Framework: **Foundation** Class: **NSString** class item: **stringWithString:**

```
use framework "Foundation"

(current application's NSMutableString's stringWithString:"Hi") --> (NSString) "Hi"
(class of (current application's NSMutableString's stringWithString:"Hi")) --> (Class) __NSCFString
(class of (current application's NSString's stringWithString:"Hi")) --> (Class) __NSCFString

class of ((current application's NSMutableString's stringWithString:"Hi") as string) --> text
class of ((current application's NSString's stringWithString:"Hi") as string) --> text

(current application's NSMutableString's stringWithString:"Hi")'s className() -->(NSString) "__NSCFString"
(current application's NSString's stringWithString:"Hi")'s className() -->(NSString) "__NSCFString"

(current application's NSMutableString's stringWithString:"Hi")'s className() as string --> "__NSCFString"
(current application's NSString's stringWithString:"Hi")'s className() as string --> "__NSCFString"
----------------------------------------------------------------------------------------------------------------------

set TheLetter to (current application's NSString's stringWithString:"T") --> (NSString) "T"
(TheLetter's isEqual:"T") --> true
(TheLetter's isEqual:"F") --> false
----------------------------------------------------------------------------------------------------------------------

set TheNum to current application's NSString's stringWithString:"22.12345678901234567890"
TheNum's integerValue --> Returns (NSNumber) 22, AppleScript's "class of" = (Class) __NSCFNumber
TheNum's floatValue --> Returns (NSNumber) 22.12346, AppleScript's "class of" = (Class) __NSCFNumber
----------------------------------------------------------------------------------------------------------------------
```

```
use framework "Foundation"

set TheDictionary to current application's NSMutableDictionary's dictionaryWithDictionary:{firstLetter:"A", secondLetter:"B"}

TheDictionary --> (NSDictionary) {secondLetter:"B", firstLetter:"A"}

TheDictionary's valueForKey:"firstLetter" --> (NSString) "A"
```

```
use framework "Foundation"

set FileManager to current application's NSFileManager's defaultManager --> <NSFileManager: 0x7f8581603da0>

-- attributesOfItemAtPath returns an NSDictionary with that describes the attributes of the file, directory, symlink, ... of the ob
-- specified by the path given to attributesOfItemAtPath.
set FileAttributes to FileManager's attributesOfItemAtPath:"/Users/bill/Desktop/Folder_7/4.pages" |error|:(missing value)

(* FileAttributes --> (NSDictionary) {
                NSFileOwnerAccountID:502,
                NSFileHFSTypeCode:0,
                NSFileSystemFileNumber:148738226,
                NSFileExtensionHidden:NO,
                NSFileSystemNumber:16777220,
                NSFileSize:697727,
                NSFileGroupOwnerAccountID:80,
                NSFileOwnerAccountName:"bill",
                NSFileCreationDate:(NSDate) "2016-08-15 08:23:59 +0000",
                NSFilePosixPermissions:420,
                NSFileHFSCreatorCode:0,
                NSFileType:"NSFileTypeRegular",
                NSFileExtendedAttributes:{
                        com.apple.iwork.documentUUID#PS:(NSData) <1f7277ef 7f664351 a4570e55 5f7e85e6>,
                        com.apple.quarantine:(NSData) <30303032 3b353762 32376234 663b5061 6765733b>
                },
                NSFileGroupOwnerAccountName:"admin",
                NSFileReferenceCount:1,
                NSFileModificationDate:(NSDate) "2016-08-16 02:32:47 +0000"
        } *)
```

```
FileAttributes's objectForKey:(current application's NSFileCreationDate) --> (NSDate) "2016-08-15 08:23:59 +0000"
FileAttributes's objectForKey:(current application's NSFileType) --> (NSString) "NSFileTypeRegular"
FileAttributes's objectForKey:(current application's NSFilePosixPermissions) --> (NSNumber) 420
```

```
use framework "Foundation"

-- Create dictionary
set TheDictionary to current application's NSDictionary's dictionaryWithObjects:{1, 2} forKeys:{"A", "B"}

-- Access dictionary value
TheDictionary's valueForKey:"A"



-- Create dictionary
set NumToLetter to current application's NSDictionary's dictionaryWithObjectsAndKeys_(1, "A", 2, "B", 3, "C", 4, "D", 5, "E", 6
value)

-- Access dictionary value
NumToLetter's valueForKey:"A" --> (NSNumber) 1

set AnotherDictionary to current application's NSDictionary's dictionaryWithObjects:{"A", "B", "C"} forKeys:{"a", "b", "c"}
AnotherDictionary --> (NSDictionary) {a:"A", b:"B", c:"C"}
-- Access dictionary properties
AnotherDictionary's valueForKey:"@count" --> (NSNumber) 3
AnotherDictionary's valueForKey:"@allKeys" --> (NSArray) {"a", "b", "c"}
AnotherDictionary's valueForKey:"@allValues" -- > (NSArray) {"A", "B", "C"}
```

Framework: **Foundation** Class: **NSMutableDictionary** class item: **removeObjectForKey:**

```
use framework "Foundation"

set TheDictionary1 to current application's NSMutableDictionary's dictionaryWithDictionary:{firstLetter:"A", secondLetter:"B"}

TheDictionary1 --> (NSDictionary) {secondLetter:"B", firstLetter:"A"}

TheDictionary1's valueForKey:"firstLetter" --> (NSString) "A"

TheDictionary1's removeObjectForKey:"firstLetter"

TheDictionary1 --> (NSDictionary) {secondLetter:"B"}
```

```
use framework "Foundation"

-- Create a dictionary
set TheDictionary to current application's NSDictionary's dictionaryWithObjects:{"value1", "value2", "value3", "value4"} forKe
"key2", "key3", "key4"}
TheDictionary --> (NSDictionary) {key3:"value3", key1:"value1", key4:"value4", key2:"value2"}

-- Example of objectForKey
TheDictionary's objectForKey:"key2" --> (NSString) "value2"

-- This also works
TheDictionary's valueForKey:"key2" --> (NSString) "value2"

-- Search for a key that doesn't exists
TheDictionary's objectForKey:"key5" --> missing value
(TheDictionary's objectForKey:"key5") = missing value --> true

-- Shows the class of the dictionary.  I stands for immutable.
TheDictionary's className() --> (NSString) (NSString) "__NSDictionaryI"
```

Framework: **Foundation** Class: **NSMutableDictionary** class item: **removeAllObjects()**

```
use framework "Foundation"

set TheDictionary to current application's NSMutableDictionary's dictionaryWithObjects:{"value1", "value2", "value3", "value4'
forKeys:{"key1", "key2", "key3", "key4"}
TheDictionary --> (NSDictionary) {key3:"value3", key1:"value1", key4:"value4", key2:"value2"}
TheDictionary's removeAllObjects() --> Doesn't return result
TheDictionary --> (NSDictionary) {}
```

```
use framework "Foundation"

-- Create dictionary
set LetterToNumberDict to current application's NSDictionary's dictionaryWithObjectsAndKeys_(1, "A", 2, "B", 3, "C", 4, "D", 5
missing value)
LetterToNumberDict --> (NSDictionary) {A:1, F:6, D:4, B:2, E:5, C:3}

-- Access dictionary value
LetterToNumberDict's valueForKey:"D" --> (NSNumber) 4
```

```
use framework "Foundation"

-- Create dictionary
set TheDictionary to current application's NSDictionary's dictionaryWithObjects:{1, 2} forKeys:{"A", "B"}

-- Access dictionary value
TheDictionary's valueForKey:"A"
```

```
use framework "Foundation"

set TheDictionary5 to current application's NSDictionary's dictionaryWithObjects:{"B", "A", "B", "C", "B", "A"} forKeys:{"key1"
"key3", "key4", "key5", "key6"}

TheDictionary5 --> (NSDictionary) {key3:"B", key1:"B", key6:"A", key4:"C", key2:"A", key5:"B"}
TheDictionary5's allKeysForObject:"B" --> (NSArray) {"key3", "key1", "key5"}
TheDictionary5's allKeysForObject:"A" --> (NSArray) {"key6", "key2"}
TheDictionary5's allKeysForObject:"C" --> (NSArray) {"key4"}
TheDictionary5's allKeysForObject:"D" --> (NSArray) {}
```

```
use framework "Foundation"

set TheDictionary1 to current application's NSDictionary's dictionaryWithObjects:{1, 2} forKeys:{"A", "B"}

set TheDictionary2 to current application's NSDictionary's dictionaryWithObjects:{1, 2} forKeys:{"A", "B"}

set TheDictionary3 to current application's NSDictionary's dictionaryWithObjects:{1, 2} forKeys:{"C", "B"}


TheDictionary1's isEqualToDictionary:TheDictionary2 --> true

TheDictionary1's isEqualToDictionary:TheDictionary3 --> false
```

Framework: **Foundation** Class: **NSFileManager** class item: **changeCurrentDirectoryPath:**

```
use framework "Foundation"

set FileManager to current application's NSFileManager's defaultManager --> <NSFileManager: 0x7f8581603da0>

-- The current directory path was point at the root directory
set TheCurrentPath to FileManager's currentDirectoryPath --> (NSString) "/"

-- Set the current directory path to the desktop of bill's folder
FileManager's changeCurrentDirectoryPath:"/Users/bill/Desktop" --> true (Returns boolean depending on sucess of operation)
```

```
use framework "Foundation"

set NewFolderURL to current application's class "NSURL"'s fileURLWithPath:"/Users/bill/Desktop/TheNewFolder"
```

Framework: **Foundation** Class: **NSFileManager** class item: **createDirectoryAtURL:withInterme**

```
use framework "Foundation"
use scripting additions

set FileManager to current application's NSFileManager's defaultManager --> <NSFileManager: 0x7f8581603da0>

set TheCurrentPath to FileManager's currentDirectoryPath
set NewFolderURL to current application's class "NSURL"'s fileURLWithPath:"/Users/bill/Desktop/TheNewFolder"

set WasSucessful to FileManager's createDirectoryAtURL:NewFolderURL withIntermediateDirectories:true attributes:(missing va
|error|:(missing value)
if (not WasSucessful) then
	display dialog "Could not create the new directory." buttons {"OK"} default button "OK"
	return false
end if



current application's ClassName's withArray:""
ClassName's methodNameParameter1:Parameter1
ClassName's methodNameParameter1:Parameter1 withParameter2:Parameter2
ClassName's methodNameParameter1:Parameter1 withParameter2:Parameter2 withParameter3:Parameter3
ClassName's methodNameParameter1:Parameter1 withParameter2:Parameter2 withParameter3:Parameter3 withParameter4:P

Object's methodNameAtIndex:
```

```
use framework "Foundation"
use scripting additions

set FileManager to current application's NSFileManager's defaultManager --> <NSFileManager: 0x7f8581603da0>

-- When testing make sure TheNewFolder exists before testing the delete script
set WasSucessful to FileManager's removeItemAtPath:"/Users/bill/Desktop/TheNewFolder" |error|:(missing value)
if (not WasSucessful) then
	display dialog "Could not remove the item." buttons {"Cancel", "OK"} default button "OK"
	return false
end if
```

```
use framework "Foundation"

set MyTimeZone to current application's NSTimeZone's timeZoneWithAbbreviation:"PST"
```

Framework: **Foundation** Class: **NSFileManager** class item: **moveItemAtURL:toURL:error:**

```
use framework "Foundation"
use scripting additions

set FileManager to current application's NSFileManager's defaultManager --> <NSFileManager: 0x7f8581603da0>

set WasSucessful to FileManager's createFileAtPath:"/Users/bill/Desktop/Test folder/DFile.txt" |contents|:("Some sample text"
attributes:(missing value)
if (not WasSucessful) then
	display dialog "Could not create the file \"DFile.txt\"." buttons {"Cancel", "OK"} default button "OK"
	return false
end if

set OldFileURL to current application's class "NSURL"'s fileURLWithPath:"/Users/bill/Desktop/Test folder/DFile.txt"
set NewFileURL to current application's class "NSURL"'s fileURLWithPath:"/Users/bill/Desktop/Test folder 2/DFile.txt"

-- Moves the "DFile.txt" from "Test folder" to "Test folder 2"
set WasSucessful to FileManager's moveItemAtURL:OldFileURL toURL:NewFileURL |error|:(missing value)
if (not WasSucessful) then
	display dialog "Could not move the file \"DFile.txt\"." buttons {"Cancel", "OK"} default button "OK"
	return false
end if
```

```
use framework "Foundation"

set FileManager to current application's NSFileManager's defaultManager --> <NSFileManager: 0x7f8581603da0>
set FileAttributes to FileManager's attributesOfItemAtPath:"/Users/bill/Desktop/Folder_7/4.pages" |error|:(missing value)

(*
	FileAttributes -->
	(NSDictionary) {
		NSFileOwnerAccountID:502,
		NSFileHFSTypeCode:1413830740,
		NSFileSystemFileNumber:628297,
		NSFileExtensionHidden:YES,
		NSFileSystemNumber:16777224,
		NSFileSize:0,
		NSFileGroupOwnerAccountID:80,
		NSFileOwnerAccountName:"bill",
		NSFileCreationDate:(NSDate) "2016-10-29 18:28:39 +0000",
		NSFilePosixPermissions:420,
		NSFileHFSCreatorCode:1061109567,
		NSFileType:"NSFileTypeRegular",
		NSFileGroupOwnerAccountName:"admin",
		NSFileReferenceCount:1,
		NSFileModificationDate:(NSDate) "2016-10-29 18:28:39 +0000"
	}
*)
```

```
use framework "Foundation"

set FileManager to current application's NSFileManager's defaultManager --> <NSFileManager: 0x7f8581603da0>
set FileAttributes to FileManager's attributesOfItemAtPath:"/Users/bill/Desktop/Folder_7/4.pages" |error|:(missing value)

set FolderList to FileManager's contentsOfDirectoryAtPath:"/Users/bill/Desktop/Folder_7" |error|:(missing value)

(*
	FolderList -->
	(NSArray) {
		".DS_Store",
		"2.webloc",
		"3.webloc",
		"4.pages",
		"4.webloc",
		"5.webloc",
		"TheItem2.webloc"
	}
*)
```

```
use framework "Foundation"
use scripting additions

set FileManager to current application's NSFileManager's defaultManager --> <NSFileManager: 0x7f8581603da0>
set FileExists to FileManager's fileExistsAtPath:"/Users/bill/Desktop/Folder_34/1.webloc"
if (FileExists) then
	display dialog "The file exists." buttons {"OK"} default button "OK"
else
	display dialog "The file could not found." buttons {"OK"} default button "OK"
end if
```

Framework: **Foundation** Class: **NSFileManager** class item: **isWritableFileAtPath:**

```
use framework "Foundation"
use scripting additions

set FileManager to current application's NSFileManager's defaultManager --> <NSFileManager: 0x7f8581603da0>
set CanWriteTo to FileManager's isWritableFileAtPath:"/Users/bill/Desktop/Folder_7/4.pages"
if (CanWriteTo) then
	display dialog "The file can be written to." buttons {"OK"} default button "OK"
else
	display dialog "The file can not be written to." buttons {"OK"} default button "OK"
	return false
end if

-- Note: This can easily be tested by running the script on a file, then locking the file in Finder and running the script again.
```

```
use framework "Foundation"

-- Files "4.pages" and "4 copy.pages" are Apple Pages document containg the single word "Test"
-- while "4 changed.pages" is an Apple Pages document contains the single word "Testing"
set FileManager to current application's NSFileManager's defaultManager --> <NSFileManager: 0x7f8581603da0>
set FilesEqual to FileManager's contentsEqualAtPath:"/Users/bill/Desktop/Folder_7/4.pages" andPath:"/Users/bill/Desktop/Fol
copy.pages"
FilesEqual --> true

set FilesEqual to FileManager's contentsEqualAtPath:"/Users/bill/Desktop/Folder_7/4.pages" andPath:"/Users/bill/Desktop/Fol
changed.pages"
FilesEqual --> false
```

```
use framework "Foundation"
use scripting additions

tell application "Finder"
	set theSelection to selection as alias list
end tell

repeat with aFile in theSelection
	set theURL to (current application's |NSURL|'s fileURLWithPath:(POSIX path of aFile))
	(theURL's setResourceValue:(current date) forKey:(current application's NSURLContentModificationDateKey) |error|:(mi
end repeat

-- NSURLCreationDateKey is a NSURLResourceKey constant.
-- NSURLContentModificationDateKey is a NSURLResourceKey constant.
```

```
use framework "Foundation"

-- This copies the file in the same folder but with a new name
set OldPath to "/Users/bill/Desktop/Test folder/Test file.scpt"
set NewPath to "/Users/bill/Desktop/Test folder/Test file 2.scpt"
set FileManager to current application's NSFileManager's defaultManager --> <NSFileManager: 0x7f8581603da0>
set WasSucessful to FileManager's copyItemAtPath:OldPath toPath:NewPath |error|:(missing value)

-- This copies the file to a new folder with the same name as the original file
set OldPath to "/Users/bill/Desktop/Test folder/Test file.scpt"
set NewPath to "/Users/bill/Desktop/Test folder 2/Test file.scpt"
set FileManager to current application's NSFileManager's defaultManager --> <NSFileManager: 0x7f8581603da0>
set WasSucessful to FileManager's copyItemAtPath:OldPath toPath:NewPath |error|:(missing value)

-- This copies the file to a new folder with a different name then the original file
set OldPath to "/Users/bill/Desktop/Test folder/Test file.scpt"
set NewPath to "/Users/bill/Desktop/Test folder 2/New Test file.scpt"
set FileManager to current application's NSFileManager's defaultManager --> <NSFileManager: 0x7f8581603da0>
set WasSucessful to FileManager's copyItemAtPath:OldPath toPath:NewPath |error|:(missing value)
```

```
use framework "Foundation"
use scripting additions

set FileManager to current application's NSFileManager's defaultManager --> <NSFileManager: 0x7f8581603da0>

-- This creates a file preloaded with the text specified in the contents parameter.  To create an empty file set |contents| to (mi
set FileCreated to FileManager's createFileAtPath:"/Users/bill/Desktop/Folder_34/DFile.txt" |contents|:("Some sample text" as
attributes:(missing value)
if (not FileCreated) then
	display dialog "The file could not be created." buttons {"OK"} default button "OK"
	return false
end if

-- This reads the data from the newly created file.
-- The results from contentsAtPath do not return the data stored in the file.  It returns the information in the file as encoded N
set TheDataBuffer to FileManager's contentsAtPath:"/Users/bill/Desktop/Folder_34/DFile.txt"
TheDataBuffer --> (NSData) <536f6d65 2073616d 706c6520 74657874>

-- initWithData decodes the NSData returned by contentsAtPath, and returns the text stored in the disk file.
set TextRead to current application's NSString's alloc()'s initWithData:(TheDataBuffer) encoding:(current application's NSUTF
--> (NSString) "Some sample text"
TextRead --> (NSString) "Some sample text"
```

```
use framework "Foundation"
use scripting additions

set FileManager to current application's NSFileManager's defaultManager --> <NSFileManager: 0x7f8581603da0>

-- This creates a file preloaded with the text specified in the contents parameter.  To create an empty file set |contents| to (mi
set FileCreated to FileManager's createFileAtPath:"/Users/bill/Desktop/Folder_34/DFile.txt" |contents|:("Some sample text" as
attributes:(missing value)
if (not FileCreated) then
	display dialog "The file could not be created." buttons {"OK"} default button "OK"
	return false
end if

-- This reads the data from the newly created file.
-- The results from contentsAtPath do not return the data stored in the file.  It returns the information in the file as encoded N
set TheDataBuffer to FileManager's contentsAtPath:"/Users/bill/Desktop/Folder_34/DFile.txt"
TheDataBuffer --> (NSData) <536f6d65 2073616d 706c6520 74657874>

-- initWithData decodes the NSData returned by contentsAtPath, and returns the text stored in the disk file.
set TextRead to current application's NSString's alloc()'s initWithData:(TheDataBuffer) encoding:(current application's NSUTF8
--> (NSString) "Some sample text"
TextRead --> (NSString) "Some sample text"
```

```
use framework "Foundation"

set FileManager to current application's NSFileManager's defaultManager --> <NSFileManager: 0x7f8581603da0>

set WasSucessful to FileManager's removeItemAtPath:"/Users/bill/Desktop/Folder_7/2.applescript" |error|:(missing value)
```

```
use framework "Foundation"

-- Create the NSString that holds the path for where to write the file to disk
set LineOfText to current application's NSString's stringWithString:"This is a test of ASObj-C file reading and writing."
set FileName to current application's NSString's stringWithString:"TheFile"
set TheFolder to current application's NSHomeDirectory() -- Get the path to the folder
set ThePath to TheFolder's stringByAppendingPathComponent:FileName -- Get the path to the file

-------------------------------------------------------------------------------------------------------

-- Write the file to disk
-- If Sucessful = true the write did not get any errors
set Sucessful to LineOfText's writeToFile:ThePath atomically:no encoding:(current application's NSUTF8StringEncoding) |error
value)

-------------------------------------------------------------------------------------------------------

-- Reads the string back from the disk
set TextFromDisk to current application's NSString's stringWithContentsOfFile:ThePath encoding:(current application's NSUTF8
|error|:(missing value)

-------------------------------------------------------------------------------------------------------
```

```
use framework "Foundation"

- The NSString to write to the disk
set LineOfText to current application's NSString's stringWithString:"This is a test of ASObj-C file reading and writing."
set FileName to current application's NSString's stringWithString:"TheFile"
set TheFolder to current application's NSHomeDirectory() -- Get the path to the folder
set ThePath to TheFolder's stringByAppendingPathComponent:FileName -- Get the path to the file

--------------------------------------------------------------------------------------------------------

-- If Sucessful = true the write did not get any errors
set Sucessful to LineOfText's writeToFile:ThePath atomically:no encoding:(current application's NSUTF8StringEncoding) |error
value)

--------------------------------------------------------------------------------------------------------

-- Reads the string back from the disk
set TextFromDisk to current application's NSString's stringWithContentsOfFile:ThePath encoding:(current application's NSUTF8
|error|:(missing value)


-- The file will be written to user's home directory
```

```
use framework "Foundation"

set TheHandle to current application's NSFileHandle's fileHandleForUpdatingAtPath:"/Users/bill/Desktop/Folder_34/TestFile.txt"
<NSConcreteFileHandle: 0x7fc0986a3b20>

TheHandle's closeFile() --> Nothing is returned.  NSFileHandle files that are opened should be closed when no longer needed.

-- The given path does not exist on the disk.
-- This returns missing value when the path does not exists
set TheHandle to current application's NSFileHandle's fileHandleForUpdatingAtPath:"/Users/bill/Desktop/Folder_??/TestFile.txt"
value

TheHandle = missing value --> true
```

```
use framework "Foundation"

-- The file at "/Users/bill/Desktop/Folder_34/TestFile.txt" initially contains the text "12345678901234567890"

set TheFolder to (current application's NSHomeDirectory() as string) & "/Desktop/Folder_34/" --> "/Users/bill/Desktop/Folder

set FilePath to TheFolder & "TestFile.txt" --> "/Users/bill/Desktop/Folder_34/TestFile.txt"

set TheHandle to current application's NSFileHandle's fileHandleForUpdatingAtPath:"/Users/bill/Desktop/Folder_34/TestFile.txt
<NSConcreteFileHandle: 0x7fcb6c6cedf0>

set TextFromDisk to current application's NSString's stringWithContentsOfFile:FilePath encoding:(my NSUTF8StringEncoding)
value)
TextFromDisk --> (NSString) "12345678901234567890" -- The text read has the same 20 characters as the file

TheHandle's truncateFileAtOffset:15 --> Nothing returned.  Keeps the first 15 characters and delete the rest.
set TextFromDisk to current application's NSString's stringWithContentsOfFile:FilePath encoding:(my NSUTF8StringEncoding)
value)
TextFromDisk --> (NSString) "123456789012345"

TheHandle's truncateFileAtOffset:0 --> Nothing returned.  Setting offset to zero deletes a data from the file
set TextFromDisk to current application's NSString's stringWithContentsOfFile:FilePath encoding:(my NSUTF8StringEncoding)
value)
TextFromDisk --> (NSString) ""

TheHandle's closeFile() --> Nothing returned
```

```
use framework "Foundation"

-- The file at "/Users/bill/Desktop/Folder_34/TestFile.txt" initially contains the text "ABCDEFGHIJKLMNOPQRSTUVWXYZ"

set FileManager to current application's NSFileManager's defaultManager --> <NSFileManager: 0x7f8581603da0>

-- Open "TestFile.txt" with NSFileManager
set TheHandle to current application's NSFileHandle's fileHandleForUpdatingAtPath:"/Users/bill/Desktop/Folder_34/TestFile.txt
<NSConcreteFileHandle: 0x7fc0986a3b20>
if (TheHandle = missing value) then
	display dialog "Could not get a handle for \"TestFile.txt.\"" buttons {"Cancel", "OK"} default button "OK"
	return false
end if

-- Get the current data pointer for "TestFile.txt"
TheHandle's offsetInFile --> (NSNumber) 0

set NSDataRead1 to TheHandle's readDataOfLength:5 --> (NSData) <41424344 45>
-- readDataOfLength returns its results in NSData format.  So the result needs to be decoded to get the text read from the disk
set TextRead1 to current application's NSString's alloc()'s initWithData:(NSDataRead1) encoding:(current application's
NSUTF8StringEncoding) --> (NSString) "ABCDE"

-- Move the data pointer to an offset of 10
-- This means move to the 11th position because the first position in the file is position zero.
-- The 11th character in file "TestFile.txt" is the letter "K"
TheHandle's seekToFileOffset:10 --> Nothing returned

set NSDataRead2 to TheHandle's readDataOfLength:5 --> (NSData) <4b4c4d4e 4f>
-- readDataOfLength returns its results in NSData format.  So the result needs to be decoded to get the text read from the disk
set TextRead2 to current application's NSString's alloc()'s initWithData:(NSDataRead2) encoding:(current application's
NSUTF8StringEncoding) --> (NSString) "KLMNO"

-- Get the current data pointer for "TestFile.txt"  The pointer is at 15 because it moved to position 10 & then read another 5 ch
TheHandle's offsetInFile --> (NSNumber) 15

TheHandle's closeFile() --> Nothing returned
```

```
use framework "Foundation"

-- The file at "/Users/bill/Desktop/Folder_34/TestFile.txt" initially contains the text "12345678901234567890"

set TheHandle to current application's NSFileHandle's fileHandleForUpdatingAtPath:"/Users/bill/Desktop/Folder_34/TestFile.txt
<NSConcreteFileHandle: 0x7fc0986a3b20>
if (TheHandle = missing value) then
	display dialog "Could not get a handle for \"TestFile.txt.\"" buttons {"Cancel", "OK"} default button "OK"
	return false
end if

set NSDataRead1 to TheHandle's readDataOfLength:5 --> (NSData) <31323334 35>
set TextRead1 to current application's NSString's alloc()'s initWithData:(NSDataRead1) encoding:(current application's
NSUTF8StringEncoding) -->(NSString) "12345"

TheHandle's seekToFileOffset:5 -- Nothing returned.  This goes to the 6th position because the first position is called number z
set NSDataRead2 to TheHandle's readDataOfLength:5 --> (NSData) <36373839 30>
set TextRead2 to current application's NSString's alloc()'s initWithData:(NSDataRead2) encoding:(current application's
NSUTF8StringEncoding) -->(NSString) "67890"

TheHandle's closeFile() --> Nothing returned
```

```
use framework "Foundation"

set FilePath to current application's NSString's stringWithString:"/Users/bill/Desktop/Folder_34/TestFile.txt"

-- Establishes a connection between NSFileHandle and a file so NSFileHandle can read from the file
set TheHandle to current application's NSFileHandle's fileHandleForReadingAtPath:FilePath --> <NSConcreteFileHandle: 0x7fcb

-- This reads the text in the file and returns NSData that represents the text read
set NSDataRead to TheHandle's readDataToEndOfFile

-- This converts the NSData to text
set TextRead to current application's NSString's alloc()'s initWithData:NSDataRead encoding:(current application's NSUTF8Stri

-- This statement breaks the connection between NSFileHandle and the file, therefore NSFileHandle can no longer perform acti
file
TheHandle's closeFile() --> Nothing returned

TextRead --> (NSString) "12345678901234567890"
```

Framework: **Foundation** Class: **NSString** class item: **dataUsingEncoding:**

```
use framework "Foundation"

-- Take a string, encode it into NSData format and convert it back to a string
set TheNSString to current application's NSString's stringWithString:"abcde" --> (NSString) "abcde"
set NSDataObject to TheNSString's dataUsingEncoding:(my NSUTF8StringEncoding) --> (NSData) <61626364 65>
set DecodedValue to current application's NSString's alloc()'s initWithData:(NSDataObject) encoding:(current application's NSUTF8StringEncoding) --> (NSString) "abcde"
```

```
use framework "Foundation"
use scripting additions



set TheNSString to current application's NSString's stringWithString:"abcde" --> (NSString) "abcde"

set NSDataObject to TheNSString's dataUsingEncoding:(my NSUTF8StringEncoding) --> (NSData) <61626364 65>

-- Initially DFile.txt is an empty text file
set TheFolder to (current application's NSHomeDirectory() as string) & "/Desktop/Folder_34/" --> (AppleScript string)
"/Users/bill/Desktop/Folder_34/"

set FilePath to TheFolder & "DFile.txt" --> (AppleScript string) "/Users/bill/Desktop/Folder_34/DFile.txt"

-- Create the NSFileHandle
set TheHandle to current application's NSFileHandle's fileHandleForWritingAtPath:FilePath --> <NSConcreteFileHandle: 0x7fcb6

if (TheHandle = missing value) then
	display dialog "The create handle failed." buttons {"OK"} default button "OK"
	return false
end if

TheHandle's writeData:NSDataObject --> Nothing returned
TheHandle's closeFile() --> Nothing returned

-- Read the data just written to "DFile.txt" to show it has now changed
set TextFromDisk to current application's NSString's stringWithContentsOfFile:FilePath encoding:(current application's NSUTF8S
|error|:(missing value)
TextFromDisk --> (NSString) "abcde"
```

```
use framework "Foundation"

-- The NSString to write to the disk
set LineOfText to current application's NSString's stringWithString:"This is a test of ASObj-C file reading and writing."
set FileName to current application's NSString's stringWithString:"TheFile"
set TheFolder to current application's NSHomeDirectory() -- Get the path to the user's home directory
set ThePath to TheFolder's stringByAppendingPathComponent:FileName -- Get the path to the file

--------------------------------------------------------------------------------------------

-- This will write "This is a test of ASObj-C file reading and writing." to a file named "TheFile" in the user's home folder
-- LineOfText is an NSString and writeToFile is a method of NSString
-- If Sucessful = true the write did not get any errors
set Sucessful to LineOfText's writeToFile:ThePath atomically:no encoding:(current application's NSUTF8StringEncoding) |error
value)

--------------------------------------------------------------------------------------------

-- Reads the string back from the disk
set TextFromDisk to current application's NSString's stringWithContentsOfFile:ThePath encoding:(current application's NSUTF
|error|:(missing value)
--------------------------------------------------------------------------------------------
```

```
use framework "Foundation"

set AppleScriptString to "Hello world"
set CocoaString to current application's NSString's stringWithString:AppleScriptString
class of AppleScriptString --> text
class of CocoaString --> __NSCFString
CocoaString's className() --> __NSCFString
----------------------------------------------------------------------------------------------------

set TheValues to current application's NSArray's arrayWithArray:{"v1", "v2", "v3", "v4"} --> (NSArray) {"v1","v2","3","v4"}
set TheKeys to current application's NSArray's arrayWithArray:{"k1", "k2", "k3", "k4"} --> (NSArray) {"k1","k2","k3","k4"}
set TheDictionary2 to current application's NSDictionary's alloc()'s initWithObjects:TheValues forKeys:TheKeys
TheDictionary2's className() --> (NSString) "__NSDictionaryI"
(TheDictionary2's className()) as string --> "__NSDictionaryI"
class of TheDictionary2 --> (Class) __NSDictionaryI

----------------------------------------------------------------------------------------------------

set TheNSString to (current application's NSString's stringWithString:"12345678")
TheNSString's |description| --> (NSString) "12345678"
TheNSString's className() --> (NSString) "__NSCFString"
----------------------------------------------------------------------------------------------------
```

```
use framework "Foundation"

set AppleScriptString to "Hello world"
set CocoaString to current application's NSString's stringWithString:AppleScriptString

-- 2 lines following indicates CocoaString is not an NSString, but is one of NSStrings subclasses
CocoaString's isMemberOfClass:(current application's NSString) --> false
CocoaString's isKindOfClass:(current application's NSString) --> true
-------------------------------------------------------------------------------------------------
```

```
use framework "Foundation"

set Test to current application's NSString's stringWithString:"test"
Test's isEqual:(Test) --> true

set Test2 to Test
Test's isEqual:(Test2) --> true

set Test3 to current application's NSString's stringWithString:"test"
Test's isEqual:(Test3) --> true
Test3's isEqual:(Test) --> true
(current application's NSString's stringWithString:"test")'s isEqual:(Test) --> true
(current application's NSString's stringWithString:"Test")'s isEqual:(Test) --> false

set ClassOf to class of Test
ClassOf's isEqual:(class of Test) --> true

----------------------------------------------------------------------------------------------------

set TheLetter to (current application's NSString's stringWithString:"T")
(TheLetter's isEqual:"T") --> true
(TheLetter's isEqual:"F") --> false
----------------------------------------------------------------------------------------------------
```

```
use framework "Foundation"

set AppleScriptString to "Hello world"
set CocoaString to current application's NSString's stringWithString:AppleScriptString
-- 2 lines following indicates CocoaString is not an NSString, but is one of NSStrings subclasses

CocoaString's isMemberOfClass:(current application's NSString) --> false
CocoaString's isKindOfClass:(current application's NSString) --> true

-- The code line following indicates CocoaString is an __NSCFString, so don't need to check for subclasses
CocoaString's isMemberOfClass:(current application's __NSCFString) --> true
------------------------------------------------------------------------------------------
```

```
use framework "Foundation"

-- Eventhough the sample acesses files in the user folder this sample will work eventhough it uses full paths
-- This returns the home directory for the user currently using the Mac
-- For a user named Bill this returns:
current application's NSHomeDirectory() --> (NSString) "/Users/bill"
------------------------------------------------------------------------------------------


-- Creates a folder named "Folder_34" on the user's desktop (user name Bill)
set TheFolder to ((current application's NSHomeDirectory()) as text) & "/Desktop/Folder_34/" --> "/Users/bill/Desktop/Folder
------------------------------------------------------------------------------------------


-- Write the NSString to the disk
set LineOfText to current application's NSString's stringWithString:"This is a test of ASObj-C file reading and writing."
set FileName to current application's NSString's stringWithString:"TheFile"
set TheFolder to current application's NSHomeDirectory() -- Get the path to the folder
set ThePath to TheFolder's stringByAppendingPathComponent:FileName -- Get the path to the file

-- If Sucessful = true the write did not get any errors
set Sucessful to LineOfText's writeToFile:ThePath atomically:no encoding:(current application's NSUTF8StringEncoding) |error
value)
------------------------------------------------------------------------------------------
```

```
use framework "Foundation"

set TheStr to current application's NSString's stringWithString:"{10, 50}"
current application's NSRangeFromString(TheStr) --> {location:10, |length|:50}

set TheStr to current application's NSString's stringWithString:"{10 50}"
current application's NSRangeFromString(TheStr) --> {location:10, |length|:50}

set TheStr to current application's NSString's stringWithString:"10-50"
current application's NSRangeFromString(TheStr) --> {location:10, |length|:50}

set TheStr to current application's NSString's stringWithString:"10"
current application's NSRangeFromString(TheStr) --> {location:10, |length|:0}

set TheStr to current application's NSString's stringWithString:"G"
current application's NSRangeFromString(TheStr) --> {location:0, |length|:0}

set TheStr to current application's NSString's stringWithString:""
current application's NSRangeFromString(TheStr) --> {location:0, |length|:0}
```

```
use framework "Foundation"

set TheList to {{|name|:"One", |color|:"Red"}, {|name|:"Two", |color|:"Blue"}}
set TheArray to current application's NSArray's arrayWithObject:TheList
TheArray --> (NSArray) {{{name:"One", color:"Red"}, {name:"Two", color:"Blue"}}}
```

Framework: **Foundation** Class: **NSArray** class item: **arrayWithArray:**

```
use framework "Foundation"

current application's NSArray's arrayWithArray:{} --> (NSArray) {}

current application's NSArray's arrayWithArray:{1, 2} --> (NSArray) {1, 2}

current application's NSMutableArray's arrayWithArray:{"Dog", "Cat", "Bird"} --> (NSArray) {"Dog", "Cat", "Bird"}

set A1 to current application's NSMutableArray's arrayWithArray:{"The Object"} --> (NSArray) {"The Object"}
set A2 to current application's NSMutableArray's arrayWithArray:{"First", A1, "Second"} --> (NSArray) {"First", {"The Object'
```

```
use framework "Foundation"

set TestArray3 to current application's NSMutableArray's arrayWithArray:{"key", "Shoe", "Door knocker", "Cup", "key", "Shoe"
	"key", "Cup", "Hose"}
TestArray3's replaceObjectAtIndex:2 withObject:"Doorbell" --> The method didn't return anything
TestArray3 --> (NSArray) {"key","Shoe","Doorbell","Cup","key","Shoe","Napkin","key","Cup","Hose"}
```

```
use framework "Foundation"

set TestArray to current application's NSArray's arrayWithArray:{"Box", "Circle", "Rectangle"}
TestArray --> (NSArray) {"Box", "Circle", "Rectangle"}
-- Arrays start at index zero
TestArray's indexOfObjectIdenticalTo:("Circle") --> 1

-- Searching for an object that doesn't exists returns NSNotFound
TestArray's indexOfObjectIdenticalTo:("Square") --> 9.22337203685478E+18
(TestArray's indexOfObjectIdenticalTo:("Square")) = current application's NSNotFound --> true
```

```
use framework "Foundation"

-- Here is the simplest case of firstObject()
set TheArray to current application's NSArray's arrayWithArray:{"One", "Two", "Three"}
TheArray's firstObject() --> (NSString) "One"
TheArray's lastObject() --> (NSString) "Three"

set TheArray to current application's NSArray's arrayWithArray:{{"A", "B", "C"}, "Middle", {1, 2, 3}}
TheArray's firstObject() --> (NSArray) {"A", "B", "C"}
TheArray's firstObject()'s firstObject() --> (NSString) "A"
TheArray's firstObject()'s lastObject() --> (NSString) "C"
------------------------------------
TheArray's lastObject() --> (NSArray) {1, 2, 3}
TheArray's lastObject()'s firstObject() --> (NSNumber) 1
TheArray's lastObject()'s lastObject() --> (NSNumber) 3

-- Create an empty Array
set AnotherArray to current application's NSArray's arrayWithArray:{}
AnotherArray's firstObject() --> missing value
AnotherArray's lastObject() --> missing value

set AppleScriptRecord1 to {|name|:"One", |color|:"Red"}
set AppleScriptRecord2 to {|name|:"Two", |color|:"Blue"}
set AppleScriptRecords3 to {{|name|:"One", |color|:"Red"}, {|name|:"Two", |color|:"Blue"}}

AppleScriptRecord1 --> (AppleScript record) {|name|:"One", |color|:"Red"}
AppleScriptRecord2 --> (AppleScript record) {|name|:"Two", |color|:"Blue"}
AppleScriptRecords3 --> (AppleScript record) {{|name|:"One", |color|:"Red"}, {|name|:"Two", |color|:"Blue"}}
```

```
set TheArray1 to current application's NSArray's arrayWithObject:{AppleScriptRecord1, AppleScriptRecord2}
TheArray1 --> (NSArray) {{{name:"One", color:"Red"}, {name:"Two", color:"Blue"}}}
class of TheArray1 --> (Class) __NSArrayI

set TheArray2 to current application's NSArray's arrayWithObject:AppleScriptRecords3
TheArray2 --> (NSArray){{{name:"One", color:"Red"}, {name:"Two", color:"Blue"}}}
class of TheArray2 --> (Class) __NSArrayI

-- Eventhough Array1 and Arrary2 were put together differently they still create the same array
TheArray1's isEqualToArray:TheArray2 --> true


TheArray1 --> (NSArray) {{{name:"One", color:"Red"}, {name:"Two", color:"Blue"}}}
class of TheArray1 --> (Class) __NSArrayI, this is an immutable NSArray
-------------------------------------
TheArray1's firstObject() --> (NSArray) {{name:"One",color:"Red"},{name:"Two",color:"Blue"}}
class of TheArray1's firstObject() --> (Class) __NSArrayM, this is a mutable NSArray
-------------------------------------
TheArray1's firstObject()'s firstObject() --> (NSDictionary) {name:"One", color:"Red"}
class of TheArray1's firstObject()'s firstObject() --> (Class) __NSDictionaryM, this is being interpreted as a NSDictionary
-------------------------------------
TheArray1's firstObject()'s lastObject() --> (NSDictionary) {name:"Two", color:"Blue"}
class of TheArray1's firstObject()'s lastObject() --> (Class) __NSDictionaryM, this is being interpreted as a NSDictionary
```

```
use framework "Foundation"

-- Here is the simplest case of firstObject()
set TheArray to current application's NSArray's arrayWithArray:{"One", "Two", "Three"}
TheArray's firstObject() --> (NSString) "One"
TheArray's lastObject() --> (NSString) "Three"

set TheArray to current application's NSArray's arrayWithArray:{{"A", "B", "C"}, "Middle", {1, 2, 3}}
TheArray's firstObject() --> (NSArray) {"A", "B", "C"}
TheArray's firstObject()'s firstObject() --> (NSString) "A"
TheArray's firstObject()'s lastObject() --> (NSString) "C"
------------------------------------
TheArray's lastObject() --> (NSArray) {1, 2, 3}
TheArray's lastObject()'s firstObject() --> (NSNumber) 1
TheArray's lastObject()'s lastObject() --> (NSNumber) 3

-- Create an empty Array
set AnotherArray to current application's NSArray's arrayWithArray:{}
AnotherArray's firstObject() --> missing value
AnotherArray's lastObject() --> missing value

set AppleScriptRecord1 to {|name|:"One", |color|:"Red"}
set AppleScriptRecord2 to {|name|:"Two", |color|:"Blue"}
set AppleScriptRecords3 to {{|name|:"One", |color|:"Red"}, {|name|:"Two", |color|:"Blue"}}

AppleScriptRecord1 --> (AppleScript record) {|name|:"One", |color|:"Red"}
AppleScriptRecord2 --> (AppleScript record) {|name|:"Two", |color|:"Blue"}
AppleScriptRecords3 --> (AppleScript record) {{|name|:"One", |color|:"Red"}, {|name|:"Two", |color|:"Blue"}}
```

```
set TheArray1 to current application's NSArray's arrayWithObject:{AppleScriptRecord1, AppleScriptRecord2}
TheArray1 --> (NSArray) {{{name:"One", color:"Red"}, {name:"Two", color:"Blue"}}}
class of TheArray1 --> (Class) __NSArrayI

set TheArray2 to current application's NSArray's arrayWithObject:AppleScriptRecords3
TheArray2 --> (NSArray){{{name:"One", color:"Red"}, {name:"Two", color:"Blue"}}}
class of TheArray2 --> (Class) __NSArrayI

-- Eventhough Array1 and Arrary2 were put together differently they still create the same array
TheArray1's isEqualToArray:TheArray2 --> true


TheArray1 --> (NSArray) {{{name:"One", color:"Red"}, {name:"Two", color:"Blue"}}}
class of TheArray1 --> (Class) __NSArrayI, this is an immutable NSArray
------------------------------------
TheArray1's firstObject() --> (NSArray) {{name:"One",color:"Red"},{name:"Two",color:"Blue"}}
class of TheArray1's firstObject() --> (Class) __NSArrayM, this is a mutable NSArray
------------------------------------
TheArray1's firstObject()'s firstObject() --> (NSDictionary) {name:"One", color:"Red"}
class of TheArray1's firstObject()'s firstObject() --> (Class) __NSDictionaryM, this is being interpreted as a NSDictionary
------------------------------------
TheArray1's firstObject()'s lastObject() --> (NSDictionary) {name:"Two", color:"Blue"}
class of TheArray1's firstObject()'s lastObject() --> (Class) __NSDictionaryM, this is being interpreted as a NSDictionary
```

Framework: **Foundation** Class: **NSMutableArray** class item: **insertObject:atIndex:**

```
use framework "Foundation"

set TestArray to current application's NSMutableArray's arrayWithArray:{"Dog", "Cat", "Bird"} --> (NSArray) {"Dog", "Cat", "
TestArray's insertObject:"Mouse" atIndex:3 --> The insertObject method does not return a result
TestArray --> (NSArray) {"Dog", "Cat", "Bird", "Mouse"}
```

```
use framework "Foundation"

set TestArray to current application's NSMutableArray's arrayWithArray:{"Box", "Circle", "Rectangle"} --> (NSArray)
{"Box","Circle","Rectangle"}

-- Array indexes start at zero
TestArray's indexOfObject:"Circle" --> 1


-- Call indexOfObject with an object that doesn't exists
TestArray's indexOfObject:"Square" --> 9.22337203685478E+18 which is the value of NSNotFound

(TestArray's indexOfObject:"Square") = current application's NSNotFound --> true

-- Call indexOfObject on an empty array
set TestArray2 to current application's NSMutableArray's arrayWithArray:{} --> (NSArray) {}
TestArray2's indexOfObject:"Circle" --> 9.22337203685478E+18 which is the value of NSNotFound

(TestArray2's indexOfObject:"Circle") = current application's NSNotFound --> true
```

```
use framework "Foundation"

set TestArray to current application's NSMutableArray's arrayWithArray:{"Box", "Circle", "Rectangle"}
TestArray --> (NSArray) {"Box", "Circle", "Rectangle"}
TestArray's exchangeObjectAtIndex:0 withObjectAtIndex:2 --> The method didn't return anything
TestArray --> (NSArray) {"Rectangle", "Circle", "Box"}
```

```
use framework "Foundation"

set TestArray1 to current application's NSMutableArray's arrayWithArray:{}
set TestArray2 to current application's NSMutableArray's arrayWithArray:{"Dog", "Cat", "Bird", "Mouse"} --> (NSArray) {"Dog"
set TestArray3 to current application's NSMutableArray's arrayWithArray:{"Horse", "Bird", "Snake"} --> (NSArray)
{"Box","Circle","Rectangle"}
set TestArray4 to current application's NSMutableArray's arrayWithArray:{"Box", "Circle", "Rectangle"}

-- If either of the arrays is an "empty array" than firstObjectCommonWithArray returns missing value
TestArray1's firstObjectCommonWithArray:TestArray2 --> missing value
TestArray2's firstObjectCommonWithArray:TestArray1 --> missing value

TestArray3's firstObjectCommonWithArray:TestArray2 --> Bird

TestArray4's firstObjectCommonWithArray:TestArray2 --> missing value
(TestArray4's firstObjectCommonWithArray:TestArray2) = missing value --> true
```

```
use framework "Foundation"

set TestArray1 to current application's NSArray's arrayWithArray:{1, 2, 3}
set TestArray2 to current application's NSArray's arrayWithArray:{"1", "2", "3"}
set TestArray3 to current application's NSArray's arrayWithArray:{1, 2, 3}

TestArray1's isEqualToArray:TestArray2 --> false
TestArray1's isEqualToArray:TestArray3 --> true
```

Framework: **Foundation** Class: **NSArray** class item: **objectAtIndex:**

```
use framework "Foundation"

set TestArray2 to current application's NSMutableArray's arrayWithArray:{"Box", "Circle", "Rectangle"}
TestArray2's objectAtIndex:2 --> (NSString) "Rectangle"
```

Framework: **Foundation** Class: **NSMutableArray** class item: **removeObject:**

```
use framework "Foundation"

set TestArray to current application's NSMutableArray's arrayWithArray:{"Dog", "Cat", "Bird", "Mouse"}
TestArray's removeObject:"Bird" --> The method doesn't return anything
TestArray --> (NSArray) {"Dog", "Cat", "Mouse"}
```

Framework: **Foundation** Class: **NSMutableArray** class item: **removeObjectAtIndex:**

```
use framework "Foundation"

set TestArray to current application's NSMutableArray's arrayWithArray:{"Dog", "Cat", "Bird", "Mouse"}
TestArray --> (NSArray) {"Dog", "Cat", "Bird", "Mouse"}
TestArray's removeObjectAtIndex:1 --> The method doesn't return anything
TestArray --> (NSArray) {"Dog","Bird","Mouse"}
```

```
use framework "Foundation"

set OriginalArray to current application's NSMutableArray's arrayWithArray:{0, 1, 2, 3, 4, 5, 6, 7}
OriginalArray --> (NSArray) {1,2,3,4,5,6,7}
OriginalArray's removeObjectIdenticalTo:3 --> The method doesn't return anything
OriginalArray --> (NSArray) (NSArray) {0, 1, 2, 4, 5, 6, 7} (since the only object equal to 3 is the 4th object, the 4th object is
```

Framework: **Foundation** Class: **NSMutableArray** class item: **removeLastObject()**

```
use framework "Foundation"

set TestArray to current application's NSMutableArray's arrayWithArray:{"Dog", "Cat", "Bird", "Mouse"}
TestArray --> (NSArray) {"Dog", "Cat", "Bird", "Mouse"}
TestArray's removeLastObject() --> The method didn't return anything
TestArray --> (NSArray) {"Dog", "Cat", "Bird"}
```

Framework: **Foundation** Class: **NSMutableArray** class item: **removeObjectsInArray:**

```
use framework "Foundation"

set AnArray to current application's NSMutableArray's arrayWithArray:{3, 6, 1, 1, 7, 6}

-- removeObjectsInArray will remove: two number ones, 2 number sixs, & 1 number 7
AnArray's removeObjectsInArray:{1, 6, 7} --> The method doesn't return anything

AnArray --> (NSArray) {3}
```

Framework: **Foundation** Class: **NSMutableArray** class item: **removeAllObjects()**

```
use framework "Foundation"

set AnArray to current application's NSMutableArray's arrayWithArray:{"Red", "Green", "Blue", "Black"}
AnArray --> (NSArray) {"Red", "Green", "Blue", "Black"}
AnArray's removeAllObjects() --> This method doesn't return anything
AnArray --> (NSArray) {}
```

Framework: **Foundation** Class: **NSMutableArray** class item: **removeObjectsInRange:**

```
use framework "Foundation"

set AnArray to current application's NSMutableArray's arrayWithArray:{0, 1, 2, 3, 4, 5, 6, 7, 8, 9}
AnArray --> (NSArray) {0, 1, 2, 3, 4, 5, 6, 7, 8, 9}
-- This will remove 3 characters starting at index 4
AnArray's removeObjectsInRange:{location:4, |length|:3}
AnArray --> (NSArray) {0, 1, 2, 3, 7, 8, 9}
```

```
use framework "Foundation"

set TheArray to current application's NSMutableArray's arrayWithArray:{1, 2, 3, 4, 5}

-- Create empty IndexSet to hold the indexes to delete
set DeleteIndexSet to current application's NSMutableIndexSet's indexSet() --> <NSMutableIndexSet: 0x......>(no indexes)

-- Specify the index of items to delete
set TheASList to {0, 1, 3} -- This is an AppleScript list

-- Add each index to delete from TheArray to DeleteIndexSet
repeat with TheIndex in TheASList
	(DeleteIndexSet's addIndex:TheIndex) --> The method doesn't return anything
end repeat

-- DeleteIndexSet specifies to delete items with indexes 0, 1, 3
TheArray's removeObjectsAtIndexes:DeleteIndexSet --> The method didn't return anything

TheArray --> (NSArray) {3, 5}
TheArray as list --> {3, 5}  (Type is Applescript list)
```

```
use framework "Foundation"

set TheArray to current application's NSMutableArray's arrayWithArray:{1, 2, 3, 4, 5}

-- Create empty IndexSet to hold the indexes to delete
set DeleteIndexSet to current application's NSMutableIndexSet's indexSet() --> <NSMutableIndexSet: 0x......>(no indexes)

-- Specify the index of items to delete
set TheASList to {0, 1, 3} -- This is an AppleScript list

-- Add each index to delete from TheArray to DeleteIndexSet
repeat with TheIndex in TheASList
	(DeleteIndexSet's addIndex:TheIndex) --> The method doesn't return anything
end repeat

DeleteIndexSet --> <NSMutableIndexSet: 0x7fec15b97700>[number of indexes: 3 (in 2 ranges), indexes: (0-1 3)]

-- DeleteIndexSet specifies to delete items with indexes 0, 1, 3 which equal the items 1, 2, 4
TheArray's removeObjectsAtIndexes:DeleteIndexSet --> The method didn't return anything

TheArray --> (NSArray) {3, 5}
TheArray as list --> (Applescript list) {3, 5}
```

```
use framework "Foundation"

set TheArray to current application's NSMutableArray's arrayWithArray:{1, 2, 3, 4, 5}

-- Create empty IndexSet to hold the indexes to delete
set DeleteIndexSet to current application's NSMutableIndexSet's indexSet() --> <NSMutableIndexSet: 0x......>(no indexes)

-- Specify the index of items to delete
set TheASList to {0, 1, 3} -- This is an AppleScript list

-- Add each index to delete from TheArray to DeleteIndexSet
repeat with TheIndex in TheASList
	(DeleteIndexSet's addIndex:TheIndex) --> The method doesn't return anything
end repeat

-- DeleteIndexSet specifies to delete items with indexes 0, 1, 3
TheArray's removeObjectsAtIndexes:DeleteIndexSet --> The method didn't return anything

TheArray --> (NSArray) {3, 5}
TheArray as list --> {3, 5}  (Type is Applescript list)
```

```
use framework "Foundation"

set RangeStr to current application's NSStringFromRange(current application's NSMakeRange(10, 50))
RangeStr --> (NSString) "{10, 50}"
```

```
use framework "Foundation"

-- The NSString to write to the disk
set LineOfText to current application's NSString's stringWithString:"This is a test of ASObj-C file reading and writing."
set FileName to current application's NSString's stringWithString:"TheFile"
set TheFolder to current application's NSHomeDirectory() -- Get the path to the folder
set ThePath to TheFolder's stringByAppendingPathComponent:FileName -- Get the path to the file

----------------------------------------------------------------------------------------------------

-- If Sucessful = true the write did not get any errors
set Sucessful to LineOfText's writeToFile:ThePath atomically:no encoding:(current application's NSUTF8StringEncoding) |error
value)

----------------------------------------------------------------------------------------------------

-- Reads the string back from the disk
set TextFromDisk to current application's NSString's stringWithContentsOfFile:ThePath encoding:(current application's NSUTF8
|error|:(missing value)
```

```
use framework "AppKit"

-- With a single Script Debugger running
set TheApp1 to current application's NSRunningApplication's runningApplicationsWithBundleIdentifier:"com.latenightsw.ScriptD
TheApp1 --> <NSRunningApplication: 0x7f8694ebb320 (com.latenightsw.ScriptDebugger6 - 4261)>

-- Changing the bundle identifier so no app will matches it causes no application to be found
set TheApp2 to current application's NSRunningApplication's runningApplicationsWithBundleIdentifier:"com.latenightsw.ScriptD
TheApp2 --> (NSArray) {}

-- A more straight forward way is to count the result
count of (current application's NSRunningApplication's runningApplicationsWithBundleIdentifier:"com.latenightsw.ScriptDebug

-- Without Script Debugger running
set SDCount to count of (current application's NSRunningApplication's
runningApplicationsWithBundleIdentifier:"com.latenightsw.ScriptDebugger6")
SDCount --> 0   (This process runs much fater then "system events")

-- With Script Debugger running
set SDCount to count of (current application's NSRunningApplication's
runningApplicationsWithBundleIdentifier:"com.latenightsw.ScriptDebugger6")
SDCount --> 1   (This process runs much fater then "system events")

-- With Script Debugger running and launching a copy of script bebugger
set SDCount to count of (current application's NSRunningApplication's
runningApplicationsWithBundleIdentifier:"com.latenightsw.ScriptDebugger6")
SDCount --> 2   (This process runs much fater then "system events")
```

Framework: **Foundation** Class: **NSString** class item: **lastPathComponent()**

```
use framework "Foundation"

-- There is a real file at the following path: "/Users/bill/Desktop/Test folder/Test file.scpt"

set FilePathStr to current application's NSString's stringWithString:"/Users/bill/Desktop/Test folder/Test file.scpt"
set FullFileName to FilePathStr's lastPathComponent() --> (NSString) "Test file.scpt"

-- If I change change the path to a file that does not exist lastPathComponent still succeds.  There is no relation to the file
-- path in FilePathStr and a real path on the disk.  It just returns returns information that would be true if the the file path did
set FilePathStr to current application's NSString's stringWithString:"/Users/bill/Desktop/Test folder/Test file 2.scpt"
set FullFileName to FilePathStr's lastPathComponent() --> (NSString) "Test file 2.scpt"

set FilePathStr to current application's NSString's stringWithString:"/Users/bill/Desktop/Test folder/"
set FullFileName to FilePathStr's lastPathComponent() --> (NSString) "Test folder"
```

Framework: **Foundation** Class: **NSString** class item: **pathExtension()**

```
use framework "Foundation"

-- There is a real file at the following path: "/Users/bill/Desktop/Test folder/Test file.scpt"

set FilePathStr to current application's NSString's stringWithString:"/Users/bill/Desktop/Test folder/Test file.scpt"
FilePathStr's pathExtension() --> (NSString) "scpt"

-- If I change change the path to a file that does not exist lastPathComponent still succeds.  There is no relation to the file
-- path in FilePathStr and a real path on the disk.  It just returns returns information that would be true if the the file path did
set FilePathStr to current application's NSString's stringWithString:"/Users/bill/Desktop/Test folder/Test file 2.scpt"
FilePathStr's pathExtension() --> (NSString) "Foundation examples.scpt" --> (NSString) "scpt"

-- If there is no file at the end of the path (i.e. path ends with folder) the method returns nothing
-- NSString knows the path ends in a folder because the last character in path is a forward slash (/)
set FilePathStr to current application's NSString's stringWithString:"/Users/bill/Desktop/Test folder/"
FilePathStr's pathExtension() --> returns nothing

-- If the extension is removed from the file name the method returns a path ending with a folder returns nothing
set FilePathStr to current application's NSString's stringWithString:"/Users/bill/Desktop/Test folder/Test file"
FilePathStr's pathExtension() --> returns nothing
```

```
use framework "Foundation"

-- There is a real file at the following path: "/Users/bill/Desktop/Test folder/Test file.scpt"

-- It returns the path with the extension removed from the file at the end of the path
set FilePathStr to current application's NSString's stringWithString:"/Users/bill/Desktop/Test folder/Test file.scpt"
FilePathStr's stringByDeletingPathExtension() --> (NSString) "/Users/bill/Desktop/Test folder/Test file"

-- If I change change the path to a file that does not exist stringByDeletingPathExtension() still succeds.  There is no relation t
-- path in FilePathStr and a real path on the disk.  It just returns returns information that would be true if the the file path did
set FilePathStr to current application's NSString's stringWithString:"/Users/bill/Desktop/Test folder/Test file 2.scpt"
FilePathStr's stringByDeletingPathExtension() --> (NSString) "/Users/bill/Desktop/Test folder/Test file 2"

-- If there is no file at the end of the path the method still works
set FilePathStr to current application's NSString's stringWithString:"/Users/bill/Desktop/Test folder/"
FilePathStr's stringByDeletingPathExtension() --> (NSString) "/Users/bill/Desktop/Test folder"

-- If the extension is moved from the file name the method still works
set FilePathStr to current application's NSString's stringWithString:"/Users/bill/Desktop/Test folder/Test file"
FilePathStr's stringByDeletingPathExtension() --> (NSString) "/Users/bill/Desktop/Test folder/Test file"
```

```
use framework "Foundation"

-- There is a real file at the following path: "/Users/bill/Desktop/Test folder/Test file.scpt"

set FilePathStr to current application's NSString's stringWithString:"/Users/bill/Desktop/Test folder/Test file.scpt"
FilePathStr's stringByDeletingLastPathComponent() --> (NSString) "/Users/bill/Desktop/Test folder"

-- If I change change the path to a file that does not exist stringByDeletingLastPathComponent() still succeds.  There is no rela
-- path in FilePathStr and a real path on the disk.  It just returns returns information that would be true if the the file path did
set FilePathStr to current application's NSString's stringWithString:"/Users/bill/Desktop/Test folder/Test file 2.scpt"
FilePathStr's stringByDeletingLastPathComponent() --> (NSString) "/Users/bill/Desktop/Test folder"

set FilePathStr to current application's NSString's stringWithString:"/Users/bill/Desktop/Test folder/"
FilePathStr's stringByDeletingLastPathComponent() --> (NSString) "/Users/bill/Desktop"
```

```
use framework "Foundation"

-- This is the most common use for valueForKey with arrays.  For more advanced operations see "valueForKeyPath" method
set MyDatabase to current application's NSArray's arrayWithArray:{{|name|:"Spot", hair:"Black", Height:{|feet|:1, |inches|:2,
totalInches:14}, weight:43}, {|name|:"Scooter", hair:"Black", Height:{|feet|:4, |inches|:7, totalInches:85}, weight:61}, {|na
hair:"Gray", Height:{|feet|:3, |inches|:2, totalInches:26}, weight:80}}

MyDatabase's valueForKey:"name" --> (NSArray) {"Spot" ,"Scooter", "FurBall"}
MyDatabase's valueForKey:"Height" --> (NSArray) {{inches:2, feet:1, totalInches:14}, {inches:7, feet:4, totalInches:55}, {in
totalInches:26}}
MyDatabase's valueForKey:"hair" --> (NSArray) {"Black", "Black", "Gray"}

----------------------------------------------------------------------------------------------------

--  Here is another way to use valueForKey
set LowerCaseLetterArray to current application's NSArray's arrayWithArray:{"a", "b", "c"}
set FloatingPointNumberArray to current application's NSArray's arrayWithArray:{1.0, 2.0, 4.0}
set NumberStringsArray to current application's NSArray's arrayWithArray:{"1.0", "2.0", "4.0"}
set StringsArray to current application's NSArray's arrayWithArray:{"1", "12", "123", "A"}

set MixedArray to current application's NSArray's arrayWithArray:{{{"1", "2"}, {"A"}, {}}, {"a"}, {"dog", "cat"}}
set EmptyArray to current application's NSArray's arrayWithArray:{}

------------------------------------------------------------------------------------------

LowerCaseLetterArray's valueForKey:"uppercaseString" --> (NSArray) {"A", "B", "C"}

FloatingPointNumberArray's valueForKey:"integerValue" --> (NSArray) {1, 2, 4}
NumberStringsArray's valueForKey:"integerValue" --> (NSArray) {1, 2, 4}
```

```
StringsArray's valueForKey:"length" --> (NSArray) {1, 2, 3, 1}
-- The integerValue property for NSString is zero if the string doesn’t start with a valid representation of a number
StringsArray's valueForKey:"integerValue" --> (NSArray) {1, 12, 123, 0}

-- It returned a {} in the 1 spot because the corresponding spot in the MixedArray didn't have an element to get the length of
MixedArray's valueForKey:"length" --> (NSArray) {{{1, 1}, {1}, {}}, {1}, {3, 3}}

EmptyArray's valueForKey:"integerValue" --> (NSArray) {}
EmptyArray's valueForKey:"length" --> (NSArray) {}

--------------------------------------------------------------------------------------------------------

-- see "valueForKeyPath" method for more advanced operations
```

```
use framework "Foundation"

set MyDatabase to current application's NSArray's arrayWithArray:{{|name|:"Spot", hair:"Black", Height:{|feet|:1, |inches|:2
totalInches:14}, weight:43}, {|name|:"Scooter", hair:"Black", Height:{|feet|:4, |inches|:7, totalInches:85}, weight:61}, {|na
hair:"Gray", Height:{|feet|:3, |inches|:2, totalInches:26}, weight:80}}

MyDatabase's valueForKeyPath:"Height.feet" --> (NSArray) {1, 4, 2}

-- How many dogs there are in the database
MyDatabase's valueForKeyPath:"@count" --> (NSNumber) 3

-- The tallest height of all the dogs
MyDatabase's valueForKeyPath:"@max.Height.totalInches" --> (NSNumber) 55

-- The smallest height of all the dogs
MyDatabase's valueForKeyPath:"@min.Height.totalInches" --> (NSNumber) 14

-- The average height of all the dogs
MyDatabase's valueForKeyPath:"@avg.Height.totalInches" --> (NSNumber) 31.666666666667

-- All together the dogs weigh 184
MyDatabase's valueForKeyPath:"@sum.weight" --> (NSNumber) 184.0

-- There were 2 dogs with black hair but black only showed up once in the array
MyDatabase's valueForKeyPath:"@distinctUnionOfObjects.hair" --> (NSArray) {"Black", "Gray"}

------------------------------------------------------------------------------------------------------------------------

-- see "valueForKey" method for a more less advanced operations
```

```
use framework "Foundation"

set theArray to current application's NSArray's arrayWithArray:{8, 2, 7, 3, 9, 1, 6, 4}
set thePred to current application's NSPredicate's predicateWithFormat:"self > 4" --> (NSComparisonPredicate) SELF > 4
(theArray's filteredArrayUsingPredicate:thePred) as list --> {8, 7, 9, 6}
--------------------------------------------------------------------------------------------


set thePred to current application's NSPredicate's predicateWithFormat:"(self != 2) AND (self != 4)" --> (NSCompoundPredica
AND SELF != 4
(theArray's filteredArrayUsingPredicate:thePred) as list --> {8, 7, 3, 9, 1, 6}
--------------------------------------------------------------------------------------------


set stringArray to current application's NSArray's arrayWithArray:{"adobe", "Apple", "microsoft", "google"}
set thePred to current application's NSPredicate's predicateWithFormat:"self BEGINSWITH 'a'" --> (NSComparisonPredicate) S
BEGINSWITH "a"
(stringArray's filteredArrayUsingPredicate:thePred) as list --> {"adobe"}
--------------------------------------------------------------------------------------------


set thePred to current application's NSPredicate's predicateWithFormat:"self BEGINSWITH 'A'" --> (NSComparisonPredicate) S
BEGINSWITH "A"
(stringArray's filteredArrayUsingPredicate:thePred) as list --> {"Apple"}
--------------------------------------------------------------------------------------------


-- [cd] means ignore case and diacriticals.
set thePred to current application's NSPredicate's predicateWithFormat:"self BEGINSWITH[cd] 'A'" --> (NSComparisonPredicat
BEGINSWITH[cd] "A"
(stringArray's filteredArrayUsingPredicate:thePred) as list --> {"adobe", "Apple"}
--------------------------------------------------------------------------------------------


set thePred to current application's NSPredicate's predicateWithFormat:"self LIKE 'Ap*'" --> (NSComparisonPredicate) SELF LI
```

```
(stringArray's filteredArrayUsingPredicate:thePred) as list --> {"Apple"}
--------------------------------------------------------------------------------------------


set stringArray to current application's NSArray's arrayWithArray:{"adobe", "22", "microsoft", "99"}
-- "\\\\d\\\\d" is a string representation of the litteral 6 characters \\d\\d.
-- When the presicate is evaluated those 6 characters will evaluate to \d\d which matches 2 digits
set thePred to current application's NSPredicate's predicateWithFormat:"self MATCHES '\\\\d\\\\d'" --> (NSComparisonPredica
MATCHES "\\d\\d"
(stringArray's filteredArrayUsingPredicate:thePred) as list --> {"22", "99"}
--------------------------------------------------------------------------------------------


set TheRecords to current application's NSArray's arrayWithArray:{{age:39, price:65}, {age:43, price:60}, {age:45, price:65

set ThePredicate to current application's NSPredicate's predicateWithFormat:"age > 38 AND price > 60"
(TheRecords's filteredArrayUsingPredicate:ThePredicate) as list --> {{age:39, price:65}, {age:45, price:65}}
--------------------------------------------------------------------------------------------


set TheRecords to current application's NSArray's arrayWithArray:{{age:39, price:{dollars:30, euros:70}}, {age:43, price:{d
euros:70}}, {age:45, price:{dollars:70, euros:70}}}

set ThePredicate to current application's NSPredicate's predicateWithFormat:"age > 38 AND price.dollars > 60"
set result to (TheRecords's filteredArrayUsingPredicate:ThePredicate) as list --> {{age:45, price:{dollars:70, euros:70}}}
--------------------------------------------------------------------------------------------


-- set args to NSArray's arrayWithArray
set TheRecords to current application's NSArray's arrayWithArray:{{age:39, price:65}, {age:43, price:60}, {age:45, price:65

set ThePredicate to current application's NSPredicate's predicateWithFormat_("%K > 38 AND %K > 60", "age", "price")
(TheRecords's filteredArrayUsingPredicate:ThePredicate) as list --> {{age:39, price:65}, {age:45, price:65}}
```

```
use framework "Foundation"

set theArray to current application's NSArray's arrayWithArray:{8, 2, 7, 3, 9, 1, 6, 4}
set thePred to current application's NSPredicate's predicateWithFormat:"self > 4" --> (NSComparisonPredicate) SELF > 4
(theArray's filteredArrayUsingPredicate:thePred) as list --> {8, 7, 9, 6}
--------------------------------------------------------------------------------


set thePred to current application's NSPredicate's predicateWithFormat:"(self != 2) AND (self != 4)" --> (NSCompoundPredica
AND SELF != 4
(theArray's filteredArrayUsingPredicate:thePred) as list --> {8, 7, 3, 9, 1, 6}
--------------------------------------------------------------------------------


set stringArray to current application's NSArray's arrayWithArray:{"adobe", "Apple", "microsoft", "google"}
set thePred to current application's NSPredicate's predicateWithFormat:"self BEGINSWITH 'a'" --> (NSComparisonPredicate) S
BEGINSWITH "a"
(stringArray's filteredArrayUsingPredicate:thePred) as list --> {"adobe"}
--------------------------------------------------------------------------------


set thePred to current application's NSPredicate's predicateWithFormat:"self BEGINSWITH 'A'" --> (NSComparisonPredicate) S
BEGINSWITH "A"
(stringArray's filteredArrayUsingPredicate:thePred) as list --> {"Apple"}
--------------------------------------------------------------------------------


-- [cd] means ignore case and diacriticals.
set thePred to current application's NSPredicate's predicateWithFormat:"self BEGINSWITH[cd] 'A'" --> (NSComparisonPredicat
BEGINSWITH[cd] "A"
(stringArray's filteredArrayUsingPredicate:thePred) as list --> {"adobe", "Apple"}
--------------------------------------------------------------------------------


set thePred to current application's NSPredicate's predicateWithFormat:"self LIKE 'Ap*'" --> (NSComparisonPredicate) SELF LI
```

```
(stringArray's filteredArrayUsingPredicate:thePred) as list --> {"Apple"}
--------------------------------------------------------------------------------


set stringArray to current application's NSArray's arrayWithArray:{"adobe", "22", "microsoft", "99"}
-- "\\\\d\\\\d" is a string representation of the litteral 6 characters \\d\\d.
-- When the presicate is evaluated those 6 characters will evaluate to \d\d which matches 2 digits
set thePred to current application's NSPredicate's predicateWithFormat:"self MATCHES '\\\\d\\\\d'" --> (NSComparisonPredica
MATCHES "\\d\\d"
(stringArray's filteredArrayUsingPredicate:thePred) as list --> {"22", "99"}
--------------------------------------------------------------------------------


set TheRecords to current application's NSArray's arrayWithArray:{{age:39, price:65}, {age:43, price:60}, {age:45, price:65

set ThePredicate to current application's NSPredicate's predicateWithFormat:"age > 38 AND price > 60"
(TheRecords's filteredArrayUsingPredicate:ThePredicate) as list --> {{age:39, price:65}, {age:45, price:65}}
--------------------------------------------------------------------------------


set TheRecords to current application's NSArray's arrayWithArray:{{age:39, price:{dollars:30, euros:70}}, {age:43, price:{d
euros:70}}, {age:45, price:{dollars:70, euros:70}}}

set ThePredicate to current application's NSPredicate's predicateWithFormat:"age > 38 AND price.dollars > 60"
set result to (TheRecords's filteredArrayUsingPredicate:ThePredicate) as list --> {{age:45, price:{dollars:70, euros:70}}}
--------------------------------------------------------------------------------


-- set args to NSArray's arrayWithArray
set TheRecords to current application's NSArray's arrayWithArray:{{age:39, price:65}, {age:43, price:60}, {age:45, price:65

set ThePredicate to current application's NSPredicate's predicateWithFormat_("%K > 38 AND %K > 60", "age", "price")
(TheRecords's filteredArrayUsingPredicate:ThePredicate) as list --> {{age:39, price:65}, {age:45, price:65}}
```

```
use framework "Foundation"

-- If you only want to call fileExistsAtPath once you can do it this way
set {ObjectExists1, IsADirectory1} to current application's NSFileManager's defaultManager()'s
fileExistsAtPath:"/Users/bill/Desktop/Folder_34/" isDirectory:(reference)

-- If you want to call NSFileManager many times then set a variable to hold the value returned by NSFileManager's defaultMana
the variable repeatedly
set TheDefaultManager to current application's NSFileManager's defaultManager() -- defaultManager() returns the FileManger's

-- Folder "Folder_34" does exist
set PathToDirectory to "/Users/bill/Desktop/Folder_34/"
set {ObjectExists1, IsADirectory1} to TheDefaultManager's fileExistsAtPath:PathToDirectory isDirectory:(reference)
ObjectExists1 --> true
IsADirectory1 --> 1
IsADirectory1 as boolean -->

-- File "Test.scptd" does exist as does folder "Folder_34"
-- Finder does consider "Test.scptd" a file but NSFileManager considers "Test.scptd" a directory because "Test.scptd" is a bundl
type of directory
set PathToFile to "/Users/bill/Desktop/Folder_34/Test.scptd"
set {ObjectExists2, IsADirectory2} to TheDefaultManager's fileExistsAtPath:PathToFile isDirectory:(reference)
ObjectExists2 --> true
IsADirectory2 --> 1
IsADirectory2 as boolean --> 1

-- "Test.scpt" does exist as does folder "Folder_34"
set PathToFile to "/Users/bill/Desktop/Folder_34/Test.scpt"
set {ObjectExists3, IsADirectory3} to TheDefaultManager's fileExistsAtPath:PathToFile isDirectory:(reference)
ObjectExists3 --> true
```

```
IsADirectory3 --> 0
IsADirectory3 as boolean --> 0

-- Folder "Folder_32" does not exist
set PathToDirectory to "/Users/bill/Desktop/Folder_32/"
set {ObjectExists1, IsADirectory1} to TheDefaultManager's fileExistsAtPath:PathToDirectory isDirectory:(reference)
ObjectExists1 --> false
IsADirectory1 --> 0
IsADirectory1 as boolean --> 0

-- Folder "Folder_34" does exist, but there is no "Test 2.scptd" in folder "Folder_34"
set PathToFile to "/Users/bill/Desktop/Folder_34/Test 2.scptd"
set {ObjectExists1, IsADirectory1} to TheDefaultManager's fileExistsAtPath:PathToDirectory isDirectory:(reference)
ObjectExists1 --> false
IsADirectory1 --> 0
IsADirectory1 as boolean --> 0

-- Note: The call to fileExistsAtPath returns 2 items because the "isDirectory" parameter is equal to "reference."
-- See Shane Stanley's "Everyday ASObjC 3.4" page 53 for a full discussion about how "reference" works.
https://www.macosxautomation.com/applescript/apps/
```

Framework: **Foundation** Class: **NSTimeZone** class item: **timeZoneWithName:**

```
use framework "Foundation"

(current application's NSTimeZone's timeZoneWithName:"America/Los_Angeles") --> (__NSTimeZone) America/Los_Angeles (
-28800

current application's NSTimeZone's defaultTimeZone() --> (__NSTimeZone) America/Los_Angeles (PST) offset -28800
```

```
use framework "Foundation"

current application's NSTimeZone's systemTimeZone() --> (__NSTimeZone) America/Los_Angeles (PST) offset -28800
```

Framework: **Foundation** Class: **NSTimeZone** class item: **defaultTimeZone()**

```
use framework "Foundation"

current application's NSTimeZone's defaultTimeZone() --> (__NSTimeZone) America/Los_Angeles (PST) offset -28800
```

Framework: **Foundation (Root)** Class: **NSObject** class item: **description()**

```
use framework "Foundation"

(current application's NSMutableString's stringWithString:"Hi")'s |description| --> (NSString) "Hi"

set NSNumber5 to (current application's NSNumber's numberWithInt:5)
NSNumber5's |description| --> (NSString) "5

set MyTimeZone to current application's NSTimeZone's timeZoneWithAbbreviation:"PST"
MyTimeZone's |description|() --> (NSString) "America/Los_Angeles (PST) offset -28800"

(current application's NSNumber's numberWithInt:(5 + 2))'s |description| --> (NSString) "7"
```

```
use framework "Foundation"

set MyTimeZone to current application's NSTimeZone's timeZoneWithAbbreviation:"PST"
MyTimeZone --> (__NSTimeZone) America/Los_Angeles (PST) offset -28800


MyTimeZone's secondsFromGMT --> (NSNumber) -28800
```

Framework: **Foundation** Class: **NSNumber** class item: **numberWithInteger:**

```
use framework "Foundation"

set ANumber to 100.5

set IntNumber to current application's NSNumber's numberWithInteger:ANumber --> (NSNumber) 100
class of IntNumber --> (Objective-C) __NSCFNumber
-- floatValue turns it to an Objective-c floating point and the scripting bridge turns it into a AppleScript floating point
set IntegerTofloat to IntNumber's floatValue() -->  (AppleScript integer) 100.0

set FloatNumber to current application's NSNumber's numberWithFloat:ANumber --> (NSNumber) 100.5
class of FloatNumber --> (Objective-C) __NSCFNumber
set FloatAsInteger to FloatNumber's intValue() --> (AppleScript integer) 100

-- NSNumber Boolean variables can only be set to 1 or zero
set BooleanValue0 to current application's NSNumber's numberWithBool:0 --> (NSNumber) NO
class of BooleanValue0 --> (Objective-C) __NSCFBoolean
set BooleanAsInteger0 to BooleanValue0's intValue() --> (AppleScript integer) 1
set BooleanAsFloat0 to BooleanValue0's floatValue() --> (AppleScript real) 0.0

-- NSNumber Boolean variables can only be set to 1 or zero.  Any other number returns an error.
set BooleanValue1 to current application's NSNumber's numberWithBool:1 --> (NSNumber) YES
class of BooleanValue1 --> (Objective-C) __NSCFBoolean
set BooleanAsInteger1 to BooleanValue1's intValue() --> (AppleScript integer) 1
set BooleanAsFloat1 to BooleanValue1's floatValue() --> (AppleScript real) 1.0

BooleanValue0's stringValue() --> (NSString) "0"
BooleanValue1's stringValue() --> (NSString) "1"
-- Since BooleanAsInteger and BooleanAsFloat are AppleScript types stringValue() can not be used with them
-- Instead they need to type cast in AppleScript to make them strings
```

```
BooleanAsInteger0 as text --> (AppleScript string) "0"
BooleanAsFloat0 as text --> (AppleScript string) "0.0"

-- NSNumber can be compared
set IntNumber1 to current application's NSNumber's numberWithInteger:3
set IntNumber2 to current application's NSNumber's numberWithInteger:3
set IntNumber3 to current application's NSNumber's numberWithInteger:6

IntNumber1's isEqualToNumber:IntNumber2 --> true
IntNumber1's isEqualToNumber:IntNumber3 --> false

-- The NSNumber compare method returns 3 possible values when comparing 2 NSNumbers
-- For purposes of clarity given FirstNSNumber's compare:SecondNSNumber
-- FirstNSNumber will be called the "receiver" and SecondNSNumber will be called the "other number"
-- if compare returns -1 then the receiver is less than "other number"
-- if compare returns 1 then the receiver is greater than "other number"
-- if compare returns zero then the receiver is equal to "other number"
IntNumber1's compare:IntNumber3 --> (AppleScript integer) -1
IntNumber3's compare:IntNumber1 --> (AppleScript integer) 1
IntNumber1's compare:IntNumber2 --> (AppleScript integer) 0

-- Another way to compare NSNumbers is to convert them to regular numbers then compare
IntNumber1's intValue() = IntNumber2's intValue() --> true
IntNumber1's intValue() = IntNumber3's intValue() --> false
IntNumber1's floatValue() = IntNumber3's floatValue() --> false
IntNumber1's intValue() < IntNumber2's intValue() --> false
IntNumber1's intValue() < IntNumber3's intValue() --> true

-- Arithmatic can be preformed using NSNumbers
(IntNumber1's intValue()) + (IntNumber2's intValue()) --> (AppleScript integer) 6
(IntNumber3's intValue()) / (IntNumber1's intValue()) --> (AppleScript real) 2.0
```

Framework: **Foundation** Class: **NSTimeZone** class item: **DaylightSavingTime()**

```
use framework "Foundation"

-- localTimeZone will return the time zone you're currently in
set LocalZone to current application's NSTimeZone's localTimeZone()
LocalZone --> (__NSLocalTimeZone) Local Time Zone (America/Los_Angeles (PST) offset -28800)
LocalZone's isDaylightSavingTime() --> false
```

```
use framework "Foundation"

set MyTimeZone to current application's NSTimeZone's timeZoneWithAbbreviation:"PST"
MyTimeZone --> (__NSTimeZone) America/Los_Angeles (PST) offset -28800

MyTimeZone's nextDaylightSavingTimeTransition() --> (NSDate) "2017-03-12 10:00:00 +0000"
```

Framework: **Foundation** Class: **NSTimeZone** class item: **isEqualToTimeZone:**

```
use framework "Foundation"

set MyTimeZone to current application's NSTimeZone's timeZoneWithAbbreviation:"PST" --> (__NSTimeZone) America/Los_A
offset -28800
MyTimeZone's isEqualToTimeZone:(current application's NSTimeZone's timeZoneWithName:"Asia/Tokyo") --> false
```

```
use framework "Foundation"

set TZAbbreviationDictionary to current application's NSTimeZone's abbreviationDictionary()
(* TZAbbreviationDictionary -->
(NSDictionary) {
	EDT:"America/New_York",
	GMT:"GMT",
	AST:"America/Halifax",
	IRST:"Asia/Tehran",
	ICT:"Asia/Bangkok",
	PET:"America/Lima",
	KST:"Asia/Seoul",
	PST:"America/Los_Angeles",
	CDT:"America/Chicago",
	EEST:"Europe/Istanbul",
	NZDT:"Pacific/Auckland",
	WEST:"Europe/Lisbon",
	EAT:"Africa/Addis_Ababa",
	HKT:"Asia/Hong_Kong",
	IST:"Asia/Calcutta",
	MDT:"America/Denver",
	NZST:"Pacific/Auckland",
	WIT:"Asia/Jakarta",
	ADT:"America/Halifax",
	BST:"Europe/London",
	ART:"America/Argentina/Buenos_Aires",
	CAT:"Africa/Harare",
	GST:"Asia/Dubai",
	PDT:"America/Los_Angeles",
    SGT:"Asia/Singapore",
    COT:"America/Bogota",
    PKT:"Asia/Karachi",
    EET:"Europe/Istanbul",
    UTC:"UTC",
    WAT:"Africa/Lagos",
    EST:"America/New_York",
    JST:"Asia/Tokyo",
    CLST:"America/Santiago",
    CET:"Europe/Paris",
    BDT:"Asia/Dhaka",
    MSK:"Europe/Moscow",
    AKDT:"America/Juneau",
    CLT:"America/Santiago",
    AKST:"America/Juneau",
    BRST:"America/Sao_Paulo",
    BRT:"America/Sao_Paulo",
    CEST:"Europe/Paris",
    CST:"America/Chicago",
    HST:"Pacific/Honolulu",
    MSD:"Europe/Moscow",
    MST:"America/Denver",
    PHT:"Asia/Manila",
    WET:"Europe/Lisbon"
}
*)
```

```
use framework "Foundation"
use scripting additions

set OldFileURL to current application's class "NSURL"'s fileURLWithPath:"/Users/bill/Desktop/Test folder/DFile.txt"
set NewFileURL to current application's class "NSURL"'s fileURLWithPath:"/Users/bill/Desktop/Test folder 2/DFile.txt"

set FileManager to current application's NSFileManager's defaultManager --> <NSFileManager: 0x7f8581603da0>

set WasSucessful to FileManager's moveItemAtURL:OldFileURL toURL:NewFileURL |error|:(missing value)
if (not WasSucessful) then
	display dialog "Could not move the item \"TheItem.webloc\"." buttons {"Cancel", "OK"} default button "OK"
	return false
end if
```

Framework: **Foundation** Class: **NSNumber** class item: **numberWithFloat:**

```
use framework "Foundation"

set ANumber to 100.5

set IntNumber to current application's NSNumber's numberWithInteger:ANumber --> (NSNumber) 100
class of IntNumber --> (Objective-C) __NSCFNumber
-- floatValue turns it to an Objective-c floating point and the scripting bridge turns it into a AppleScript floating point
set IntegerTofloat to IntNumber's floatValue() -->  (AppleScript integer) 100.0

set FloatNumber to current application's NSNumber's numberWithFloat:ANumber --> (NSNumber) 100.5
class of FloatNumber --> (Objective-C) __NSCFNumber
set FloatAsInteger to FloatNumber's intValue() --> (AppleScript integer) 100

-- NSNumber Boolean variables can only be set to 1 or zero
set BooleanValue0 to current application's NSNumber's numberWithBool:0 --> (NSNumber) NO
class of BooleanValue0 --> (Objective-C) __NSCFBoolean
set BooleanAsInteger0 to BooleanValue0's intValue() --> (AppleScript integer) 1
set BooleanAsFloat0 to BooleanValue0's floatValue() --> (AppleScript real) 0.0

-- NSNumber Boolean variables can only be set to 1 or zero.  Any other number returns an error.
set BooleanValue1 to current application's NSNumber's numberWithBool:1 --> (NSNumber) YES
class of BooleanValue1 --> (Objective-C) __NSCFBoolean
set BooleanAsInteger1 to BooleanValue1's intValue() --> (AppleScript integer) 1
set BooleanAsFloat1 to BooleanValue1's floatValue() --> (AppleScript real) 1.0

BooleanValue0's stringValue() --> (NSString) "0"
BooleanValue1's stringValue() --> (NSString) "1"
-- Since BooleanAsInteger and BooleanAsFloat are AppleScript types stringValue() can not be used with them
-- Instead they need to type cast in AppleScript to make them strings
```

```
BooleanAsInteger0 as text --> (AppleScript string) "0"
BooleanAsFloat0 as text --> (AppleScript string) "0.0"

-- NSNumber can be compared
set IntNumber1 to current application's NSNumber's numberWithInteger:3
set IntNumber2 to current application's NSNumber's numberWithInteger:3
set IntNumber3 to current application's NSNumber's numberWithInteger:6

IntNumber1's isEqualToNumber:IntNumber2 --> true
IntNumber1's isEqualToNumber:IntNumber3 --> false

-- The NSNumber compare method returns 3 possible values when comparing 2 NSNumbers
-- For purposes of clarity given FirstNSNumber's compare:SecondNSNumber
-- FirstNSNumber will be called the "receiver" and SecondNSNumber will be called the "other number"
-- if compare returns -1 then the receiver is less than "other number"
-- if compare returns 1 then the receiver is greater than "other number"
-- if compare returns zero then the receiver is equal to "other number"
IntNumber1's compare:IntNumber3 --> (AppleScript integer) -1
IntNumber3's compare:IntNumber1 --> (AppleScript integer) 1
IntNumber1's compare:IntNumber2 --> (AppleScript integer) 0

-- Another way to compare NSNumbers is to convert them to regular numbers then compare
IntNumber1's intValue() = IntNumber2's intValue() --> true
IntNumber1's intValue() = IntNumber3's intValue() --> false
IntNumber1's floatValue() = IntNumber3's floatValue() --> false
IntNumber1's intValue() < IntNumber2's intValue() --> false
IntNumber1's intValue() < IntNumber3's intValue() --> true

-- Arithmatic can be preformed using NSNumbers
(IntNumber1's intValue()) + (IntNumber2's intValue()) --> (AppleScript integer) 6
(IntNumber3's intValue()) / (IntNumber1's intValue()) --> (AppleScript real) 2.0
```

```
use framework "Foundation"

set ANumber to 100.5

set IntNumber to current application's NSNumber's numberWithInteger:ANumber --> (NSNumber) 100
class of IntNumber --> (Objective-C) __NSCFNumber
-- floatValue turns it to an Objective-c floating point and the scripting bridge turns it into a AppleScript floating point
set IntegerTofloat to IntNumber's floatValue() -->  (AppleScript integer) 100.0

set FloatNumber to current application's NSNumber's numberWithFloat:ANumber --> (NSNumber) 100.5
class of FloatNumber --> (Objective-C) __NSCFNumber
set FloatAsInteger to FloatNumber's intValue() --> (AppleScript integer) 100

-- NSNumber Boolean variables can only be set to 1 or zero
set BooleanValue0 to current application's NSNumber's numberWithBool:0 --> (NSNumber) NO
class of BooleanValue0 --> (Objective-C) __NSCFBoolean
set BooleanAsInteger0 to BooleanValue0's intValue() --> (AppleScript integer) 1
set BooleanAsFloat0 to BooleanValue0's floatValue() --> (AppleScript real) 0.0

-- NSNumber Boolean variables can only be set to 1 or zero.  Any other number returns an error.
set BooleanValue1 to current application's NSNumber's numberWithBool:1 --> (NSNumber) YES
class of BooleanValue1 --> (Objective-C) __NSCFBoolean
set BooleanAsInteger1 to BooleanValue1's intValue() --> (AppleScript integer) 1
set BooleanAsFloat1 to BooleanValue1's floatValue() --> (AppleScript real) 1.0

BooleanValue0's stringValue() --> (NSString) "0"
BooleanValue1's stringValue() --> (NSString) "1"
-- Since BooleanAsInteger and BooleanAsFloat are AppleScript types stringValue() can not be used with them
-- Instead they need to type cast in AppleScript to make them strings
```

```
BooleanAsInteger0 as text --> (AppleScript string) "0"
BooleanAsFloat0 as text --> (AppleScript string) "0.0"

-- NSNumber can be compared
set IntNumber1 to current application's NSNumber's numberWithInteger:3
set IntNumber2 to current application's NSNumber's numberWithInteger:3
set IntNumber3 to current application's NSNumber's numberWithInteger:6

IntNumber1's isEqualToNumber:IntNumber2 --> true
IntNumber1's isEqualToNumber:IntNumber3 --> false

-- The NSNumber compare method returns 3 possible values when comparing 2 NSNumbers
-- For purposes of clarity given FirstNSNumber's compare:SecondNSNumber
-- FirstNSNumber will be called the "receiver" and SecondNSNumber will be called the "other number"
-- if compare returns -1 then the receiver is less than "other number"
-- if compare returns 1 then the receiver is greater than "other number"
-- if compare returns zero then the receiver is equal to "other number"
IntNumber1's compare:IntNumber3 --> (AppleScript integer) -1
IntNumber3's compare:IntNumber1 --> (AppleScript integer) 1
IntNumber1's compare:IntNumber2 --> (AppleScript integer) 0

-- Another way to compare NSNumbers is to convert them to regular numbers then compare
IntNumber1's intValue() = IntNumber2's intValue() --> true
IntNumber1's intValue() = IntNumber3's intValue() --> false
IntNumber1's floatValue() = IntNumber3's floatValue() --> false
IntNumber1's intValue() < IntNumber2's intValue() --> false
IntNumber1's intValue() < IntNumber3's intValue() --> true

-- Arithmatic can be preformed using NSNumbers
(IntNumber1's intValue()) + (IntNumber2's intValue()) --> (AppleScript integer) 6
(IntNumber3's intValue()) / (IntNumber1's intValue()) --> (AppleScript real) 2.0
```

```
use framework "Foundation"

set ANumber to 100.5

set IntNumber to current application's NSNumber's numberWithInteger:ANumber --> (NSNumber) 100
class of IntNumber --> (Objective-C) __NSCFNumber
-- floatValue turns it to an Objective-c floating point and the scripting bridge turns it into a AppleScript floating point
set IntegerTofloat to IntNumber's floatValue() -->  (AppleScript integer) 100.0

set FloatNumber to current application's NSNumber's numberWithFloat:ANumber --> (NSNumber) 100.5
class of FloatNumber --> (Objective-C) __NSCFNumber
set FloatAsInteger to FloatNumber's intValue() --> (AppleScript integer) 100

-- NSNumber Boolean variables can only be set to 1 or zero
set BooleanValue0 to current application's NSNumber's numberWithBool:0 --> (NSNumber) NO
class of BooleanValue0 --> (Objective-C) __NSCFBoolean
set BooleanAsInteger0 to BooleanValue0's intValue() --> (AppleScript integer) 1
set BooleanAsFloat0 to BooleanValue0's floatValue() --> (AppleScript real) 0.0

-- NSNumber Boolean variables can only be set to 1 or zero.  Any other number returns an error.
set BooleanValue1 to current application's NSNumber's numberWithBool:1 --> (NSNumber) YES
class of BooleanValue1 --> (Objective-C) __NSCFBoolean
set BooleanAsInteger1 to BooleanValue1's intValue() --> (AppleScript integer) 1
set BooleanAsFloat1 to BooleanValue1's floatValue() --> (AppleScript real) 1.0

BooleanValue0's stringValue() --> (NSString) "0"
BooleanValue1's stringValue() --> (NSString) "1"
-- Since BooleanAsInteger and BooleanAsFloat are AppleScript types stringValue() can not be used with them
-- Instead they need to type cast in AppleScript to make them strings
```

```
BooleanAsInteger0 as text --> (AppleScript string) "0"
BooleanAsFloat0 as text --> (AppleScript string) "0.0"

-- NSNumber can be compared
set IntNumber1 to current application's NSNumber's numberWithInteger:3
set IntNumber2 to current application's NSNumber's numberWithInteger:3
set IntNumber3 to current application's NSNumber's numberWithInteger:6

IntNumber1's isEqualToNumber:IntNumber2 --> true
IntNumber1's isEqualToNumber:IntNumber3 --> false

-- The NSNumber compare method returns 3 possible values when comparing 2 NSNumbers
-- For purposes of clarity given FirstNSNumber's compare:SecondNSNumber
-- FirstNSNumber will be called the "receiver" and SecondNSNumber will be called the "other number"
-- if compare returns -1 then the receiver is less than "other number"
-- if compare returns 1 then the receiver is greater than "other number"
-- if compare returns zero then the receiver is equal to "other number"
IntNumber1's compare:IntNumber3 --> (AppleScript integer) -1
IntNumber3's compare:IntNumber1 --> (AppleScript integer) 1
IntNumber1's compare:IntNumber2 --> (AppleScript integer) 0

-- Another way to compare NSNumbers is to convert them to regular numbers then compare
IntNumber1's intValue() = IntNumber2's intValue() --> true
IntNumber1's intValue() = IntNumber3's intValue() --> false
IntNumber1's floatValue() = IntNumber3's floatValue() --> false
IntNumber1's intValue() < IntNumber2's intValue() --> false
IntNumber1's intValue() < IntNumber3's intValue() --> true

-- Arithmatic can be preformed using NSNumbers
(IntNumber1's intValue()) + (IntNumber2's intValue()) --> (AppleScript integer) 6
(IntNumber3's intValue()) / (IntNumber1's intValue()) --> (AppleScript real) 2.0
```

```
use framework "Foundation"

set ANumber to 100.5

set IntNumber to current application's NSNumber's numberWithInteger:ANumber --> (NSNumber) 100
class of IntNumber --> (Objective-C) __NSCFNumber
-- floatValue turns it to an Objective-c floating point and the scripting bridge turns it into a AppleScript floating point
set IntegerTofloat to IntNumber's floatValue() -->  (AppleScript integer) 100.0

set FloatNumber to current application's NSNumber's numberWithFloat:ANumber --> (NSNumber) 100.5
class of FloatNumber --> (Objective-C) __NSCFNumber
set FloatAsInteger to FloatNumber's intValue() --> (AppleScript integer) 100

-- NSNumber Boolean variables can only be set to 1 or zero
set BooleanValue0 to current application's NSNumber's numberWithBool:0 --> (NSNumber) NO
class of BooleanValue0 --> (Objective-C) __NSCFBoolean
set BooleanAsInteger0 to BooleanValue0's intValue() --> (AppleScript integer) 1
set BooleanAsFloat0 to BooleanValue0's floatValue() --> (AppleScript real) 0.0

-- NSNumber Boolean variables can only be set to 1 or zero.  Any other number returns an error.
set BooleanValue1 to current application's NSNumber's numberWithBool:1 --> (NSNumber) YES
class of BooleanValue1 --> (Objective-C) __NSCFBoolean
set BooleanAsInteger1 to BooleanValue1's intValue() --> (AppleScript integer) 1
set BooleanAsFloat1 to BooleanValue1's floatValue() --> (AppleScript real) 1.0

BooleanValue0's stringValue() --> (NSString) "0"
BooleanValue1's stringValue() --> (NSString) "1"
-- Since BooleanAsInteger and BooleanAsFloat are AppleScript types stringValue() can not be used with them
-- Instead they need to type cast in AppleScript to make them strings
```

```
BooleanAsInteger0 as text --> (AppleScript string) "0"
BooleanAsFloat0 as text --> (AppleScript string) "0.0"

-- NSNumber can be compared
set IntNumber1 to current application's NSNumber's numberWithInteger:3
set IntNumber2 to current application's NSNumber's numberWithInteger:3
set IntNumber3 to current application's NSNumber's numberWithInteger:6

IntNumber1's isEqualToNumber:IntNumber2 --> true
IntNumber1's isEqualToNumber:IntNumber3 --> false

-- The NSNumber compare method returns 3 possible values when comparing 2 NSNumbers
-- For purposes of clarity given FirstNSNumber's compare:SecondNSNumber
-- FirstNSNumber will be called the "receiver" and SecondNSNumber will be called the "other number"
-- if compare returns -1 then the receiver is less than "other number"
-- if compare returns 1 then the receiver is greater than "other number"
-- if compare returns zero then the receiver is equal to "other number"
IntNumber1's compare:IntNumber3 --> (AppleScript integer) -1
IntNumber3's compare:IntNumber1 --> (AppleScript integer) 1
IntNumber1's compare:IntNumber2 --> (AppleScript integer) 0

-- Another way to compare NSNumbers is to convert them to regular numbers then compare
IntNumber1's intValue() = IntNumber2's intValue() --> true
IntNumber1's intValue() = IntNumber3's intValue() --> false
IntNumber1's floatValue() = IntNumber3's floatValue() --> false
IntNumber1's intValue() < IntNumber2's intValue() --> false
IntNumber1's intValue() < IntNumber3's intValue() --> true

-- Arithmatic can be preformed using NSNumbers
(IntNumber1's intValue()) + (IntNumber2's intValue()) --> (AppleScript integer) 6
(IntNumber3's intValue()) / (IntNumber1's intValue()) --> (AppleScript real) 2.0
```

```
use framework "Foundation"

set ANumber to 100.5

set IntNumber to current application's NSNumber's numberWithInteger:ANumber --> (NSNumber) 100
class of IntNumber --> (Objective-C) __NSCFNumber
-- floatValue turns it to an Objective-c floating point and the scripting bridge turns it into a AppleScript floating point
set IntegerTofloat to IntNumber's floatValue() -->  (AppleScript integer) 100.0

set FloatNumber to current application's NSNumber's numberWithFloat:ANumber --> (NSNumber) 100.5
class of FloatNumber --> (Objective-C) __NSCFNumber
set FloatAsInteger to FloatNumber's intValue() --> (AppleScript integer) 100

-- NSNumber Boolean variables can only be set to 1 or zero
set BooleanValue0 to current application's NSNumber's numberWithBool:0 --> (NSNumber) NO
class of BooleanValue0 --> (Objective-C) __NSCFBoolean
set BooleanAsInteger0 to BooleanValue0's intValue() --> (AppleScript integer) 1
set BooleanAsFloat0 to BooleanValue0's floatValue() --> (AppleScript real) 0.0

-- NSNumber Boolean variables can only be set to 1 or zero.  Any other number returns an error.
set BooleanValue1 to current application's NSNumber's numberWithBool:1 --> (NSNumber) YES
class of BooleanValue1 --> (Objective-C) __NSCFBoolean
set BooleanAsInteger1 to BooleanValue1's intValue() --> (AppleScript integer) 1
set BooleanAsFloat1 to BooleanValue1's floatValue() --> (AppleScript real) 1.0

BooleanValue0's stringValue() --> (NSString) "0"
BooleanValue1's stringValue() --> (NSString) "1"
-- Since BooleanAsInteger and BooleanAsFloat are AppleScript types stringValue() can not be used with them
-- Instead they need to type cast in AppleScript to make them strings
```

```
BooleanAsInteger0 as text --> (AppleScript string) "0"
BooleanAsFloat0 as text --> (AppleScript string) "0.0"

-- NSNumber can be compared
set IntNumber1 to current application's NSNumber's numberWithInteger:3
set IntNumber2 to current application's NSNumber's numberWithInteger:3
set IntNumber3 to current application's NSNumber's numberWithInteger:6

IntNumber1's isEqualToNumber:IntNumber2 --> true
IntNumber1's isEqualToNumber:IntNumber3 --> false

-- The NSNumber compare method returns 3 possible values when comparing 2 NSNumbers
-- For purposes of clarity given FirstNSNumber's compare:SecondNSNumber
-- FirstNSNumber will be called the "receiver" and SecondNSNumber will be called the "other number"
-- if compare returns -1 then the receiver is less than "other number"
-- if compare returns 1 then the receiver is greater than "other number"
-- if compare returns zero then the receiver is equal to "other number"
IntNumber1's compare:IntNumber3 --> (AppleScript integer) -1
IntNumber3's compare:IntNumber1 --> (AppleScript integer) 1
IntNumber1's compare:IntNumber2 --> (AppleScript integer) 0

-- Another way to compare NSNumbers is to convert them to regular numbers then compare
IntNumber1's intValue() = IntNumber2's intValue() --> true
IntNumber1's intValue() = IntNumber3's intValue() --> false
IntNumber1's floatValue() = IntNumber3's floatValue() --> false
IntNumber1's intValue() < IntNumber2's intValue() --> false
IntNumber1's intValue() < IntNumber3's intValue() --> true

-- Arithmatic can be preformed using NSNumbers
(IntNumber1's intValue()) + (IntNumber2's intValue()) --> (AppleScript integer) 6
(IntNumber3's intValue()) / (IntNumber1's intValue()) --> (AppleScript real) 2.0
```

```
use framework "Foundation"

set ANumber to 100.5

set IntNumber to current application's NSNumber's numberWithInteger:ANumber --> (NSNumber) 100
class of IntNumber --> (Objective-C) __NSCFNumber
-- floatValue turns it to an Objective-c floating point and the scripting bridge turns it into a AppleScript floating point
set IntegerTofloat to IntNumber's floatValue() -->  (AppleScript integer) 100.0

set FloatNumber to current application's NSNumber's numberWithFloat:ANumber --> (NSNumber) 100.5
class of FloatNumber --> (Objective-C) __NSCFNumber
set FloatAsInteger to FloatNumber's intValue() --> (AppleScript integer) 100

-- NSNumber Boolean variables can only be set to 1 or zero
set BooleanValue0 to current application's NSNumber's numberWithBool:0 --> (NSNumber) NO
class of BooleanValue0 --> (Objective-C) __NSCFBoolean
set BooleanAsInteger0 to BooleanValue0's intValue() --> (AppleScript integer) 1
set BooleanAsFloat0 to BooleanValue0's floatValue() --> (AppleScript real) 0.0

-- NSNumber Boolean variables can only be set to 1 or zero.  Any other number returns an error.
set BooleanValue1 to current application's NSNumber's numberWithBool:1 --> (NSNumber) YES
class of BooleanValue1 --> (Objective-C) __NSCFBoolean
set BooleanAsInteger1 to BooleanValue1's intValue() --> (AppleScript integer) 1
set BooleanAsFloat1 to BooleanValue1's floatValue() --> (AppleScript real) 1.0

BooleanValue0's stringValue() --> (NSString) "0"
BooleanValue1's stringValue() --> (NSString) "1"
-- Since BooleanAsInteger and BooleanAsFloat are AppleScript types stringValue() can not be used with them
-- Instead they need to type cast in AppleScript to make them strings
```

```
BooleanAsInteger0 as text --> (AppleScript string) "0"
BooleanAsFloat0 as text --> (AppleScript string) "0.0"

-- NSNumber can be compared
set IntNumber1 to current application's NSNumber's numberWithInteger:3
set IntNumber2 to current application's NSNumber's numberWithInteger:3
set IntNumber3 to current application's NSNumber's numberWithInteger:6

IntNumber1's isEqualToNumber:IntNumber2 --> true
IntNumber1's isEqualToNumber:IntNumber3 --> false

-- The NSNumber compare method returns 3 possible values when comparing 2 NSNumbers
-- For purposes of clarity given FirstNSNumber's compare:SecondNSNumber
-- FirstNSNumber will be called the "receiver" and SecondNSNumber will be called the "other number"
-- if compare returns -1 then the receiver is less than "other number"
-- if compare returns 1 then the receiver is greater than "other number"
-- if compare returns zero then the receiver is equal to "other number"
IntNumber1's compare:IntNumber3 --> (AppleScript integer) -1
IntNumber3's compare:IntNumber1 --> (AppleScript integer) 1
IntNumber1's compare:IntNumber2 --> (AppleScript integer) 0

-- Another way to compare NSNumbers is to convert them to regular numbers then compare
IntNumber1's intValue() = IntNumber2's intValue() --> true
IntNumber1's intValue() = IntNumber3's intValue() --> false
IntNumber1's floatValue() = IntNumber3's floatValue() --> false
IntNumber1's intValue() < IntNumber2's intValue() --> false
IntNumber1's intValue() < IntNumber3's intValue() --> true

-- Arithmatic can be preformed using NSNumbers
(IntNumber1's intValue()) + (IntNumber2's intValue()) --> (AppleScript integer) 6
(IntNumber3's intValue()) / (IntNumber1's intValue()) --> (AppleScript real) 2.0
```

```
use framework "Foundation"

set ANumber to 100.5

set IntNumber to current application's NSNumber's numberWithInteger:ANumber --> (NSNumber) 100
class of IntNumber --> (Objective-C) __NSCFNumber
-- floatValue turns it to an Objective-c floating point and the scripting bridge turns it into a AppleScript floating point
set IntegerTofloat to IntNumber's floatValue() -->  (AppleScript integer) 100.0

set FloatNumber to current application's NSNumber's numberWithFloat:ANumber --> (NSNumber) 100.5
class of FloatNumber --> (Objective-C) __NSCFNumber
set FloatAsInteger to FloatNumber's intValue() --> (AppleScript integer) 100

-- NSNumber Boolean variables can only be set to 1 or zero
set BooleanValue0 to current application's NSNumber's numberWithBool:0 --> (NSNumber) NO
class of BooleanValue0 --> (Objective-C) __NSCFBoolean
set BooleanAsInteger0 to BooleanValue0's intValue() --> (AppleScript integer) 1
set BooleanAsFloat0 to BooleanValue0's floatValue() --> (AppleScript real) 0.0

-- NSNumber Boolean variables can only be set to 1 or zero.  Any other number returns an error.
set BooleanValue1 to current application's NSNumber's numberWithBool:1 --> (NSNumber) YES
class of BooleanValue1 --> (Objective-C) __NSCFBoolean
set BooleanAsInteger1 to BooleanValue1's intValue() --> (AppleScript integer) 1
set BooleanAsFloat1 to BooleanValue1's floatValue() --> (AppleScript real) 1.0

BooleanValue0's stringValue() --> (NSString) "0"
BooleanValue1's stringValue() --> (NSString) "1"
-- Since BooleanAsInteger and BooleanAsFloat are AppleScript types stringValue() can not be used with them
-- Instead they need to type cast in AppleScript to make them strings
```

```
BooleanAsInteger0 as text --> (AppleScript string) "0"
BooleanAsFloat0 as text --> (AppleScript string) "0.0"

-- NSNumber can be compared
set IntNumber1 to current application's NSNumber's numberWithInteger:3
set IntNumber2 to current application's NSNumber's numberWithInteger:3
set IntNumber3 to current application's NSNumber's numberWithInteger:6

IntNumber1's isEqualToNumber:IntNumber2 --> true
IntNumber1's isEqualToNumber:IntNumber3 --> false

-- The NSNumber compare method returns 3 possible values when comparing 2 NSNumbers
-- For purposes of clarity given FirstNSNumber's compare:SecondNSNumber
-- FirstNSNumber will be called the "receiver" and SecondNSNumber will be called the "other number"
-- if compare returns -1 then the receiver is less than "other number"
-- if compare returns 1 then the receiver is greater than "other number"
-- if compare returns zero then the receiver is equal to "other number"
IntNumber1's compare:IntNumber3 --> (AppleScript integer) -1
IntNumber3's compare:IntNumber1 --> (AppleScript integer) 1
IntNumber1's compare:IntNumber2 --> (AppleScript integer) 0

-- Another way to compare NSNumbers is to convert them to regular numbers then compare
IntNumber1's intValue() = IntNumber2's intValue() --> true
IntNumber1's intValue() = IntNumber3's intValue() --> false
IntNumber1's floatValue() = IntNumber3's floatValue() --> false
IntNumber1's intValue() < IntNumber2's intValue() --> false
IntNumber1's intValue() < IntNumber3's intValue() --> true

-- Arithmatic can be preformed using NSNumbers
(IntNumber1's intValue()) + (IntNumber2's intValue()) --> (AppleScript integer) 6
(IntNumber3's intValue()) / (IntNumber1's intValue()) --> (AppleScript real) 2.0
```

```
use framework "Foundation"

set TheBehavior to current application's NSDecimalNumberHandler's decimalNumberHandlerWithRoundingMode:(current appli
NSRoundBankers) scale:4 raiseOnExactness:yes raiseOnOverflow:yes raiseOnUnderflow:yes raiseOnDivideByZero:yes

--   15 + 0.12355 = 15.12355
set NumberToAdd to (current application's NSDecimalNumber's decimalNumberWithString:"0.12355") --> (NSNumber) 0.123
class of NumberToAdd --> (Class) NSDecimalNumber
set ANumber to current application's NSDecimalNumber's decimalNumberWithString:"15" --> (NSNumber) 15.0
class of ANumber --> (Class) NSDecimalNumber
set ANumber to ANumber's decimalNumberByAdding:NumberToAdd withBehavior:TheBehavior --> (NSNumber) 15.1236
```

```
use framework "Foundation"

set TheBehavior to current application's NSDecimalNumberHandler's decimalNumberHandlerWithRoundingMode:(current appli
NSRoundBankers) scale:4 raiseOnExactness:yes raiseOnOverflow:yes raiseOnUnderflow:yes raiseOnDivideByZero:yes

set ANumber to current application's NSDecimalNumber's decimalNumberWithString:"15" --> (NSNumber) 15.0
class of ANumber --> (Class) NSDecimalNumber
set NumberToMultiplyBy to (current application's NSDecimalNumber's decimalNumberWithString:"3") --> (NSNumber) 3.0
class of NumberToMultiplyBy --> (Class) NSDecimalNumber
ANumber's decimalNumberByMultiplyingBy:NumberToMultiplyBy withBehavior:TheBehavior --> (NSNumber) 45.0
```

```
use framework "Foundation"

current application's NSDecimalNumberHandler's defaultDecimalNumberHandler()
```

Framework: **Foundation** Class: **NSDecimalNumberHandler** class item: **initWithRoundingMode:scale:rais**

```
use framework "Foundation"

current application's NSDecimalNumberHandler's alloc()'s initWithRoundingMode:(current application's NSRoundDown) scale:2
raiseOnExactness:yes raiseOnOverflow:yes raiseOnUnderflow:yes raiseOnDivideByZero:yes --> (Class) NSDecimalNumberHan
```

```
use framework "Foundation"

current application's NSDecimalNumber's decimalNumberWithString:"1047.853" --> (NSNumber) 1047.853
current application's NSDecimalNumber's decimalNumberWithString:"1000.001" --> (NSNumber) 1000.001

set ScientificNotation to current application's NSDecimalNumber's decimalNumberWithString:"2.33333E+5"  --> (NSNumber)
class of ScientificNotation --> (Class) NSDecimalNumber

current application's NSDecimalNumber's decimalNumberWithString:"2.1111E-5" --> (NSNumber) 0.000021111

current application's NSDecimalNumber's decimalNumberWithString:"-2.6666E-5" --> (NSNumber) -0.000026666

current application's NSDecimalNumber's decimalNumberWithString:"1E-5" --> (NSNumber) 0.00001
```

```
use framework "Foundation"

set ANumber to current application's NSDecimalNumber's decimalNumberWithString:"15" --> (NSNumber) 15.0
class of ANumber --> (Class) NSDecimalNumber
set NumberToSubtract to (current application's NSDecimalNumber's decimalNumberWithString:"3") --> (NSNumber) 3.0
class of NumberToSubtract --> (Class) NSDecimalNumber
set ANumber to ANumber's decimalNumberBySubtracting:NumberToSubtract --> (NSNumber) 12.0
```

```
use framework "Foundation"

set NumberToAdd to (current application's NSDecimalNumber's decimalNumberWithString:"3") --> (NSNumber) 3.0
class of NumberToAdd --> (Class) NSDecimalNumber
set ANumber to current application's NSDecimalNumber's decimalNumberWithString:"15"
class of ANumber --> (Class) NSDecimalNumber
set ANumber to ANumber's decimalNumberByAdding:NumberToAdd --> (NSNumber) 18.0
```

```
use framework "Foundation"

set ANumber to current application's NSDecimalNumber's decimalNumberWithString:"15" --> (NSNumber) 15.0
class of ANumber --> (Class) NSDecimalNumber
set NumberToSubtract to (current application's NSDecimalNumber's decimalNumberWithString:"3") --> (NSNumber) 3.0
class of NumberToSubtract --> (Class) NSDecimalNumber
set ANumber to ANumber's decimalNumberByMultiplyingBy:NumberToSubtract --> (NSNumber) 45.0
```

```
use framework "Foundation"

current application's NSDecimalNumber's alloc()'s initWithMantissa:10 exponent:-2 isNegative:no --> (NSNumber) 0.1
```

```
use framework "Foundation"

current application's NSDecimalNumber's alloc()'s initWithString:"2468" --> (NSNumber) 2468.0
```

```
use framework "Foundation"

set ANumber to current application's NSDecimalNumber's decimalNumberWithString:"5" --> (NSNumber) 5.0
class of ANumber --> (Class) NSDecimalNumber
set NumberToSubtract to (current application's NSDecimalNumber's decimalNumberWithString:"3") --> (NSNumber) 3.0
class of NumberToSubtract --> (Class) NSDecimalNumber
set ANumber to ANumber's decimalNumberByRaisingToPower:NumberToSubtract --> (NSNumber) 125.0
```

```
use framework "Foundation"

set ANumber to current application's NSDecimalNumber's decimalNumberWithString:"1.235675" --> (NSNumber) 1.235675
class of ANumber --> (Class) NSDecimalNumber
set TheExponent to (current application's NSDecimalNumber's decimalNumberWithString:"2") --> (NSNumber) 2.0
class of TheExponent --> (Class) NSDecimalNumber
set TheNumber to ANumber's decimalNumberByMultiplyingByPowerOf10:TheExponent --> (NSNumber) 123.5675
```

```
use framework "Foundation"

set ANumber to current application's NSDecimalNumber's decimalNumberWithString:"15" --> (NSNumber) 15.0
class of ANumber --> (Class) NSDecimalNumber
set NumberToDivideBy to (current application's NSDecimalNumber's decimalNumberWithString:"3") --> (NSNumber) 3.0
class of NumberToDivideBy --> (Class) NSDecimalNumber
set ANumber to ANumber's decimalNumberByDividingBy:NumberToDivideBy --> (NSNumber) 5.0
```

```
use framework "Foundation"

set TheNumber to current application's NSDecimalNumber's decimalNumberWithString:"45.5656" --> (NSNumber) 45.5656
class of TheNumber --> (Class) NSDecimalNumber
set DValue to TheNumber's doubleValue() --> (AppleScript real) 45.5656
```

```
use framework "Foundation"

current application's NSMutableIndexSet's superclass() --> (Class) NSIndexSet
```

```
use framework "Foundation"

current application's NSDecimalNumber's decimalNumberWithMantissa:949 exponent:-2 isNegative:no --> (NSNumber) 9.49
current application's NSDecimalNumber's decimalNumberWithMantissa:10 exponent:-2 isNegative:no --> (NSNumber) 0.1
current application's NSDecimalNumber's decimalNumberWithMantissa:8 exponent:2 isNegative:yes --> (NSNumber) -800.0
current application's NSDecimalNumber's decimalNumberWithMantissa:56 exponent:0 isNegative:no --> (NSNumber) 56.0
current application's NSDecimalNumber's decimalNumberWithMantissa:0 exponent:0 isNegative:no --> (NSNumber) 0.0
-- NaN = notANumber
current application's NSDecimalNumber's decimalNumberWithMantissa:0 exponent:0 isNegative:yes --> (NSNumber) NaN
```

```
use framework "Foundation"

set TheBehavior to current application's NSDecimalNumberHandler's decimalNumberHandlerWithRoundingMode:(current appli
NSRoundBankers) scale:4 raiseOnExactness:yes raiseOnOverflow:yes raiseOnUnderflow:yes raiseOnDivideByZero:yes

--   15 - 0.12345 = 14.87655
set ANumber to current application's NSDecimalNumber's decimalNumberWithString:"15" --> (NSNumber) 15.0
class of ANumber --> (Class) NSDecimalNumber
set NumberToSubtract to (current application's NSDecimalNumber's decimalNumberWithString:"0.12345") --> (NSNumber) 0
class of NumberToSubtract --> (Class) NSDecimalNumber
set ANumber to ANumber's decimalNumberBySubtracting:NumberToSubtract withBehavior:TheBehavior --> (NSNumber) 14.8
```

```
use framework "Foundation"

set TheBehavior to current application's NSDecimalNumberHandler's decimalNumberHandlerWithRoundingMode:(current appli
NSRoundPlain) scale:3 raiseOnExactness:yes raiseOnOverflow:yes raiseOnUnderflow:yes raiseOnDivideByZero:yes

set ANumber to current application's NSDecimalNumber's decimalNumberWithString:"1.235675" --> (NSNumber) 1.235675
class of ANumber --> (Class) NSDecimalNumber
set TheExponent to (current application's NSDecimalNumber's decimalNumberWithString:"2") --> (NSNumber) 2.0
class of TheExponent --> (Class) NSDecimalNumber
set TheNumber to ANumber's decimalNumberByMultiplyingByPowerOf10:TheExponent withBehavior:TheBehavior --> (NSNum
```

```
use framework "Foundation"

set TheBehavior to current application's NSDecimalNumberHandler's decimalNumberHandlerWithRoundingMode:(current appli
NSRoundBankers) scale:3 raiseOnExactness:yes raiseOnOverflow:yes raiseOnUnderflow:yes raiseOnDivideByZero:yes

set ANumber to current application's NSDecimalNumber's decimalNumberWithString:"2" --> (NSNumber) 2.0
class of ANumber --> (Class) NSDecimalNumber
set TheExponent to (current application's NSDecimalNumber's decimalNumberWithString:"3") --> (NSNumber) 3.0
class of TheExponent --> (Class) NSDecimalNumber
ANumber's decimalNumberByRaisingToPower:TheExponent withBehavior:TheBehavior --> (NSNumber) 8.0
```

```
use framework "Foundation"

set TheBehavior to current application's NSDecimalNumberHandler's decimalNumberHandlerWithRoundingMode:(current appli
NSRoundPlain) scale:3 raiseOnExactness:yes raiseOnOverflow:yes raiseOnUnderflow:yes raiseOnDivideByZero:yes

set ANumber to current application's NSDecimalNumber's decimalNumberWithString:"15" --> (NSNumber) 15.0
class of ANumber --> (Class) NSDecimalNumber
set NumberToDivideBy to (current application's NSDecimalNumber's decimalNumberWithString:"3") --> (NSNumber) 3.0
class of NumberToDivideBy --> (Class) NSDecimalNumber
set ANumber to ANumber's decimalNumberByDividingBy:NumberToDivideBy withBehavior:TheBehavior --> (NSNumber) 5.0
```

```
use framework "Foundation"

set TheBehavior to current application's NSDecimalNumberHandler's decimalNumberHandlerWithRoundingMode:(current appli
NSRoundBankers) scale:4 raiseOnExactness:yes raiseOnOverflow:yes raiseOnUnderflow:yes raiseOnDivideByZero:yes

set ANumber to current application's NSDecimalNumber's decimalNumberWithString:"15" --> (NSNumber) 15.0
class of ANumber --> (Class) NSDecimalNumber
set NumberToMultiplyBy to (current application's NSDecimalNumber's decimalNumberWithString:"3") --> (NSNumber) 3.0
class of NumberToMultiplyBy --> (Class) NSDecimalNumber
ANumber's decimalNumberByMultiplyingBy:NumberToMultiplyBy withBehavior:TheBehavior --> (NSNumber) 45.0
```

```
use framework "AppKit"
```

```
use framework "Foundation"

set TheNumber to current application's NSNumber's numberWithInteger:3 --> (NSNumber) 3
current application's NSStringFromClass(TheNumber's |class|()) as text --> __NSCFNumber

set TheString to current application's NSString's stringWithString:"AppleScripting" --> (NSString) "AppleScript"
current application's NSStringFromClass(TheString's |class|()) as text --> "__NSCFString"
```

```
use framework "AppKit"

on applicationShouldTerminate:sender
	-- This is where the code goes to do any cleanup before the application quits
	return current application's NSTerminateNow
end applicationShouldTerminate:
```

Framework: **Foundation** Class: **NSMetadataQuery** class item: **resultCount**

Framework: **Foundation** Class: **NSRect** class item: **NSMakeRect()**

Framework: **Foundation** Class: **NSRange** class item: **NSMakeRange()**

```
use framework "AppKit"

set TheView to current application's NSView's alloc()'s initWithFrame:(current application's NSMakeRect(0, 0, 500, 200))

set TheButton to current application's NSPopUpButton's alloc()'s initWithFrame:(current application's NSMakeRect(140, 110, 1
pullsDown:false

set TheButton to (current application's NSButton's alloc()'s initWithFrame:(current application's NSMakeRect(110, 10, 180, 40
TheButton's setButtonType:(current application's NSMomentaryLightButton)
TheButton's setBezelStyle:(current application's NSRoundedBezelStyle)
TheButton's setTitle:"Button title text"
TheButton's setTarget:me
TheButton's setAction:("clicked:")
TheButton's setKeyEquivalent:(return)

set ATextField to current application's NSTextField's alloc()'s initWithFrame:(current application's NSMakeRect(60, 110, 80, 20
ATextField's setEditable:false
ATextField's setStringValue:"Text to put in text field:"
ATextField's setDrawsBackground:false
ATextField's setBordered:false
```

```
use framework "Foundation"

set procInfo to current application's NSProcessInfo's processInfo()
procInfo --> <NSProcessInfo: 0x7fe779f1cc60>

set ArgumentList to procInfo's arguments()
ArgumentList --> /Applications/Script Debugger.app/Contents/MacOS/Script Debugger

set EnvironmentList to procInfo's environment()
(*
-->(NSDictionary) {
            PATH:"/usr/bin:/bin:/usr/sbin:/sbin",
            TMPDIR:"/var/folders/gx/swff6_0s6k5_6vh5qh0tn6000000gp/T/",
            LOGNAME:"bill",
            XPC_FLAGS:"0x0",
            HOME:"/Users/bill",
            Apple_PubSub_Socket_Render:"/private/tmp/com.apple.launchd.cYbyfFLjHp/Render",
            USER:"bill",
            SSH_AUTH_SOCK:"/private/tmp/com.apple.launchd.fNV1ZX5zDN/Listeners",
            DISPLAY:"/private/tmp/com.apple.launchd.nxspR6RPka/org.macosforge.xquartz:0",
            XPC_SERVICE_NAME:"com.latenightsw.ScriptDebugger6.805472",
            SHELL:"/bin/bash",
            __CF_USER_TEXT_ENCODING:"0x1F6:0x0:0x0" }
*)

-- Returns a global unique identifier for the process.
set AUniqueStr to procInfo's globallyUniqueString()
AUniqueStr --> (NSString) "50B0DDD9-6807-4736-A566-33AC413ED98F-3153-000042859372C56B"
```

```
set ProcessID to procInfo's processIdentifier() --> 3153

set ProcName to procInfo's processName() --> (NSString) "Script Debugger"

set TheHostName to procInfo's hostName() --> (NSString) "bills-second-imac.local"

set AVersionStr to procInfo's operatingSystemVersionString()
AVersionStr --> (NSString) "Version 10.11.6 (Build 15G1217)"

set OSVersion to procInfo's operatingSystemVersion()
OSVersion --> {majorVersion:10, minorVersion:11, patchVersion:6}

set CPUCoresCount to procInfo's processorCount() --> 4

set ActiveCPUCoresCount to procInfo's activeProcessorCount()

set TheRAMcapacity to procInfo's physicalMemory()
TheRAMcapacity --> 1.073741824E+9 = 1,073,741,824 = 1 gigabyte
-- 2^10 = 1,024 = 1K
-- 2^20 = 1,048,576 = 1mb
-- 2^30 = 1,073,741,824 = 1 gigabyte

set anRAMcapacity to procInfo's systemUptime() --> 7.5854402117396E+4
-- 7.5854402117396E+4 / (60 * 60) = 21.070667254832 hours ≈ 21 hours & 4.24 minutes

set aThermalState to procInfo's thermalState()
(* The result for procInfo's thermalState() are:
        NSProcessInfoThermalStateNominal
        NSProcessInfoThermalStateFair
        NSProcessInfoThermalStateSerious
        NSProcessInfoThermalStateCritical
```

```
use framework "Foundation"

set SDBundle to current application's NSBundle's mainBundle()

-- You can get all kinds of information from the info dictionary.
set InfoDictionaryVersion to (SDBundle's objectForInfoDictionaryKey:"CFBundleInfoDictionaryVersion") as string
InfoDictionaryVersion --> "6.0"

set DocumentTypes to (SDBundle's objectForInfoDictionaryKey:"CFBundleDocumentTypes")
CFBundleTypeExtensions of DocumentTypes -->
-- (NSArray) {{"scpt"},{"scptd"},{"app"},{"applescript"},{"sdef"},{"osax"},{"*"},{"sdtemplate"}}

set ApplicationCategory to (SDBundle's objectForInfoDictionaryKey:"LSApplicationCategoryType")
ApplicationCategory --> (NSString) "public.app-category.developer-tools"

set CopyrightText to (SDBundle's objectForInfoDictionaryKey:"NSHumanReadableCopyright")
return CopyrightText --> (NSString) "Copyright © 1993-2016 Late Night Software Ltd. All rights reserved."

-- This prints out the entire contents of the dictionary.  To check something out use the key to the left of the colon
-- for the input to objectForInfoDictionaryKey and you can get the value.
SDBundle's infoDictionary:"SDMainBundle's infoDictionary"
SDMainBundle's infoDictionary
(*
SDMainBundle's infoDictionary -->

(NSDictionary) {
        DTCompiler:"com.apple.compilers.llvm.clang.1_0",
        CFBundleURLTypes:{
                {
```

```
				CFBundleURLName:"AppleScript",
				CFBundleURLSchemes:{
					"applescript"
				},
				CFBundleTypeRole:"Editor"
			},
			{
				CFBundleURLName:"AppleScript (Script Debugger)",
				CFBundleURLSchemes:{
					"sdapplescript"
				},
				CFBundleTypeRole:"Editor"
			}
		},
		CFBundleInfoDictionaryVersion:"6.0",
		NSAppleScriptEnabled:YES,
		DTPlatformVersion:"GM",
		CFBundleIconFile:"AppIcon",
		CFBundleName:"Script Debugger",
		DTSDKName:"macosx10.12",
		NSContactsUsageDescription:"Your name will be used to populate template fields in newly created scripts.",
		SUEnableSystemProfiling:YES,
		NSPrincipalClass:"ScriptDebuggerApplication",
		LSApplicationCategoryType:"public.app-category.developer-tools",
		SUEnableAutomaticChecks:YES,
		SUFeedURL:"https://river.yodns.com/~alldritt/versions/com.latenightsw.ScriptDebugger6.php",
		CFBundleDocumentTypes:{
			{
				CFBundleTypeExtensions:{
					"scpt"
				},
use framework "Foundation"

-- There are many ways to create an NSBundle object
set TheURL to current application's NSURL's fileURLWithPath:"/Applications/Script Debugger.app"
set TheBundle to current application's NSBundle's bundleWithURL:TheURL
TheBundle --> NSBundle </Applications/Script Debugger.app> (loaded)
class of TheBundle --> (Class) NSBundle

set NewBundle to current application's NSBundle's mainBundle()
NewBundle --> NSBundle </Applications/Script Debugger.app> (loaded)
class of NewBundle --> (Class) NSBundle

set TheBundlePath to (NewBundle's bundlePath()) --> (NSString) "/Applications/Script Debugger.app"
set AnotherBundle to current application's NSBundle's bundleWithPath:TheBundlePath
AnotherBundle --> NSBundle </Applications/Script Debugger.app> (loaded)

set ThePath to (current application's NSWorkspace's sharedWorkspace()'s
absolutePathForAppBundleWithIdentifier:"com.latenightsw.ScriptDebugger6")
ThePath --> (NSString) "/Applications/Script Debugger.app"
set YetAnotherBundle to current application's NSBundle's bundleWithPath:ThePath --> NSBundle </Applications/Script Debugg
(loaded)

-- All 4 bundle object produce the same application Id
TheBundle's bundleIdentifier() --> (NSString) "com.latenightsw.ScriptDebugger6"
NewBundle's bundleIdentifier() --> (NSString) "com.latenightsw.ScriptDebugger6"
AnotherBundle's bundleIdentifier() --> (NSString) "com.latenightsw.ScriptDebugger6"
YetAnotherBundle's bundleIdentifier() --> (NSString) "com.latenightsw.ScriptDebugger6"

-- Numerous way to get a URL to Script Debugger
set TheURL to current application's NSWorkspace's sharedWorkspace()'s
```

```
URLForApplicationWithBundleIdentifier:"com.latenightsw.ScriptDebugger6"
TheURL --> (NSString) "/Applications/BBEdit.app"
class of TheURL --> (Class) NSURL
```

```
use framework "Foundation"

-- There are many ways to create an NSBundle object
set TheURL to current application's NSURL's fileURLWithPath:"/Applications/Script Debugger.app"
set TheBundle to current application's NSBundle's bundleWithURL:TheURL
TheBundle --> NSBundle </Applications/Script Debugger.app> (loaded)
class of TheBundle --> (Class) NSBundle

set NewBundle to current application's NSBundle's mainBundle()
NewBundle --> NSBundle </Applications/Script Debugger.app> (loaded)
class of NewBundle --> (Class) NSBundle

set TheBundlePath to (NewBundle's bundlePath()) --> (NSString) "/Applications/Script Debugger.app"
set AnotherBundle to current application's NSBundle's bundleWithPath:TheBundlePath
AnotherBundle --> NSBundle </Applications/Script Debugger.app> (loaded)

set ThePath to (current application's NSWorkspace's sharedWorkspace()'s
absolutePathForAppBundleWithIdentifier:"com.latenightsw.ScriptDebugger6")
ThePath --> (NSString) "/Applications/Script Debugger.app"
set YetAnotherBundle to current application's NSBundle's bundleWithPath:ThePath --> NSBundle </Applications/Script Debugg
(loaded)

-- All 4 bundle object produce the same application Id
TheBundle's bundleIdentifier() --> (NSString) "com.latenightsw.ScriptDebugger6"
NewBundle's bundleIdentifier() --> (NSString) "com.latenightsw.ScriptDebugger6"
AnotherBundle's bundleIdentifier() --> (NSString) "com.latenightsw.ScriptDebugger6"
YetAnotherBundle's bundleIdentifier() --> (NSString) "com.latenightsw.ScriptDebugger6"

-- Numerous way to get a URL to Script Debugger
set TheURL to current application's NSWorkspace's sharedWorkspace()'s
```

```
URLForApplicationWithBundleIdentifier:"com.latenightsw.ScriptDebugger6"
TheURL --> (NSString) "/Applications/BBEdit.app"
class of TheURL --> (Class) NSURL
```

```
use framework "Foundation"

-- There are many ways to create an NSBundle object
set TheURL to current application's NSURL's fileURLWithPath:"/Applications/Script Debugger.app"
set TheBundle to current application's NSBundle's bundleWithURL:TheURL
TheBundle --> NSBundle </Applications/Script Debugger.app> (loaded)
class of TheBundle --> (Class) NSBundle

set NewBundle to current application's NSBundle's mainBundle()
NewBundle --> NSBundle </Applications/Script Debugger.app> (loaded)
class of NewBundle --> (Class) NSBundle

set TheBundlePath to (NewBundle's bundlePath()) --> (NSString) "/Applications/Script Debugger.app"
set AnotherBundle to current application's NSBundle's bundleWithPath:TheBundlePath
AnotherBundle --> NSBundle </Applications/Script Debugger.app> (loaded)

set ThePath to (current application's NSWorkspace's sharedWorkspace()'s
absolutePathForAppBundleWithIdentifier:"com.latenightsw.ScriptDebugger6")
ThePath --> (NSString) "/Applications/Script Debugger.app"
set YetAnotherBundle to current application's NSBundle's bundleWithPath:ThePath --> NSBundle </Applications/Script Debug
(loaded)

-- All 4 bundle object produce the same application Id
TheBundle's bundleIdentifier() --> (NSString) "com.latenightsw.ScriptDebugger6"
NewBundle's bundleIdentifier() --> (NSString) "com.latenightsw.ScriptDebugger6"
AnotherBundle's bundleIdentifier() --> (NSString) "com.latenightsw.ScriptDebugger6"
YetAnotherBundle's bundleIdentifier() --> (NSString) "com.latenightsw.ScriptDebugger6"

-- Numerous way to get a URL to Script Debugger
set TheURL to current application's NSWorkspace's sharedWorkspace()'s
```

```
URLForApplicationWithBundleIdentifier:"com.latenightsw.ScriptDebugger6"
TheURL --> (NSString) "/Applications/BBEdit.app"
class of TheURL --> (Class) NSURL

TheBundle's bundleURL --> (NSURL) file:///Applications/Script%20Debugger.app/
TheBundle's bundlePath --> (NSString) "/Applications/Script Debugger.app"
```

```
use framework "AppKit"

set AppID to "com.latenightsw.ScriptDebugger6"
set AppPOSIXPath to (current application's NSWorkspace's sharedWorkspace()'s absolutePathForAppBundleWithIdentifier:App
AppPOSIXPath --> "/Applications/Script Debugger.app"

-- There are many ways to create an NSBundle object
set TheURL to current application's NSURL's fileURLWithPath:"/Applications/Script Debugger.app"
set TheBundle to current application's NSBundle's bundleWithURL:TheURL
TheBundle --> NSBundle </Applications/Script Debugger.app> (loaded)
class of TheBundle --> (Class) NSBundle

set NewBundle to current application's NSBundle's mainBundle()
NewBundle --> NSBundle </Applications/Script Debugger.app> (loaded)
class of NewBundle --> (Class) NSBundle

set TheBundlePath to (NewBundle's bundlePath()) --> (NSString) "/Applications/Script Debugger.app"
set AnotherBundle to current application's NSBundle's bundleWithPath:TheBundlePath
AnotherBundle --> NSBundle </Applications/Script Debugger.app> (loaded)

set ThePath to (current application's NSWorkspace's sharedWorkspace()'s
absolutePathForAppBundleWithIdentifier:"com.latenightsw.ScriptDebugger6")
ThePath --> (NSString) "/Applications/Script Debugger.app"
set YetAnotherBundle to current application's NSBundle's bundleWithPath:ThePath --> NSBundle </Applications/Script Debugg
(loaded)

-- All 4 bundle object produce the same application Id
TheBundle's bundleIdentifier() --> (NSString) "com.latenightsw.ScriptDebugger6"
NewBundle's bundleIdentifier() --> (NSString) "com.latenightsw.ScriptDebugger6"
AnotherBundle's bundleIdentifier() --> (NSString) "com.latenightsw.ScriptDebugger6"
```

```
YetAnotherBundle's bundleIdentifier() --> (NSString) "com.latenightsw.ScriptDebugger6"

-- Numerous way to get a URL to Script Debugger
set TheURL to current application's NSWorkspace's sharedWorkspace()'s
URLForApplicationWithBundleIdentifier:"com.latenightsw.ScriptDebugger6"
TheURL --> (NSString) "/Applications/BBEdit.app"
class of TheURL --> (Class) NSURL
```

```
use framework "Foundation"

-- There are many ways to create an NSBundle object
set TheURL to current application's NSURL's fileURLWithPath:"/Applications/Script Debugger.app"
set TheBundle to current application's NSBundle's bundleWithURL:TheURL
TheBundle --> NSBundle </Applications/Script Debugger.app> (loaded)
class of TheBundle --> (Class) NSBundle

set NewBundle to current application's NSBundle's mainBundle()
NewBundle --> NSBundle </Applications/Script Debugger.app> (loaded)
class of NewBundle --> (Class) NSBundle

set TheBundlePath to (NewBundle's bundlePath()) --> (NSString) "/Applications/Script Debugger.app"
set AnotherBundle to current application's NSBundle's bundleWithPath:TheBundlePath
AnotherBundle --> NSBundle </Applications/Script Debugger.app> (loaded)

set ThePath to (current application's NSWorkspace's sharedWorkspace()'s
absolutePathForAppBundleWithIdentifier:"com.latenightsw.ScriptDebugger6")
ThePath --> (NSString) "/Applications/Script Debugger.app"
set YetAnotherBundle to current application's NSBundle's bundleWithPath:ThePath --> NSBundle </Applications/Script Debugg
(loaded)

-- All 4 bundle object produce the same application Id
TheBundle's bundleIdentifier() --> (NSString) "com.latenightsw.ScriptDebugger6"
NewBundle's bundleIdentifier() --> (NSString) "com.latenightsw.ScriptDebugger6"
AnotherBundle's bundleIdentifier() --> (NSString) "com.latenightsw.ScriptDebugger6"
YetAnotherBundle's bundleIdentifier() --> (NSString) "com.latenightsw.ScriptDebugger6"

-- Numerous way to get a URL to Script Debugger
set TheURL to current application's NSWorkspace's sharedWorkspace()'s
```

```
URLForApplicationWithBundleIdentifier:"com.latenightsw.ScriptDebugger6"
TheURL --> (NSString) "/Applications/BBEdit.app"
class of TheURL --> (Class) NSURL
```

```
use framework "Foundation"

set theURL to current application's NSWorkspace's sharedWorkspace()'s
URLForApplicationWithBundleIdentifier:"com.latenightsw.ScriptDebugger6"
theURL --> (NSURL) file:///Applications/Script%20Debugger.app/

-- There are many ways to create an NSBundle object
set TheURL to current application's NSURL's fileURLWithPath:"/Applications/Script Debugger.app"
set TheBundle to current application's NSBundle's bundleWithURL:TheURL
TheBundle --> NSBundle </Applications/Script Debugger.app> (loaded)
class of TheBundle --> (Class) NSBundle

set NewBundle to current application's NSBundle's mainBundle()
NewBundle --> NSBundle </Applications/Script Debugger.app> (loaded)
class of NewBundle --> (Class) NSBundle

set TheBundlePath to (NewBundle's bundlePath()) --> (NSString) "/Applications/Script Debugger.app"
set AnotherBundle to current application's NSBundle's bundleWithPath:TheBundlePath
AnotherBundle --> NSBundle </Applications/Script Debugger.app> (loaded)

set ThePath to (current application's NSWorkspace's sharedWorkspace()'s
absolutePathForAppBundleWithIdentifier:"com.latenightsw.ScriptDebugger6")
ThePath --> (NSString) "/Applications/Script Debugger.app"
set YetAnotherBundle to current application's NSBundle's bundleWithPath:ThePath --> NSBundle </Applications/Script Debugg
(loaded)

-- All 4 bundle object produce the same application Id
TheBundle's bundleIdentifier() --> (NSString) "com.latenightsw.ScriptDebugger6"
NewBundle's bundleIdentifier() --> (NSString) "com.latenightsw.ScriptDebugger6"
AnotherBundle's bundleIdentifier() --> (NSString) "com.latenightsw.ScriptDebugger6"
YetAnotherBundle's bundleIdentifier() --> (NSString) "com.latenightsw.ScriptDebugger6"
```

```
-- Numerous way to get a URL to Script Debugger
set TheURL to current application's NSWorkspace's sharedWorkspace()'s URLForApplicationWithBundleIdentifier:"com.latenightsw.ScriptDebugger6"
TheURL --> (NSString) "/Applications/BBEdit.app"
class of TheURL --> (Class) NSURL
```

Framework: **Foundation** Class: **NSBundle** class item: **objectForInfoDictionaryKey**

```
use framework "Foundation"

set TheURL to current application's |NSURL|'s fileURLWithPath:"/Applications/Script Debugger.app"
TheURL --> (NSURL) file:///Applications/Script%20Debugger.app/

set TheBundle to current application's NSBundle's bundleWithURL:TheURL
TheBundle --> NSBundle </Applications/Script Debugger.app> (loaded)

set ThebundleIdentifier to TheBundle's bundleIdentifier() as text
ThebundleIdentifier --> "com.latenightsw.ScriptDebugger6"

set TheIconFileName to (TheBundle's objectForInfoDictionaryKey:"CFBundleIconFile")
TheIconFileName --> (NSString) "AppIcon"
```

```
use framework "Foundation"

TheBundle's executablePath() --> /Applications/Script Debugger.app/Contents/MacOS/Script Debugger
TheBundle's resourcePath() --> /Applications/Script Debugger.app/Contents/Resources
TheBundle's sharedFrameworksPath() --> /Applications/Script Debugger.app/Contents/SharedFrameworks
```

```
use framework "Foundation"

TheBundle's executablePath() --> /Applications/Script Debugger.app/Contents/MacOS/Script Debugger
TheBundle's resourcePath() --> /Applications/Script Debugger.app/Contents/Resources
TheBundle's sharedFrameworksPath() --> /Applications/Script Debugger.app/Contents/SharedFrameworks
```

Framework: **Foundation** Class: **NSBundle** class item: **sharedFrameworksPath()**

```
use framework "Foundation"

TheBundle's executablePath() --> /Applications/Script Debugger.app/Contents/MacOS/Script Debugger
TheBundle's resourcePath() --> /Applications/Script Debugger.app/Contents/Resources
TheBundle's sharedFrameworksPath() --> /Applications/Script Debugger.app/Contents/SharedFrameworks
```

Framework: **Foundation** Class: **NSBundle** class item: **bundleURL**

```
use framework "Foundation"

TheBundle's bundleURL --> (NSURL) file:///Applications/Script%20Debugger.app/
TheBundle's bundlePath --> (NSString) "/Applications/Script Debugger.app"
```

Framework: **Foundation** Class: **NSBundle** class item: **localizations**

```
use framework "Foundation"

set TheBundle to current application's NSBundle's mainBundle()
TheBundle's localizations --> (NSArray) {"English"}
```

```
use framework "Foundation"

set TheBundle to current application's NSBundle's mainBundle()
TheBundle's loaded --> (NSNumber) YES
TheBundle's unload --> (NSNumber) YES
TheBundle's loaded --> (NSNumber) NO
TheBundle's load --> (NSNumber) YES
TheBundle's loaded --> (NSNumber) YES
```

```
use framework "Foundation"

set TheBundle to current application's NSBundle's mainBundle()
TheBundle's loaded --> (NSNumber) YES
TheBundle's unload --> (NSNumber) YES
TheBundle's loaded --> (NSNumber) NO
TheBundle's load --> (NSNumber) YES
TheBundle's loaded --> (NSNumber) YES
```

Framework: **Foundation** Class: **NSBundle** class item: **unload**

```
use framework "Foundation"

set TheBundle to current application's NSBundle's mainBundle()
TheBundle's loaded --> (NSNumber) YES
TheBundle's unload --> (NSNumber) YES
TheBundle's loaded --> (NSNumber) NO
TheBundle's load --> (NSNumber) YES
TheBundle's loaded --> (NSNumber) YES
```

```
use framework "Foundation"

-- There are many ways to create an NSBundle object
set TheURL to current application's NSURL's fileURLWithPath:"/Applications/Script Debugger.app"
set TheBundle to current application's NSBundle's bundleWithURL:TheURL
TheBundle --> NSBundle </Applications/Script Debugger.app> (loaded)
class of TheBundle --> (Class) NSBundle

set NewBundle to current application's NSBundle's mainBundle()
NewBundle --> NSBundle </Applications/Script Debugger.app> (loaded)
class of NewBundle --> (Class) NSBundle

set TheBundlePath to (NewBundle's bundlePath()) --> (NSString) "/Applications/Script Debugger.app"
set AnotherBundle to current application's NSBundle's bundleWithPath:TheBundlePath
AnotherBundle --> NSBundle </Applications/Script Debugger.app> (loaded)

set ThePath to (current application's NSWorkspace's sharedWorkspace()'s
absolutePathForAppBundleWithIdentifier:"com.latenightsw.ScriptDebugger6")
ThePath --> (NSString) "/Applications/Script Debugger.app"
set YetAnotherBundle to current application's NSBundle's bundleWithPath:ThePath --> NSBundle </Applications/Script Debugg
(loaded)

-- All 4 bundle object produce the same application Id
TheBundle's bundleIdentifier() --> (NSString) "com.latenightsw.ScriptDebugger6"
NewBundle's bundleIdentifier() --> (NSString) "com.latenightsw.ScriptDebugger6"
AnotherBundle's bundleIdentifier() --> (NSString) "com.latenightsw.ScriptDebugger6"
YetAnotherBundle's bundleIdentifier() --> (NSString) "com.latenightsw.ScriptDebugger6"

-- Numerous way to get a URL to Script Debugger
set TheURL to current application's NSWorkspace's sharedWorkspace()'s
```

```
URLForApplicationWithBundleIdentifier:"com.latenightsw.ScriptDebugger6"
TheURL --> (NSString) "/Applications/BBEdit.app"
class of TheURL --> (Class) NSURL
```

```
use framework "Foundation"

set ResourcePath to "/Applications/Script Debugger.app/Contents/Resources/English.lproj"
ResourcePath --> "/Applications/Script Debugger.app/Contents/Resources/English.lproj"

set TheBundle to (current application's NSBundle's bundleWithPath:ResourcePath)
TheBundle --> NSBundle </Applications/Script Debugger.app/Contents/Resources/English.lproj> (not yet loaded)

set StringList to (TheBundle's pathsForResourcesOfType:"strings" inDirectory:"") as list

(*
        return StringList -->
        {
                "/Applications/Script Debugger.app/Contents/Resources/English.lproj/Application.strings",
                "/Applications/Script Debugger.app/Contents/Resources/English.lproj/InfoPlist.strings",
                "/Applications/Script Debugger.app/Contents/Resources/English.lproj/LNSProjectEditor.strings",
                "/Applications/Script Debugger.app/Contents/Resources/English.lproj/LNSPSMTabWindow.strings",
                "/Applications/Script Debugger.app/Contents/Resources/English.lproj/MVPreferencePaneGroups.strings"
        }
*)
```