AMAction Class Reference

Apple Applications > Automator

2007-03-01

Apple Inc.
© 2007 Apple Inc.
All rights reserved.

No part of this publication may be reproduced, stored in a retrieval system, or transmitted, in any form or by any means, mechanical, electronic, photocopying, recording, or otherwise, without prior written permission of Apple Inc., with the following exceptions: Any person is hereby authorized to store documentation on a single computer for personal use only and to print copies of documentation for personal use provided that the documentation contains Apple's copyright notice.

The Apple logo is a trademark of Apple Inc.

Use of the "keyboard" Apple logo (Option-Shift-K) for commercial purposes without the prior written consent of Apple may constitute trademark infringement and unfair competition in violation of federal and state laws.

No licenses, express or implied, are granted with respect to any of the technology described in this document. Apple retains all intellectual property rights associated with the technology described in this document. This document is intended to assist application developers to develop applications only for Apple-labeled computers.

Every effort has been made to ensure that the information in this document is accurate. Apple is not responsible for typographical errors.

Apple Inc.
1 Infinite Loop
Cupertino, CA 95014
408-996-1010

Apple, the Apple logo, AppleScript, Carbon, Mac, Mac OS, Objective-C, and Xcode are trademarks of Apple Inc., registered in the United States and other countries.

Simultaneously published in the United States and Canada.

Even though Apple has reviewed this document, APPLE MAKES NO WARRANTY OR REPRESENTATION, EITHER EXPRESS OR IMPLIED, WITH RESPECT TO THIS DOCUMENT, ITS QUALITY, ACCURACY, MERCHANTABILITY, OR FITNESS FOR A PARTICULAR PURPOSE. AS A RESULT, THIS DOCUMENT IS PROVIDED "AS IS," AND YOU, THE READER, ARE ASSUMING THE ENTIRE RISK AS TO ITS QUALITY AND ACCURACY.

IN NO EVENT WILL APPLE BE LIABLE FOR DIRECT, INDIRECT, SPECIAL, INCIDENTAL, OR CONSEQUENTIAL DAMAGES RESULTING FROM ANY DEFECT OR INACCURACY IN THIS DOCUMENT, even if advised of the possibility of such damages.

THE WARRANTY AND REMEDIES SET FORTH ABOVE ARE EXCLUSIVE AND IN LIEU OF ALL OTHERS, ORAL OR WRITTEN, EXPRESS OR IMPLIED. No Apple dealer, agent, or employee is authorized to make any modification, extension, or addition to this warranty.

Some states do not allow the exclusion or limitation of implied warranties or liability for incidental or consequential damages, so the above limitation or exclusion may not apply to you. This warranty gives you specific legal rights, and you may also have other rights which vary from state to state.

Contents

2007-03-01 | © 2007 Apple Inc. All Rights Reserved.

2007-03-01 | © 2007 Apple Inc. All Rights Reserved.

AMAction Class Reference

Inherits from	NSObject
Conforms to	NSObject (NSObject)
Framework	/System/Library/Frameworks/Automator.framework
Availability	Available in Mac OS X v10.4 and later.
Companion guide	Automator Programming Guide
Declared in	AMAction.h
Related sample code	Apply Firmware Password CoreRecipes UnsharpMask

Overview

`AMAction` is an abstract class that defines the interface and general characteristics of Automator actions. Automator is an Apple-provided application that allows users to construct and execute workflows consisting of a sequence of discrete modules called actions. An action performs a specific task, such as copying a file or cropping an image, and passes its output to Automator to give to the next action in the workflow. Actions are currently implemented as loadable bundles owned by objects of the `AMBundleAction` class, a subclass of `AMAction`.

The critically important method declared by `AMAction` is `runWithInput:fromAction:error:` (page 12). When Automator executes a workflow, it sends this message to each action object in the workflow (in workflow sequence), in most cases passing in the output of the previous action as input. The action object performs its task in this method and ends by returning an output object for the next action in the workflow.

Subclassing Notes

Subclassing `AMAction` is not recommended. For most situations requiring an enhancement to the Automator framework, it is sufficient to subclass `AMBundleAction` or one of its public subclasses, `AMAppleScriptAction` or `AMShellScriptAction`.

2007-03-01 | © 2007 Apple Inc. All Rights Reserved.

Tasks

Initialization and Encoding

- `initWithDefinition:fromArchive:` (page 9)
  Initializes the receiver with the specified definition.
- `initWithContentsOfURL:error:` (page 9)
  Loads an Automator action from a file URL.
- `writeToDictionary:` (page 14)
  Examines the parameters and other configuration information specified in the passed dictionary and add its own information to it if appropriate.
- `definition` (page 17) Deprecated in Mac OS X v10.4
  Returns the definition of the receiver. (Deprecated. Removed for performance reasons. There is no replacement.)

Controlling the Action

- `reset` (page 11)
  Resets the receiver to its initial state.
- `runAsynchronouslyWithInput:` (page 12)
  Causes Automator to wait for notification that the receiver has completed execution, which allows the receiver to perform an asynchronous operation.
- `runWithInput:fromAction:error:` (page 12)
  Requests the receiver to perform its task using the specified input from the specified action.
- `stop` (page 13)
  Stops the receiver from running.

Initializing and Synchronizing the Action User Interface

- `activated` (page 7)
  Invoked when the window of the Automator workflow to which the receiver belongs becomes the main window. This allows the receiver to synchronize its information with settings in another application.
- `opened` (page 10)
  Invoked when the receiver is first added to a workflow, allowing it to initialize its user interface.

Updating Action Parameters

- `parametersUpdated` (page 11)
  Requests the receiver to update its user interface from its stored parameters, which have changed.
- `updateParameters` (page 14)
  Requests the receiver to update its stored set of parameters from the settings in the action's user interface.

2007-03-01 | © 2007 Apple Inc. All Rights Reserved.

Getting Information About the Action

- `closed` (page 7)
  Invoked by Automator when the receiving action is removed from a workflow, allowing it to perform cleanup operations.
- `didFinishRunningWithError:` (page 8)
  Sent by the receiver to itself when it has finished running asynchronously.
- `ignoresInput` (page 8)
  Returns a Boolean value that indicates whether the action acts upon its input or the input is ignored.
- `name` (page 10)
  Returns the name of the action.
- `output` (page 10)
  Returns the receiver's output.
- `willFinishRunning` (page 14)
  Invoked by Automator when the receiver has essentially completed its run phase.

Instance Methods

activated

Invoked when the window of the Automator workflow to which the receiver belongs becomes the main window. This allows the receiver to synchronize its information with settings in another application.

```
- (void)activated
```

Discussion
Be sure to invoke the superclass implementation of this method as the last thing in your implementation.

Availability
Available in Mac OS X v10.4 and later.

See Also
- `opened` (page 10)

Declared In
`AMAction.h`

closed

Invoked by Automator when the receiving action is removed from a workflow, allowing it to perform cleanup operations.

```
- (void)closed
```

Discussion
This method is intended to be overridden, so that your action can perform its specific cleanup operations.

2007-03-01 | © 2007 Apple Inc. All Rights Reserved.

Availability
Available in Mac OS X v10.5 and later.

Declared In
AMAction.h

didFinishRunningWithError:

Sent by the receiver to itself when it has finished running asynchronously.

```
- (void)didFinishRunningWithError:(NSDictionary *)errorInfo
```

Parameters

errorInfo

If an error occurred during asynchronous running of the action, upon return contains an instance of NSError that describes the problem.

Discussion
An action that overrides runAsynchronouslyWithInput: (page 12) should invoke didFinishRunningWithError: on completion, so that Automator can resume running the workflow that the action is part of.

Availability
Available in Mac OS X v10.5 and later.

See Also
- willFinishRunning (page 14)

Declared In
AMAction.h

ignoresInput

Returns a Boolean value that indicates whether the action acts upon its input or the input is ignored.

```
- (BOOL)ignoresInput
```

Return Value
YES if the action acts upon its input, otherwise NO.

Discussion
Many actions act upon their input, but an action may merely pass on its input or, rarely, ignore it.

Special Considerations

Although this method was documented in Mac OS X version 10.4, and an action would respond to this message, the method was not made public, and using it would generate a warning in Xcode.

Availability
Available in Mac OS X v10.5 and later.

Declared In
AMAction.h

2007-03-01 | © 2007 Apple Inc. All Rights Reserved.

initWithContentsOfURL:error:

Loads an Automator action from a file URL.

```
- (id)initWithContentsOfURL:(NSURL *)fileURLerror:(NSError **)outError
```

Parameters

fileURL

URL that specifies the location of an action file.

outError

If no action is found or if an error occurs in initializing or running it, upon return contains an instance of `NSError` that describes the problem. For keys and error constants used with action errors, see *Automator Constants Reference*.

Return Value
The initialized action.

Discussion
This method is typically invoked by applications that use the `AMWorkflow` class to embed Automator workflows. It is used to allow creation of actions for a workflow.

Availability
Available in Mac OS X v10.5 and later.

Declared In
`AMAction.h`

initWithDefinition:fromArchive:

Initializes the receiver with the specified definition.

```
- (id)initWithDefinition:(NSDictionary *)dict fromArchive:(BOOL)archived
```

Parameters

dict

Describes the action, including any custom definition properties.

archived

If the receiver is being unarchived, `YES`, otherwise `NO`.

Return Value
The initialized action.

Discussion
This is the primary initializer for all Automator classes. The Automator application sends this message to instances of `AMAction` both when it loads actions bundles and when it unarchives them.

The `AMAction` object being instantiated should perform whatever initializations are necessary after invoking `super`'s implementation of this method. It can then examine the values in *dict*, particularly if the receiver had been archived with custom definition properties.

Availability
Available in Mac OS X v10.4 and later.

See Also
`- definition` (page 17)

2007-03-01 | © 2007 Apple Inc. All Rights Reserved.

- writeToDictionary: (page 14)

Declared In
AMAction.h

name

Returns the name of the action.

```
- (NSString *)name
```

Return Value
The name of the receiving action.

Availability
Available in Mac OS X v10.5 and later.

Declared In
AMAction.h

opened

Invoked when the receiver is first added to a workflow, allowing it to initialize its user interface.

```
- (void)opened
```

Discussion
You should perform all initializations of an action's user interface in this method and not in `awakeFromNib`. Be sure to invoke the superclass implementation of this method as the final step of your implementation.

Availability
Available in Mac OS X v10.4 and later.

See Also
- activated (page 7)

Declared In
AMAction.h

output

Returns the receiver's output.

```
- (id)output
```

Return Value
The receiving action's output, or `nil` if called before the action is run.

2007-03-01 | © 2007 Apple Inc. All Rights Reserved.

Discussion
This method is used in conjunction with the `AMWorkflow` class, which allows access to the actions in a workflow. Within a workflow, for example, you might iteratively inspect the output of each action. Or, on completion of a workflow, you might examine the output of the last action, to determine the output of the workflow.

Availability
Available in Mac OS X v10.5 and later.

Related Sample Code
AutomatorHandsOn
CoreRecipes
UnsharpMask

Declared In
`AMAction.h`

parametersUpdated

Requests the receiver to update its user interface from its stored parameters, which have changed.

```
- (void)parametersUpdated
```

Availability
Available in Mac OS X v10.4 and later.

See Also
- `updateParameters` (page 14)

Declared In
`AMAction.h`

reset

Resets the receiver to its initial state.

```
- (void)reset
```

Discussion
Resetting causes the action to release its output generated from the current execution of the workflow.

Availability
Available in Mac OS X v10.4 and later.

See Also
- `stop` (page 13)

Declared In
`AMAction.h`

2007-03-01 | © 2007 Apple Inc. All Rights Reserved.

runAsynchronouslyWithInput:

Causes Automator to wait for notification that the receiver has completed execution, which allows the receiver to perform an asynchronous operation.

```
- (void)runAsynchronouslyWithInput:(id)input
```

Parameters

input

The input for the action. Should contain one or more objects compatible with one of the types specified in the action's `AMAccepts` property.

Discussion

This method should be overridden only by actions that need to make asynchronous calls. After `runAsynchronouslyWithInput:` is invoked, Automator does not continue until the action invokes `didFinishRunningWithError:` (page 8). So in your override of this method, you can make an asynchronous call, wait to be notified of its completion, then invoke `didFinishRunningWithError:` to signal to Automator that the action has completed.

Warning: Failure to invoke `didFinishRunningWithError:` can cause a workflow to stall indefinitely.

For actions that do not need to make asynchronous calls, the preferred method is `runWithInput:fromAction:error:` (page 12).

Availability

Available in Mac OS X v10.5 and later.

Declared In

`AMAction.h`

runWithInput:fromAction:error:

Requests the receiver to perform its task using the specified input from the specified action.

```
- (id)runWithInput:(id)input fromAction:(AMAction *)anAction error:(NSDictionary
    **)errorInfo
```

Parameters

input

The input for the receiving action. Should contain one or more objects compatible with one of the types specified in the action's `AMAccepts` property.

By default, actions can only accept and provide the following types. However, by overriding this method, you can change the types your action can use:

- Objective-C actions: Accepts and provides types must inherit from com.apple.cocoa.string, com.apple.cocoa.path, com.apple.cocoa.url, or, starting in Mac OS X version 10.5 (v10.5), com.apple.cocoa.data.
- Shell script actions: Accepts and provides types must inherit from com.apple.cocoa.string or, starting in Mac OS X v10.5, com.apple.cocoa.data.
- AppleScript actions: Accepts and provides types must inherit from com.apple.applescript.object.

2007-03-01 | © 2007 Apple Inc. All Rights Reserved.

anAction

The action from which the *input* object was obtained.

errorInfo

If an error occurs, the action returns an error dictionary in this parameter. The keys and values for this dictionary are:

- `OSAScriptErrorNumber` (a string constant) — The value for this key is an instance of `NSNumber` whose integer value indicates an error code. See the header file `MacErrors.h` in the Carbon Core framework for a list of valid error codes, particularly the section on OSA errors.
- `OSAScriptErrorMessage` (a string constant) —The value for this key is an instance of `NSString` describing the error.

For an example of how to create such a dictionary, see "Implementing runWithInput:fromAction:error:" in "Implementing an Objective-C Action" in *Automator Programming Guide*.

Return Value

An object containing one or more objects of a data type compatible with a type specified in the receiving action's `AMProvides` property. If the receiver does not modify the data passed in *input*, it should return it unchanged.

Discussion

The input and output objects for actions are usually instances of `NSArray`. If the receiver encounters problems, it should return by indirection an error dictionary that describes the error.

This method is intended to be overridden. AppleScript actions, however, usually will not need to override this method because the same functionality is provided by an AppleScript script.

Availability

Available in Mac OS X v10.4 and later.

See Also

- `reset` (page 11)
- `runAsynchronouslyWithInput:` (page 12)
- `stop` (page 13)

Declared In

`AMAction.h`

stop

Stops the receiver from running.

```
- (void)stop
```

Discussion

The output acquired by the action during execution of the current workflow is still accessible to Automator.

Availability

Available in Mac OS X v10.4 and later.

See Also

- `reset` (page 11)

2007-03-01 | © 2007 Apple Inc. All Rights Reserved.

Declared In
AMAction.h

updateParameters

Requests the receiver to update its stored set of parameters from the settings in the action's user interface.

```
- (void)updateParameters
```

Discussion
This message is sent just before an action is saved, copied, or run. Preferably, an action's settings should not solely reside in the controls of its view, but if they do, the action can fetch and save them in this method.

Availability
Available in Mac OS X v10.4 and later.

See Also
- parametersUpdated (page 11)

Declared In
AMAction.h

willFinishRunning

Invoked by Automator when the receiver has essentially completed its run phase.

```
- (void)willFinishRunning
```

Discussion
This method is intended to be overridden. An action can use this method to perform cleanup operations, such as closing windows and deallocating memory.

Availability
Available in Mac OS X v10.5 and later.

Declared In
AMAction.h

writeToDictionary:

Examines the parameters and other configuration information specified in the passed dictionary and add its own information to it if appropriate.

```
- (void)writeToDictionary:(NSMutableDictionary *)dictionary
```

Parameters
dictionary
Possibly contains parameter and other configuration information about the receiver.

Discussion
Automator sends this message to an action object prior to archiving it. In its implementation of this method, the action object should first invoke the superclass implementation.

2007-03-01 | © 2007 Apple Inc. All Rights Reserved.

Availability
Available in Mac OS X v10.4 and later.

See Also
`- initWithDefinition:fromArchive:` (page 9)

Declared In
`AMAction.h`

2007-03-01 | © 2007 Apple Inc. All Rights Reserved.

2007-03-01 | © 2007 Apple Inc. All Rights Reserved.

Deprecated AMAction Methods

A method identified as deprecated has been superseded and may become unsupported in the future.

Deprecated in Mac OS X v10.4

definition

Returns the definition of the receiver. (Deprecated in Mac OS X v10.4. Removed for performance reasons. There is no replacement.)

```
- (NSMutableDictionary *)definition
```

Return Value
A mutable dictionary containing the current parameters of the action as well as other information.

Discussion
If your action has non-persistent data, it may override this method to append that data to the definition supplied by the superclass and return it.

Special Considerations

This method was removed to allow for performance improvements in the underlying implementation. There is no replacement.

Availability
Available in Mac OS X v10.4 and later.

Deprecated in Mac OS X v10.4.

Not available to 64-bit applications.

See Also
`- initWithDefinition:fromArchive:` (page 9)

Declared In
`AMAction.h`

2007-03-01 | © 2007 Apple Inc. All Rights Reserved.

2007-03-01 | © 2007 Apple Inc. All Rights Reserved.

Document Revision History

This table describes the changes to *AMAction Class Reference*.

Date	Notes
2007-03-01	Updated for methods added in Mac OS X v10.5.
2006-05-23	First publication of this content as a separate document.

2007-03-01 | © 2007 Apple Inc. All Rights Reserved.

Document Revision History

2007-03-01 | © 2007 Apple Inc. All Rights Reserved.

Index

2007-03-01 | © 2007 Apple Inc. All Rights Reserved.