // Match an image
- (void) flipHorizontal:(B00L)horizontal vertical:(B00L)vertical; // Flip an image
profile
- (void) embedWithSource:(ImageEventsProfile *)withSource; // Embed an image with an ICC
- (void) cropToDimensions:(NSArray<NSNumber *> *)toDimensions; // Crop an image
image
- (void) closeSaving:(ImageEventsSavo)saving savingIn:(NSString *)savingIn; // Close an
- (void) moveTo:(SBObject *)to; // Move an object to a new location.
Copy an object.
- (void) duplicateTo:(SBObject *)to withProperties:(NSDictionary *)withProperties; //
- (void) delete; // Delete an object.
(BOOL)printDialog; // Print a document.
- (void) printWithProperties:(NSDictionary *)withProperties printDialog:
document.
- (void) saveIn:(ImageEventsFile *)in_ as:(ImageEventsSaveableFileFormat)as; // Save a
// Close a document.
- (void) closeSaving:(ImageEventsSaveOptions)saving savingIn:(ImageEventsFile *)savingIn;
@protocol ImageEventsGenericMethods
typedef enum ImageEventsSaveableFileFormat ImageEventsSaveableFileFormat;
};
   ImageEventsSaveableFileFormatText = 'ctxt' /* Text File Format */
enum ImageEventsSaveableFileFormat {
typedef enum ImageEventsTypv ImageEventsTypv;
}:
   ImageEventsTypvTIFF = 'TIFF' /* TIFF */
   ImageEventsTypvQuickTimeImage = 'qtif' /* QuickTime Image */,
```

- (id) open:(id)x; // Open a document.
- (void) print:(id)x withProperties:(NSDictionary *)withProperties printDialog:
 (B00L)printDialog; // Print a document.
- (void) quitSaving:(ImageEventsSaveOptions)saving; // Quit the application.
- (B00L) exists:(id)x; // Verify that an object exists.

- (SBElementArray<ImageEventsDiskItem *> *) diskItems;- (SBElementArray<ImageEventsDomain *> *) domains;

@interface ImageEventsApplication (DiskFolderFileSuite)

// The Disk-Folder-File specific extensions to the application

- (SBElementArray<ImageEventsAlias *> *) aliases;- (SBElementArray<ImageEventsDisk *> *) disks;

- (id) move:(id)x to:(id)to; // Move disk item(s) to a new location.

@interface ImageEventsDocument : SBObject <ImageEventsGenericMethods>

- (ImageEventsFile *) open:(id)x; // Open disk item(s) with the appropriate application.

@end

// A document.

*\

*

* Disk-Folder-File Suite

```
@end
are displayed in the window.
@property (copy, readonly) ImageEventsDocument *document; // The document whose contents
@property BOOL zoomed; // Is the window zoomed right now?
@property (readonly) BOOL zoomable; // Does the window have a zoom button?
@property BOOL visible; // Is the window visible right now?
@property (readonly) B00L resizable; // Can the window be resized?
@property BOOL miniaturized; // Is the window minimized right now?
@property (readonly) BOOL miniaturizable; // Does the window have a minimize button?
@property (readonly) BOOL closeable; // Does the window have a close button?
@property NSRect bounds; // The bounding rectangle of the window.
@property NSInteger index; // The index of the window, ordered front to back.
- (NSInteger) id; // The unique identifier of the window.
@property (copy, readonly) NSString *name; // The title of the window.
@interface ImageEventsWindow : SBObject <ImageEventsGenericMethods>
// A window.
@end
one.
@property (copy, readonly) ImageEventsFile *file; // Its location on disk, if it has
@property (readonly) BOOL modified; // Has it been modified since the last save?
@property (copy, readonly) NSString *name; // Its name.
```

folders belonging to the User

collection of folders belonging to the System

Items folder

was loaded

```
Items folder
@property (copy, readonly) ImageEventsFolder *speakableItemsFolder; // The Speakable
@property (copy, readonly) ImageEventsFolder *sitesFolder; // The user's Sites folder
Documents folder
@property (copy, readonly) ImageEventsFolder *sharedDocumentsFolder; // The Shared
folder
@property (copy, readonly) ImageEventsFolder *scriptsFolder; // The user's Scripts
Additions folder
@property (copy, readonly) ImageEventsFolder *scriptingAdditionsFolder; // The Scripting
@property (copy, readonly) ImageEventsFolder *publicFolder; // The user's Public folder
Preferences folder
@property (copy, readonly) ImageEventsFolder *preferencesFolder; // The user's
folder
@property (copy, readonly) ImageEventsFolder *picturesFolder; // The user's Pictures
collection of folders residing on the Network
@property (copy, readonly) ImageEventsNetworkDomainObject *networkDomain; // the
@property (copy, readonly) ImageEventsFolder *musicFolder; // The user's Music folder
@property (copy, readonly) ImageEventsFolder *moviesFolder; // The user's Movies folder
of folders residing on the Local machine
@property (copy, readonly) ImageEventsLocalDomainObject *localDomain; // the collection
@property (copy, readonly) ImageEventsFolder *libraryFolder; // The Library folder
currently logged in user
@property (copy, readonly) ImageEventsFolder *homeFolder; // The Home folder of the
@property (copy, readonly) ImageEventsFolder *fontsFolder; // The Fonts folder
Folder Action Scripts folder
@property (copy, readonly) ImageEventsFolder *FolderActionScriptsFolder; // The user's
@property (copy, readonly) ImageEventsFolder *favoritesFolder; // The user's Favorites
folder
@property (copy, readonly) ImageEventsFolder *downloadsFolder; // The user's Downloads
folder
@property (copy, readonly) ImageEventsFolder *documentsFolder; // The user's Documents
Pictures folder
@property (copy, readonly) ImageEventsFolder *desktopPicturesFolder; // The Desktop
tolder
@property (copy, readonly) ImageEventsFolder *desktopFolder; // The user's Desktop
collection of folders belonging to the Classic System
@property (copy, readonly) ImageEventsClassicDomainObject *ClassicDomain; // the
Applications folder
@property (copy, readonly) ImageEventsFolder *applicationsFolder; // The user's
Application Support folder
@property (copy, readonly) ImageEventsFolder *applicationSupportFolder; // The
- (SBElementArray<ImageEventsFolder *> *) folders;
- (SBElementArray<ImageEventsFilePackage *> *) filePackages;
- (SBElementArray<ImageEventsFile *> *) files;
```

@property (copy, readonly) ImageEventsUserDomainObject *userDomain; // the collection of

@property (copy, readonly) ImageEventsDisk *startupDisk; // the disk from which Mac OS X

@property (copy, readonly) ImageEventsFolder *temporaryItemsFolder; // The Temporary

@property (copy, readonly) ImageEventsFolder *trash; // The user's Trash folder

@property (copy, readonly) ImageEventsSystemDomainObject *systemDomain; // the

```
@end
(id) moveTo:(id)to; // Move disk item(s) to a new location.
- (void) delete; // Delete disk item(s).
resides
@property (copy, readonly) NSString *volume; // the volume on which the disk item
@property BOOL visible; // Is the disk item visible?
@property (copy, readonly) NSString *URL; // the URL of the disk item
@property (readonly) NSInteger size; // the logical size of the disk item
disk item
@property (copy, readonly) NSString *POSIXPath; // the POSIX file system path of the
@property (readonly) NSInteger physicalSize; // the actual space used by the disk item
@property (copy, readonly) NSString *path; // the file system path of the disk item
@property (readonly) BOOL packageFolder; // Is the disk item a package?
@property (copy, readonly) NSString *nameExtension; // the extension portion of the name
@property (copy) NSString *name; // the name of the disk item
modified
@property (copy) NSDate *modificationDate; // the date on which the disk item was last
- (NSString *) id; // the unique ID of the disk item
displayed in the User Interface
@property (copy, readonly) NSString *displayedName; // the name of the disk item as
created
@property (copy, readonly) NSDate *creationDate; // the date on which the disk item was
has this disk item as an element
@property (copy, readonly) ImageEventsDiskItem *container; // the folder or disk which
@property (readonly) BOOL busyStatus; // Is the disk item busy?
@interface ImageEventsDiskItem : SBObject <ImageEventsGenericMethods>
// An item stored in the file system
@end
Workflows folder
@property (copy, readonly) ImageEventsFolder *workflowsFolder; // The Automator
@property (copy, readonly) ImageEventsFolder *utilitiesFolder; // The Utilities folder
```

```
// An alias in the file system
@interface ImageEventsAlias : ImageEventsDiskItem
```

- (SBElementArray<ImageEventsAlias *> *) aliases;
- (SBElementArray<ImageEventsDiskItem *> *) diskItems;
- (SBElementArray<ImageEventsFile *> *) files;
- (SBElementArray<ImageEventsFilePackage *> *) filePackages;
- (SBElementArray<ImageEventsFolder *> *) folders;

```
@property (copy) id creatorType; // the OSType identifying the application that created
the alias
@property (copy) id defaultApplication; // the application that will launch if the alias
is opened
@property (copy) id fileType; // the OSType identifying the type of data contained in
```

the alias (copy, readonly) NSString *kind; // The kind of alias, as shown in Finder

@property (copy, readonly) NSString *productVersion; // the version of the product

```
@property (copy, readonly) ImageEventsFolder *fontsFolder; // The Fonts folder
Action Scripts folder
@property (copy, readonly) ImageEventsFolder *FolderActionScriptsFolder; // The Folder
Pictures folder
@property (copy, readonly) ImageEventsFolder *desktopPicturesFolder; // The Desktop
folder
@property (copy, readonly) ImageEventsFolder *applicationsFolder; // The Applications
Application Support folder
@property (copy, readonly) ImageEventsFolder *applicationSupportFolder; // The
- (SBElementArray<ImageEventsFolder *> *) folders;
@interface ImageEventsDomain : SBObject <ImageEventsGenericMethods>
// A domain in the file system
@end
volumes only
@property (copy, readonly) id zone; // the zone in which the disk's server resides, AFP
@property (readonly) BOOL startup; // Is this disk the boot disk?
volumes only
@property (copy, readonly) id server; // the server on which the disk resides, AFP
file server)?
@property (readonly) BOOL localVolume; // Is the media a local volume (as opposed to a
@property BOOL ignorePrivileges; // Ignore permissions on this disk?
disk
@property (copy, readonly) NSNumber *freeSpace; // the number of free bytes left on the
@property (readonly) ImageEventsEdfm format; // the file system format of this disk
0U);
@property (readonly) BOOL ejectable; // Can the media be ejected (floppies, CD's, and so
used) on the disk
@property (copy, readonly) NSNumber *capacity; // the total number of bytes (free or
- (SBElementArray<ImageEventsFolder *> *) folders;
- (SBElementArray<ImageEventsFilePackage *> *) filePackages;
- (SBElementArray<ImageEventsFile *> *) files;
- (SBElementArray<ImageEventsDiskItem *> *) diskItems;
- (SBElementArray<ImageEventsAlias *> *) aliases;
@interface ImageEventsDisk : ImageEventsDiskItem
// A disk in the file system
@end
referenced by the alias (visible at the bottom of the "Get Info" window)
@property (copy, readonly) NSString *version; // the version of the application bundle
@property (copy, readonly) NSString *typeIdentifier; // The type identifier of the alias
@property BOOL stationery; // Is the alias a stationery pad?
application bundle referenced by the alias
@property (copy, readonly) NSString *shortVersion; // the short version of the
(visible at the top of the "Get Info" window)
```

@property (copy, readonly) ImageEventsFolder *libraryFolder; // The Library folder

- (NSString *) id; // the unique identifier of the domain

```
@property (copy, readonly) NSString *name; // the name of the domain
@property (copy, readonly) ImageEventsFolder *preferencesFolder; // The Preferences
folder
@property (copy, readonly) ImageEventsFolder *scriptingAdditionsFolder; // The Scripting
Additions folder
@property (copy, readonly) ImageEventsFolder *scriptsFolder; // The Scripts folder
@property (copy, readonly) ImageEventsFolder *sharedDocumentsFolder; // The Shared
Documents folder
Items folder
@property (copy, readonly) ImageEventsFolder *speakableItemsFolder; // The Speakable
@property (copy, readonly) ImageEventsFolder *utilitiesFolder; // The Utilities folder
@property (copy, readonly) ImageEventsFolder *workflowsFolder; // The Automator
@property (copy, readonly) ImageEventsFolder *workflowsFolder; // The Automator
@property (copy, readonly) ImageEventsFolder *workflowsFolder; // The Automator
@end
```

```
folder
@property (copy, readonly) ImageEventsFolder *startupItemsFolder; // The StartupItems
folder
@property (copy, readonly) ImageEventsFolder *shutdownFolder; // The Shutdown Items
Preferences folder
@property (copy, readonly) ImageEventsFolder *preferencesFolder; // The Classic
folder
@property (copy, readonly) ImageEventsFolder *launcherItemsFolder; // The Launcher Items
@property (copy, readonly) ImageEventsFolder *fontsFolder; // The Fonts folder
@property (copy, readonly) ImageEventsFolder *extensionsFolder; // The Extensions folder
folder
@property (copy, readonly) ImageEventsFolder *desktopFolder; // The Classic Desktop
Strip Modules folder
@property (copy, readonly) ImageEventsFolder *controlStripModulesFolder; // The Control
folder
@property (copy, readonly) ImageEventsFolder *controlPanelsFolder; // The Control Panels
folder
@property (copy, readonly) ImageEventsFolder *appleMenuFolder; // The Apple Menu Items
```

@end

// A file in the file system

- (SBElementArray<ImageEventsFolder *> *) folders;

@interface ImageEventsClassicDomainObject : ImageEventsDomain

```
@interface ImageEventsFile : ImageEventsDiskItem
@property (copy) id creatorType; // the OSType identifying the application that created
the file
@property (copy) id defaultApplication; // the application that will launch if the file
is opened
@property (copy) id fileType; // the OSType identifying the type of data contained in
the file
@property (copy, readonly) NSString *kind; // The kind of file, as shown in Finder
```

@property (copy, readonly) ImageEventsFolder *systemFolder; // The System folder

```
@property (copy, readonly) NSString *productVersion; // the version of the product
(visible at the top of the "Get Info" window)
@property (copy, readonly) NSString *shortVersion; // the short version of the file
@property BOOL stationery; // Is the file a stationery pad?
@property (copy, readonly) NSString *typeIdentifier; // The type identifier of the file
@property (copy, readonly) NSString *version; // the version of the file (visible at the
bottom of the "Get Info" window)
```

- (id) open; // Open a document.
- (ImageEventsFile *) open; // Open disk item(s) with the appropriate application.

@end

```
// A file package in the file system
@interface ImageEventsFilePackage : ImageEventsFile
```

- (SBElementArray<ImageEventsAlias *> *) aliases;
- (SBElementArray<ImageEventsDiskItem *> *) diskItems;
- (SBElementArray<ImageEventsFile *> *) files;
- (SBElementArray<ImageEventsFilePackage *> *) filePackages;
- (SBElementArray<ImageEventsFolder *> *) folders;

@end

```
// A folder in the file system
@interface ImageEventsFolder : ImageEventsDiskItem
```

- (SBElementArray<ImageEventsAlias *> *) aliases;
- (SBElementArray<ImageEventsDiskItem *> *) diskItems;
- (SBElementArray<ImageEventsFile *> *) files;
- (SBElementArray<ImageEventsFilePackage *> *) filePackages;
- (SBElementArray<ImageEventsFolder *> *) folders;

@end

```
// The local domain in the file system 
 @interface ImageEventsLocalDomainObject : ImageEventsDomain
```

- (SBElementArray<ImageEventsFolder *> *) folders;

@end

```
// The network domain in the file system
@interface ImageEventsNetworkDomainObject : ImageEventsDomain
```

- (SBElementArray<ImageEventsFolder *> *) folders;

@end

```
// The system domain in the file system
@interface ImageEventsSystemDomainObject : ImageEventsDomain
```

- (SBElementArray<ImageEventsFolder *> *) folders;

- (SBElementArray<ImageEventsProfile *> *) profiles;

// An image contained in a file

@end

- (SBElementArray<ImageEventsMetadataTag *> *) metadataTags;

@interface ImageEventsImage : SBObject <ImageEventsGenericMethods>

```
@end
```

```
@property (copy, readonly) NSString *name; // the name of the display
display
@property (copy, readonly) ImageEventsProfile *displayProfile; // the profile for the
@property (readonly) NSInteger displayNumber; // the number of the display
@interface ImageEventsDisplay : SBObject <ImageEventsGenericMethods>
// A monitor connected to the computer
*\
* Image Suite
\*
@end
Items folder
@property (copy, readonly) ImageEventsFolder *temporaryItemsFolder; // The Temporary
@property (copy, readonly) ImageEventsFolder *sitesFolder; // The user's Sites folder
@property (copy, readonly) ImageEventsFolder *publicFolder; // The user's Public folder
folder
@property (copy, readonly) ImageEventsFolder *picturesFolder; // The user's Pictures
@property (copy, readonly) ImageEventsFolder *musicFolder; // The user's Music folder
@property (copy, readonly) ImageEventsFolder *moviesFolder; // The user's Movies folder
@property (copy, readonly) ImageEventsFolder *homeFolder; // The user's Home folder
folder
@property (copy, readonly) ImageEventsFolder *favoritesFolder; // The user's Favorites
folder
@property (copy, readonly) ImageEventsFolder *downloadsFolder; // The user's Downloads
@property (copy, readonly) ImageEventsFolder *documentsFolder; // The user's Documents
folder
@property (copy, readonly) ImageEventsFolder *desktopFolder; // The user's Desktop
- (SBElementArray<ImageEventsFolder *> *) folders;
@interface ImageEventsUserDomainObject : ImageEventsDomain
// The user domain in the file system
```

```
@property (readonly) ImageEventsPPCS connectionSpace; // the connection space of the
@property (readonly) ImageEventsPSpc colorSpace; // the color space of the profile
@interface ImageEventsProfile : SBObject <ImageEventsGenericMethods>
// A ColorSync ICC profile.
@end
@property (copy, readonly) id value; // the current setting of the tag
@property (copy, readonly) NSString *name; // the name of the tag
@interface ImageEventsMetadataTag : SBObject <ImageEventsGenericMethods>
// A metadata tag: EXIF, IPTC, etc.
@end
vertical pixel density of the image, respectively, in dots per inch
@property (copy, readonly) NSArray<NSNumber *> *resolution; // the horizontal and
@property (copy, readonly) NSString *name; // the name of the image
encloses the file that contains the image
@property (copy, readonly) ImageEventsDiskItem *location; // the folder or disk that
image
@property (copy, readonly) ImageEventsFile *imageFile; // the file that contains the
@property (copy, readonly) id fileType; // file type of the image's file
embedded in the image
@property (copy, readonly) ImageEventsProfile *embeddedProfile; // the profile, if any,
the image, respectively, in pixels
@property (copy, readonly) NSArray<NSNumber *> *dimensions; // the width and height of
representation
@property (readonly) ImageEventsPSpc colorSpace; // color space of the image's color
representation
@property (readonly) ImageEventsBitz bitDepth; // bit depth of the image's color
```

@property (copy, readonly) NSDate *creationDate; // the creation date of the profile profile

the profile @property (copy, readonly) NSString *deviceManufacturer; // the device manufacturer of @property (readonly) ImageEventsPCla deviceClass; // the device class of the profile @property (copy, readonly) NSString *creator; // the creator type of the profile

profile @property (readonly) ImageEventsPRdr renderingIntent; // the rendering intent of the @property (readonly) ImageEventsPQua quality; // the quality of the profile @property (copy, readonly) NSString *preferredCMM; // the preferred CMM of the profile @property (copy, readonly) NSString *platform; // the intended platform of the profile @property (copy, readonly) NSString *name; // the description text of the profile @property (copy, readonly) id location; // the file location of the profile @property (readonly) NSInteger deviceModel; // the device model of the profile

@property (copy, readonly) NSString *version; // the version number of the profile @property (readonly) NSInteger size; // the size of the profile in bytes

```
location
@property (copy) ImageEventsFile *systemProfileLocation; // the default system profile
@property (copy) ImageEventsProfile *systemProfile; // the default system profile
quitting; if set to zero, idle time will not cause the application to quit
@property NSInteger quitDelay; // the time in seconds the application will idle before
@property (copy, readonly) id profileFolder; // the ColorSync profile folder
to use, or "automatic"
@property (copy) NSString *preferredCMM; // specifies preferred Color Management Module
location
@property (copy) ImageEventsFile *defaultXYZProfileLocation; // the default XYZ profile
@property (copy) ImageEventsProfile *defaultXYZProfile; // the default XYZ profile
location
@property (copy) ImageEventsFile *defaultRGBProfileLocation; // the default RGB profile
@property (copy) ImageEventsProfile *defaultRGBProfile; // the default RGB profile
location
@property (copy) ImageEventsFile *defaultLabProfileLocation; // the default Lab profile
@property (copy) ImageEventsProfile *defaultLabProfile; // the default Lab profile
profile location
@property (copy) ImageEventsFile *defaultGrayProfileLocation; // the default Gray
@property (copy) ImageEventsProfile *defaultGrayProfile; // the default Gray profile
profile location
@property (copy) ImageEventsFile *defaultCMYKProfileLocation; // the default CMYK
@property (copy) ImageEventsProfile *defaultCMYKProfile; // the default CMYK profile
- (SBElementArray<ImageEventsProfile *> *) profiles;
- (SBElementArray<ImageEventsImage *> *) images;
- (SBElementArray<ImageEventsDisplay *> *) displays;
@interface ImageEventsApplication (ImageEventsSuite)
// The Image Events application
*\
```

@end

* Image Events Suite
