AppleScriptObjC relies on the automatic conversion of text, numbers, booleans, lists and records to and from their Cocoa counterparts. However, there is no conversion of dates, files, or data objects. It also loses precision when converting floating-point Cocoa numbers to reals. This framework provides these extra conversions and precision with reals, as well as some trigonometry functions, several text- and list-handling routines based on those in ASObjC Runner, and a handful of other methods.

You can use the framework by putting it in ~/Library/Frameworks/ or /Library/Frameworks/, or by adding it to a script bundle or applet.

Most methods will return missing value where there is an error, and if they have an error parameter the returned error will have a description in localizedError:.

You can use and distribute this framework free of charge, but it must include this header file. There are no guarantees or warranties whatsoever. Use entirely at your own risk. Feedback welcome to <sstanley@myriad-com.com.au>.

Shane Stanley

---

Spread of AppleScriptObjC World and The main target of this scripting guide (blue)

<table>
<thead>
<tr>
<th>OS X ver</th>
<th>ASObjC (Xcode)</th>
<th>Cocoa Applet</th>
<th>AppleScript Libraries</th>
<th>ASObjC on Script Editor</th>
</tr>
</thead>
<tbody>
<tr>
<td>10.6</td>
<td>Xcode</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>10.7</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>10.8</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>10.9</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>10.10</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Copyright Notice

Mac, OS X, Mac OS X, Siri, Newton, AppleScript and AppleScriptObjC are registered trade mark of US Apple Inc. UDing and UDing Simulator are registered trade mark of Toyo Ink SC Holdings.co.,Ltd. Adobe Photoshop is registered trade mark of US Adobe Software. ASObjCExtra.framework and ASObjCExplorer 4 are copyrighted software by Shane Stanley. "AS Hole" (http://piyocast.com/as) is an AppleScript blog written by Takaaki Naganoya.
About this “Scripting Guide”

One day, I received a private e-mail from Shane Stanley. He wanted me to evaluate his framework “ASObjCExtras”. All of AppleScripters enjoy his e-book “AppleScriptObjC Explored” and “Everyday AppleScriptObjC”. His devotion to AppleScript world is huge, important and wonderful (I remember same great people: Bill Cheeseman and Mark Alldritt). Shane has been the only & absolute lighthouse for us for this several years. Who can decline his request? My answer was “Yes I do”. There was no choice.

In historical perspective, there were some open member project in AppleScript world. I deserted them all. One project did not work on my mother tongue (Japanese) environment, another seemed so charmless. We did not help each other. I thought it was our mistake. It is the time to make action.

ASObjCExtras is tidy and powerful framework. It made more safer and faster to use Cocoa framework. It is very fun to write scripts with ASObjCExtras. Shane’s powerful editor “ASObjCEplorer 4”’s cocoa-object/event logging made it easier, too. AppleScript gets 10-200 times faster speed by using Cocoa framework with keeping its simplicity and easiness. ASObjCExtras includes many useful methods for replacing frequently used AppleScript routines. You can replace pure-AppleScript-based sort routine with ASObjCExtras-based high-speed version. Many heavily looped scripts will be shorten by ASObjCExtras and shorten its processing period dramatically.

When I wrote a lot of AppleScript with earlier version of ASObjCExtras, sometimes I faced to the lack of scripting guide. So, I wrote this document in my free time. That’s all. I made many questions to Shane and received his great advices. But all of this document and sample scripts are written and checked by me. If this document is wrong, it is not Shane’s fault. It is mine. All of scripts are checked with ASObjCEplorer 4 and Apple’s Script Editor on OS X, Yosemite. Welcome your comment or feedback to me (mailto:maro@piyocast.com). Day by day, I write AppleScript on my blog (http://piyocast.com/as/) . So, you can see the latest samples there.

Takaaki Naganoya
(Project Manager, Programmer, Planner, Writer and Editor)
My earlier and most important project was "Newt On" / "Newt Off" (2002). This full AppleScript-written program is a small "Knowledge Navigator". It understands and execute natural language (Japanese). Similar to Apple’s ancient PDA “Newton”’s assistant function. Voice control version existed named “Kotodama” (2003). Over 10 years ago, I made a kind of Apple’s “Siri” by only AppleScript. These technologies are put into my later works.

I wrote long and many AppleScript routines in “Newt On” and “Kotodama” project. It was too hard to maintain them all by myself. So, I opened them on my AppleScript Blog “AS Hole” (2008-Today). Today, this site is listed in the first page of Google’s search engine result with keyword "AppleScript" and gains over 1,000,000 page views/year.

TOYO INK, the second largest ink maker in the world has been releasing “UDing” series softwares. I’m an external project manager and programmer in this project. “UDing Simulator” software is made for color blind simulation. It is written in Objective-C++ and AppleScriptObjC; controls Adobe Photoshop and keep correct ICC profiles. This project is evaluated very well in Japan.

—— My Books ——

LEFT: “The Road to REALbasic Master”, Softbank Creative, 1999★


What is this?

(1) ASObjCExtras (&AppleScriptObjC) brings **High-speed data processing** ability to AppleScripters.

**1D List Uniquefy**

Over 30 times Faster!

First Run: 0.446 seconds  
Average: 0.197 seconds

<table>
<thead>
<tr>
<th>Time (sec)</th>
<th>AppleScript</th>
<th>ASOC with ASObjCExtras</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>5.712</td>
<td>5.735</td>
</tr>
<tr>
<td>1.5</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4.5</td>
<td></td>
<td></td>
</tr>
<tr>
<td>6</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

(2) ASObjCExtras provides the way to avoid Scripting Bridge's **casting bug**.

Cocoa NSArray: [1.1, 2.1]

**Error**

ASify() as list

AppleScript List: \{1.1000012, 2.10000123\}

More Safer!

Cocoa NSArray: [1.1, 2.1]

**More Safer!**

AppleScript List: \{1.1, 2.1\}

Cocoa Object

AppleScript Object

(3) ASObjCExtras reflects the skillful scripters' (= your) requests.

Request Reflected
ASObjCExtras.framework

How To Install

http://www.macosxautomation.com/applescript/apps/ASObjCExtras.html

Put “ASObjCExtras.framework” to

~/Library/Frameworks

or

/Library/Frameworks

You can freely re-distribute and code-signing your script with ASObjCExtras framework (so, this framework is not code-signed).

ASObjCExtras.framework supports with:

OS X 10.10 or later

OS X Yosemite
ASObjCExtras.framework

How To Use

Just write: use framework “ASObjCExtras”

Script Editor

```
use AppleScript version "2.4"
use scripting additions
use framework "Foundation"
use framework "ASObjCExtras"

set theVersionInfo to current application's SMSFord's ldescription() as text
```

ASObjCExplorer 4

```
-- Created 2014-11-26 16:26:04 +0900 by Takaaki Naganoya
-- 2014 Piyomaru Software

use AppleScript version "2.4"
use scripting additions
use framework "Foundation"
use framework "ASObjCExtras"

current application's SMSFord's ldescription()
---> SMSFord 1.2.0b1
```
How To Receive error details

Where a method has an "error" parameter, it should say that it takes missing value or reference. And if you use Shane Stanley's ASObjCExplorer 4, you can check the error details directly.

ASObjCExplorer 4

Methods can return error (Example. Not all)

<table>
<thead>
<tr>
<th>Method</th>
<th>Error Parameter</th>
</tr>
</thead>
<tbody>
<tr>
<td>subarraysIn: paddedWith:</td>
<td>error</td>
</tr>
<tr>
<td>subarraysFrom: groupedBy:</td>
<td>error</td>
</tr>
<tr>
<td>subarraysIn: withItems: insertedAtIndex:</td>
<td>error</td>
</tr>
<tr>
<td>arrayByMovingItemAt: toIndex: inArray:</td>
<td>error</td>
</tr>
<tr>
<td>sumMaxMinOf:</td>
<td>error</td>
</tr>
<tr>
<td>subarraysIn: sortedByIndexes: ascending:</td>
<td>error</td>
</tr>
<tr>
<td>sortTypes:</td>
<td>error</td>
</tr>
</tbody>
</table>

ASObjCExtras.framework

1. Select "Log" tab
2. Click "Run & Log"
If you use **Script Editor** to write ASOC scripts, you can get error details with some additional descriptions.

```applescript
-- Created 2015-01-13 by Takaaki Naganoya
-- 2015 Piyomaru Software
use AppleScript version "2.4"
use scripting additions
use framework "Foundation"
use framework "ASObjCExtras"

set listOfLists to {{1, 2, 3, 4}, {11, 22, 33}}
setat aRes to current application's SMSFord's colsToRowsIn:listOfLists lerror: (missing value)
setat {theArray, theError} to current application's SMSFord's
colsToRowsIn:listOfLists lerror:(reference)
set errorDetails to theError's localizedDescription()
set errorRes to errorDetails as string

--> "Can't perform cols to rows when lists have differing item counts."
```

**Requires additional descriptions**
How To Distribute with Applet

You can contain & re-distribute ASObjCExtras.framework within AppleScript applet’s bundle
<table>
<thead>
<tr>
<th>Kind</th>
<th>Method</th>
<th>Appearance version</th>
<th>Dim</th>
</tr>
</thead>
<tbody>
<tr>
<td>Other</td>
<td>description()</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>openHeaderFile()</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Class Conversion</td>
<td>ASify()</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>ASifyInList()</td>
<td>1.2</td>
<td></td>
</tr>
<tr>
<td></td>
<td>ASifyFor()</td>
<td>1.2</td>
<td></td>
</tr>
<tr>
<td></td>
<td>ASifyListFor()</td>
<td>1.2</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Cocoaify:</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Cocoaify: forTypes:</td>
<td>1.2</td>
<td></td>
</tr>
<tr>
<td>Path Manipulation</td>
<td>URLFrom:</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>fileFromURL:</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>HFSPathFromURL:</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>HFSPathFromURL: colonForPackages:</td>
<td>1.2</td>
<td></td>
</tr>
<tr>
<td>String Manipulation</td>
<td>datesFromStrings: format:</td>
<td>1D→0D, 1D→1D</td>
<td></td>
</tr>
<tr>
<td></td>
<td>stringFrom: ICUTransform: inverse:</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>stringFrom: makingIt:</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>arrayFromTSV:</td>
<td>→2D</td>
<td></td>
</tr>
<tr>
<td></td>
<td>arrayFromCSV: commals:</td>
<td>→2D</td>
<td></td>
</tr>
<tr>
<td>List Manipulation</td>
<td>subarraysIn: paddedWith: error:</td>
<td>2D</td>
<td></td>
</tr>
<tr>
<td></td>
<td>colsToRowsIn: error:</td>
<td>2D</td>
<td></td>
</tr>
<tr>
<td></td>
<td>arrayByFlattening:</td>
<td>2D→1D</td>
<td></td>
</tr>
<tr>
<td></td>
<td>arrayByFullyFlattening:</td>
<td>1.2</td>
<td>nD→1D</td>
</tr>
<tr>
<td>Kind</td>
<td>Method</td>
<td>Appearance version</td>
<td>Dim</td>
</tr>
<tr>
<td>-----------------</td>
<td>------------------------------------------------------------------------</td>
<td>--------------------</td>
<td>------</td>
</tr>
<tr>
<td>List Manipulation</td>
<td>subarraysFrom: groupedBy: error:</td>
<td></td>
<td>1D→2D</td>
</tr>
<tr>
<td></td>
<td>subarraysIn: withItems: insertedAtIndex: error:</td>
<td></td>
<td>2D</td>
</tr>
<tr>
<td></td>
<td>arrayByInsertingItems: inArray: atIndex: error:</td>
<td></td>
<td>1D,2D</td>
</tr>
<tr>
<td></td>
<td>arrayByMovingItemAt: toIndex: inArray: error:</td>
<td></td>
<td>1D,2D</td>
</tr>
<tr>
<td></td>
<td>arrayByDeletingBlanksIn:</td>
<td></td>
<td>1D</td>
</tr>
<tr>
<td></td>
<td>arrayByTrimmingTrailingBlanksFrom:</td>
<td></td>
<td>1D</td>
</tr>
<tr>
<td></td>
<td>arrayByTrimmingBlanksFrom:</td>
<td></td>
<td>1D</td>
</tr>
<tr>
<td></td>
<td>arrayByReplacingNullsIn: withItem:</td>
<td></td>
<td>1D</td>
</tr>
<tr>
<td></td>
<td>arrayWithPattern: startNumber: endNumber: minDigits:</td>
<td></td>
<td>1D</td>
</tr>
<tr>
<td></td>
<td>sumMaxMinOf: error:</td>
<td></td>
<td>1D</td>
</tr>
<tr>
<td></td>
<td>subarraysIn: sortedByIndexes: ascending: sortTypes: error:</td>
<td></td>
<td>2D</td>
</tr>
<tr>
<td></td>
<td>subarraysIn: sortedByIndexes: ascending: sortTypes: sortKeys: error:</td>
<td></td>
<td>2D→2D</td>
</tr>
<tr>
<td></td>
<td>arrayByMergingTextAtIndexes: inArray: inserting: error:</td>
<td></td>
<td>1D</td>
</tr>
<tr>
<td></td>
<td>indexesOfItem: inArray: inverting:</td>
<td></td>
<td>1D</td>
</tr>
<tr>
<td></td>
<td>indexesOfItems: inArray: inverting:</td>
<td></td>
<td>1D</td>
</tr>
<tr>
<td></td>
<td>subarraysIn: asDictionariesUsingLabels: error:</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
### Table Of Contents:

<table>
<thead>
<tr>
<th>Kind</th>
<th>Method</th>
<th>Appearance version</th>
<th>Dim</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>List Manipulation</strong></td>
<td>subarraysFrom: usingKeys: outKeys: error:</td>
<td>1.2</td>
<td>2D/1D</td>
</tr>
<tr>
<td></td>
<td>arrayWithIndexSet:</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>arrayByAddingInteger: inArray:</td>
<td>1.2</td>
<td>1D</td>
</tr>
<tr>
<td><strong>Trigonometry</strong></td>
<td>tanValueOf:</td>
<td>0D/1D</td>
<td></td>
</tr>
<tr>
<td></td>
<td>sinValueOf:</td>
<td>0D/1D</td>
<td></td>
</tr>
<tr>
<td></td>
<td>cosValueOf:</td>
<td>0D/1D</td>
<td></td>
</tr>
<tr>
<td></td>
<td>atanValueOf:</td>
<td>0D/1D</td>
<td></td>
</tr>
<tr>
<td></td>
<td>asinValueOf:</td>
<td>0D/1D</td>
<td></td>
</tr>
<tr>
<td></td>
<td>acosValueOf:</td>
<td>0D/1D</td>
<td></td>
</tr>
<tr>
<td></td>
<td>tanhValueOf:</td>
<td>0D/1D</td>
<td></td>
</tr>
<tr>
<td></td>
<td>sinhValueOf:</td>
<td>0D/1D</td>
<td></td>
</tr>
<tr>
<td></td>
<td>coshValueOf:</td>
<td>0D/1D</td>
<td></td>
</tr>
<tr>
<td></td>
<td>atanhValueOf:</td>
<td>0D/1D</td>
<td></td>
</tr>
<tr>
<td></td>
<td>asinhValueOf:</td>
<td>0D/1D</td>
<td></td>
</tr>
<tr>
<td></td>
<td>acoshValueOf:</td>
<td>0D/1D</td>
<td></td>
</tr>
<tr>
<td></td>
<td>logValueOf:</td>
<td>0D/1D</td>
<td></td>
</tr>
<tr>
<td></td>
<td>log10ValueOf:</td>
<td>0D/1D</td>
<td></td>
</tr>
<tr>
<td><strong>Misc.</strong></td>
<td>metadataFromImage: error:</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>infoForFile:</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>sizeInfoForFile:</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Every syntax element’s colors are different a quite little. So, scripts can detect each syntax element by itself. My “Context menu assistant” acknowledges each syntax element and replace only variable’s name or make routine caller-relation map from handler names.

My routine caller-relation mapping generator AppleScript make it easier to understand large size of AppleScript program.

This Script detects syntax elements by its colors. My Script Editor settings are optimized for such a self-analysis solutions & my color taste.
ASObjCExtras.framework
Utility methods

- Utility
- Misc
- File Info
- String
- Path
- Class
- Trigonometry
- List
<table>
<thead>
<tr>
<th>method</th>
<th>description()</th>
</tr>
</thead>
<tbody>
<tr>
<td>Introduction</td>
<td>Returns ASObjCExtras.framework version information</td>
</tr>
<tr>
<td>kind</td>
<td>Utility</td>
</tr>
<tr>
<td>Input</td>
<td>No</td>
</tr>
<tr>
<td>Output</td>
<td>NSString / ( &quot;SMSFord &quot; &amp; version No String. )</td>
</tr>
<tr>
<td>Notes</td>
<td>Use this method to check ASObjCExtras.framework is installed or not. And check which version is installed. This &quot;description&quot; word conflicts with another application's (ex. Apple's &quot;Script Editor&quot;) reserved word. So, this function requires &quot;</td>
</tr>
</tbody>
</table>

```apple_script
set theVersionInfo to current application's SMSFord's |description|() as text
---> "SMSFord 1.2.2"
```
<table>
<thead>
<tr>
<th>Method</th>
<th>openHeaderFile()</th>
</tr>
</thead>
<tbody>
<tr>
<td>Introduction</td>
<td>Open ASObjCExtras.framework's header file with Xcode</td>
</tr>
<tr>
<td>Kind</td>
<td>Utility</td>
</tr>
<tr>
<td>Input</td>
<td>No</td>
</tr>
<tr>
<td>Output</td>
<td>No (Open Header file with Xcode)</td>
</tr>
<tr>
<td>Notes</td>
<td>If Xcode is not installed, nothing happen.</td>
</tr>
</tbody>
</table>

--- Sample Code

```appleScript
use AppleScript version "2.4"
use scripting additions
use framework "Foundation"
use framework "ASObjCExtras"

current application's SMSFord's openHeaderFile()
```
ASObjCExtras.framework

Class Conversion methods
### Method

#### Introduction
Converts any NSDate objects to AS dates, and any floating-point numbers to reals, to work around the imprecision of the built-in conversion. Result must still be coerced.

<table>
<thead>
<tr>
<th>Kind</th>
<th>Class Conversion</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Input</strong></td>
<td>1: Cocoa Object</td>
</tr>
<tr>
<td>2: AppleScript class</td>
<td></td>
</tr>
<tr>
<td><strong>Output</strong></td>
<td>Cocoa Object</td>
</tr>
<tr>
<td><strong>Notes</strong></td>
<td>Typical use: set theList to anNSArray's ASify() as list</td>
</tr>
</tbody>
</table>

---

#### Sample Code

```
use AppleScript version "2.4"
use framework "Foundation"
use framework "ASObjCExtras"
use scripting additions

set aList to {1, 2, 3, 4}
set aArray to current application's NSArray's arrayWithArray:aList
set bList to (aArray's ASify()) as list
--> {1, 2, 3, 4}
```

---

<table>
<thead>
<tr>
<th>Description</th>
<th>Before (b)</th>
<th>After (a)</th>
</tr>
</thead>
<tbody>
<tr>
<td>set a to (b's ASify()) as list</td>
<td>NSArray</td>
<td>list</td>
</tr>
<tr>
<td>set a to (b's ASify()) as list</td>
<td>NSMuttableArray</td>
<td>list</td>
</tr>
<tr>
<td>set a to (b's ASify()) as record</td>
<td>NSDictionary</td>
<td>record</td>
</tr>
<tr>
<td>set a to (b's ASify()) as record</td>
<td>NSMuttableDictionary</td>
<td>record</td>
</tr>
<tr>
<td>set a to (b's ASify()) as date</td>
<td>NSDate</td>
<td>date</td>
</tr>
<tr>
<td>set a to (b's ASify()) as real</td>
<td>floating-point numbers</td>
<td>real</td>
</tr>
<tr>
<td>method</td>
<td>1’s ASifyInList() as 2</td>
<td></td>
</tr>
<tr>
<td>------------------------</td>
<td>------------------------</td>
<td></td>
</tr>
<tr>
<td>Introduction</td>
<td>As ASify(), but returns the result in a single-item list. Use this when unsure of the class of the result, and extract the first item from the result coerced to a list. Works around recursion problem.</td>
<td></td>
</tr>
<tr>
<td>kind</td>
<td>Class Conversion</td>
<td></td>
</tr>
<tr>
<td>Input</td>
<td>1:Cocoa Object 2:AppleScript class</td>
<td></td>
</tr>
<tr>
<td>Output</td>
<td>NSArray</td>
<td></td>
</tr>
<tr>
<td>Notes</td>
<td>Typical use: set the Thing to item 1 of (an NSThing’s ASifyInList() as list)</td>
<td></td>
</tr>
</tbody>
</table>

---

Sample Code

```applecript
-- Sample Code
-- Created 2014-12-29 by Takaaki Naganoya
-- 2014 Piyomaru Software
use AppleScript version "2.4"
use framework "Foundation"
use framework "ASObjCExtras"
use scripting additions

set aText to "ABCED"
set aNSString to current application's NSString's stringWithString:aText
set bRes to item 1 of ((aNSString's ASifyInList) as list)
--> "ABCED"
```
### NFUIPE's ASifyFor:

<table>
<thead>
<tr>
<th>Method</th>
<th>➊’s ASifyFor: ➋</th>
</tr>
</thead>
<tbody>
<tr>
<td>Introduction</td>
<td>Convert Cocoa object to AppleScript object</td>
</tr>
<tr>
<td>Kind</td>
<td>Class Conversion</td>
</tr>
<tr>
<td>Input</td>
<td>➊Cocoa Object  ➋AppleScript string</td>
</tr>
<tr>
<td>Output</td>
<td>NSArray</td>
</tr>
<tr>
<td>Notes</td>
<td>Similar to ASify, but you can control which classes get converted. The string you pass can be comma- or space-delimited.</td>
</tr>
</tbody>
</table>

---

**Sample Code**

-- Sample Code

-- Created 2014-12-29 by Takaaki Naganoya

---

```
use AppleScript version "2.4"
use scripting additions
use framework "Foundation"
use framework "ASObjCExtras"

set aList to {1.1, 2.1, 3.1, 4.1, current date}
set aArray to current application's SMSFord's Cocoaify:aList

---

set bList to (aArray's ASifyFor:"reals, dates") as list

--> {1.1, 2.1, 3.1, 4.1, date "Friday, January 9, 2015 at 8:28:21 PM"}
```

---

<table>
<thead>
<tr>
<th>Description</th>
<th>After (AppleScript Object)</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;reals&quot;, &quot;floating-point numbers&quot;</td>
<td>real</td>
</tr>
<tr>
<td>&quot;dates&quot;, &quot;NSDate&quot;</td>
<td>date</td>
</tr>
<tr>
<td>&quot;data&quot;, &quot;NSData&quot;</td>
<td>data</td>
</tr>
<tr>
<td>&quot;files&quot;, &quot;NSURL&quot;</td>
<td>files («class furl»)</td>
</tr>
</tbody>
</table>
### 1’s ASifyListFor()

<table>
<thead>
<tr>
<th>method</th>
<th>Convert Cocoa object to AppleScript object</th>
</tr>
</thead>
<tbody>
<tr>
<td>kind</td>
<td>Class Conversion</td>
</tr>
<tr>
<td>Input</td>
<td>1Cocoa Object (NSArray)</td>
</tr>
<tr>
<td>Output</td>
<td>NSArray</td>
</tr>
<tr>
<td>Notes</td>
<td>As ASifyFor:, but returns the result in a single-item list. See ASifyInList. In this case, you don’t have to know what class to coerce the result to—you coerce it to a list every time. So you can use it in a general handler for several classes.</td>
</tr>
</tbody>
</table>

---

Sample Code

-- Created 2015-01-06 by Takaaki Naganoya

```applescript
-- 2015 Piyomaru Software
use AppleScript version "2.4"
use scripting additions
use framework "Foundation"
use framework "ASObjCExtras"

set aList to {1.1, 2.1, 3.1, 4.1, current date}
set aArray to current application's CMSFord's Cocoaify:aList
set aRes to item 1 of ((aArray's ASifyInListFor:"dates reals") as list)
--> {1.1, 2.1, 3.1, 4.1, date "Friday, January 9, 2015 at 8:29:33 PM"}
```

<table>
<thead>
<tr>
<th>description</th>
<th>After (AppleScript Object)</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;reals&quot;, &quot;floating-point numbers&quot;</td>
<td>real</td>
</tr>
<tr>
<td>&quot;dates&quot;, &quot;NSDates&quot;</td>
<td>date</td>
</tr>
<tr>
<td>&quot;data&quot;, &quot;NSData&quot;</td>
<td>data</td>
</tr>
<tr>
<td>&quot;files&quot;, &quot;NSURLs&quot;</td>
<td>files («class furl»)</td>
</tr>
<tr>
<td>Method</td>
<td>Cocoaify:①</td>
</tr>
<tr>
<td>----------</td>
<td>------------</td>
</tr>
<tr>
<td>Introduction</td>
<td>Convert AppleScript object to Cocoa Object</td>
</tr>
<tr>
<td>Kind</td>
<td>Class Conversion</td>
</tr>
<tr>
<td>Input</td>
<td>AppleScript Object</td>
</tr>
<tr>
<td>Output</td>
<td>Cocoa Object</td>
</tr>
<tr>
<td>Notes</td>
<td>For converting from AppleScript objects to Cocoa objects. Conversion is recursive through any contained records and lists.</td>
</tr>
</tbody>
</table>

---

Sample Code

```plaintext
-- Sample Code
-- Created 2014-11-29 by Takaaki Naganoya
-- 2014 Piyomaru Software
use AppleScript version "2.4"
use scripting additions
use framework "Foundation"
use framework "ASObjCExtras"

set aList to {1, 2, 3, 4.1}
set aArray to current application's SMSFord's Cocoaify:aList
set bList to aArray's ASify() as list
--> {1, 2, 3, 4.1}
```

<table>
<thead>
<tr>
<th>Before</th>
<th>After</th>
</tr>
</thead>
<tbody>
<tr>
<td>list</td>
<td>NSArray, NSMutabileArray</td>
</tr>
<tr>
<td>record</td>
<td>NSDictionary, NSMutabileDictionary</td>
</tr>
<tr>
<td>date</td>
<td>NSDate</td>
</tr>
<tr>
<td>real</td>
<td>floating-point numbers</td>
</tr>
</tbody>
</table>
**Cocoaify:**① forTypes:②

**Introduction**
Convert AppleScript object to Cocoa Object

**Kind**
Class Conversion

**Input**
①AppleScript Object ②Cocoa Object Class

**Output**
NSArray

**Notes**
This method is for converting from AppleScript objects to Cocoa objects.
Conversion is recursive through any contained records and lists.

---

**Sample Code**
-- Created 2014-11-29 by Takaaki Naganoya
-- 2014 Piyomaru Software

```applescript
use AppleScript version "2.4"
use scripting additions
use framework "Foundation"
use framework "ASObjCExtras"

set aDate to {current date, (current date) + 1, (current date) + 2}
set bNSDate to (current application's SMSFord's Cocoaify:aDate forTypes:"date")
set bDateList to bNSDate's ASify() as list

--> {date "Friday, January 9, 2015 at 8:30:15 PM", date "Friday, January 9, 2015 at 8:30:16 PM", date "Friday, January 9, 2015 at 8:30:17 PM"}  
```

<table>
<thead>
<tr>
<th>Before (AppleScript Object)</th>
<th>After (Cocoa Object)</th>
</tr>
</thead>
<tbody>
<tr>
<td>dates</td>
<td>NSDates</td>
</tr>
<tr>
<td>data</td>
<td>NSData</td>
</tr>
<tr>
<td>files, files and aliases</td>
<td>NSURLs</td>
</tr>
</tbody>
</table>
ASObjCExtras.framework
Path Manipulation methods
<table>
<thead>
<tr>
<th>method</th>
<th>URLFrom:①</th>
</tr>
</thead>
<tbody>
<tr>
<td>Introduction</td>
<td>Pass an alias, file, HFS path or POSIX path, and an NSURL will be returned</td>
</tr>
<tr>
<td>kind</td>
<td>Path Manipulation</td>
</tr>
<tr>
<td>Input</td>
<td>①AppleScript alias, file, HFS path string, POSIX path</td>
</tr>
<tr>
<td>Output</td>
<td>NSURL</td>
</tr>
<tr>
<td>Notes</td>
<td></td>
</tr>
</tbody>
</table>

—Sample Code
-- Created 2014-11-29 by Takaaki Naganoya
-- 2014 Piyomaru Software
use AppleScript version "2.4"
use scripting additions
use framework "Foundation"
use framework "ASObjCExtras"

set a to choose file
set aURL to current application's SMSFord's URLFrom:a
--> file:///Applications/Automator.app/
--This is NSURL data
method `fileFromURL`:

**Introduction**
Pass an NSURL, and a file («class furl») will be returned. You will need to coerce the result with "as «class furl»"

**kind**
Path Manipulation

**Input**
1. NSURL

**Output**
AppleScript’s file

**Notes**

--- Sample Code
-- Created 2014-11-29 by Takaaki Naganoya
-- 2014 Piyomaru Software
use AppleScript version "2.4"
use scripting additions
use framework "Foundation"
use framework "ASObjCExtras"

set a to choose file
set aURL to current application's SMSFord's URLFrom:a
set aHFSPath to current application's SMSFord's fileFromURL:aURL
--> file "Macintosh HD:Applications:Automator.app:"
HFSPathFromURL:

- Sample Code
-- Created 2014-11-29 by Takaaki Naganoya
-- 2014 Piyomaru Software
use AppleScript version "2.4"
use scripting additions
use framework "Foundation"
use framework "ASObjCExtras"

set a to choose file
set aURL to current application's SMSFord's URLFrom:a
set aHFSPath to current application's SMSFord's HFSPathFromURL:aURL
set aFile to aHFSPath's ASify() as text
--> "Macintosh HD:Applications:Automator.app"
### HFSPathFromURL:colonForPackages:

<table>
<thead>
<tr>
<th>Method</th>
<th>HFSPathFromURL:colonForPackages:</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Introduction</strong></td>
<td>Pass an NSURL, and the HFS path is returned. Directories will have a trailing colon; packages will not. You can specify whether package paths have colons appended.</td>
</tr>
<tr>
<td><strong>Kind</strong></td>
<td>Path Manipulation</td>
</tr>
<tr>
<td><strong>Input</strong></td>
<td>①NSURL ②AppleScript boolean (true/false)</td>
</tr>
<tr>
<td><strong>Output</strong></td>
<td>NSString (HFS path string)</td>
</tr>
<tr>
<td><strong>Notes</strong></td>
<td></td>
</tr>
</tbody>
</table>

---

--- Sample Code
-- Created 2014-11-26 by Takaaki Naganoya
-- 2014 Piyomaru Software

```applescript
use AppleScript version "2.4"
use scripting additions
use framework "Foundation"
use framework "ASObjCExtras"

set a to choose file
set aURL to current application's SMSFord's URLFrom:a
set aHFSPath to current application's SMSFord's HFSPathFromURL:aURL:colonForPackages:true
set aFile to aHFSPath's ASify() as text
--> "Macintosh HD:Applications:Automator.app:"
```
ASObjCExtras.framework

String Manipulation methods

- List
- Trigonometry
- Class
- Path
- File Info
- Utility
- Misc
- String
```
--Sample Code
-- Created 2015-01-03 by Takaaki Naganoya
-- 2015 Piyomaru Software
use AppleScript version "2.4"
use scripting additions
use framework "Foundation"
use framework "ASObjCExtras"

set d1List to {"2015.01.03 00:00:00"}
set aNSDate to (current application's SMSFord's datesFromStrings:d1List | format:"yyyy.MM.dd HH:mm:ss")
set aDate to aNSDate's ASify() as date
-- date "Saturday, January 3, 2015 at 12:00:00 AM"

set d2List to {"2015.01.03 00:00:00", "2015.01.02 00:00:00", "2015.01.01 00:00:00"}
set aNSDate to (current application's SMSFord's datesFromStrings:d2List | format:"yyyy.MM.dd HH:mm:ss")
set aDate to (aNSDate's ASifyFor:"dates") as list
-- {date "Saturday, January 3, 2015 at 12:00:00 AM", date "Friday, January 2, 2015 at 12:00:00 AM", date "Thursday, January 1, 2015 at 12:00:00 AM"}
```
--- Sample Code
-- Created 2015-01-06 by Takaaki Naganoya
-- 2015 Piyomaru Software
use AppleScript version "2.4"
use scripting additions
use framework "Foundation"
use framework "ASObjCExtras"

set theString to "ながのや, たかあき" --Hiragana
set aRes to (current application's SMSFord's stringFrom:theString
  ICUTransform:"Hiragana-Latin" inverse:false) as text
--> "naganoya, takaaki"

set theString to "ながのや, たかあき" --Hiragana
set aRes to (current application's SMSFord's stringFrom:theString
  ICUTransform:"Hiragana-Katakana" inverse:false) as text
--> "ナガノヤ, タカアキ"--Katakana

set theString to "Takaaki, Naganoya"
set aRes to (current application's SMSFord's stringFrom:theString
  ICUTransform:"Hiragana-Latin" inverse:true) as text
--> "たかあき、 ながのや"--Hiragana

set theString to "Takaaki, Naganoya"
set aRes to (current application's SMSFord's stringFrom:theString
  ICUTransform:"Katakana-Latin" inverse:true) as text
--> "タカアキ、ナガノヤ"--Katakana

set theString to "Shane, Stanley"
set aRes to (current application's SMSFord's stringFrom:theString
  ICUTransform:"Katakana-Latin" inverse:true) as text
--> "シャネ、スタンレイ"--Katakana.....this seems odd. "シェーン, スタンリー" will be a
right spelling
method | stringFrom:① makingIt:②
---|---
Introduction | Useful string conversion routines
kind | String Manipulation
Input | ①AppleScript string ②AppleScript string
Output | NSString
Notes | 

Sample Code

-- Created 2015-01-04 by Takaaki Naganoya
-- 2015 Piyomaru Software
use AppleScript version "2.4"
use scripting additions
use framework "Foundation"
use framework "ASObjCExtras"

set res0 to current application's SMSFord's description() as text
--> "SMSFord 1.2.2"

--Calculates the MD5 hash of a string
set aStr to "0123456789"
set a to (current application's SMSFord's stringFrom:aStr makingIt:"MD5") as string
--> "781e5e245d69b566979b86e28d23f2c7"

--Converts straight quote marks into typographer's quote marks
set bStr to "'a'"
set b to (current application's SMSFord's stringFrom:bStr makingIt:"SmartQuoted") as string
--> "‘a’"

--Converts typographer's quote marks into straight quote marks
set cStr to ""'a""
set c to (current application's SMSFord's stringFrom:cStr makingIt:"UnsmartQuoted") as string
--> ""'a"

34 / 88
--Encodes the five reserved XML characters only
set d1Str to "&"<>"
set d1 to (current application's SMSFord's stringFrom:d1Str makingIt:"EncodedXML") as string
--> "&amp;&quot;&lt;&gt;&apos;"

--Decode the five reserved XML characters only
set d2Str to "&"<>"
set d2 to (current application's SMSFord's stringFrom:d2Str makingIt:"UnencodedForXML") as string
--> "&"<>"

--Encodes characters outside ASCII 32-126 in hexadecimal form (&#xHHHH;)
set eStr to "あいうえお"
set e to (current application's SMSFord's stringFrom:eStr makingIt:"EncodedHex") as string
--> "&#x3042;&#x3044;&#x3046;&#x3048;&#x304A;"

--Encodes characters outside ASCII 32-126 in decimal form (&#DD;), for use in HTML
set fStr to "あいうえお"
set f to (current application's SMSFord's stringFrom:fStr makingIt:"EncodedDecimal") as string
--> "&#12354;&#12356;&#12358;&#12360;&#12362;"

--Decodes characters that appear in decimal form (&#DD;) or hexadecimal form (&#xHHHH;), as used in XML and HTML
set gStr to "&#12354;&#12356;&#12358;&#12360;&#12362;"
set g to (current application's SMSFord's stringFrom:gStr makingIt:"DecodedDecimal") as string
--> "あいうえお"
--Deletes any paragraphs that are empty or contain only spaces and/or tabs
set hStr to "a
aaa
a
a
a"
set h to (current application's SMSFord's stringFrom:hStr makingIt:"EmptyLineFree") as string
--> (*
"a
aaa
a
a
a"
*)

--Converts runs of more than one space to a single space character, and trims spaces from the beginning and end of paragraphs.
set hStr to "aaaaa bbbb ccccc"
set h to (current application's SMSFord's stringFrom:hStr makingIt:"CleanSpaced") as string
--> "aaaaa bbbb ccccc"
**makingIt: option strings**

<table>
<thead>
<tr>
<th>option string</th>
<th>descriptions</th>
</tr>
</thead>
<tbody>
<tr>
<td>“MD5”</td>
<td>Calculates the MD5 hash of a string</td>
</tr>
<tr>
<td>“SmartQuoted”</td>
<td>Converts straight quote marks into typographer’s quote marks</td>
</tr>
<tr>
<td>“UnsmartQuoted”</td>
<td>Converts typographer’s quote marks into straight quote marks</td>
</tr>
<tr>
<td>“EncodedXML”</td>
<td>Encodes the five reserved XML characters only</td>
</tr>
<tr>
<td>“UnencodedForXML”</td>
<td>Decode the five reserved XML characters only</td>
</tr>
<tr>
<td>“EncodedHex”</td>
<td>Encodes characters outside ASCII 32-126 in hexadecimal form (&amp;#xHHHH;);</td>
</tr>
<tr>
<td>“EncodedDecimal”</td>
<td>Encodes characters outside ASCII 32-126 in decimal form (&amp;#DD;), for use in HTML</td>
</tr>
<tr>
<td>“DecodedDecimal”</td>
<td>Decodes characters that appear in decimal form (&amp;#DD;) or hexadecimal form (&amp;#xHHHH;), as used in XML and HTML</td>
</tr>
<tr>
<td>“EmptyLineFree”</td>
<td>Deletes any paragraphs that are empty or contain only spaces and/or tabs</td>
</tr>
<tr>
<td>“CleanSpaced”</td>
<td>Converts runs of more than one space to a single space character, and trims spaces from the beginning and end of paragraphs.</td>
</tr>
</tbody>
</table>
---Sample Code
-- Created 2014-11-26 by Takaaki Naganoya
-- 2014 Piyomaru Software
use AppleScript version "2.4"
use scripting additions
use framework "Foundation"
use framework "ASObjCExtras"

set a to "1  2  3
2  2  3"
set b to (current application's SMSFord's arrayFromTSV:a) as list
--> {{"1", "2", "3"}, {{"2", "2", "3"}}

---

<table>
<thead>
<tr>
<th>method</th>
<th>arrayFromTSV:①</th>
</tr>
</thead>
<tbody>
<tr>
<td>Introduction</td>
<td>Converts tab-separated values to list of lists</td>
</tr>
<tr>
<td>kind</td>
<td>String Manipulation</td>
</tr>
<tr>
<td>Input</td>
<td>AppleScript string</td>
</tr>
<tr>
<td>Output</td>
<td>NSArray (2D)</td>
</tr>
<tr>
<td>Notes</td>
<td></td>
</tr>
</tbody>
</table>
**Method**: arrayFromCSV:① commals:②

**Introduction**: Converts comma-separated values to a list of lists. The commals parameter defines (single) character used

<table>
<thead>
<tr>
<th>kind</th>
<th>String Manipulation</th>
</tr>
</thead>
</table>

**Input**: ①AppleScript string ②AppleScript string

**Output**: NSArray (2D)

**Notes**

---

---Sample Code
-- Created 2014-11-26 by Takaaki Naganoya
-- 2014 Piyomaru Software
use AppleScript version "2.4"
use scripting additions
use framework "Foundation"
use framework "ASObjCExtras"

```applescript
set a to "1, 2, 3
2, 2, 3"

set b to (current application's SMSFord's arrayFromCSV:a commals:",")) as list
--> {"1", "2", "3"}, {"2", "2", "3"}"
```
ASObjCExtras.framework

List Manipulation methods

- List
- Trigonometry
- Class
- Path
- String
- File Info
- Utility
- Misc
subarraysIn:① paddedWith:② error:③

Introduction
Assumes the array is a list of arrays, and the pad item will be used to pad out any subarrays that are shorter than the longest. This method return an array. You should use ASify() or similar to convert the results to lists if they include real numbers.

Kind
List Manipulation

Input
①AppleScript list ②AppleScript object (number, string, boolean etc.) ③AppleScript missing value or reference

Output
NSArray

Notes
This method return an array. You should use ASify() or similar to convert the results to lists if they include real numbers.

—Sample Code
-- Created 2014-11-26 by Takaaki Naganoya
-- 2014 Piyomaru Software
use AppleScript version "2.4"
use framework "Foundation"
use framework "ASObjCExtras"
use scripting additions

set aList to {{1, 2, 3, 4, 5}, {1, 2, 3}, {1, 2}}
set aPad to 0

set b to (current application's SMSFord's subarraysIn:aList paddedWith:aPad lerror:
  (missing value)) as list
--> {{1, 2, 3, 4, 5}, {1, 2, 3, 0, 0}, {1, 2, 0, 0, 0}}
Introduction

- 3x3 list

```
set aList to {{1, 2, 3}, {11, 12, 13}, {21, 22, 23}}
set b to (current application's SMSFord's colsToRowsIn:aList error:(missing value)) as list
--> {{1, 11, 21}, {2, 12, 22}, {3, 13, 23}}--3x3
```

- 3x5 list

```
set aList to {{1, 2, 3}, {11, 12, 13}, {21, 22, 23}, {31, 32, 33}, {41, 42, 43}}
set b to (current application's SMSFord's colsToRowsIn:aList error:(missing value)) as list
--> {{1, 11, 21, 31, 41}, {2, 12, 22, 32, 42}, {3, 13, 23, 33, 43}}--5x3
```
**arrayByFlattening:**

<table>
<thead>
<tr>
<th>method</th>
<th>Flattens a list of lists to a single list: <code>[[1, 2], [3, 4]]</code> -&gt; <code>1, 2, 3, 4</code>. Flattens one level only.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Introduction</td>
<td>List Manipulation</td>
</tr>
<tr>
<td>kind</td>
<td></td>
</tr>
<tr>
<td>Input</td>
<td>AppleScript list (2D)</td>
</tr>
<tr>
<td>Output</td>
<td>NSArray (1D)</td>
</tr>
<tr>
<td>Notes</td>
<td>This method return an array. You should use <code>ASify()</code> or similar to convert the results to lists if they include real numbers.</td>
</tr>
</tbody>
</table>

--Sample Code
-- Created 2014-11-26 by Takaaki Naganoya
-- 2014 Piyomaru Software

```applescript
use AppleScript version "2.4"
use framework "Foundation"
use framework "ASObjCExtras"
use scripting additions

set aList to {{1, 2, 3}, {11, 12, 13}, {21, 22, 23}}

set b to (current application's SMSFord's arrayByFlattening:aList) as list
---> {1, 2, 3, 11, 12, 13, 21, 22, 23}
```
<table>
<thead>
<tr>
<th>method</th>
<th>arrayByFullyFlattening:①</th>
</tr>
</thead>
<tbody>
<tr>
<td>Introduction</td>
<td>Flattens a list to a single list: (1, 2, (3, {4, 5})) --&gt; (1, 2, 3, 4, 5). Flattens all levels</td>
</tr>
<tr>
<td>kind</td>
<td>List Manipulation</td>
</tr>
<tr>
<td>Input</td>
<td>AppleScript list (nD)</td>
</tr>
<tr>
<td>Output</td>
<td>NSArray (1D)</td>
</tr>
<tr>
<td>Notes</td>
<td>This method return an array. You should use ASify() or similar to convert the results to lists if they include real numbers.</td>
</tr>
</tbody>
</table>

--- Sample Code
-- Created 2014-11-26 by Takaaki Naganoya
-- 2014 Piyomaru Software
use AppleScript version "2.4"
use framework "Foundation"
use framework "ASObjCExtras"
use scripting additions

```plaintext
set aList to {{{1, 2, 3}, {11, 12, 13}, {21, 22, 23}, {24, 25, 26}}}  

set b to (current application's SMSFord's arrayByFullyFlattening:aList) as list
--> \{1, 2, 3, 11, 12, 13, 21, 22, 23, 24, 25, 26\}
```
<table>
<thead>
<tr>
<th>method</th>
<th>subarraysFrom:① groupedBy:② error:③</th>
</tr>
</thead>
<tbody>
<tr>
<td>Introduction</td>
<td>Breaks list into subarrays of aNumber items</td>
</tr>
<tr>
<td>kind</td>
<td>List Manipulation</td>
</tr>
<tr>
<td>Input</td>
<td>①AppleScript list (1D) ②AppleScript number ③AppleScript missing value or reference</td>
</tr>
<tr>
<td>Output</td>
<td>NSArray (2D)</td>
</tr>
<tr>
<td>Notes</td>
<td>This method return an array. You should use ASify() or similar to convert the results to lists if they include real numbers.</td>
</tr>
</tbody>
</table>

--Sample Code
-- Created 2014-11-26 by Takaaki Naganoya
-- 2014 Piyomaru Software
use AppleScript version "2.4"
use framework "Foundation"
use framework "ASObjCExtras"
use scripting additions

```
set aList to {1, 2, 3, 11, 12, 13, 21, 22, 23}

set aGroupNum to 3
set b to (current application's SMSFord's subarraysFrom:aList
  groupedBy:aGroupNum error:((missing value)) as list
---> {{1, 2, 3}, {11, 12, 13}, {21, 22, 23}}

set aGroupNum to 5
set c to (current application's SMSFord's subarraysFrom:aList
  groupedBy:aGroupNum error:((missing value)) as list
---> {{1, 2, 3, 11, 12}, {13, 21, 22, 23}}
```


--Sample Code
-- Created 2014-11-26 by Takaaki Naganoya
-- 2014 Piyomaru Software
use AppleScript version "2.4"
use framework "Foundation"
use framework "ASObjCExtras"
use scripting additions

--Case: Number of every item is same
set aList to {{{1, 2, 3}, {11, 12, 13}, {21, 22, 23}}
set insArray to {100, 101, 102}
set anIndex to 0
set b to (current application's SMSFord's subarraysIn:aList withItems:insArray
insertedAtIndex:anIndex lerrorl:(missing value)) as list
--> {{100, 1, 2, 3}, {101, 11, 12, 13}, {102, 21, 22, 23}}

--Case: Number of every item is not same
set aList to {{{1, 2, 3}, {11, 12, 13}, {21, 22, 23}}
set insArray to {100, 101}
set anIndex to 0
set b to (current application's SMSFord's subarraysIn:aList withItems:insArray
insertedAtIndex:anIndex lerrorl:(missing value)) as list
--> {missing value}
arrayByInsertingItems:① inArray:② atIndex:③ error:④

Introduction
Kind: List Manipulation
Input:
① AppleScript list (2D) ② AppleScript list (2D) ③ AppleScript number ④ missing value
Output: NSArray
Notes: This method return an array. You should use ASify() or similar to convert the results to lists if they include real numbers.

--Sample Code
-- Created 2014-12-30 by Takaaki Naganoya
-- 2014 Piyomaru Software
use AppleScript version "2.4"
use framework "Foundation"
use framework "ASObjCExtras"
use scripting additions

set aList to {{1, 2, 3}, {11, 12, 13}, {21, 22, 23}}
set insArray to {{100, 101, 102}}
set anIndex to 1
set b to (current application's SMSFord's arrayByInsertingItems:insArray
inArray:aList atIndex:anIndex lenrorl:(missing value)) as list
--> {{1, 2, 3}, {100, 101, 102}, {11, 12, 13}, {21, 22, 23}}
arrayByMovingItemAt:① toIndex:②
inArray:③ error:④

Introduction

Does what it says. Indexes are zero-based.

kind

List Manipulation

Input

① AppleScript number ② AppleScript number ③ AppleScript list (1D/2D) ④ AppleScript missing value or reference

Output

NSArray

Notes

This method return an array. You should use ASify() or similar to convert the results to lists if they include real numbers.

--Sample Code
-- Created 2014-12-30 by Takaaki Naganoya
-- 2014 Piyomaru Software
use AppleScript version "2.4"
use framework "Foundation"
use framework "ASObjCEXtras"
use scripting additions

set aList to {1, 2, 3, 11, 12, 13, 21, 22, 23}
set b to (current application's SMSFord's arrayByMovingItemAt:1 toIndex:3
    inArray:aList error:(missing value)) as list
--> {1, 3, 11, 2, 12, 13, 21, 22, 23}

set aList to {{1, 2, 3}, {11, 12, 13}, {21, 22, 23}}
set b to (current application's SMSFord's arrayByMovingItemAt:1 toIndex:2
    inArray:aList error:(missing value)) as list
--> {{1, 2, 3}, {21, 22, 23}, {11, 12, 13}}
**method**

arrayByDeletingBlanksIn:①

**Introduction**

Delete any blank items. Blanks are missing value, empty strings, empty lists, consisiting only of the above.

**kind**

List Manipulation

**Input**

①AppleScript list

**Output**

NSArray

**Notes**

This method return an array. You should use ASify() or similar to convert the results to lists if they include real numbers.

—Sample Code

-- Created 2014-12-30 by Takaaki Naganoya
-- 2014 Piyomaru Software

use AppleScript version "2.4"

use framework "Foundation"

use framework "ASObjCExtras"

use scripting additions

set aList to {1, 2, 3, 11, 12, 13, 21, 22, 23, "", {}}

set b to (current application's SMSFord's arrayByDeletingBlanksIn:aList) as list

-->{1, 2, 3, 11, 12, 13, 21, 22, 23}
<table>
<thead>
<tr>
<th>method</th>
<th>arrayByTrimmingTrailingBlanksFrom:①</th>
</tr>
</thead>
<tbody>
<tr>
<td>Introduction</td>
<td>Trim any blank items from the trailing end of the list. Blanks are missing value, empty strings, empty lists, and lists consising only of the above.</td>
</tr>
<tr>
<td>kind</td>
<td>List Manipulation</td>
</tr>
<tr>
<td>Input</td>
<td>①AppleScript list</td>
</tr>
<tr>
<td>Output</td>
<td>NSArray</td>
</tr>
<tr>
<td>Notes</td>
<td>This method return an array. You should use ASify() or similar to convert the results to lists if they include real numbers.</td>
</tr>
</tbody>
</table>

```
-- Created 2014-12-30 by Takaaki Naganoya
-- 2014 Piyomaru Software
use AppleScript version "2.4"
use scripting additions
use framework "Foundation"
use framework "ASObjCExtras"

set aList to {1, 2, 3, 11, 12, 13, 21, 22, 23, "", {}}

set b to (current application's SMSFord's
           arrayByTrimmingTrailingBlanksFrom:aList) as list
-->{1, 2, 3, 11, 12, 13, 21, 22, 23}
```
method **arrayByTrimmingBlanksFrom:**

Trim any blank items from both ends of the list. Blanks are missing value, empty strings, empty lists, and lists consisting only of the above.

<table>
<thead>
<tr>
<th>Introduction kind</th>
<th>List Manipulation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Input</td>
<td>AppleScript list</td>
</tr>
<tr>
<td>Output</td>
<td>NSArray</td>
</tr>
<tr>
<td>Notes</td>
<td>This method return an array. You should use ASify() or similar to convert the results to lists if they include real numbers.</td>
</tr>
</tbody>
</table>

---

Sample Code

---

-- Created 2014-12-30 by Takaaki Naganoya
-- 2014 Piyomaru Software

use AppleScript version "2.4"
use scripting additions
use framework "Foundation"
use framework "ASObjCExtras"

```script
set aList to {missing value, 1, 2, 3, 11, 12, 13, 21, 22, 23, "", {}}

set b to (current application's SMSFord's arrayByTrimmingBlanksFrom:aList) as list
--> {1, 2, 3, 11, 12, 13, 21, 22, 23}
```
<table>
<thead>
<tr>
<th>method</th>
<th>arrayByReplacingNullsIn:① withItem:②</th>
</tr>
</thead>
<tbody>
<tr>
<td>Introduction</td>
<td>Instances of 'missing value' will be replaced with whatever you pass to withItem:</td>
</tr>
<tr>
<td>kind</td>
<td>List Manipulation</td>
</tr>
<tr>
<td>Input</td>
<td>AppleScript list, AppleScript Object</td>
</tr>
<tr>
<td>Output</td>
<td>NSArray</td>
</tr>
<tr>
<td>Notes</td>
<td>This method return an array. You should use ASify() or similar to convert the results to lists if they include real numbers.</td>
</tr>
</tbody>
</table>

--- Sample Code ---

-- Created 2014-12-30 by Takaaki Naganoya
-- 2014 Piyomaru Software

use AppleScript version "2.4"
use framework "Foundation"
use framework "ASObjCExtras"
use scripting additions

set aList to {1, 2, 3, 4, missing value}
set replItem to 0

set b to (current application's SMSFord's arrayByReplacingNullsIn:aList withItem:replItem) as list
---> {1, 2, 3, 4, 0}
<table>
<thead>
<tr>
<th></th>
<th>arrayWithPattern:① startNumber:② endNumber:③ minDigits:④</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Introduction</strong></td>
<td>Pattern should be a string where every instance of %@ will be replaced by a number. So a pattern of &quot;label%@&quot;, a startNumber of 3, an endNumber of 1 and minDigits 3 will return (&quot;label003&quot;, &quot;label002&quot;, &quot;label001&quot;)</td>
</tr>
<tr>
<td><strong>kind</strong></td>
<td>List Manipulation</td>
</tr>
<tr>
<td><strong>Input</strong></td>
<td>① AppleScript string ② AppleScript number ③ AppleScript number ④ AppleScript number</td>
</tr>
<tr>
<td><strong>Output</strong></td>
<td>NSArray</td>
</tr>
<tr>
<td><strong>Notes</strong></td>
<td>This method return an array. You should use ASify() or similar to convert the results to lists if they include real numbers.</td>
</tr>
</tbody>
</table>

---Sample Code
-- Created 2014-11-29 by Takaaki Naganoya
-- 2014 Piyomaru Software
use AppleScript version "2.4"
use scripting additions
use framework "Foundation"
use framework "ASObjCExtras"

```
set aArray to current application's SMSFord's arrayWithPattern:"Piyomaru%@" startNumber:1 endNumber:100 minDigits:3
set aList to aArray's ASify() as list
```
### sumMaxMinOf: ① error: ②

**Introduction**
Returns an array of {sum, max, min} of the list.

**kind**
List Manipulation

**Input**
① AppleScript list ② AppleScript missing value or reference

**Output**
NSArray

**Notes**
This method return an array. You should use ASify() or similar to convert the results to lists if they include real numbers.

---

**Sample Code**

```
-- Sample Code
-- Created 2014-12-30 by Takaaki Naganoya
-- 2014 Piyomaru Software
use AppleScript version "2.4"
use framework "Foundation"
use framework "ASObjCExtras"
use scripting additions

set aList to {1, 2, 3, 4, 5, 6, 7, 8, 9, 10}

set {sumNum, maxNum, minimumNum} to (current application's SMSFord's sumMaxMinOf:(aList) lerrorl:(missing value)) as list

--> {55.0, 10.0, 1.0}
```
---Sample Code
-- Created 2014-12-30 by Takaaki Naganoya
-- 2014 Piyomaru Software
use AppleScript version "2.4"
use scripting additions
use framework "Foundation"
use framework "ASObjCExtras"

set aList to {{1, 2, 3}, {11, 12, 13}, {21, 22, 23}, {21, 23, 23}}

set sortIndexes to {0, 1} --Key Item id: begin from 0
set sortOrders to {false, true}
set sortTypes to {"compare:", "compare:"}
set rList to (current application's SMSFord's subarraysIn:(aList) sortedByIndexes: (sortIndexes) ascending: (sortOrders) sortTypes: (sortTypes) lerror!: (missing value)) as list

--> {{21, 22, 23}, {21, 23, 23}, {11, 12, 13}, {1, 2, 3}}
subarraysIn:① sortedByIndexes:② ascending:③ sortTypes:④ sortKeys:⑤ error:⑥

As "subarraysIn:① sortedByIndexes:② ascending:③ sortTypes:④ error:⑤", but you can also pass a list of key strings to be used for the sort. Pass an empty list to use "self" throughout, which is what the above method uses. For example, if you pass ["length"], the sorting will be done on the basis of the length. The values being sorted must support the key or an error will be thrown.

This method return an array. You should use ASIfy() or similar to convert the results to lists if they include real numbers.

--- Sample Code
-- Created 2014-12-30 by Takaaki Naganoya
-- 2014 Piyomaru Software
use AppleScript version "2.4"
use scripting additions
use framework "Foundation"
use framework "ASObjCExtras"

set aList to {{1, 2, "Piyomaru"}, {2, 1, "Hiyoko"}, {1, 1, "Glass"}, {4, 1, "Battery"}}

set sortIndexes to {2, 1, 0} --Key Item id: begin from 0
set sortOrders to {false, true, true}
set sortTypes to {"compare:", "compare:", "compare:"}
set rList to (current application's SMSFord's subarraysIn:(aList) sortedByIndexes: (sortIndexes) ascending:(sortOrders) sortTypes:(sortTypes) sortKeys:{"length", "self", "self"} error:((missing value)) as list
--> {{1, 2, "Piyomaru"}, {4, 1, "Battery"}, {2, 1, "Hiyoko"}, {1, 1, "Glass"}}
<table>
<thead>
<tr>
<th>method</th>
<th>arrayByMergingTextAtIndexes:① inArray:② inserting:③ error:④</th>
</tr>
</thead>
<tbody>
<tr>
<td>Introduction</td>
<td>Concatenate two text items into one. Provide a list of the (zero-based) indexes of the items to merge; they will be joined using the separator in the order provided, and appear at the first of the indexes. Empty strings will be ignored.</td>
</tr>
<tr>
<td>kind</td>
<td>List Manipulation</td>
</tr>
<tr>
<td>Input</td>
<td>①AppleScript list ②AppleScript list ③NSString or AppleScript string ④AppleScript missing value or reference</td>
</tr>
<tr>
<td>Output</td>
<td>NSArray</td>
</tr>
<tr>
<td>Notes</td>
<td>This method return an array. You should use ASify() or similar to convert the results to lists if they include real numbers.</td>
</tr>
</tbody>
</table>

---

Sample Code
-- Created 2014-12-30 by Takaaki Naganoya
-- 2014 Piyomaru Software
use AppleScript version "2.4"
use scripting additions
use framework "Foundation"
use framework "ASObjCExtras"

set aList to \"Piyomaru\", \"Hiyoko\", \"Glass\", \"Battery\"
set mergeIndexes to \{2, 1\} --Key Item id: begin from 0
set rList to (current application's SMSFord's arrayByMergingTextAtIndexes:
    (mergeIndexes) inArray:(aList) inserting:("____") error:(missing value)) as list
--> \"Piyomaru\", \"Glass____Hiyoko\", \"Battery\"
<table>
<thead>
<tr>
<th><strong>method</strong></th>
<th>**indexesOfItem:**① **inArray:**② **inverting:**③</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Introduction</strong></td>
<td>Returns list of indexes where item is found; inverting true returns indexes where the item is not found. Indexes are zero-based.</td>
</tr>
<tr>
<td><strong>kind</strong></td>
<td>List Manipulation</td>
</tr>
<tr>
<td><strong>Input</strong></td>
<td>① NSDictionary or AppleScript string ② AppleScript list ③ AppleScript boolean</td>
</tr>
<tr>
<td><strong>Output</strong></td>
<td>NSArray</td>
</tr>
<tr>
<td><strong>Notes</strong></td>
<td>This method return an array. You should use ASify() or similar to convert the results to lists if they include real numbers.</td>
</tr>
</tbody>
</table>

---

Sample Code

-- Created 2014-12-06 by Takaaki Naganoya
-- 2014 Piyomaru Software

```applescript
use AppleScript version "2.4"
use framework "Foundation"
use framework "ASObjCExtras"
use scripting additions

set aList to {"1", "2", "3", "4", "5", "6", "7", "8", "9", "0", "1", "2", "3", "4", "5", "6", "7", "8", "9", "0"}

set aRes to (current application's SMSFord's indexesOfItem:"0" inArray:aList inverting:false) as list

--> {9, 19, 29}--0 based
```
<table>
<thead>
<tr>
<th>method</th>
<th>indexesOfItems:① inArray:② inverting:③</th>
</tr>
</thead>
<tbody>
<tr>
<td>Introduction</td>
<td>Returns list of indexes where any of the items in the list are found; inverting true returns indexes where the items are not found. Indexes are zero-based.</td>
</tr>
<tr>
<td>kind</td>
<td>List Manipulation</td>
</tr>
<tr>
<td>Input</td>
<td>①NSArray or AppleScript list ②NSArray or AppleScript list ③AppleScript boolean</td>
</tr>
<tr>
<td>Output</td>
<td>NSArray</td>
</tr>
<tr>
<td>Notes</td>
<td>This method return an array. You should use ASify() or similar to convert the results to lists if they include real numbers.</td>
</tr>
</tbody>
</table>

---

---Sample Code
-- Created 2014-12-06 by Takaaki Naganoya
-- 2014 Piyomaru Software
use AppleScript version "2.4"
use framework "Foundation"
use framework "ASObjCExtras"
use scripting additions

```
set aList to {"1", "2", "3", "4", "5", "6", "7", "8", "9", "0", "1", "2", "3", "4", "5", "6", "7", "8", "9", "0"}

set aRes to (current application's SMSFord's indexesOfItems:{"0", "8"}) inArray:aList inverting:false) as list
---> {7, 9, 17, 19, 27, 29}--0 based
```
### subarraysIn:①

#### asDictionariesUsingLabels:② error:③

<table>
<thead>
<tr>
<th>method</th>
<th>Subarrays in a list of lists, using labels to create dictionaries.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Introduction</strong></td>
<td>Assumes the array is a list of lists, and that each list has the same number of items as the list of labels. The result will be an array of records/dictionaries that use the supplied labels in order.</td>
</tr>
<tr>
<td><strong>Kind</strong></td>
<td>List Manipulation</td>
</tr>
<tr>
<td><strong>Input</strong></td>
<td>① AppleScript list ② AppleScript list ③ AppleScript missing value or reference</td>
</tr>
<tr>
<td><strong>Output</strong></td>
<td>NSArray</td>
</tr>
<tr>
<td><strong>Notes</strong></td>
<td>This method returns an array. You should use ASify() or similar to convert the results to lists if they include real numbers.</td>
</tr>
</tbody>
</table>

---

```applescript
use AppleScript version "2.4"
use scripting additions
use framework "Foundation"
use framework "ASObjCExtras"

set aList to {"aLabel", "bLabel"}
set bList to {{1, 2}, {2, 3}, {3, 4}}

set aArray to current application's SMSFord's subarraysIn:bList
    asDictionariesUsingLabels:aList lerror:(missing value)
set bList to aArray's ASify() as list
--> {{aLabel:1, bLabel:2}, {aLabel:2, bLabel:3}, {aLabel:3, bLabel:4}}
```
### Introduction

**Method:** subarraysFrom: ① usingKeys:② outKeys:③ error:④

**Pass a list of records/dictionaries and a list of labels, and a list of lists will be returned, with the order of the values in each sublist matching the order of the labels. If an empty list is passed for usingKeys, the keys of the first item will be used, sorted in case-insensitive alphabetical order, and these keys will be listed as strings in outKeys.**

**Kind:** List Manipulation

**Input:**

- ① AppleScript list of records
- ② AppleScript list
- ③ AppleScript list or reference
- ④ AppleScript missing value or reference

**Output:**

- NSArray (2D/1D)

**Notes:**

This method return an array. You should use ASify() or similar to convert the results to lists if they include real numbers.

---

---Sample Code
-- Created 2014-12-06 by Takaaki Naganoya
-- 2014 Piyomaru Software
use AppleScript version "2.4"
use scripting additions
use framework "Foundation"
use framework "ASObjCExtras"

```applescript
set aList to {{age:10, aName:"ccc", weight:70}, {age:20, aName:"bbb", weight:80}, {age:3, aName:"aaaa", weight:10}}

set aArray to current application's SMSFord's subarraysFrom:aList usingKeys: ④ error: ④
   "weight", "aName"} outKeys:"aName"} lerrorl:(missing value)
set bList to aArray's ASify() as list
-- {{70, "ccc"}, {80, "bbb"}, {10, "aaaa"}}

--another sample
set aList to {{aName:"ccc", weight:70}, {age:20, aName:"bbb", weight:80}, {age:3, aName:"aaaa", weight:10}}
set {aArray, theKeys} to current application's SMSFord's subarraysFrom:aList usingKeys:{ } outKeys:(reference) lerrorl:(missing value)
set bList to aArray's ASify() as list
set cList to theKeys's ASify() as list
-- {{"aName", "weight"}}
```

---
```plaintext
**Introduction**

ASObjC can't normally extract the values from an IndexSet; this method makes it possible

**kind**
List Manipulation

**Input**
①NSMutableIndexSet, NSIndexSet

**Output**
NSArray

**Notes**
This method return an array. You should use ASify() or similar to convert the results to lists if they include real numbers.

```AppleScript
-- Sample Code
-- Created 2014-12-30 by Takaaki Naganoya
-- 2014 Piyomaru Software
use AppleScript version "2.4"
use scripting additions
use framework "Foundation"
use framework "ASObjCExtras"

set aindexSet to current application's NSMutableIndexSet's alloc()'s init()

repeat 100 times
    set aRandom to random number from 1 to 100
    aindexSet's addIndex:aRandom
end repeat

set aList to (current application's SMSFord's arrayWithIndexSet:aindexSet) as list
---> {2, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 23, 24, 25, 28, 29, 30, 31, 32, 33, 34, 37, 38, 39, 40, 43, 52, 53, 56, 57, 59, 61, 63, 67, 70, 74, 76, 78, 79, 80, 81, 83, 84, 85, 87, 88, 89, 90, 91, 92, 94, 96

-- Another Sample
set anIndexSet to current application's NSMutableIndexSet's indexSet()
anIndexSet's addIndexesInRange:(current application's NSMakeRange(5, 20))
set aList to (current application's SMSFord's arrayWithIndexSet:anIndexSet) as list
---> {5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24}
set aList to {0, 1, 2, 3, 4, 5, 6, 7, 8, 9}

set aArray to current application's SMSFord's arrayByAddingInteger:1 inArray:aList
set bList to aArray's ASify() as list
--> {1, 2, 3, 4, 5, 6, 7, 8, 9, 10}

set bArray to current application's SMSFord's arrayByAddingInteger:-1 inArray:aList
set cList to bArray's ASify() as list
--> {-1, 0, 1, 2, 3, 4, 5, 6, 7, 8}
ASObjCExtras.framework

Trigonometry methods
---Sample Code
-- Created 2014-11-26 by Takaaki Naganoya
-- 2014 Piyomaru Software
use AppleScript version "2.4"
use framework "Foundation"
use framework "ASObjCExtras"
use scripting additions

set aList to {1, 2, 3, 4, 5, 6, 7, 8, 9, 10}

--Real Value
set aVal to current application's SMSFord's tanValueOf:30
set bVal to aVal's ASify() as real
--> -6.405331196646

--List of Real Value
set aArray to (current application's SMSFord's tanValueOf:aList)
set bList to aArray's ASify() as list
--> {1.557407724655, -2.185039863262, -0.142546543074, 1.15782128235, -3.380515006247, -0.291006191385, 0.871447982724, -6.79971145522, -0.452315659442, 0.648360827459}
<table>
<thead>
<tr>
<th><strong>method</strong></th>
<th><strong>sinValueOf:①</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Introduction</strong></td>
<td>Pass a single number or a list of numbers. Results must be coerced using &quot;as real&quot; or &quot;as list&quot;. Angles are in radians, and errors are returned as missing value.</td>
</tr>
<tr>
<td><strong>kind</strong></td>
<td>Trigonometry</td>
</tr>
<tr>
<td><strong>Input</strong></td>
<td>①AppleScript number (real) or AppleScript list</td>
</tr>
<tr>
<td><strong>Output</strong></td>
<td>NSNumber or NSArray</td>
</tr>
<tr>
<td><strong>Notes</strong></td>
<td></td>
</tr>
</tbody>
</table>

--Sample Code
-- Created 2014-11-30 by Takaaki Naganoya
-- 2014 Piyomaru Software

use AppleScript version "2.4"
use framework "Foundation"
use framework "ASObjCExtras"
use scripting additions

```
set aList to {1, 2, 3, 4, 5, 6, 7, 8, 9, 10}

--Real Value
set aVal to current application's SMSFord's sinValueOf:30
set bVal to aVal's ASify() as real
--> -0.988031624093

--List of Real Value
set aArray to (current application's SMSFord's sinValueOf:aList)
set bList to aArray's ASify() as list
--> {0.841470984808, 0.909297426826, 0.14112000806, -0.756802495308, -0.958924274663, -0.279415498199, 0.656986598719, 0.989358246623, 0.412118485242, -0.544021110889}
```
### Method
**cosValueOf:**

Pass a single number or a list of numbers. Results must be coerced using "as real" or "as list". Angles are in radians, and errors are returned as missing value.

**Kind:** Trigonometry

**Input:** AppleScript number (real) or AppleScript list

**Output:** NSNumber or NSArray

**Notes:**

```
-- Sample Code
-- Created 2014-11-30 by Takaaki Naganoya
-- 2014 Piyomaru Software
use AppleScript version "2.4"
use framework "Foundation"
use framework "ASObjCExtras"
use scripting additions

-- Real Value
set aVal to current application's SMSFord's cosValueOf: -0.5
set bVal to aVal's ASify() as real
--> 0.87758256189

-- List of Real Value
set aList to {0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9}
set aArray to (current application's SMSFord's cosValueOf: aList)
set bList to aArray's ASify() as list
--> {0.995004165278, 0.980066577841, 0.955336489126, 0.921060994003, 0.87758256189, 0.82533561491, 0.764842187284, 0.696706709347, 0.621609968271}
```
method atanValueOf:①

Introduction
Pass a single number or a list of numbers. Results must be coerced using "as real" or "as list". Angles are in radians, and errors are returned as missing value.

kind Trigonometry

Input ①AppleScript number (real) or AppleScript list

Output NSNumber or NSArray

Notes

--Sample Code
-- Created 2014-11-30 by Takaaki Naganoya
-- 2014 Piyomaru Software
use AppleScript version "2.4"
use framework "Foundation"
use framework "ASObjCExtras"
use scripting additions

--Real Value
set aVal to current application's SMSFord's atanValueOf:-0.5
set bVal to aVal's ASify() as real
--> -0.463647609001

--List of Real Value
set aList to {0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9}
set aArray to (current application's SMSFord's atanValueOf:aList)
set bList to aArray's ASify() as list
--> {0.099668652491, 0.19739555985, 0.291456794478, 0.380506377112,
0.463647609001, 0.540419500271, 0.610725964389,
0.674740942224, 0.732815101787}
---Sample Code
-- Created 2014-11-30 by Takaaki Naganoya
-- 2014 Piyomaru Software
use AppleScript version "2.4"
use framework "Foundation"
use framework "ASObjCExtras"
use scripting additions

---Real Value
set aVal to current application's SMSFord's asinValueOf:-0.5
set bVal to aVal's ASify() as real
--> -0.523598775598

---List of Real Value
set aList to {0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9}
set aArray to (current application's SMSFord's asinValueOf:aList)
set bList to aArray's ASify() as list
--> {0.100167421162, 0.20135792079, 0.304692654015, 0.411516846067, 0.523598775598, 0.643501108793, 0.775397496611, 0.927295218002, 1.119769514999}
--- Sample Code
-- Created 2014-11-30 by Takaaki Naganoya
-- 2014 Piyomaru Software
use AppleScript version "2.4"
use framework "Foundation"
use framework "ASObjCExtras"
use scripting additions

-- Real Value
set aVal to current application's SMSFord's acosValueOf:-0.5
set bVal to aVal's ASify() as real
--> 2.094395102393

-- List of Real Value
set aList to \{0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9\}
set aArray to (current application's SMSFord's acosValueOf:aList)
set bList to aArray's ASify() as list
--> \{1.470628905633, 1.369438406005, 1.26610367278, 1.159279480727, 1.047197551197, 0.927295218002, 0.795398830184, 0.643501108793, 0.451026811796\}
### Sample Code

-- Created 2014-11-30 by Takaaki Naganoya
-- 2014 Piyomaru Software
use AppleScript version "2.4"
use framework "Foundation"
use framework "ASObjCExtras"
use scripting additions

---

**Introduction**

Pass a single number or a list of numbers. Results must be coerced using "as real" or "as list". Angles are in radians, and errors are returned as missing value.

**kind**
Trigonometry

**Input**
AppleScript number (real) or AppleScript list

**Output**
NSNumber or NSArray

---

**--Real Value**

```applescript
set aVal to current application's SMSFord's tanhValueOf:-0.5
set bVal to aVal's ASify() as real
--> -0.46211715726
```

**--List of Real Value**

```applescript
set aList to {0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9}
set aArray to (current application's SMSFord's tanhValueOf:aList)
set bList to aArray's ASify() as list
--> {0.099667994625, 0.197375320225, 0.291312612452, 0.379948962255, 0.46211715726, 0.537049566998, 0.604367777117, 0.664036770268, 0.716297870199}
```
method sinhValueOf:

Introduction
Pass a single number or a list of numbers. Results must be coerced using "as real" or "as list". Angles are in radians, and errors are returned as missing value.

kind Trigonometry

Input AppleScript number (real) or AppleScript list

Output NSNumber or NSArray

Notes

--Sample Code
-- Created 2014-11-30 by Takaaki Naganoya
-- 2014 Piyomaru Software
use AppleScript version "2.4"
use framework "Foundation"
use framework "ASObjCExtras"
use scripting additions

--Real Value
set aVal to current application's SMSFord's sinhValueOf:-0.5
set bVal to aVal's ASify() as real
--> -0.521095305494

--List of Real Value
set aList to {0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9}
set aArray to (current application's SMSFord's sinhValueOf:aList)
set bList to aArray's ASify() as list
--> {0.10016675002, 0.201336002541, 0.304520293447, 0.410752325803, 0.521095305494, 0.636653582148, 0.75858370184, 0.888105982188, 1.026516725708}
```
--Sample Code
-- Created 2014-11-30 by Takaaki Naganoya
-- 2014 Piyomaru Software
use AppleScript version "2.4"
use framework "Foundation"
use framework "ASObjCExtras"
use scripting additions

--Real Value
set aVal to current application's SMSFord's coshValueOf:-0.5
set bVal to aVal's ASify() as real
--> 1.127625965206

--List of Real Value
set aList to {0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9}
set aArray to (current application's SMSFord's coshValueOf:aList)
set bList to aArray's ASify() as list
--> {1.005004168056, 1.020066755619, 1.045338514129, 1.081072371838, 1.127625965206, 1.185465218242, 1.255169005631, 1.337434946305, 1.433086385449}
```
---Sample Code
-- Created 2014-11-30 by Takaaki Naganoya
-- 2014 Piyomaru Software
use AppleScript version "2.4"
use framework "Foundation"
use framework "ASObjCExtras"
use scripting additions

--Real Value
set aVal to current application's SMSFord's atanhValueOf:-0.5
set bVal to aVal's ASify() as real
--> -0.549306144334

--List of Real Value
set aList to {0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9}
set aArray to (current application's SMSFord's atanhValueOf:aList)
set bList to aArray's ASify() as list
-->{0.100335347731, 0.202732554054, 0.309519604203, 0.423648930194, 0.549306144334, 0.69314718056, 0.867300527694, 1.098612288668, 1.472219489583}
**asinhValueOf:**

Pass a single number or a list of numbers. Results must be coerced using "as real" or "as list". Angles are in radians, and errors are returned as missing value.

---

**Sample Code**

-- Created 2014-11-30 by Takaaki Naganoya
-- 2014 Piyomaru Software

```applescript
use AppleScript version "2.4"
use framework "Foundation"
use framework "ASObjCExtras"
use scripting additions

--Real Value
set aVal to current application's SMSFord's asinhValueOf:-0.5
set bVal to aVal's ASify() as real
--> -0.48121182506

--List of Real Value
set aList to {0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9}
set aArray to (current application's SMSFord's asinhValueOf:aList)
set bList to aArray's ASify() as list
-->{0.099834078899, 0.198690110349, 0.295673047563, 0.390035319771, 0.48121182506, 0.568824898732, 0.652666566082, 0.732668256045, 0.808866935653}
```
<table>
<thead>
<tr>
<th>Method</th>
<th>acoshValueOf: ①</th>
</tr>
</thead>
<tbody>
<tr>
<td>Introduction</td>
<td>Pass a single number or a list of numbers. Results must be coerced using &quot;as real&quot; or &quot;as list&quot;. Angles are in radians, and errors are returned as missing value.</td>
</tr>
<tr>
<td>Kind</td>
<td>Trigonometry</td>
</tr>
<tr>
<td>Input</td>
<td>①AppleScript number (real) or AppleScript list</td>
</tr>
<tr>
<td>Output</td>
<td>NSNumber or NSArray</td>
</tr>
<tr>
<td>Notes</td>
<td></td>
</tr>
</tbody>
</table>

--Sample Code
-- Created 2014-11-30 by Takaaki Naganoya
-- 2014 Piyomaru Software
use AppleScript version "2.4"
use framework "Foundation"
use framework "ASObjCExtras"
use scripting additions

--Real Value
set aVal to current application's SMSFord's acoshValueOf:20
set bVal to aVal's ASify() as real
--> 3.688253867361

--List of Real Value
set aList to {10, 20, 30, 40, 50, 60, 70, 80, 90}
set aArray to (current application's SMSFord's acoshValueOf:aList)
set bList to aArray's ASify() as list
--> {2.993222846126, 3.688253867361, 4.094066668632, 4.38187034804, 4.605070170985, 4.787422291103, 4.941591398296, 5.075134750445, 5.192925985264}
<table>
<thead>
<tr>
<th>Method</th>
<th>logValueOf: ①</th>
</tr>
</thead>
<tbody>
<tr>
<td>Introduction</td>
<td>Pass a single number or a list of numbers. Results must be coerced using &quot;as real&quot; or &quot;as list&quot;. Angles are in radians, and errors are returned as missing value.</td>
</tr>
<tr>
<td>Kind</td>
<td>Trigonometry</td>
</tr>
<tr>
<td>Input</td>
<td>①AppleScript number (real) or AppleScript list</td>
</tr>
<tr>
<td>Output</td>
<td>NSNumber or NSArray</td>
</tr>
<tr>
<td>Notes</td>
<td></td>
</tr>
</tbody>
</table>

--Sample Code
-- Created 2014-11-30 by Takaaki Naganoya
-- 2014 Piyomaru Software
use AppleScript version "2.4"
use framework "Foundation"
use framework "ASObjCExtras"
use scripting additions

--Real Value
set aVal to current application's SMSFord's logValueOf:10
set bVal to aVal's ASify() as real
--> 2.302585092994

--List of Real Value
set aList to {10, 20, 30, 40, 50, 60, 70, 80, 90}
set aArray to (current application's SMSFord's logValueOf:aList)
set bList to aArray's ASify() as list
--> {2.302585092994, 2.995732273554, 3.401197381662, 3.68879454114, 3.912023005428, 4.094344562222, 4.248495242049, 4.382026634674, 4.49980967033}
method `log10ValueOf:`

Introduction

Pass a single number or a list of numbers. Results must be coerced using "as real" or "as list". Angles are in radians, and errors are returned as missing value.

kind Trigonometry

Input ①AppleScript number (real) or AppleScript list

Output NSNumber or NSArray

Notes

--Sample Code
-- Created 2014-11-30 by Takaaki Naganoya
-- 2014 Piyomaru Software
use AppleScript version "2.4"
use framework "Foundation"
use framework "ASObjCExtras"
use scripting additions

--Real Value
set aVal to current application's SMSFord's log10ValueOf:10
set bVal to aVal's ASify() as real
--> 1.0

--List of Real Value
set aList to {10, 20, 30, 40, 50, 60, 70, 80, 90}
set aArray to (current application's SMSFord's log10ValueOf:aList)
set bList to aArray's ASify() as list
--> {1.0, 1.30102995664, 1.47712125472, 1.602059991328, 1.698970004336, 1.778151250384, 1.845098040014, 1.90308986992, 1.954242509439}
ASObjCExtras.framework

Misc. methods
<table>
<thead>
<tr>
<th>method</th>
<th>metadataFromImage:① error:②</th>
</tr>
</thead>
<tbody>
<tr>
<td>Introduction</td>
<td>Pass an alias, file, HFS path, POSIX path or NSURL for an image file, and receive back a dictionary/record of the metadata.</td>
</tr>
<tr>
<td>kind</td>
<td>Misc.</td>
</tr>
<tr>
<td>Input</td>
<td>① AppleScript alias ② AppleScript missing value or reference</td>
</tr>
<tr>
<td>Output</td>
<td>NSDictionary</td>
</tr>
<tr>
<td>Notes</td>
<td></td>
</tr>
</tbody>
</table>

-- Sample Code
-- Created 2014-11-15 by Takaaki Naganoya
-- 2014 Piyomaru Software

use AppleScript version "2.4"
use scripting additions
use framework "Foundation"
use framework "ASObjCExtras"

set anAlias to choose file
set b to (current application's SMSFord's metadataFromImage:anAlias |error|: (missing value))
set c to b's ASify() as record

--Jpeg
--> {[jiffy]:{XDensity:1, YDensity:1, JFIFVersion:{1, 0, 1}, DensityUnit:0, ColorModel:"RGB", PixelHeight:1920, PixelWidth:1080, Depth:8}

--Jpeg (From digital Camera 1)

--Jpeg (From Digital Camera 2)
--> {[tiff]:{YResolution:72.0, ResolutionUnit:2, Software:"CX3 Firmware", DateTime:"2010:08:05 16:12:20", XResolution:72.0, ImageDescription:"Exif_JPEG_PICTURE", Orientation:1, Copyright:"", Model:"CX3", Imakel:"RICOH"}, ProfileName:"sRGB IEC61966-2.1", DPIWidth:72.0, {[exif]:{[DateTimeOriginal:"2010:08:05 16:12:20", ComponentsConfiguration:{1, 2, 3, 0}, MaxApertureValue:3.5, BrightnessValue:8.1, ExposureBiasValue:0.0, FNumber:5.1, CompressedBitsPerPixel:2.0, FocalLength:23.6, SceneCaptureType:0, ApertureValue:4.7, Sharpness:0, ColorSpace:1, PixelYDimension:480, WhiteBalance:0, UserComment:""}, XResolution:72.0, LightSource:0, FlashPixVersion:{1, 0}, DateTimeDigitized:"2010:08:05 16:12:20", ISOSpeedRatings:{400}, ExposureMode:0, ExifVersion:{2, 2, 1}, PixelXDimension:640, ExposureProgram:2, Flash:16, ExposureTime:0.001754385965, MeteringMode:5}, Depth:8, ColorModel:"RGB", PixelHeight:480, PixelWidth:640, Orientation:1, DPIHeight:72.0}}

--bmp

83 / 88
method infoForFile:

Takes an alias, file, HFS path, POSIX path or NSURL and returns a
dictionary/record. The labels can be found by looking up NSURL; the prefix
"NSURL" and suffix "Key" are removed, and the initial character made
lowercase.

-- Sample Code
-- Created 2014-11-15 by Takaaki Naganoya
-- 2014 Piyomaru Software

use AppleScript version "2.4"
use scripting additions
use framework "Foundation"
use framework "ASObjCExtras"

set anAlias to choose file
set b to (current application's SMSFord's infoForFile:anAlias)
set c to b's ASify() as record
-- {isAliasFile:false, fileResourceType:"NSURLFileResourceTypeRegular",
isExcludedFromBackup:false, contentModificationDate:date "Friday, January
9, 2015 at 8:24:50 PM", isReadable:true, isSystemImmutable:false,
isSymbolicLink:false, isExecutable:false, parentDirectoryURL:<class ocid id
«data optr0000000070109B40EA7F0000», hasHiddenExtension:true,
labelNumber:0, isWritable:true, |path|:"/Users/maro/Desktop/Screen Shot
2015-01-09 at 8.24.50 PM.png", isPackage:false, |name|:"Screen Shot
2015-01-09 at 8.24.50 PM.png", isDirectory:false, localizedName:"Screen
Shot 2015-01-09 at 8:24:50 PM", localizedTypeDescription:"Portable
Network Graphics image", linkCount:1, attributeModificationDate:date "Friday,
January 9, 2015 at 8:24:50 PM", creationDate:date "Friday, January 9, 2015
at 8:24:50 PM", isHidden:false, isRegularFile:true, contentAccessDate:date
"Friday, January 9, 2015 at 8:24:56 PM", isUserImmutable:false}
<table>
<thead>
<tr>
<th>method</th>
<th>sizeInfoForFile: ①</th>
</tr>
</thead>
<tbody>
<tr>
<td>Introduction</td>
<td>Takes an alias, file, HFS path, POSIX path or NSURL and returns a dictionary/record.</td>
</tr>
<tr>
<td>kind</td>
<td>Misc.</td>
</tr>
<tr>
<td>Input</td>
<td>AppleScript alias</td>
</tr>
<tr>
<td>Output</td>
<td>NSDitionary</td>
</tr>
<tr>
<td>Notes</td>
<td>For files, the values are returned like this: (totalFileSize:94927, totalFileAllocatedSize:98304, fileAllocatedSize:98304, fileSize:94927) The &quot;total&quot; values include metadata; &quot;Allocated&quot; means disk space used. Finder reports totalFileSize. For directories and packages, only the total values are returned: (totalFileSize:271230605, totalFileAllocatedSize:286121984).</td>
</tr>
</tbody>
</table>

-- Sample Code
-- Created 2014-11-15 by Takaaki Naganoya
-- 2014 Piyomaru Software

```applescript
use AppleScript version "2.4"
use scripting additions
use framework "Foundation"
use framework "ASObjCExtras"

set anAlias to choose file
set b to (current application's SMSFord's sizeInfoForFile:anAlias) as record
--> {totalFileSize:199432, totalFileAllocatedSize:200704, fileAllocatedSize:200704, fileSize:199432}
```
Piyomaru Software
A Professional AppleScript Developer in Tokyo, Japan

We develop applications/scripts with visual specification documents!

mailto: maro@piyocast.com
twitter: @Piyomaru
iMessage: piyomarusoft@mac.com

My speciality will be seen at my blog (http://piyocast.com/as)