

---

# NSProcessInfo Class Reference

[Cocoa](#) > [Process Management](#)



2007-03-26



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, Mac, and Mac OS are trademarks of Apple Inc., registered in the United States and other countries.

iPhone is a trademark of Apple Inc.

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
Tasks	6
Getting the Process Information Agent	6
Accessing Process Information	6
Getting Host Information	6
Getting Computer Information	6
Class Methods	7
processInfo	7
Instance Methods	7
activeProcessorCount	7
arguments	8
environment	8
globallyUniqueString	8
hostName	9
operatingSystem	9
operatingSystemName	9
operatingSystemVersionString	10
physicalMemory	10
processIdentifier	10
processName	11
processorCount	11
setProcessName:	11
Constants	12
NSProcessInfo—Operating Systems	12

## **Document Revision History 15**

---

## **Index 17**

---



# NSProcessInfo Class Reference

---

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

## 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 7) 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 9) 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.

## Tasks

### Getting the Process Information Agent

- + [processInfo](#) (page 7)  
Returns the process information agent for the process.

### Accessing Process Information

- [arguments](#) (page 8)  
Returns the command-line arguments for the process.
- [environment](#) (page 8)  
Returns the variable names and their values in the environment from which the process was launched.
- [processIdentifier](#) (page 10)  
Returns the identifier of the process.
- [globallyUniqueString](#) (page 8)  
Returns a global unique identifier for the process.
- [processName](#) (page 11)  
Returns the name of the process.
- [setProcessName:](#) (page 11)  
Sets the name of the process.

### Getting Host Information

- [hostName](#) (page 9)  
Returns the name of the host computer.
- [operatingSystem](#) (page 9)  
Returns a constant to indicate the operating system on which the process is executing.
- [operatingSystemName](#) (page 9)  
Returns a string containing the name of the operating system on which the process is executing.
- [operatingSystemVersionString](#) (page 10)  
Returns a string containing the version of the operating system on which the process is executing.

### Getting Computer Information

- [physicalMemory](#) (page 10)  
Provides the amount of physical memory on the computer.
- [processorCount](#) (page 11)  
Provides the number of processing cores available on the computer.
- [activeProcessorCount](#) (page 7)  
Provides the number of active processing cores available on the computer.

## 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

Quartz Composer WWDC 2005 TextEdit

TextEditPlus

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 11)

#### **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.

### 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 11)

### Related Sample Code

Quartz Composer WWDC 2005 TextEdit  
TextEditPlus



**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.

**Declared In**

NSProcessInfo.h

## operatingSystem

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

- (unsigned int)operatingSystem

**Return Value**

Operating system identifier. See [“Constants”](#) (page 12) for a list of possible values. In Mac OS X, it's `NSMACH0peratingSystem`.

**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 @"NSMACH0peratingSystem"

**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 11)

### Related Sample Code

Quartz Composer WWDC 2005 TextEdit  
TextEditPlus

### 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 10)
- [setProcessName:](#) (page 11)

### Related Sample Code

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 7)

### 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 11)

## Declared In

NSProcessInfo.h

# Constants

## NSProcessInfo—Operating Systems

The following constants are provided by the `NSProcessInfo` class as return values for [operatingSystem](#) (page 9).

```
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.

**Declared In**

NSProcessInfo.h



# Document Revision History

---

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

Date	Notes
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 7](#)  
`arguments` [instance method 8](#)

## E

---

`environment` [instance method 8](#)

## G

---

`globallyUniqueString` [instance method 8](#)

## H

---

`hostName` [instance method 9](#)

## N

---

`NSHPUX0peratingSystem` [constant 12](#)  
`NSMACH0peratingSystem` [constant 12](#)  
`NSOSF10peratingSystem` [constant 12](#)  
**NSProcessInfo—Operating Systems** [12](#)  
`NSSolaris0peratingSystem` [constant 13](#)  
`NSSunOS0peratingSystem` [constant 13](#)  
`NSWindows950peratingSystem` [constant 13](#)  
`NSWindowsNT0peratingSystem` [constant 13](#)

## O

---

`operatingSystem` [instance method 9](#)  
`operatingSystemName` [instance method 9](#)  
`operatingSystemVersionString` [instance method 10](#)

## P

---

`physicalMemory` [instance method 10](#)  
`processIdentifier` [instance method 10](#)  
`processInfo` [class method 7](#)  
`processName` [instance method 11](#)  
`processorCount` [instance method 11](#)

## S

---

`setProcessName:` [instance method 11](#)