MOLE Class Reference

MOLE 0.1 22-05-2009

MOLE Class Reference

Inherits from	NSObject
Conforms to	NSObject(NSObject)
Framework	/Library/Frameworks/MOLE.framework
Availability	Available from 45RPM Software.
Declared in	MOLE.h
Related sample code	

Overview

mole ¹ |məʊl| noun

1 a small burrowing insectivorous mammal with dark velvety fur, a long muzzle, and very small eyes. Family Talpidae: several genera and species, including the **eastern mole** (Scalopus aquaticus) of North America.
2 a spy who achieves over a long period an important position within the security defenses of a country. Someone within an organization who anonymously betrays confidential information.
3 a Mac OS X framework to read and, eventually, write Microsoft OLE format documents.

`MOLE` provides a simple means to read and, eventually, write Microsoft OLE format documents and also the structures that are commonly found inside. Whilst `MOLE` removes the need for the programmer to understand how the OLE format is constructed, it will still be necessary for the programmer to understand what structures a particular program (for example, Microsoft Word) writes and how these individual structures are formatted.

`MOLE` is written in Objective C for i386 and PPC architectures.

Acknowledgements

I owe a very great debt of gratitude to Daniel Rentz of the OpenOffice project for his excellent document covering the OLE format, and also for the advice that he gave to me in conversations via email. http://sc.openoffice.org/compdocfileformat.pdf

I am also indebted to Anja Schaffhirt for her guide to the Summary Information Stream. http://sedna-soft.de/summary-information-stream/

I could not have written this software without Hex Fiend by RidiculousFish.

I also relied on the following books for reference:

Programming in Objective-C by Stephen Kochan

Cocoa Programming for Mac OS X by Aaron Hillegass

Advanced Mac OS X Programming by Aaron Hillegass and Mark Dalrymple

I wouldn't have written this software if it weren't for the fact that my wife wrote beautiful e-mails to me before we were married, and these e-mails were unfortunately archived in MS Outlook format. Well. I had to extract the contents somehow, didn't I? And, having written an extractor program, I decided it would form a good basis for an Objective C OLE framework.

Tasks

Initialization

- `initialize:filename:` (page n)

  `MOLE` currently requires initialization to be carried out as a task distinct from the '`init`'ing of the class. Developers who use this early alpha version of `MOLE` need to be aware that this method will be deprecated in the future although, for the sake of compatibility, I do not plan to remove it entirely.

- `initWithContentsOfFile:filename:` (page n)

  Returns an object initialized with the path to the OLE file for reading.

- `initWithData:dataname:` (page n)

  Returns an object initialized with an NSData object.

- `openOLE:filename:` (page n)

  Prepares the object with the path to the OLE file for reading, and returns a boolean value indicating success or failure.

- `openData:dataname:` (page n)

  Prepares the object with an NSData object, and returns a boolean value indicating success or failure.

Accessing Data

- `getDirectory:` (page n)

  Returns an `NSArray` containing the storage structure of the OLE file open in the receiver.

- `dumpStream:streamName:width:` (page n)

  Returns an `NSArray` containing a hex dump of a specified stream from the OLE file open in the receiver.

- `getStream:streamName:` (page n)

  Returns an `NSData` object containing the data contained within a specified stream from the OLE file open in the receiver.

- `extractToFile:streamName:filename:` (page n)

  Saves the data contained within a specified stream to the path specified by the filename, and returns a boolean indicating success or failure.

- `getOLEDictionary:` (page n)

  Returns the entire contents of an OLE file as an `NSDictionary`.

- `getOLEArray:` (page n)

  Returns the entire contents of an OLE file as an `NSArray`.

Methods

initialize

Initializes `MOLE`.

`- initialize:`

Parameters

None

Return Value

Void

Discussion

Early versions of `MOLE` required initialization to be carried out as a task distinct from the construction of the class instance. Developers who used this early alpha version of `MOLE` need to be aware that this method will be deprecated in the future, although it will be retained for backwards compatibility. If the `initialize` method is to be used, then `MOLE` will need to be inited and have the file opened separately as well.

```
Mole* reader = [[Mole alloc] init];
[reader initialize];
[reader openOLE:@"/afile.doc"];
```

Availability

Available in MOLE 0.1a and later.

See Also

`- openOLE:filename:`

`- initWithContentsOfFile:filename:`

Declared In

`MOLE.h`

initWithContentsOfFile

Initializes `MOLE` with the path of the OLE file to be opened.

`- initWithContentsOfFile:filename:`

Parameters

filename `NSString`

The path of the OLE file to be opened.

Return Value

Void

Discussion

This is a 'correct' method for initializing `MOLE`. Use of this method negates the need to call `initialize` and `openOLE` separately.

```
Mole* reader = [[Mole alloc] initWithContentsOfFile:@"/afile.doc"];
```

Availability

Available in MOLE 0.2a and later.

See Also

- `initialize:`

Declared In

`MOLE.h`

initWithData

Initializes `MOLE` with an `NSData` oject.

- `initWithData:dataname:`

Parameters

dataname `NSData`

An `NSData` object containing the bytes from an OLE document.

Return Value

Void

Discussion

This is a 'correct' method for initializing `MOLE`. Use of this method negates the need to call `initialize` and `openData` separately.

```
Mole* reader = [[Mole alloc] initWithData:nsdataobject];
```

Availability

Available in MOLE 0.1a and later.

See Also

- `initialize:`

Declared In

`MOLE.h`

openOLE

Specifies the path of an OLE document file for opening.

`- openOLE:filename:`

Parameters

filename `NSString`

The path of the OLE file to be opened.

Return Value

Boolean indicating success or failure.

Discussion

In the event that `MOLE` is instantiated using `init` and `initialize`, `openOLE` is used to specify the filename that `MOLE` will be used to read.

```
Mole* reader = [[Mole alloc] init];
[reader initialize];
[reader openOLE:@"/afile.doc"];
```

Availability

Available in MOLE 0.1a and later.

See Also

`- initialize:`

Declared In

`MOLE.h`

openData

Specifies the path of an OLE document file for opening.

`- openData:dataname:`

Parameters

dataname `NSData`

An `NSData` object containing the bytes from an OLE document.

Return Value

Boolean indicating success or failure.

Discussion

In the event that `MOLE` is instantiated using `init` and `initialize`, `openData` is used to specify the `NSData` object that `MOLE` will be used to read.

```
Mole* reader = [[Mole alloc] init];
[reader initialize];
[reader openData:nsdataobject];
```

Availability

Available in MOLE 0.2a and later. Provided by special request for an NSData analogue to openOLE.

See Also

`- initialize:`

Declared In

`MOLE.h`

getDirectory

Returns the 'directory structure' of the OLE document open in the receiver.

`- getDirectory:`

Parameters

None.

Return Value

`NSArray` of `NSDictionary` containing the following entries.

`entryname`	the path of the directory entry (streamName)
`entrysize`	the size of the stream, which is always zero for a directory.

Discussion

`getDirectory` is used to verify what streams exist in an OLE file. Once the directory structure `NSArray` has been retrieved, the entries within the `NSDictionary` are used by other methods in the `MOLE` framework to retrieve the actual content.

```
NSArray* dirarray= [[NSArray alloc]
                    initWithArray:[reader getDirectory]];
```

Availability

Available in MOLE 0.1a and later.

Declared In

`MOLE.h`

dumpStream

Generates a hex dump of the specified stream.

- `dumpStream:streamName:width:`

Parameters

streamName `NSString`

The OLE path of the stream to be dumped. For example, a root level stream might be `__substg1.0_0070001F`. A nested stream might be `/__nameid_version1.0/__substg1.0_10090102`. The 'entryname' entry of the `NSArray` returned by `getDirectory` can be used to provide this parameter.

width `int`

The 'window' width of the required dump - i.e. the number of bytes to be dumped on each line.

Return Value

`NSArray` of `NSDictionary` containing the following entries.

`offset`	the offset within the current stream.
`bytes`	the contents of the current offset 'window', in bytes.
`ascii`	the contents of the current offset 'window', in ascii.

Discussion

dumpStream is provided as an aid to debugging software that is intended to read OLE files. It permits the developer to look at the contents of an OLE file easily without having to manually hunt through the file using a Hex editor.

```
NSArray* dumparray = [[NSArray alloc] initWithArray:
                         [reader dumpStream:@"/astream":16]];
```

Availability

Available in MOLE 0.1a and later.

Declared In

`MOLE.h`

getStream

Returns the contents of an OLE stream.

- `getStream:streamName:`

Parameters

streamName `NSString`

The OLE path of the stream to be retrieved. For example, a root level stream might be `__substg1.0_0070001F`. A nested stream might be `/__nameid_version1.0/`

`__substg1.0_10090102`. The 'entryname' entry of the `NSArray` returned by `getDirectory` can be used to provide this parameter.

Return Value

`NSData` containing the contents of the requested stream.

Discussion

`getStream` is used to retrieve the contents of an OLE stream for further processing.

```
NSData* streamdata = [[NSData alloc]
                          initWithData:[reader getStream:@"/astream"]];
```

Availability

Available in MOLE 0.1a and later.

See Also

`- extractToFile:streamName:filename:`

Declared In

`MOLE.h`

extractToFile

Returns the contents of an OLE stream.

`- extractToFile:streamName:filename:`

Parameters

streamName `NSString`

The OLE path of the stream to be retrieved. For example, a root level stream might be `__substg1.0_0070001F`. A nested stream might be `/__nameid_version1.0/__substg1.0_10090102`. The 'entryname' entry of the `NSArray` returned by `getDirectory` can be used to provide this parameter.

filename `NSString`

The path of the output file that the stream will be dumped into.

Return Value

Boolean indicating success or failure.

Discussion

`extractToFile` is used to save the contents of an OLE stream to file.

```
[reader extractToFile:@"/astream" :@"/outputfile"];
```

Availability

Available in MOLE 0.1a and later.

See Also

`- getStream:streamName:`

Declared In

`MOLE.h`

getOLEDictionary

Returns the entire contents of an OLE file as an `NSDictionary`.

`- getOLEDictionary:`

Parameters

None

Return Value

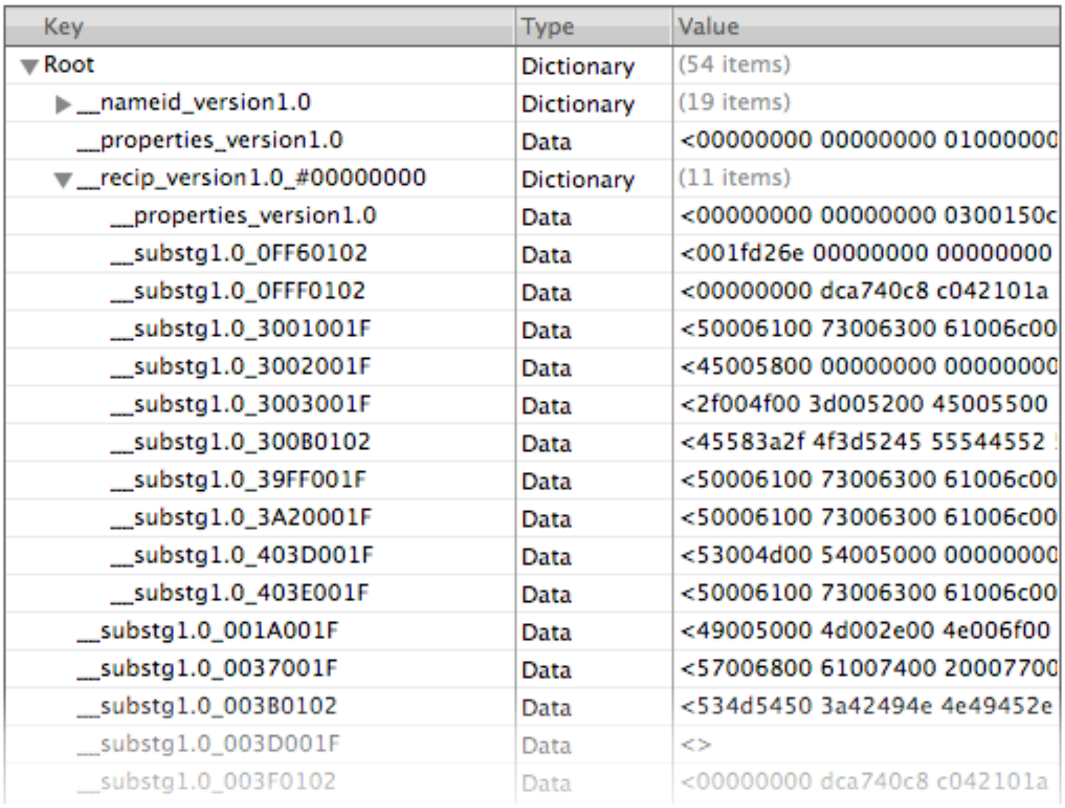
`NSDictionary` containing the OLE document tree. If the item in the tree is a data stream, it is stored as an `NSData` object. If the item is a sub-directory, it is stored as an `NSDictionary` object. In either case, the object key is equal to the name of the OLE stream.

Discussion

`getOLEDictionary` is used to return the entire contents of the OLE file, with its internal directory tree intact, as an `NSDictionary`.

```
NSDictionary* oledict = [[NSDictionary alloc] initWithDictionary:
                         [reader getOLEDictionary]];
```

The generated `NSDictionary` can be written to file using the `NSDictionary` method `writeToFile` and then opened using ‘Property List Editor’. As an example, doing so on a Microsoft Outlook .msg file yields the following structure:



Key	Type	Value
▼ Root	Dictionary	(54 items)
▶ __nameid_version1.0	Dictionary	(19 items)
__properties_version1.0	Data	<00000000 00000000 01000000
▼ __recip_version1.0_#00000000	Dictionary	(11 items)
__properties_version1.0	Data	<00000000 00000000 0300150c
__substg1.0_0FF60102	Data	<001fd26e 00000000 00000000
__substg1.0_0FFF0102	Data	<00000000 dca740c8 c042101a
__substg1.0_3001001F	Data	<50006100 73006300 61006c00
__substg1.0_3002001F	Data	<45005800 00000000 00000000
__substg1.0_3003001F	Data	<2f004f00 3d005200 45005500
__substg1.0_300B0102	Data	<45583a2f 4f3d5245 55544552
__substg1.0_39FF001F	Data	<50006100 73006300 61006c00
__substg1.0_3A20001F	Data	<50006100 73006300 61006c00
__substg1.0_403D001F	Data	<53004d00 54005000 00000000
__substg1.0_403E001F	Data	<50006100 73006300 61006c00
__substg1.0_001A001F	Data	<49005000 4d002e00 4e006f00
__substg1.0_0037001F	Data	<57006800 61007400 20007700
__substg1.0_003B0102	Data	<534d5450 3a42494e 4e49452e
__substg1.0_003D001F	Data	<>
__substg1.0_003F0102	Data	<00000000 dca740c8 c042101a

Availability

Available in MOLE 0.1a and later.

See Also

- `getDirectory:`
- `getOLEArray:`
- `getStream:streamName:`

Declared In

`MOLE.h`

getOLEArray

Returns the entire contents of an OLE file as an `NSArray`. Usage of this method is not advised - in most situations getOLEDictionary should be used instead.

`- getOLEArray:`

Parameters

None

Return Value

`NSArray` of `NSDictionary` containing the OLE document tree in the following entries:

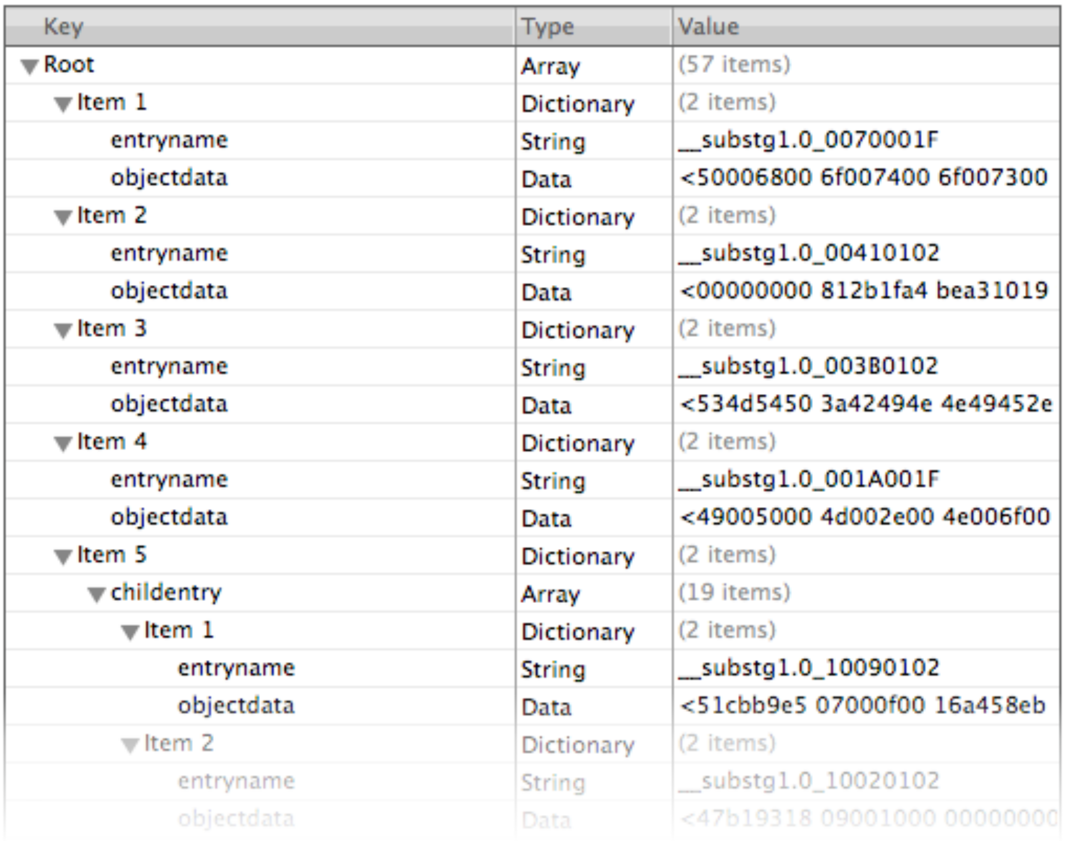
`entryname`	(Always Present) `NSString` containing the name of the current OLE entry, excluding the path.
`objectdata`	(Present for a data stream) `NSData` containing the content of the current OLE entry.
`childentry`	(Present for a 'subdirectory') `NSArray` containing the contents of the OLE subdirectory.

Discussion

`getOLEArray` is used to return the entire contents of the OLE file, with its internal directory tree intact, as an `NSArray`. Whilst this may on occasion be useful and has been included by request, I suggest that you might find getOLEDictionary more useful because it replicates the directory exactly.

```
NSArray* olearray = [[NSArray alloc] initWithArray:
                     [reader getOLEArray]];
```

The generated `NSArray` can be written to file using the `NSArray` method `writeToFile` and then opened using 'Property List Editor'. As an example, doing so on a Microsoft Outlook .msg file yields the following structure:



Key	Type	Value
▼Root	Array	(57 items)
▼Item 1	Dictionary	(2 items)
entryname	String	__substg1.0_0070001F
objectdata	Data	<50006800 6f007400 6f007300
▼Item 2	Dictionary	(2 items)
entryname	String	__substg1.0_00410102
objectdata	Data	<00000000 812b1fa4 bea31019
▼Item 3	Dictionary	(2 items)
entryname	String	__substg1.0_003B0102
objectdata	Data	<534d5450 3a42494e 4e49452e
▼Item 4	Dictionary	(2 items)
entryname	String	__substg1.0_001A001F
objectdata	Data	<49005000 4d002e00 4e006f00
▼Item 5	Dictionary	(2 items)
▼childentry	Array	(19 items)
▼Item 1	Dictionary	(2 items)
entryname	String	__substg1.0_10090102
objectdata	Data	<51cbb9e5 07000f00 16a458eb
▼Item 2	Dictionary	(2 items)
entryname	String	__substg1.0_10020102
objectdata	Data	<47b19318 09001000 0000000

Availability

Available in MOLE 0.3a and later.

See Also

- `getDirectory:`
- `getOLEDictionary:`
- `getStream:streamName:`

Declared In

`MOLE.h`