
NSProcessInfo Class Reference

Data Management: Event Handling



2009-05-22



Apple Inc.
© 2009 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, Cocoa, Mac, Mac OS, and Quartz 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

NSProcessInfo Class Reference 5

Overview	5
Sudden Termination	5
Tasks	6
Getting the Process Information Agent	6
Accessing Process Information	7
Sudden Application Termination	7
Getting Host Information	7
Getting Computer Information	7
Class Methods	8
processInfo	8
Instance Methods	8
activeProcessorCount	8
arguments	9
disableSuddenTermination	9
enableSuddenTermination	9
environment	10
globallyUniqueString	10
hostName	11
operatingSystem	11
operatingSystemName	11
operatingSystemVersionString	12
physicalMemory	12
processIdentifier	12
processName	13
processorCount	13
setProcessName:	13
systemUptime	14
Constants	14
NSProcessInfo—Operating Systems	14

Document Revision History 17

Index 19

NSProcessInfo Class Reference

Inherits from	NSObject
Conforms to	NSObject (NSObject)
Framework	/System/Library/Frameworks/Foundation.framework
Availability	Available in Mac OS X v10.0 and later.
Companion guide	Interacting with the Operating System
Declared in	NSProcessInfo.h
Related sample code	CocoaEcho CocoaHTTPServer GLUT Quartz Composer WWDC 2005 TextEdit URL CacheInfo

Overview

The `NSProcessInfo` class provides methods to access information about the current process. Each process has a single, shared `NSProcessInfo` object, known as **process information agent**.

The process information agent can return such information as the arguments, environment variables, host name, or process name. The `processInfo` (page 8) class method returns the shared agent for the current process—that is, the process whose object sent the message. For example, the following line returns the `NSProcessInfo` object, which then provides the name of the current process:

```
NSString *processName = [[NSProcessInfo processInfo] processName];
```

The `NSProcessInfo` class also includes the `operatingSystem` (page 11) method, which returns an enum constant identifying the operating system on which the process is executing.

`NSProcessInfo` objects attempt to interpret environment variables and command-line arguments in the user's default C string encoding if they cannot be converted to Unicode as UTF-8 strings. If neither conversion works, these values are ignored by the `NSProcessInfo` object.

Sudden Termination

Mac OS X v10.6 includes a new mechanism that allows the system to log out or shut down more quickly by, whenever possible, killing applications instead of requesting that they quit themselves.

Your application can enable this capability on a global basis and then manually override its availability during actions that could cause data corruption or a poor user experience by allowing sudden termination. Alternately, your application can just manually enable and disable this functionality.

The methods [enableSuddenTermination](#) (page 9) and [disableSuddenTermination](#) (page 9) decrement or increment, respectively, a counter whose value is 1 when the process is first created. When the counter's value is 0 the application is considered to be safely killable and may be killed by the system without any notification or event being sent to the process first.

Your application can support sudden termination upon launch by adding a key to the application's `Info.plist`. If the `NSSupportsSuddenTermination` key exists in the `Info.plist` and has a value of `YES`, it is the equivalent of calling [enableSuddenTermination](#) (page 9) during your application launch. This renders the application process killable right away. You can still override this behavior by invoking [disableSuddenTermination](#) (page 9).

Typically, you will disable sudden termination whenever your application defers work that must be done before the application terminates. If, for example, your application defers writing data to disk, and sudden termination is enabled, you should bracket the sensitive operations with a call to [disableSuddenTermination](#) (page 9), perform the necessary operations, and then send a balancing [enableSuddenTermination](#) (page 9) message.

In agents or daemon executables that don't depend on Application Kit you can manually invoke [enableSuddenTermination](#) (page 9) right away. You can then use the enable and disable methods whenever the process has work it must do before it terminates.

Some Application Kit functionality automatically disables sudden termination on a temporary basis to ensure data integrity.

- `NSUserDefaults` temporarily disables sudden termination to prevent process killing between the time at which a default has been set and the time at which the preferences file including that default has been written to disk.
- `NSDocument` temporarily disables sudden termination to prevent process killing between the time at which the user has made a change to a document and the time at which the user's change has been written to disk.

Debugging tip: You can determine the value of the sudden termination using the following `gdb` command.

```
print (long)[[NSClassFromString(@"NSProcessInfo") processInfo]
_suddenTerminationDisablingCount
```

Do not attempt to invoke or override `suddenTerminationDisablingCount` (a private method) in your application. It is there just for this debugging purpose, and may disappear at any time.

Tasks

Getting the Process Information Agent

+ [processInfo](#) (page 8)

Returns the process information agent for the process.

Accessing Process Information

- [arguments](#) (page 9)
Returns the command-line arguments for the process.
- [environment](#) (page 10)
Returns the variable names and their values in the environment from which the process was launched.
- [processIdentifier](#) (page 12)
Returns the identifier of the process.
- [globallyUniqueString](#) (page 10)
Returns a global unique identifier for the process.
- [processName](#) (page 13)
Returns the name of the process.
- [setProcessName:](#) (page 13)
Sets the name of the process.

Sudden Application Termination

- [disableSuddenTermination](#) (page 9)
Disables the application for quickly killing using sudden termination.
- [enableSuddenTermination](#) (page 9)
Enables the application for quick killing using sudden termination.

Getting Host Information

- [hostName](#) (page 11)
Returns the name of the host computer.
- [operatingSystem](#) (page 11)
Returns a constant to indicate the operating system on which the process is executing.
- [operatingSystemName](#) (page 11)
Returns a string containing the name of the operating system on which the process is executing.
- [operatingSystemVersionString](#) (page 12)
Returns a string containing the version of the operating system on which the process is executing.

Getting Computer Information

- [physicalMemory](#) (page 12)
Provides the amount of physical memory on the computer.
- [processorCount](#) (page 13)
Provides the number of processing cores available on the computer.
- [activeProcessorCount](#) (page 8)
Provides the number of active processing cores available on the computer.
- [systemUptime](#) (page 14)
Returns the how long it has been since the computer has been restarted.

Class Methods

processInfo

Returns the process information agent for the process.

```
+ (NSProcessInfo *)processInfo
```

Return Value

Shared process information agent for the process.

Discussion

An [NSProcessInfo](#) (page 5) object is created the first time this method is invoked, and that same object is returned on each subsequent invocation.

Availability

Available in Mac OS X v10.0 and later.

Related Sample Code

CocoaEcho

CocoaHTTPServer

GLUT

Quartz Composer WWDC 2005 TextEdit

URL CacheInfo

Declared In

NSProcessInfo.h

Instance Methods

activeProcessorCount

Provides the number of active processing cores available on the computer.

```
- (NSUInteger)activeProcessorCount
```

Return Value

Number of active processing cores.

Availability

Available in Mac OS X v10.5 and later.

See Also

- [processorCount](#) (page 13)

Declared In

NSProcessInfo.h

arguments

Returns the command-line arguments for the process.

- (NSArray *)arguments

Return Value

Array of strings with the process's command-line arguments.

Availability

Available in Mac OS X v10.0 and later.

Related Sample Code

SimpleImageFilter

Declared In

NSProcessInfo.h

disableSuddenTermination

Disables the application for quickly killing using sudden termination.

- (void)disableSuddenTermination

Discussion

This method increments the sudden termination counter. When the termination counter reaches 0 the application allows sudden termination.

By default the sudden termination counter is set to 1. This can be overridden in your application Info.plist. See [“Sudden Termination”](#) (page 5) for more information and debugging suggestions.

Availability

Available in Mac OS X v10.6 and later.

See Also

- [enableSuddenTermination](#) (page 9)

Declared In

NSProcessInfo.h

enableSuddenTermination

Enables the application for quick killing using sudden termination.

- (void)enableSuddenTermination

Discussion

This method decrements the sudden termination counter. When the termination counter reaches 0 the application allows sudden termination.

By default the sudden termination counter is set to 1. This can be overridden in your application Info.plist. See [“Sudden Termination”](#) (page 5) for more information and debugging suggestions.

Availability

Available in Mac OS X v10.6 and later.

See Also

- [disableSuddenTermination](#) (page 9)

Declared In

NSProcessInfo.h

environment

Returns the variable names and their values in the environment from which the process was launched.

- (NSDictionary *)environment

Return Value

Dictionary of environment-variable names (keys) and their values.

Availability

Available in Mac OS X v10.0 and later.

Declared In

NSProcessInfo.h

globallyUniqueString

Returns a global unique identifier for the process.

- (NSString *)globallyUniqueString

Return Value

Global ID for the process. The ID includes the host name, process ID, and a time stamp, which ensures that the ID is unique for the network.

Discussion

This method generates a new string each time it is invoked, so it also uses a counter to guarantee that strings created from the same process are unique.

Availability

Available in Mac OS X v10.0 and later.

See Also

- [processName](#) (page 13)

Related Sample Code

Quartz Composer WWDC 2005 TextEdit

Declared In

NSProcessInfo.h

hostName

Returns the name of the host computer.

- (NSString *)hostName

Return Value

Host name of the computer.

Availability

Available in Mac OS X v10.0 and later.

Related Sample Code

CocoaEcho

CocoaSOAP

Declared In

NSProcessInfo.h

operatingSystem

Returns a constant to indicate the operating system on which the process is executing.

- (NSUInteger)operatingSystem

Return Value

Operating system identifier. See [“Constants”](#) (page 14) for a list of possible values. In Mac OS X, it's `NSMACHOperatingSystem`.

Availability

Available in Mac OS X v10.0 and later.

Declared In

NSProcessInfo.h

operatingSystemName

Returns a string containing the name of the operating system on which the process is executing.

- (NSString *)operatingSystemName

Return Value

Operating system name. In Mac OS X, it's @"NSMACHOperatingSystem"

Availability

Available in Mac OS X v10.0 and later.

Declared In

NSProcessInfo.h

operatingSystemVersionString

Returns a string containing the version of the operating system on which the process is executing.

- (NSString *)operatingSystemVersionString

Return Value

Operating system version. This string is human readable, localized, and is appropriate for displaying to the user. This string is *not* appropriate for parsing.

Availability

Available in Mac OS X v10.2 and later.

Declared In

NSProcessInfo.h

physicalMemory

Provides the amount of physical memory on the computer.

- (unsigned long long)physicalMemory

Return Value

Amount of physical memory in bytes.

Availability

Available in Mac OS X v10.5 and later.

Declared In

NSProcessInfo.h

processIdentifier

Returns the identifier of the process.

- (int)processIdentifier

Return Value

Process ID of the process.

Availability

Available in Mac OS X v10.0 and later.

See Also

- [processName](#) (page 13)

Related Sample Code

Quartz Composer WWDC 2005 TextEdit

Declared In

NSProcessInfo.h

processName

Returns the name of the process.

- (NSString *)processName

Return Value

Name of the process.

Discussion

The process name is used to register application defaults and is used in error messages. It does not uniquely identify the process.

Availability

Available in Mac OS X v10.0 and later.

See Also

- [processIdentifier](#) (page 12)
- [setProcessName:](#) (page 13)

Related Sample Code

GLUT

URL CacheInfo

Declared In

NSProcessInfo.h

processorCount

Provides the number of processing cores available on the computer.

- (NSUInteger)processorCount

Return Value

Number of processing cores.

Availability

Available in Mac OS X v10.5 and later.

See Also

- [activeProcessorCount](#) (page 8)

Declared In

NSProcessInfo.h

setProcessName:

Sets the name of the process.

- (void)setProcessName:(NSString *)name

Parameters*name*

New name for the process.

Discussion**Warning:** User defaults and other aspects of the environment might depend on the process name, so be very careful if you change it. Setting the process name in this manner is not thread safe.**Availability**

Available in Mac OS X v10.0 and later.

See Also- [processName](#) (page 13)**Declared In**

NSProcessInfo.h

systemUptime

Returns the how long it has been since the computer has been restarted.

- (NSTimeInterval)systemUptime

Return ValueAn `NSTimeInterval` indicating how long system the computer has been restarted.**Availability**

Available in Mac OS X v10.6 and later.

Declared In

NSProcessInfo.h

Constants

NSProcessInfo—Operating SystemsThe following constants are provided by the `NSProcessInfo` class as return values for [operatingSystem](#) (page 11).

```
enum {
    NSWindowsNTOperatingSystem = 1,
    NSWindows95OperatingSystem,
    NSSolarisOperatingSystem,
    NSHPUXOperatingSystem,
    NSMACHOperatingSystem,
    NSSunOSOperatingSystem,
    NSOSF1OperatingSystem
};
```

Constants

NSHPUXOperatingSystem

Indicates the HP UX operating system.

Available in Mac OS X v10.0 and later.

Declared in NSProcessInfo.h.

NSMACHOperatingSystem

Indicates the Mac OS X operating system.

Available in Mac OS X v10.0 and later.

Declared in NSProcessInfo.h.

NSOSF1OperatingSystem

Indicates the OSF/1 operating system.

Available in Mac OS X v10.0 and later.

Declared in NSProcessInfo.h.

NSSolarisOperatingSystem

Indicates the Solaris operating system.

Available in Mac OS X v10.0 and later.

Declared in NSProcessInfo.h.

NSSunOSOperatingSystem

Indicates the Sun OS operating system.

Available in Mac OS X v10.0 and later.

Declared in NSProcessInfo.h.

NSWindows95OperatingSystem

Indicates the Windows 95 operating system.

Available in Mac OS X v10.0 and later.

Declared in NSProcessInfo.h.

NSWindowsNTOperatingSystem

Indicates the Windows NT operating system.

Available in Mac OS X v10.0 and later.

Declared in NSProcessInfo.h.

Document Revision History

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

Date	Notes
2009-05-22	Updated for Mac OS X v10.6. Added description of sudden termination support.
2007-03-26	Updated for Mac OS X v10.5.
2006-05-23	First publication of this content as a separate document.

REVISION HISTORY

Document Revision History

Index

A

`activeProcessorCount` [instance method 8](#)
`arguments` [instance method 9](#)

D

`disableSuddenTermination` [instance method 9](#)

E

`enableSuddenTermination` [instance method 9](#)
`environment` [instance method 10](#)

G

`globallyUniqueString` [instance method 10](#)

H

`hostName` [instance method 11](#)

N

`NSHPUX0peratingSystem` [constant 15](#)
`NSMACH0peratingSystem` [constant 15](#)
`NSOSF10peratingSystem` [constant 15](#)
`NSProcessInfo—Operating Systems` [14](#)
`NSSolaris0peratingSystem` [constant 15](#)
`NSSunOS0peratingSystem` [constant 15](#)
`NSWindows950peratingSystem` [constant 15](#)
`NSWindowsNT0peratingSystem` [constant 15](#)

O

`operatingSystem` [instance method 11](#)
`operatingSystemName` [instance method 11](#)
`operatingSystemVersionString` [instance method 12](#)

P

`physicalMemory` [instance method 12](#)
`processIdentifier` [instance method 12](#)
`processInfo` [class method 8](#)
`processName` [instance method 13](#)
`processorCount` [instance method 13](#)

S

`setProcessName:` [instance method 13](#)
`systemUptime` [instance method 14](#)